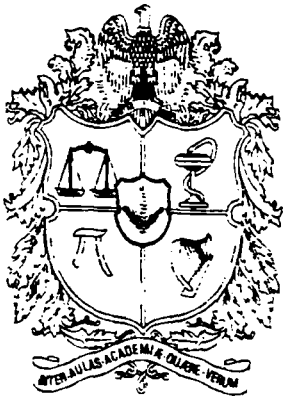
Universidad Nacional de Colombia
Sede Manizales

Facultad de Ciencias y Administración



PROGRAMACIÓN ESTRUCTURADA

UN ENFOQUE ALGORÍTMICO

ALONSO TAMAYO ALZATE
Profesor Asociado

I.S.B.N 958-9322-34-4

Autor:
Alonso Tamayo Alzate
Administrador de Empresas. Esp.
Profesor Asociado
Universidad Nacional de Colombia
Sede Manizales

Revisado por:
Eduardo Villegas Jaramillo
Ingeniero de Sistemas. Ms. Sc.
Profesor Asistente
Luis Fernando Montes López
Ingeniero Electricista
Instructor Asociado

Impreso por:
Centro de Publicaciones
Universidad Nacional de Colombia
Sede Manizales

Marzo de 1997
Primera Edición

CONTENIDO

PRÓLOGO 9
CAPÍTULO I. CONCEPTOS BÁSICOS
1.1 INTRODUCCIÓN 11
1.2 GENERALIDADES 12
1.3 DEFINICIÓN 16
1.4 CLASIFICACIÓN DE LOS ALGORITMOS 16
1.4.1 DIRECTOS 16
1.4.2 ITERATIVOS O CÍCLICOS 16
1.4.3 ANIDADOS 16
PROBLEMAS PROPUESTOS 18
1.5 RECOMENDACIONES A SEGUIR PARA LA SOLUCIÓN DE UN PROBLEMA 19
1.6 DIAGRAMAS DE FLUJO Y SEUDOCÓDIGO 21
1.6.1 DIAGRAMAS DE FLUJO 21
1.6.1.1 SIMBOLOGÍA 21
1.6.2 SEUDOCÓDIGO 25
1.7 CONSIDERACIONES ADICIONALES 25
1.7.1 CONSTANTE 25
1.7.2 VARIABLE 26
1.7.3 OPERACIÓN DE ASIGNACIÓN 26
1.7.4 OPERADORES ARITMÉTICOS 27
1.7.5 OPERADORES RELACIONALES 27
1.7.6 OPERADORES LÓGICOS 27
1.7.7 EXPRESIONES 28
1.7.8 EXPRESIÓN ARITMÉTICA 28
1.7.9 EXPRESIÓN LÓGICA 28
1.7.10 REGLAS PARA EXPRESIONES ARITMÉTICAS 28
1.7.11 JERARQUÍA DE LAS OPERACIONES ARITMÉTICAS 29
PROBLEMAS PROPUESTOS 30

CAPÍTULO II. ESTRUCTURAS ALGORÍTMICAS

2.1 INTRODUCCIÓN 33
2.2 SINTAXIS FORTRAN Y PASCAL 34
2.2.1 ELEMENTOS DE LENGUAJES DE PROGRAMACIÓN 34
2.2.2 INSTRUCCIONES DE ENTRADA Y SALIDA EN FORTRAN 38
2.2.3 INSTRUCCIONES DE ENTRADA Y SALIDA EN TURBO PASCAL 44
2.2.4 ESTRUCTURA GENERAL DE UN PROGRAMA FORTRAN 47
2.2.5 ESTRUCTURA GENERAL DE UN PROGRAMA TURBO PASCAL 48
2.3 ESTRUCTURAS ALGORÍTMICAS 54
2.3.1 ESTRUCTURA SECUENCIAL 54
2.3.2 ESTRUCTURAS CONDICIONALES 63
2.3.2.1 ESTRUCTURA IF_THEN (SI_ENTONCES) 64
2.3.2.2 ESTRUCTURA IF_THEN_ELSE (SI_ENTONCES_DE LO CONTRARIO) 68
2.3.2.2.1 ESTRUCTURA IF ANIDADA 77
2.3.2.3 ESTRUCTURA CASE (EN CASO DE_HACER) 91
PROBLEMAS PROPUESTOS 101
2.3.3 ESTRUCTURAS REPETITIVAS O CÍCLICAS 105
2.3.3.1 WHILE_DO (MIENTRAS_HACER) 105
2.3.3.1.1 ACUMULADOR 109
PROBLEMAS PROPUESTOS 115
2.3.3.1.2 CONTADOR 115
2.3.3.2 REPEAT_UNTIL (REPETIR HASTA_QUE) 125
2.3.3.3 FOR (PARA) 141
PROBLEMAS PROPUESTOS 156
2.3.4 ESTRUCTURAS REPETITIVAS ANIDADAS 160
2.4 REGISTRO DUMMY O REGISTRO CENTINELA 175
PROBLEMAS RESUELTOS 181
PROBLEMAS PROPUESTOS 208
EJERCICIOS COMPILADOS 214

CAPÍTULO III. ARREGLOS

3.1 INTRODUCCIÓN 225
3.2 DEFINICIÓN 226
3.3 ARREGLOS DE UNA DIMENSIÓN 226
PROBLEMAS RESUELTOS 247
PROBLEMAS PROPUESTOS 259
3.4 ARREGLOS DE DOS DIMENSIONES 263
PROBLEMAS RESUELTOS 273
PROBLEMAS PROPUESTOS 291
EJERCICIOS COMPILADOS 292

CAPÍTULO IV. DISEÑO DESCENDENTE
4.1 INTRODUCCIÓN 309
4.2 PROGRAMACIÓN MODULAR 310
4.3 CLASIFICACIÓN DE LOS MÓDULOS 311
4.3.1 FUNCIONES 311
4.3.2 PROCEDIMIENTOS 317
PROBLEMAS RESUELTOS 350
PROBLEMAS PROPUESTOS 374
EJERCICIOS COMPILADOS 376

BIBLIOGRAFÍA 441

PRÓLOGO

Bien es cierto que programar es una tarea compleja y exigente que involucra gran creatividad y dedicación, en donde no existe un conjunto de reglas, ni directrices que le indiquen a la persona cómo solucionar un problema para llevarlo al computador, ni solución única a un mismo problema y lo más interesante aún, es que toda solución presentada es susceptible de ser mejorada; me ha motivado a escribir este libro texto con el propósito de hacer de la programación una asignatura, además de formativa y apasionante, fácil de entender y aplicar.

En este material se hace énfasis en el empleo del lenguaje algorítmico, cuyo soporte fundamental es el seudocódigo en español, sin embargo también se utilizan diagramas de flujo como herramienta algorítmica, pero en justa medida, ya que considero que un problema solucionado en seudocódigo es más transparente e inmediato su traslado a un lenguaje de programación de alto nivel, que pasarlo directamente desde el diagrama de flujo.

Una de las inquietudes y expectativas cuando se está enseñando a programar, consiste en el deseo casi unánime de todos los participantes en interactuar con el computador lo más tempranamente posible, pero este deseo no es posible hacerlo realidad por el momento, ya que se debe teorizar varias horas en la enseñanza de algún lenguaje de programación de alto nivel, para que el estudiante tenga la posibilidad de llevar al computador aquel problema que ha solucionado con tanta dedicación y empeño, por lo tanto he querido acompañar el estudio algorítmico con una introducción a los lenguajes de programación Fortran /77 y Turbo Pascal, aportando los elementos esenciales para que el estudiante pueda correr en el computador sus propios algoritmos e induciéndolo a profundizar en el estudio de su sintaxis, para lo cual existe gran disponibilidad de material al respecto.

He plasmado en este libro mi amplia experiencia y metodología empleada en los salones de clase durante los cursos de programación de computadores, con el propósito de facilitarle al estudiante la comprensión de la asignatura mediante la resolución de

problemas, definiendo las entradas, procesos y salidas que se requieren en la solución de estos, para luego plantear el algoritmo bien sea en seudocódigo o en diagramas de flujo o en ambos, llegándose a presentar varias versiones algorítmicas de un mismo problema, con el fin único que el estudiante pueda apreciar más de una solución al mismo ejercicio y por último, para que el esfuerzo realizado se vea convertido en una realidad palpable, se presentan las compilaciones en Fortran y Pascal; en resumen se le muestra al estudiante la evolución que va teniendo el problema a través de sus diferentes etapas hasta llevarlo al computador, haciendo especial énfasis en la parte algorítmica y magistralmente llevado de una manera tal, que el tema sea asimilable a través de la variada gama de ejercicios explicados y desarrollados, llegando al punto de la redundancia, pero la experiencia ha demostrado la necesidad de ser reiterativos con el estudiante, sobre todo en temas que demandan de amplia comprensión y creatividad.

El Autor.

CAPITULO I

CONCEPTOS BÁSICOS

1.1 INTRODUCCIÓN
1.2 GENERALIDADES
1.3 DEFINICIÓN
1.4 CLASIFICACIÓN DE LOS ALGORITMOS
PROBLEMAS PROPUESTOS
1.5 RECOMENDACIONES A SEGUIR PARA LA SOLUCIÓN DE UN PROBLEMA
1.6 DIAGRAMAS DE FLUJO Y SEUDOCÓDIGO
1.7 CONSIDERACIONES ADICIONALES
PROBLEMAS PROPUESTOS

1.1 INTRODUCCIÓN

Este capítulo más que ocuparse de entrar directamente a profundizar en el estudio de las diversas herramientas para la construcción de algoritmos y programación, proporciona al estudiante una serie de recomendaciones y conceptos elementales pero importantes, partiendo entre otros de la definición y clasificación de los algoritmos, presentación de éstos en seudocódigo, diagramas de flujo, hasta cubrir elementos de programación como constantes, variables, expresiones, etc., que se convertirán en el soporte fundamental para la mejor comprensión de los próximos temas a ser tratados en este libro.

1.2 GENERALIDADES

La palabra Algoritmo tiene su origen en el nombre del matemático árabe del siglo IX llamado MOHAMMED AL-KHOWARIZMI, quien enunció las reglas a seguir para la realización de las operaciones aritméticas básicas como la suma, resta, multiplicación y división, así mismo desarrolló varios métodos para la solución de problemas con ecuaciones. La traducción al latín de su apellido "KHOWARIZMI" en "ALGORITMUS", derivó en la palabra ALGORITMO.

1.3 DEFINICIÓN

Algoritmo es un conjunto de acciones precisas y lógicas que se deben realizar en un orden determinado, para así obtener la solución a un problema en un número finito de pasos. Un ejemplo obvio de algoritmo lo constituye la apertura de una caja fuerte; o el seguimiento de los instructivos que traen consigo los juegos como el parqués, damas o ajedrez, entre otros, o aún más simple lo es una receta de cocina.
Un ejemplo sutil consiste en hacer una llamada telefónica, el cual se puede expresar en seis pasos sencillos a saber:

Paso 1 **INICIO**
Paso 2 Levantar la bocina
Paso 3 Marcar número
Paso 4 Hablar
Paso 5 Descargar bocina
Paso 6 **FIN**

Esta solución, aunque aparentemente buena, sólo es una aproximación, ya que no tiene presente sucesos como: no encontrar tono, estar ocupado el número solicitado ó simplemente no contestan la llamada.

Lo anterior conlleva a que un Algoritmo debe reunir las siguientes características:

- No se debe prestar a **ambigüedades**.
- El número de pasos debe ser cuantificable, es decir, **finito**.
- Debe alcanzar el fin propuesto, es decir, **efectivo**.

Una segunda versión del ejercicio se puede presentar así:

Paso 1 **INICIO**
Paso 2 Levantar bocina
Paso 3 Esperar hasta obtener tono
Paso 4 Marcar número
Paso 5 Esperar hasta que contesten la llamada
Paso 6 Hablar
Paso 7 Descargar la bocina
Paso 8 **FIN**

Pero se continúa con ambigüedades como se aprecia en los pasos 3 y 5. No se debe mantener levantada la bocina esperando tono, porque el tiempo transcurrido puede ser demasiado, por lo tanto no es práctico. Igualmente sucede con el paso 5, en donde no se debe esperar hasta que contesten la llamada porque se puede dar el caso de no recibir contestación.
Lo anterior obliga a analizar el ejercicio con un mayor nivel de profundidad, evitando así que se presenten situaciones indefinidas en un momento dado.

La tercera versión del ejercicio muestra un conjunto de actividades en operaciones más detalladas, buscando siempre dar mayor precisión y claridad a cada acción que se deba ejecutar.

Paso 1 **INICIO**
Paso 2 **Levantar bocina**
Paso 3 **¿Hay tono?**
No hay tono, repítase el siguiente proceso hasta lograrlo:
*** Descargar bocina**
*** Esperar un tiempo prudente**
*** Levantar bocina nuevamente**
Paso 4 **Marcar número**
Paso 5 **¿Contestan?**
No contestan, repítase el siguiente proceso hasta lograrlo:
*** Esperar un tiempo prudente**
*** Interrumpir comunicación**
*** ¿Hay tono?**
No hay tono, repítase el siguiente proceso:
* Descargar bocina

* Esperar un tiempo prudente
* Levantar bocina nuevamente
* Marcar número nuevamente

Paso 6 Hablar
Paso 7 Descargar bocina
Paso 8 **FIN**

En el anterior algoritmo aparece el concepto de **DECISIÓN** representado en las preguntas formuladas, las cuales tienen dos respuestas, una cierta o positiva y la otra falsa o negativa.

En el paso 3, si la respuesta es positiva continúa con el paso 4, en caso contrario aparece otro concepto y es el de **REPETICIÓN**, el cual agrupa una ó más actividades que se repiten hasta satisfacer una condición determinada.

El ejercicio anterior ofrece la solución al problema que se plantea con un mínimo de condiciones, pero no siempre se van a presentar problemas tan elementales y es así como el grado de complejidad aumenta, en la medida en que entran a participar de la problemática un mayor número de restricciones.

El ejercicio que se ha estado analizando, cambia ostensiblemente si se tienen en cuenta los siguientes interrogantes:

- ¿Es teléfono público ?
- ¿Se dispone de cambio suficiente ?
- ¿Es teléfono directo ?
- ¿Es llamada local ?

Como conclusión de lo que se ha visto hasta el momento, se puede resumir lo siguiente:

- Entre mayor conocimiento se tenga de una situación o problema, más facilidad se presentará en el momento de darle solución algorítmica.
- Todo algoritmo debe propender por brindar una solución precisa más no exacta a un problema o situación planteada.

EJERCICIO

Conocidos el nombre y la edad de una persona, imprimir el nombre de ella sólo si es mayor de edad.

Solución

a) Análisis

Para determinar si la persona es mayor de edad, se debe comparar la edad contra 18 años.

b) Algoritmo

Paso 1 **INICIO**
Paso 2 **Conocer** nombre y edad
Paso 3 ¿Es edad mayor de 18 años?
No, acabar el ejercicio
Sí, **Imprimir** el nombre
Paso 4 **FIN**

EJERCICIO

Conocido un número, calcular su raíz cuadrada. Si el número es negativo, imprimir el número y un mensaje que indique "tiene raíz imaginaria".

Solución

a) Análisis

Para determinar si una cantidad es negativa, es preciso compararla contra cero.

b) Algoritmo

Paso 1 **INICIO**
Paso 2 **Conocer** N
Paso 3 ¿Es N menor que cero?
Sí, **imprimir** N, "tiene raíz imaginaria"
No, RAIZ = $\sqrt{N}$
Imprimir N, RAÍZ
Paso 4 **FIN**

1.4 CLASIFICACIÓN DE LOS ALGORITMOS.

Los algoritmos se clasifican en tres grandes grupos a saber:

1.4.1 DIRECTOS: Conformado por el seguimiento de una serie de pasos elementales secuenciales, consistente en la mayoría de las veces de una entrada seguida de uno o más procesos, incluyendo decisiones y terminando con una salida.

1.4.2 ITERATIVOS O CÍCLICOS: Cuando se repite una serie de pasos o actividades hasta satisfacer una condición predeterminada.

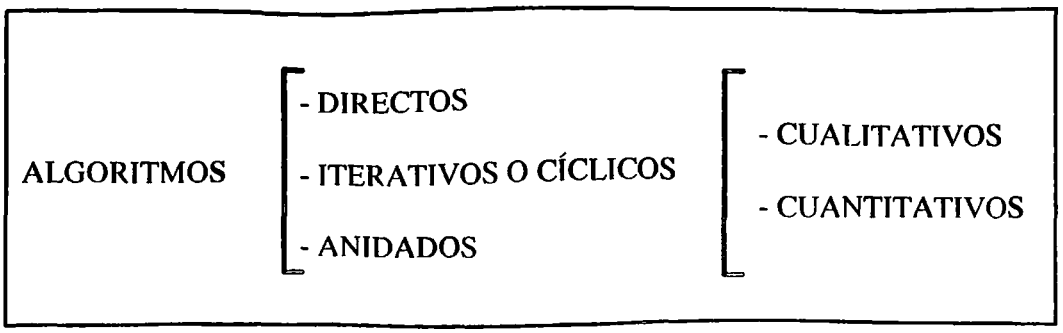
1.4.3 ANIDADOS: Cuando dentro de las instrucciones que comprende un ciclo, están incluidos a su vez otros ciclos o iteraciones.

La solución a un problema se puede presentar en forma cualitativa o cuantitativa, sin importar su clasificación.

Cualitativa: Son aquellos problemas que son solucionados a través de la descripción o cualificación de cada uno de sus pasos o actividades.

Cuantitativa: Son aquellas soluciones que son expresadas directamente en forma matemática; de hecho involucra cálculos numéricos.

CUADRO SINÓPTICO



Ejemplo de algoritmo cualitativo.

Enunciado: Elaborar un algoritmo para ingresar a la Universidad a realizar estudios superiores.

Solución

a) Algoritmo

Paso 1 **INICIO**
Paso 2 ¿Reúne todos los requisitos?
No, completar requisitos.
Paso 3 Comprar formulario
Paso 4 Diligenciar formulario
Paso 5 ¿Formulario bien diligenciado?
No, repetir proceso de diligenciamiento hasta que sea correcto.
Paso 6 Presentar examen de admisión
Paso 7 ¿Ganó examen?
No, repetir proceso el próximo semestre
Sí, ingresar a la Universidad
Paso 8 **FIN**

Ejemplo de algoritmo cuantitativo.

Enunciado: Conocidos dos números o cantidades, calcular la suma, resta y producto de ellos, e imprimir los resultados.

Solución

a) Algoritmo

Paso 1 **INICIO**
Paso 2 **Conocer** los números A y B
Paso 3 Calcular A + B y el resultado almacenarlo en C. ($C \leftarrow A + B$)
Paso 4 Calcular A - B y el resultado almacenarlo en D. ($D \leftarrow A - B$)
Paso 5 Calcular A * B y el resultado almacenarlo en E. ($E \leftarrow A * B$)
Paso 6 **Imprimir** el contenido de A, B, C, D, E
Paso 7 **FIN**

En los ejemplos anteriores se presentan dos algoritmos, uno cualitativo y el otro cuantitativo respectivamente, pero ambos son directos, puesto que consisten de proposiciones que son ejecutadas una a una secuencialmente y puede estar compuesta de una o varias entradas, seguida de uno o varios procesos y terminando con una o varias salidas.

Resumiendo los dos ejercicios anteriores se tiene:

	INGRESO A LA UNIVERSIDAD	**CONOCIDAS DOS CANTIDADES**
ENTRADA	Reunir requisitos.	Conocer los número A y B.
PROCESO	Comprar y diligenciar formulario. Presentar examen.	Calcular suma, resta y producto de A y B.
SALIDA	Ingresar o no a la universidad.	Imprimir A, B, C, D, E.

PROBLEMAS PROPUESTOS

1- Ordenar las siguientes actividades en forma lógica.

- Tiene cigarrillos ?
- Comprar fósforos.
- Empezar.
- Tiene fósforos ?
- Terminar.
- Tomar un cigarrillo.
- Encender fósforo y prender cigarrillo.
- Comprar cigarrillos.

2- Ordenar las siguientes actividades en la forma más conveniente, con el propósito de abastecer un carro con gasolina.

- ¿Desea llenar tanque de gasolina?
- ¿Desea gasolina corriente?
- Empezar.
- Llevar carro a la bomba de gasolina.
- Terminar.
- Pagar consumo de gasolina.

- Utilizar gasolina extra.
- Consumir lo deseado.

3- Elaborar un algoritmo para cambiar la llanta de un vehículo.

4- Elaborar un algoritmo para despachar una encomienda por correo.

5- Elaborar un algoritmo que indique como sembrar una planta en el jardín.

6- Leer un número y calcular el 7% del valor leído, imprimir tanto el número como el porcentaje calculado.

7- Leer dos valores en un registro e imprimir ambos valores sólo si son de diferente signo y distintos de cero.

8- Describir el proceso que se debe realizar cuando un cliente le paga a la cajera de un almacén, artículos por valor de $3.785 con un billete de $10.000

9- Elaborar un algoritmo para calcular el neto a pagar a un empleado de una compañía, en donde se conoce el valor pagado por cada día trabajado y el número de días laborados en el mes. Las deducciones corresponden al 3% del salario bruto.

10- Leer en un registro el código de un empleado, nombre, salario básico por hora, número de horas trabajadas en el período y porcentaje de retención en la fuente. Calcular el salario bruto, el valor de la retención en la fuente (5%) y el salario neto. Imprimir código del empleado, nombre, salario bruto, valor de la retención en la fuente y el salario neto.

11- Calcular el área del triángulo según las siguientes fórmulas:

$$AREA = \sqrt{S(S-A)(S-B)(S-C)}$$

$$donde\ S = \frac{A+B+C}{2}$$

1.5 RECOMENDACIONES A SEGUIR PARA LA SOLUCIÓN DE UN PROBLEMA

1- Entender el problema claramente.

Es necesario leer el enunciado del problema que se va a resolver, dos, tres o más veces hasta tanto se entienda. Sería inútil tratar de darle solución a un problema, cuando ni

siquiera el enunciado se entiende. Entre mayor claridad se tenga del enunciado, más facilidad se presentará en la solución.

2- Buscar la solución del problema.

- Determinar o definir la información requerida.
- Determinar los cálculos o cómputos necesarios.
- Definir la información que va a producir el sistema.

Se resumen estas tres actividades como:

Es necesario establecer prioridades, porque así como hay cálculos que se tendrán que realizar primero que otros, así mismo ocurrirá tanto para la información de entrada, como para la información de salida.

3- Definir variables.

Definir las variables que se van a utilizar en la solución del problema.

4- Diseñar la solución algorítmica.

5- Codificar la solución algorítmica en un lenguaje de programación.

Se debe seleccionar un lenguaje de alto nivel, bien puede ser FORTRAN, COBOL, PASCAL, C, etc., siempre buscando concordancia entre el tipo de problema a resolver y el lenguaje que pueda tener una mayor afinidad, como es el caso del FORTRAN que está orientado a la solución de problemas de ingeniería en donde su desempeño es muy fuerte, el COBOL que está orientado a la solución de problemas de gestión o comerciales y el PASCAL entre otros, que es de propósito múltiple.

6- Compilar el programa.

Consiste en determinar si la codificación está acorde con la sintaxis del compilador seleccionado; es decir, si se está corriendo un programa codificado en PASCAL, éste debe

estar ajustado a la sintaxis (conjunto de reglas y normas) del lenguaje PASCAL. Esta tarea la realiza el compilador, pero debe tenerse presente que el compilador chequea sólo la sintaxis de la codificación correspondiente, más no la lógica que implícitamente se plasma en la misma, es decir, una codificación puede tener cero errores de compilación y no funcionar correctamente por presentar problemas de lógica en el planteamiento de la solución o algoritmo.

Con base en lo explicado en el numeral sexto, se puede concluir que un **programa de computador** es un conjunto de instrucciones, que involucran la solución de un problema, las cuales son suministradas al computador conforme a la sintaxis del lenguaje de programación seleccionado.

7- Ejecutar el programa.

Cuando las etapas anteriores no presentan inconsistencias, se procede a alimentar el programa con los datos correspondientes, para así obtener la respuesta definitiva al problema planteado.

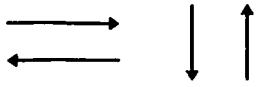
1.6 La solución de un problema se puede presentar de varias formas, sin embargo las más conocidas son mediante DIAGRAMAS DE FLUJO y SEUDOCÓDIGO.

1.6.1 DIAGRAMAS DE FLUJO.

Consiste en un conjunto de notaciones y símbolos geométricos que están comunicadas entre sí y que permiten expresar en forma clara y comprensible la solución de un problema.

1.6.1.1 Simbología.

Los símbolos empleados en los diagramas de flujo y de mayor frecuencia de uso son los siguientes:

FLECHA 

Permite efectuar uniones entre símbolos e indica la dirección del flujo. Es necesario marcar la punta de la flecha, para que no se presenten ambigüedades dentro del diagrama.

LÍNEA Sirve para unir dos símbolos.

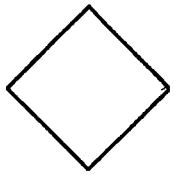
ÓVALO

Indica el comienzo y terminación del diagrama de flujo.
Generalmente se explica dentro del símbolo si se trata del comienzo o final, así:

RECTÁNGULO

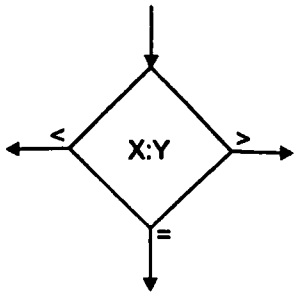
Se señala dentro de él todos los procesos, tanto cálculos como asignaciones, que se deben realizar.

ROMBO DE DECISIONES

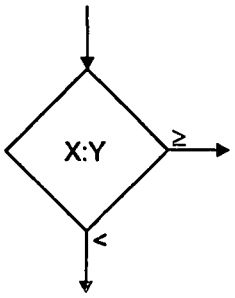


Se emplea para tomar decisiones, alterando la secuencia normal del flujo de datos. El rombo de decisiones puede ser aritmético o lógico.

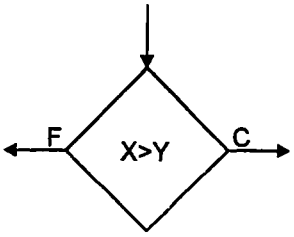
El rombo de decisiones artimético tiene tres salidas, así:



Los dos puntos ':' indica que se está comparando X contra Y. Los operadores relacionales se ubican por fuera del rombo.
El rombo de tres salidas se puede convertir en un rombo de dos salidas, así:



El rombo de decisiones lógico tiene dos salidas, SI o NO, CIERTO o FALSO, y el operador relacional se coloca dentro del rombo, así:



Usualmente la salida cierta de la comparación es direccionada hacia la derecha del rombo.

ENTRADA / SALIDA

Permite la entrada o salida de información, pero como el símbolo es genérico, se hace necesario señalar si se trata de entrada o de salida.

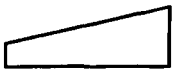
IMPRESORA

Se utiliza cuando la salida consiste de informes a ser escritos en papel.

PANTALLA

Se emplea para salidas orientadas a pantalla.

TECLADO



Permite la entrada de datos por teclado.

CONECTOR DE PÁGINA

Permite hacer conecciones dentro de una misma página.

CONECTOR FUERA DE PÁGINA

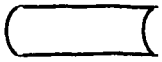
Permite efectuar conecciones fuera de página. Dentro de los conectores se coloca cualquier caracter y en el lugar donde se desea empalmar, se coloca otro conector con el mismo caracter antes indicado.

PROCESO PREDEFINIDO

Indica la utilización de funciones o procedimientos.

CINTA MAGNÉTICA

ARCHIVO DE ACCESO DIRECTO



ARCHIVO FUERA DE LÍNEA

DOCUMENTACIÓN

Sirve para hacer comentarios a un programa.

1.6.2 SEUDOCÓDIGO

Describe en lenguaje natural, mediante la utilización de ciertas palabras claves similares a las empleadas en los lenguajes de programación, la solución de un problema en forma clara, precisa y óptima.

Observación.

Los diagramas de flujo se diseñan de arriba hacia abajo y de izquierda a derecha, manteniendo el mismo estándar en la simbología empleada. El seudocódigo debe ser escrito con indentación para facilitar su comprensión y seguimiento.

Aunque los diagramas de flujo han sido empleados durante mucho tiempo para representar soluciones algorítmicas a problemas planteados, su utilización ha mermado ostensiblemente con la aparición de los lenguajes de programación estructurados, no obstante durante la presente obra serán aplicados en la resolución de ejercicios, de tal manera que el estudiante pueda apreciar tanto la solución a través de diagramas de flujo como en seudocódigo.

Aunque es posible que para algunas personas resulte el planteamiento de un algoritmo más comprensible cuando es expresado en diagramas de flujo que en seudocódigo, lo cierto es que la codificación de un algoritmo expresado en seudocódigo, es una tarea más fácil e inmediata, que hacerlo desde el diagrama de flujo.

1.7 CONSIDERACIONES ADICIONALES

1.7.1 CONSTANTE

Es una cantidad fija en donde el valor no cambia durante la ejecución del problema.

Ejemplo:

PI=3.14159 A=759 e=2.718182

1.7.2 VARIABLE

Representa una localización de memoria compuesta de un nombre y un valor. El valor que contiene la variable es susceptible de modificación; a su vez a las variables se les da un nombre para poderlas identificar. Este nombre tiene restricciones en cuanto a su longitud se refiere, por ejemplo, algunos lenguajes de programación (compiladores) aceptan variables máximo de 5 caracteres de longitud, otros de 6, otros de 8, etc., pero siempre el primer caracter debe ser una letra y no puede contener caracteres especiales como * , ; / + - () @ > ? $ etc., ni espacios en blanco. Lo recomendable en este caso es que debe hacer referencia con lo que representa, es decir, debe ser **nemotécnica.**

Ejemplos Válidos:

A	NETO	A124X
XYZ	SALARIO	X1Z
KARLA	PESO	K123

Ejemplos No válidos:

1XYZ	No comienza por letra
A$B	Contiene caracter especial
TOTAL NETO	Contiene espacio en blanco.

1.7.3 OPERACIÓN DE ASIGNACIÓN

Se produce cuando a una variable se le señala un valor determinado, bien puede ser por asignación directa, o como resultante de la evaluación de una expresión.

La operación de asignación se denota por el símbolo: ←, e indica que el valor o variable que está a la derecha del símbolo de asignación, se almacena en la variable que está a la izquierda del mismo símbolo. Si en lugar de tener un valor o variable tiene una expresión, ella se almacena de idéntica forma, una vez sea evaluada.

Ejemplos:

$K \leftarrow 3$	Indica que a la variable K se le ha asignado un valor de 3.
$C \leftarrow A + B$	Indica que el resultado de sumar A+B se almacene en la variable C.

$L \leftarrow M$	Indica que el contenido de la variable M se almacene en la variable L.
$Z \leftarrow Z + 1$	Indica que el valor de la variable Z se incremente en una unidad y el resultado se almacene nuevamente en la misma variable Z.

Al lado izquierdo del símbolo de asignación irá siempre una variable, no se admite una constante , ni tampoco otra expresión.

Ejemplos incorrectos:

$A + B * C \leftarrow D$

$57 \leftarrow E$

$X * Y \leftarrow Z + W$

1.7.4 OPERADORES ARITMÉTICOS

+	Suma ó adición
-	Resta ó sustracción
*	Multiplicación
/	División
↑	Potenciación ó Exponenciación

1.7.5 OPERADORES RELACIONALES

>	Mayor que
>=	Mayor o igual
<	Menor que
<=	Menor o igual
<>	No igual o diferente

1.7.6 OPERADORES LÓGICOS

AND	Conjunción lógica
OR	Disyunción lógica
NOT	Negación lógica

1.7.7 EXPRESIONES

Son constantes y variables unidas entre sí por operadores lógicos y aritméticos. Se usan para expresar cálculos y tienen como función asignar valores a una variable.

1.7.8 EXPRESIÓN ARITMÉTICA

Es una secuencia de constantes numéricas, variables y operadores aritméticos que indican la cantidad de cálculos a efectuar.

1.7.9 EXPRESIÓN LÓGICA

Está conformada por la combinación de elementos lógicos y operadores lógicos. Un elemento lógico puede ser una constante lógica, variable lógica, o expresión relacional, ésta última constituida por la combinación de dos expresiones aritméticas con un operador relacional.

1.7.10 REGLAS PARA EXPRESIONES ARITMÉTICAS

- Las cantidades pueden estar precedidas por los signos más o menos (+,-), ó pueden estar conectadas por cualquiera de los símbolos operacionales (*,/,+,-).
- Una expresión puede contener cantidades enteras, cantidades reales, o ambas.
- Dos símbolos de operación no deben aparecer consecutivamente.

Ejemplos:

Forma incorrecta	**Forma correcta**
A / - B	A / (-B)
A + - B	A + (-B)

- Los símbolos operacionales no son asumidos, por lo tanto, no deben aparecer consecutivamente.

Ejemplos:

Forma incorrecta	**Forma correcta**
3I	3 * I
A(B + C)	A * (B + C)
B↑2 - 4AC	B↑2 - 4 * A * C

- Los paréntesis sólo indican agrupación y no multiplicación.
- Las expresiones ambigüas se deben aclarar con el uso del paréntesis.

Ejemplo:
A^{BC} se expresa como $A \uparrow (B * C)$

1.7.11 JERARQUÍA DE LAS OPERACIONES ARITMÉTICAS

1. Evaluación de las operaciones encerradas entre paréntesis y dentro de éstos, los paréntesis más internos.
2. Evaluación de funciones.
3. Exponenciación o potenciación.
4. Multiplicación y división.
5. Suma y resta.

Cuando aparece en una expresión dos operaciones que son del mismo orden o prioridad, se procede a evaluar de izquierda a derecha.

Es de aclarar que no todos los lenguajes de programación (compiladores) proceden de acuerdo a ésta jerarquía, pero sí la gran mayoría.

Ejemplos:

Expresión algebraica	**Expresión computacional**
$A = \frac{X+Y}{C^5}$	$A \leftarrow (X+Y)/C \uparrow 5$
$D = \left(\frac{H}{5}\right)(X+Y+Z)$	$D \leftarrow H*(X+Y+Z)/5$
$L = 80W^2 + 60W - 5$	$L \leftarrow 80*W \uparrow 2 + 60*W - 5$
$Q = \frac{X^5}{5!}$	$Q \leftarrow X \uparrow 5/120$
$Y = \sqrt{B^2 - 4AC}$	$Y \leftarrow (B \uparrow 2 - 4*A*C) \uparrow 0.5$
$Y = \frac{\sqrt{A}}{\left(4+\sqrt{A}\right)}$	$Y \leftarrow A \uparrow (1/2)/(4 + A \uparrow (1/2))$
$Z = \frac{A+B}{7-C}$	$Z \leftarrow (A+B)/(7-C)$

PROBLEMAS PROPUESTOS

1. Reemplazar las siguientes expresiones algebraicas con expresiones computacionales:

a) $A = X + \frac{Y}{Z} + W$

b) $B = X + 1 + \frac{X^5}{5!} + \frac{X^7}{7!}$

c) $C = \left(\frac{X+Y}{X-Y}\right)^3 - 3$

d) $D = \frac{X+Y}{\left(\frac{W+A}{B+C}\right)}$

e) $E = \left(\frac{X}{Y}\right)^{W-3}$

2. Las siguientes expresiones computacionales corresponden a expresiones algebraicas, pero se encuentran erradas. Escribir las expresiones correctas.

Expresión computacional	**Expresión algebráica**
a) $X \leftarrow A/B+(C+D)\uparrow 2$	$X = \frac{A}{\left[B+(C+D)^2\right]}$
b) $Y \leftarrow A+7/B-3$	$Y = \frac{A+7}{B-3}$
c) $W \leftarrow (B+3.14159)/(B+7)\uparrow 5$	$W = \left(\frac{B+\pi}{B+7}\right)^5$
d) $Z \leftarrow 9/(1/X)-(4/Y)$	$Z = \frac{9}{\left[\frac{1}{X}-\frac{4}{Y}\right]}$
e) $X \leftarrow A\uparrow L+2+B\uparrow L+4$	$X = A^{L+2} + B^{L+4}$

3. EJERCICIO

Sea: X = 10; Y = 5; Z = 2; W = 3

Evaluar las siguientes expresiones:

a) $A \leftarrow X + Y \uparrow Z * W$

Solución

$A = 10 + 5^2 * 3$

$A = 10 + 25 * 3$

$A = 10 + 75$

$A = 85$

b) $B \leftarrow \big(X * (Y + Z)\big) / W$

Solución

$B = \big(10 * (5 + 2)\big) / 3$

$B = (10 * 7) / 3$

$B = 70 / 3$

$B = 23.23$

c) $C \leftarrow (X * Y - Z) / W$

Solución

$C = (10 * 5 - 2) / 3$

$C = (50 - 2) / 3$

$C = 48 / 3$

$C = 16$

Evaluar las siguientes expresiones:

d) $D \leftarrow X * (Y * Z / W)$

e) $E \leftarrow X * (Y - Z) / W$

f) $F \leftarrow X * Y - Z / W$

g) $G \leftarrow X * Y \uparrow Z / W$

h) $H \leftarrow X \uparrow Y * Z / W$

i) $I \leftarrow X * Y * Z / W$

j) $J \leftarrow X \uparrow (Y + Z) / W$

k) $K \leftarrow (X \uparrow Y) \uparrow Z / W$

4. Las siguientes expresiones algorítmicas corresponden a expresiones algebraicas. En caso de encontrar algún error, escriba la expresión correcta.

a) $\left(\frac{X}{Y}\right)^{R-1}$	$(X/Y) \uparrow R-1$
b) $\left(\frac{C+\pi}{C-9}\right)^{3}$	$(C+3.14159)/(C-9) \uparrow 3$
c) $3+\frac{A+5}{7}$	$3+A+5/7$
d) AZ^2+bY^2/c	$(A \uparrow Z \uparrow 2+b*Y \uparrow 2)/c$
e) $A^{I+2}+B^{I+3}$	$(A*(I+2)+B \uparrow (I+3))$

5. Escriba las expresiones algebraicas equivalentes para cada una de las siguientes expresiones algorítmicas.

a) L ← I / K * KI ↑ 3
b) X ← A + B / C ↑ 2 - D
c) Y ← A - B * C ↑ 3 / D
d) Z ← (B ↑ 2 - 4 * A * C) ↑ (1 / 2) / (2 * A)
e) A ← (C + 3.14159) / (C - 9) ↑ 3

CAPITULO II

ESTRUCTURAS ALGORÍTMICAS

2.1 INTRODUCCIÓN
2.2 SINTAXIS FORTRAN Y PASCAL
2.3 ESTRUCTURAS ALGORÍTMICAS
2.4REGISTRO DUMMY O REGISTRO CENTINELA
PROBLEMAS RESUELTOS
PROBEMAS PROPUESTOS
EJERCICIOS COMPILADOS

2.1 INTRODUCCIÓN

Todo problema de computación tiene dos fases muy demarcadas como son la solución del problema, que se describe por medio de un algoritmo y la puesta en marcha del mismo, que se expresa en un lenguaje de programación de alto nivel, convirtiéndose ésta en una labor sencilla y rutinaria, pero no tan importante como la primera etapa, porque el éxito o fracaso de una aplicación radica en el diseño algorítmico.

Dado lo anterior, se ha procurado no vincular el lenguaje algorítmico con un lenguaje de programación en particular o equipo de computación en específico, tratando de conservar siempre la transparencia, característica que debe mantener todo algoritmo, aunque existen influencias obvias, se ha dejado la decisión relacionada con la escogencia del lenguaje de programación al usuario final.

Adicionalmente se incorporó en este capítulo la introducción a la sintaxis de los lenguajes de programación FORTRAN 77 y TURBO PASCAL, lenguajes de una amplia difusión a nivel académico debido a la estructuración y fortalezas que presentan, permitiendo la solución de gran variedad de problemas de toda índole tanto ingenieriles como de gestión.

2.2 SINTAXIS FORTRAN Y PASCAL

Para obtener soluciones más cercanas al computador, es necesario conocer la sintaxis de los lenguajes de alto nivel, por lo tanto, a partir del momento se explicarán los elementos básicos de los lenguajes Fortran y Pascal en la medida que se requieran y se presentará la codificación de gran cantidad de los algoritmos planteados, permitiéndole al estudiante confrontar sus conocimientos a través de la corrida de ejercicios en el computador.

2.2.1 ELEMENTOS DE LENGUAJES DE PROGRAMACIÓN

Operaciones aritméticas	**Fortran/77**	**Turbo Pascal**
Suma	+	+
Resta	-	-
Multiplicación	*	*
División	/	/ División real
		div División entera

Los compiladores tanto de Fortran como de Pascal, ofrecen funciones internas para obtener raíz cuadrada y exponente al cuadrado de la siguiente forma:

Función	**Fortran/77**	**Turbo Pascal**
Exponente cuadrado		SQR(X)
Raíz cuadrada	SQRT(X)	SQRT(X)

Como Pascal no tiene un operador aritmético que permita elevar a una potencia diferente al cuadrado, se hace necesario obtenerla de la siguiente forma:

POTENCIA = Exp(N*(ln(I)))
siendo ln : función interna logaritmo natural
N : exponente a ser elevado
I : base

Operadores relacionales	Fortran/77	Turbo Pascal
Mayor que	.GT.	>
Mayor o igual que	.GE.	>=
Menor que	.LT.	<
Menor o igual que	.LE.	<=
Igual	.EQ.	=
No igual o diferente	.NE.	<>

Operadores lógicos	Fortran/77	Turbo Pascal
Conjunción lógica	.AND.	AND
Disyunción lógica	.OR.	OR
Negación lógica	.NOT.	NOT

TIPOS DE DATOS EN FORTRAN

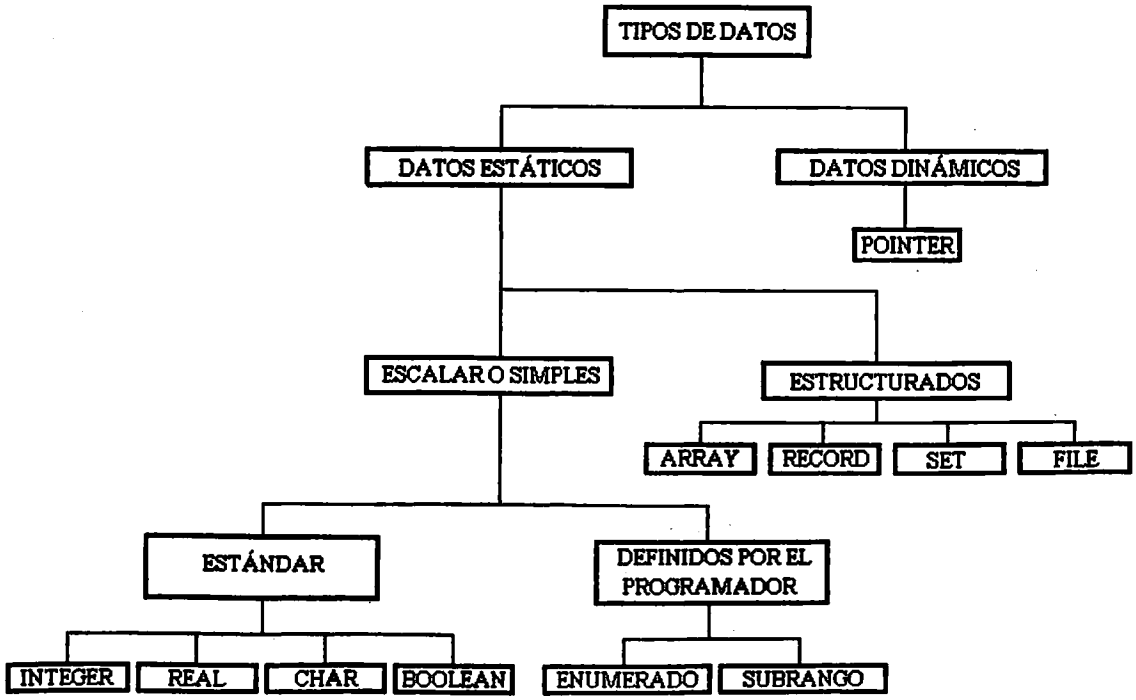
- Cuando el primer caracter de una variable comienza por cualquiera de las siguientes letras I,J,K,L,M,N, la variable será **Entera** y solamente podrá tomar valores enteros, siendo sus máximos valores:
  32767 (ó 2^{15}-1) y -32768 (ó -2^{15}) en precisión normal,
  2^{31} y -2^{31} en precisión doble.
- Cuando el primer caracter de una variable comienza por cualquier letra comprendida entre los intervalos (A...H) o (O...Z), la variable será **Real** y podrá tomar valores comprendidos entre 10^{75} y -10^{78}.
- Las variables **Lógicas** son aquellas que contienen expresiones lógicas y se definen con la proposición LOGICAL.
- Las variables pueden ser **Alfanuméricas** y se definen con la proposición explícita CHARACTER.
- El número máximo de caracteres que conforman el nombre de una variable es de ocho.

Lo que en Fortran se denomina Variable, en Pascal se le conoce como Identificador, que no es más que una posición de memoria que se le llama por su nombre y está en capacidad de almacenar un valor.

Tanto los compiladores Fortran como Pascal no admiten la utilización de palabras reservadas como definición de variables. Ejemplo: PROGRAM, BEGIN, STOP, IF, FOR, DO, WHILE, REPEAT, GOTO ... etc.

En Pascal:
La longitud de una línea de programa en Turbo Pascal puede contener hasta 128 caracteres, sin embargo, sólo los 63 primeros caracteres son significativos.

TIPOS DE DATOS EN TURBO PASCAL



En Pascal existen 5 tipos predefinidos de datos enteros, ellos son:

- *Byte*: son los números comprendidos entre 0 y 255, y requieren de 1 byte para poder ser representados en memoria.
- *Integer*: son los números comprendidos entre -32767 y 32768, y requieren de 2 bytes para ser representados en memoria.
- *Longint*: es una ampliación del rango de los datos enteros y su valor oscila entre
  -2147483648 y 2147483648; requiere de 4 bytes para ser representado en memoria.
- *Shortint*: pueden tomar valores entre -128 y 127 y requiere de 1 byte para ser representado en memoria.
- *Word*: pueden tomar valores entre 0 y 65535. Permiten accesar una dirección de memoria desde programa y requiere de 2 bytes de memoria para su representación.

Otros tipos de datos estándar son:

- *Real*: pueden tomar valores entre 2.9E-39 y 1.7E38 y requieren de 6 bytes para ser representados en memoria.
- Existen otros 4 tipos de datos reales, pero requieren de una versión de Turbo Pascal 5.0 en adelante o disponer de coprocesador matemático, ellos son:

- SINGLE: son los números comprendidos entre 1.5E-45 y 3.4E38; requiere de 4 bytes para ser representados en memoria.
- DOUBLE: son aquellos valores comprendidos entre 5.0E-307 y 1.7E307; requiere de 8 bytes para ser representados en memoria.
- EXTENDED: pueden tomar valores entre 1.9E-4932 y 1.1E4932. Requiere de 10 bytes para ser representados en memoria.
- COMP: pueden tomar valores entre -9.2E18 y 9.2E18. Requiere de 8 bytes para ser representados en memoria.

- *Boolean*: pueden tomar valores verdadero o falso. Requiere de 1 byte para ser representado en memoria.
- *Char*: está compuesto por todos los caracteres ASCII comprendidos entre 0 y 255. Requiere de un byte de memoria para su representación. Sólo puede contener un caracter, el cual va encerrado entre comillas simples. Cuando es necesario emplear más de un caracter, se requiere definir una cadena de caracteres.

  - CADENA DE CARACTERES (STRING).

  Es una secuencia de caracteres cuya longitud máxima puede ser de 255 caracteres y van encerrados entre comillas simples. Cada caracter ocupa un byte de memoria. La cadena de caracteres se define utilizando la palabra reservada STRING y la cantidad de caracteres a ser empleada en el programa.

  Ejemplo:

  'UNIVERSIDAD NACIONAL DE COLOMBIA', ocupa 32 bytes de memoria.
  'MANIZALES', ocupa 9 bytes de memoria.

Asignación

Para asignar un valor a una variable o identificador, se utiliza uno de los siguientes símbolos:

Algoritmo	**Fortran/77**	**Turbo Pascal**
←	=	:=

Delimitadores

Las instrucciones tanto de Fortran como de Pascal deben estar delimitadas unas de otras y se logra de la siguiente manera:

Fortran/77	**Turbo Pascal**
Un final de línea	Un final de línea. Un punto y coma. Un comentario.

Comentarios

Provee al programa fuente de la documentación explicativa necesaria como objetivos y procedimientos, para que sea más comprensible y fácil de hacerle seguimiento por parte del programador o cualquier persona que pueda interesarse en lo que hace el programa.

Los comentarios no son ejecutables, por lo tanto no alteran la lógica del programa; pero son de gran ayuda y se recomienda su incorporación en la confección de todo programa.

En Fortran se coloca una C o * en la primera columna y a continuación el comentario deseado. Si el comentario es muy extenso, se emplea tantas líneas como sea necesario colocando la C o * al comienzo de cada línea.

En Pascal el comentario va encerrado entre llaves {} o (* *) y no lleva punto y coma (;) al final del mismo.

2.2.2 INSTRUCCIONES DE ENTRADA Y SALIDA EN FORTRAN

READ(*,F)LISTA
WRITE(*,F)LISTA

READ : permite el ingreso de datos al sistema.
WRITE : produce la salida de información del sistema.
* : unidad lógica del dispositivo de lectura o escritura.
F o un : número de la instrucción FORMAT que describe los datos a ser leídos escritos. Debe ser una constante entera, sin signo y diferente de cero.
LISTA : lista de entrada o salida, que contiene el nombre de variables, nombre de arreglo o elementos de un arreglo (ver capítulo 3, por favor).

La finalidad de la proposición FORMAT es describir la forma como se están leyendo o escribiendo los datos. Así:

Formato	Tipo	Forma general
F	Real	Fw.d
G	Real	Gw.d
E	Exponencial de un número real	Ew.d
I	Entero	Iw
A	Caracter	Aw

donde: w: tamaño del campo.
d: posición hacia la derecha del punto decimal.

W debe contener los espacios suficientes para la parte entera, parte decimal, el punto decimal, y si el valor es negativo para el signo.

Ejemplo:

```
      READ(*,100)RADIO
100   FORMAT(F5.2)
```

Indica que se debe leer la variable RADIO por el teclado con el formato F5.2, donde 5 es la longitud total de la cantidad a ser leída y 2 son los decimales que contiene esa cantidad.

Ejemplo:

```
      OPEN(7,FILE='LISTA.DAT')
      READ(7,170)X,Y,Z
170   FORMAT(2F5.2,F10.2)
```

En este caso las variables X,Y,Z se leen de un archivo llamado LISTA.DAT. Las variables X,Y se leen con el mismo formato F5.2, y la variable Z se lee con el formato F10.2.

Ejemplo:

```
      WRITE(*,100)AREA,RADIO
100   FORMAT(F8.2,F7.3)
```

En este caso la salida de información se produce a través del monitor o pantalla, siendo leída la variable AREA con formato F8.2, y la variable RADIO con formato F7.3.

Es posible leer o imprimir con formato libre de la siguiente forma:

```
READ(*,*)A,B
WRITE(*,*)A,B
```

El primer asterisco indica que las variables A,B se leerán por teclado y se imprimirán por pantalla. El segundo asterisco indica que se leerán o imprimirán con formato libre.

FORMATOS DE EDICIÓN

Las comillas simples en una instrucción WRITE sirven para escribir títulos.

Ejemplo:

```
      WRITE(*,1)
1     FORMAT('  UNIVERSIDAD NACIONAL' // 'SEDE MANIZALES   ')
```

Las diagonales (//) sirven para dejar líneas entre títulos.

Formato X: permite ignorar grupos de caracteres durante operaciones de lectura o generar espacios en blanco durante operaciones de escritura.

Ejemplo:

```
      WRITE(*,3)A,J
3     FORMAT(30X,F7.2,5X,I4)
```

Salta 30 espacios e imprime la variable A con formato F7.2, luego deja 5 espacios e imprime la variable J con formato I4.

Formato T: permite leer o escribir en cualquier posición de un registro.

Ejemplo:

```
      WRITE(*,200)K
200   FORMAT(T30,I5)
```

Deja 30 espacios e imprime la variable K con formato I5.

Observación

Cuando se edita una línea, el primer caracter de cada línea es utilizado por la impresora para el control de carro, si no se utiliza el formato respectivo.
En los dos ejemplos anteriores cuando se utilizó tanto el formato X como el formato T, los espacios reales dejados al comienzo de la línea no fueron 30 sino 29, ya que el primer caracter fue utilizado por la impresora como control de carro.

Formato de control de carro de las impresoras:

'*b*' o 1H*b* Avanza una línea antes de imprimir.
'ϕ' o 1Hϕ Avanza dos líneas antes de imprimir.
'1' o 1H1 Avanza a la primera línea de la página siguiente.
'+' o 1H+ No avanza. Se emplea para sombrear o repisar.

Ejemplo:

```
      WRITE(*,15)
15    FORMAT(1H1,20X,'UNIVERSIDAD NACIONAL'//26X,'SEDE MANIZALES')
```

Avanza a la primera línea de la página siguiente, deja 20 espacios e imprime UNIVERSIDAD NACIONAL; luego deja una línea en blanco, se desplaza 25 espacios

e imprime SEDE MANIZALES.

Observación

Si N diagonales consecutivas aparecen al comienzo o al final de una instrucción FORMAT, ellas son totalmente efectuadas; si aparecen en cualquier otro lugar de la instrucción FORMAT, el número de diagonales realizadas es N-1.

Ejemplo:

```
      WRITE(*,5)KOD,PP,SP,EF
5     FORMAT('1' /// 20X, 'CÓDIGO', 5X, 'PRIMER PARCIAL', 2X, 'SEGUNDO
     *PARCIAL', 2X, 'EXAMEN FINAL' /// 20X, I6, 9X, F3.1, 13X, F3.1, 13X, F3.1)
```

Produce la siguiente salida:

Deja 3 líneas en blanco

CÓDIGO PRIMER PARCIAL SEGUNDO PARCIAL EXAMEN FINAL

Deja 2 líneas en blanco

XXXXXX X.X X.X X.X

Observación

En Fortran la codificación se efectúa así: de la columna 1 a la columna 5 inclusive, se coloca el número de identificación de instrucciones (etiqueta) cuando la lleva, la columna 6 se deja en blanco y puede ser utilizada marcando cualquier caracter en ella (excepto ϕ o espacio en blanco) para indicar continuación de la instrucción anterior, como sucede en la instrucción FORMAT del ejemplo previo, y las proposiciones se codifican de la columna 7 a la columna 72 inclusive. Una línea de codificación sólo debe contener una instrucción.

Diagonales consecutivas (/) producen líneas en blanco cuando se utilizan en instrucciones WRITE, o permiten saltar registros cuando se aplican en las instrucciones READ.

Ejemplo:

```
      WRITE(*,1)A,I
1     FORMAT(/// F5.2 // I4 // )
```

Produce tres líneas en blanco, luego imprime en otra línea los datos correspondientes a la

variable A, deja una línea en blanco e imprime los datos de la variable I y finalmente deja dos líneas en blanco.

La contradiagonal (\ ) permite ubicar el cursor al frente del título.

Ejemplo:

```
      WRITE(*,10)
10    FORMAT(10X, 'ESCRIBA EL VALOR DE A' \ )
      READ(*,20)A
20    FORMAT(F10.2)
```

Ejemplo:

```
      READ(*,30)P,Q,I,R,S,J,T,U,K
30    FORMAT(F7.2 / F5.0 / I3)
```

Se leen nueve registros, el primero con el valor de la variable P especificada con formato F7.2, el segundo registro conteniendo el valor de la variable Q con formato F5.0 y así sucesivamente hasta llegar al noveno registro donde se lee el valor correspondiente a la variable K con formato I3, repitiéndose automáticamente el formato hasta leer todas las variables.

Ejemplo:

```
      WRITE(*,5)KOD,PP,SP,EF
5     FORMAT('1'/// 20X, 'CÓDIGO', 13X, I6 // 20X, 'PRIMER PARCIAL', 9X, F3.1
     *// 20X, 'SEGUNDO PARCIAL', 7X, F3.1 // 20X, 'EXAMEN FINAL', 10X, F3.1)
```

Produce la siguiente salida:

```
CÓDIGO              XXXXXX
PRIMER PARCIAL         X.X
SEGUNDO PARCIAL        X.X
EXAMEN FINAL           X.X
```

El asterisco ubicado en la columna sexta indica continuación de la instrucción FORMAT.

Ejemplo:

```
      WRITE(*,7)A,I,B,J,C,K,D,L
7     FORMAT(4(F8.2,I3))
```

Indica que existen cuatro grupos sucesivos de datos compuestos de un valor real, seguido de un valor entero.

Ejemplo:

```
      WRITE(*,8)A,B,C,D,I,J,K,L
8     FORMAT(4F8.2,4I3)
```

Se imprimen 4 variables reales consecutivas con formato F8.2, seguidá de 4 variables enteras que se imprimen con el mismo formato I3.

Ejemplo:

```
      READ(*,20)A,B,C
20    FORMAT(F7.2)
```

Se están leyendo 3 variables, una por registro, con el formato F7.2. Aquí el formato se repite tantas veces como variables estén especificadas en las proposiciones de lectura o escritura.

Ejemplo:

```
      READ(*,6)A,I,B,J
6     FORMAT(F7.0,I3)
```

Se leen dos registros, en el primero los valores correspondientes a las variables A,I y en el segundo registro los valores de las variables B,J.

Ejemplo:

```
      WRITE(*,3)I,A,J,B,K,C,L,D
3     FORMAT(I2,F7.2,(I3,F3.0))
```

Se imprime la variable I con formato I2, luego la variable A con formato F7.2, seguido de la variable J con formato I3, luego la variable B con formato F3.0. De aquí en adelante el formato que está encerrado en el paréntesis interno se repite tantas veces como variables estén especificadas en las instrucciones de lectura o escritura.

Ejemplo:

```
      READ(*,5)AB,IJ,AC,IL
5     FORMAT((E12.3,I5),F7.2)
```

La variable AB se está leyendo con formato E12.3, la variable IJ con formato I5, la variable AC con formato E12.3 y la variable IL con formato I5. El formato F7.2 no es ejecutado, ya que toda especificación a la derecha de un formato repetitivo, no es tenida en cuenta.

2.2.3 INSTRUCCIONES DE ENTRADA Y SALIDA EN TURBO PASCAL

READ(Lista de identificadores);
READLN(Lista de identificadores);

READ, READLN:	Indican entrada de datos al sistema y se hace un valor cada vez, es decir, para cada valor introducido se debe pulsar la tecla <ENTER>.
READ:	Mantiene el cursor después del último caracter introducido.
READLN:	Envía el cursor al comienzo de la siguiente línea una vez pulsada la tecla <ENTER>.
LISTA DE IDENTIFICADORES:	Se colocan los identificadores entre paréntesis y separados por comas.

WRITE(Lista de identificadores);
WRITELN(Lista de identificadores);
WRITE(Identificador:longitud de campo);
WRITE(Identificador:longitud de campo:parte decimal);

WRITE, WRITELN:	Indican salida de información del sistema, permitiendo visualizarla en la pantalla.
WRITELN:	Avanza el cursor a la línea siguiente después de editar en pantalla.
LONGITUD DE CAMPO:	Especifica la longitud total del identificador.
PARTE DECIMAL:	Indica el total de dígitos decimales a imprimir para el identificador referenciado.

Ejemplo:

READ(A,B,C);

Si los identificadores A,B,C fueron declarados como enteros, la proposición READ ordena que se alimente a la computadora con 3 valores enteros, los cuales son almacenados en las localizaciones de memoria que corresponden a los identificadores relacionados en la proposición READ, por ejemplo, si los valores ingresados fueron 1, 7, 5, entonces el valor 1 es asignado al identificador A, el valor 7 al identificador B y el valor 5 al identificador C.

Ejemplo:

READLN(Z);

En este caso los valores deben ser ingresados uno por línea, ya que el READLN cambia de línea una vez ha leído el dato, así:

1
7
5

Ejemplo:

READ;

Se puede utilizar el READ vacío, es decir, sin relacionar identificadores y produce como efecto una pausa en la ejecución del programa. Para continuar con la ejecución es necesario oprimir cualquier tecla, barra espaciadora o Enter.

Ejemplo:

WRITE(A,B,C);

Imprimir en pantalla el contenido de los identificadores A,B,C.
Cuando no se especifica la forma como se van a imprimir los identificadores, estos son editados en notación científica por el compilador TURBO PASCAL, si son del tipo REAL.

Ejemplo:

WRITE(A:5,B:7:2);

Imprime en pantalla el identificador A con 5 posiciones enteras, y el identificador B con 7 posiciones enteras y 2 posiciones decimales.

El WRITELN vacío, es decir, sin referenciar identificadores, permite avanzar líneas.

Ejemplo:

WRITELN;WRITELN;WRITELN;

Deja 3 líneas o renglones en blanco en la pantalla.

Se puede utilizar la impresora mediante la definición en la sección de USES de la unidad PRINTER y agregando a cada instrucción WRITE o WRITELN la palabra LST.

Ejemplo:

```
USES PRINTER;
  .
  .
WRITELN(Lst, 'UNIVERSIDAD NACIONAL');
```

Imprime el título UNIVERSIDAD NACIONAL a través de la impresora y luego cambia de línea.

Ejemplo:

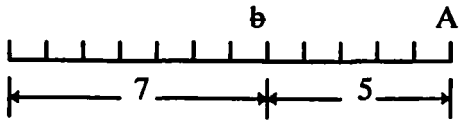
WRITELN(Lst); WRITELN(Lst);WRITELN(Lst);

Deja tres líneas o renglones en blanco en la impresora.

Ejemplo:

WRITELN(Lst, ' ':7, 'A':5);

Produce un espacio en blanco en la columna siete, luego imprime la letra A con cinco posiciones, ajustándola a la derecha, así:



Ejemplo:

WRITELN(Lst, 'UNIVERSIDAD NACIONAL':40);

Escribe por impresora UNIVERSIDAD NACIONAL terminando en la columna 40.

Ejemplo:

WRITELN(Lst, '':40, 'UNIVERSIDAD NACIONAL');

Escribe por impresora UNIVERSIDAD NACIONAL a partir de la columna 41.

Ejemplo:

WRITELN(Lst, '*':10, '*':70);

Produce un asterisco en la posición 10 y otro en la posición 70.

Es posible ubicar el cursor en un punto determinado de la pantalla mediante la utilización de la proposición GOTOXY, que opera de la siguiente forma:

GOTOXY(C,F) : donde C es columna y F es fila.

Para poder utilizar el GOTOXY es necesario definir en la sección de USES la unidad CRT

Ejemplo:

```
USES CRT;
   .
   .
   .
GOTOXY(10,12);
WRITE('UNIVERSIDAD NACIONAL');
```

Ubica el cursor en la columna 10, fila 12 de la pantalla e imprime el título UNIVERSIDAD NACIONAL.

2.2.4 ESTRUCTURA GENERAL DE UN PROGRAMA FORTRAN

Está compuesta de tres secciones:

- Sección de encabezamiento.
- Sección de declaraciones.
- Sección de proposiciones.

La sección de encabezamiento está conformada por la palabra reservada PROGRAM y el nombre asignado al programa, el cual servirá de identificación, éste debe empezar por un caracter alfabético y puede estar seguido de una secuencia de caracteres alfanuméricos, sin sobrepasar de un total de ocho.

Luego le sigue *la sección de declaraciones*, compuesta de proposiciones no ejecutables, las cuales no siempre van en todo ejercicio, ya que depende de las características propias del problema que se está resolviendo; algunas declaraciones son:

- DIMENSION
- DOUBLE PRECISION
- CHARACTER
- DATA
- EQUIVALENCE
- COMPLEX

La sección de proposiciones va a continuación de la última proposición no ejecutable y termina con las proposiciones STOP y END; permitiendo el STOP la terminación del programa y señalando la proposición END el fin de la compilación.

Ejemplo:

```
      PROGRAM EJEMPLO
      READ(*,*)A,B
      SUMA=A + B
      RESTA= A - B
      WRITE (*,100)SUMA
100   FORMAT(' La suma de A y B es :',F6.2)
      WRITE(*,150)RESTA
150   FORMAT(' La resta de A y B es :',F6.2)
      STOP
      END
```

En este programa se leen las variables A y B y se realiza la suma de ellas almacenando su resultado en la variable SUMA; luego se efectúa la resta de A y B y el resultado se almacena en la variable RESTA, para luego imprimir ambas salidas.

2.2.5 ESTRUCTURA GENERAL DE UN PROGRAMA TURBO PASCAL

Está compuesta de:

- Encabezamiento.
- Bloque del programa.

El encabezamiento empieza con la palabra reservada PROGRAM seguida del nombre asignado al programa y terminando con punto y coma (;). El nombre del programa debe comenzar con un caracter alfabético, seguido de una secuencia de caracteres alfanuméricos y aceptando solamente como caracter especial el guión inferior (_).

Ejemplo:

PROGRAM XYZ;

PROGRAM UNIVERSIDAD_NACIONAL;

El bloque del programa está compuesto a su vez de:

- Sección de definiciones y declaraciones.
- Sección de proposiciones.

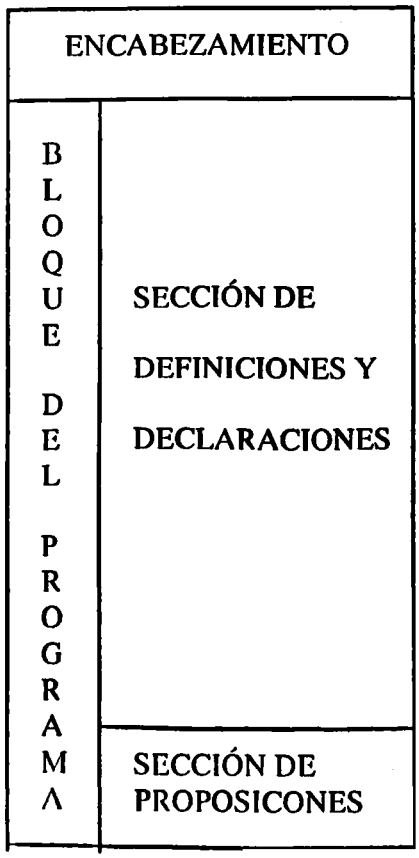
La sección de definiciones y declaraciones la integran:

- Declaración de usos.
- Declaración de rótulos.
- Definición de constantes.
- Definición de tipos.
- Declaración de variables
- Declaración de subprogramas.

La sección de proposiciones comienza con la palabra reservada BEGIN y termina con la proposición END.

Ampliando se tiene que la forma general de un programa PASCAL es la siguiente:

```
PROGRAM NOMBRE_PROGRAM;

USES
     DECLARACIÓN DE USOS;

LABEL
      DECLARACIÓN DE RÓTULOS;

CONST
      DEFINICIÓN DE CONSTANTES;

TYPE
     DEFINICIÓN DE TIPOS;

VAR
    DECLARACIÓN DE VARIABLES;

PROCEDURE O FUNCTION
      DECLARACIÓN DE SUBPROGRAMAS;

BEGIN
      PROPOSICIONES DEL PROGRAMA;
END.
```



El **Encabezamiento** está compuesto de la sentencia PROGRAM, la cual es obligatoria en todo programa Pascal.

Forma General:

```
PROGRAM NOMBRE_PROG;
```

NOMBRE_PROG : Especifica el nombre del programa, este obedece al gusto del programador, pero debe cumplir con las normas que existen para la definición de identificadores.

Ejemplo:

```
PROGRAM EJERCICIO;
```

La **Sección de definiciones y declaraciones** está compuesta de las sentencias USES, LABEL, CONST, TYPE, VAR y PROCEDURE y/o FUNCTION. Todas ellas son opcionales, es decir, pueden ir o no incluídas en un programa Turbo Pascal.

Sentencia USES.

Indica las unidades a ser utilizadas por el programa y debe ser codificada a continuación de la sentencia PROGRAM.

Forma General.

```
USES
   LISTA DE UNIDADES;
```

Ejemplo:

```
USES
     CRT;
```

Indica que se utilizará como dispositivo de entrada al programa el teclado y como unidad de salida la pantalla o monitor.

Ejemplo:

```
USES
     DOS,PRINTER;
```

Indica que serán empleadas funciones del sistema operativo D.O.S y como unidad de salida será utilizada la impresora.

*Sentencia **LABEL**.*

Permite la transferencia incondicional del flujo del programa, mediante la declaración de etiquetas que van separadas entre sí por comas.

Forma General.

```
LABEL
    ETIQUETA1,ETIQUETA2;
```

Las etiquetas pueden ser identificadores o valores enteros comprendidos entre 0 y 9999. La transferencia a una determinada etiqueta se logra mediante el empleo de la sentencia GOTO, la cual no es recomendable, salvo en casos extremos.

Ejemplo:

```
LABEL
    1,200,FIN;
```

*Sentencia **CONST**.*

Permite representar un valor que no cambia durante la ejecución del programa.

Forma General.

```
CONST
    NOMBRE = VALOR;
```

NOMBRE : Nombre de la constante.
VALOR : Valor asignado a la constante.

Una vez se ha asignado el valor al nombre de la constante, queda fijado y no puede cambiarse durante la ejecución del programa.

Ejemplo:

```
CONST
    A = 375;
    X = 0.035;
    NOM = 'UNIVERSIDAD NACIONAL';
    ESTRELLA = '*******';
```

Cuando el valor a ser asignado se trata de una cadena de caracteres, ésta debe ir encerrada entre comillas simples. Existen constantes con nombre predefinidos que se pueden utilizar directamente.

Ejemplo:

PI	3.14159265
TRUE	Verdadero
FALSE	Falso
MAXINT	Máximo entero = 32767
MAXLONGINT	Máximo entero largo = 2147'483.647

*Sentencia **TYPE**.*

Permite definir los distintos tipos de datos que van a ser utilizados durante el programa.
Forma General.

```
TYPE
    IDENTIFICADOR = TIPO;
```

Ejemplo:

```
TYPE
    FACTORIAL   = INTEGER;
    A,B         = REAL;
    ALPHA       = STRING[10];
```

*Sentencia **VAR**.*

Representa aquella información que cambia durante la ejecución de un programa. Todas las variables que intervienen durante la ejecución de un programa deben ser declaradas previamente a su utilización.

Forma General.

```
VAR
    IDENTIFICADOR : TIPO;
```

Ejemplo:

```
VAR
    I,J,K               :BYTE;
    SALARIO_BRUTO       :REAL;
    CODIGO              :INTEGER;
```

*Sentencia **PROCEDURE** y **FUNCTION**.*

Son programas que realizan una determinada tarea o proceso específico y que son ejecutados desde el programa principal o programa invocador, tantas veces como sea necesario. Este tema es analizado detalladamente en el capítulo IV Diseño Descendente.

Las sentencias antes descritas pueden ser declaradas dentro del programa Pascal en cualquier orden, sin embargo, es recomendable mantener el esquema planteado.

La **Sección de proposiciones** está compuesta de todas aquellas instrucciones que tienen como fin realizar una tarea determinada y van encerradas entre las proposiciones BEGIN y END.

Forma general.

```
BEGIN
    { Cuerpo del programa
END.
```

Ejemplo:

Codificando el ejercicio de la página 48 se tiene:

```
PROGRAM EJEMPLO;
USES CRT;
VAR
    A,B,SUMA,RESTA : REAL;

BEGIN
      READ(A,B);
      SUMA:=A+B;
      RESTA:=A - B;
      WRITE('La suma de A y B es :',SUMA:6:2);
      WRITE('La resta de A y B es :',RESTA:6:2);
END.
```

En esta codificación se declaró en la sección de USES la unidad CRT, que es una unidad estándar del TURBO PASCAL y permite entre otros usos, la pantalla o monitor como unidad de salida.

2.3 ESTRUCTURAS ALGORÍTMICAS

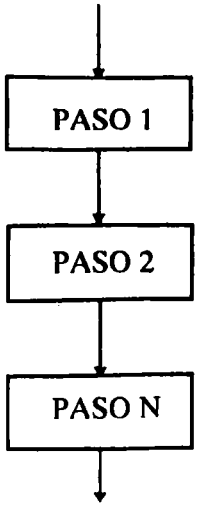
Las estructuras se pueden clasificar de la siguiente forma:

ESTRUCTURAS ALGORÍTMICAS	- SECUENCIAL - CONDICIONAL - REPETITIVAS O CÍCLICAS - REPETITIVAS ANIDADAS

2.3.1 ESTRUCTURA SECUENCIAL:

Es aquella en la que un paso o acción es ejecutado antes de continuar con el inmediatamente siguiente. Se caracteriza por que el flujo del algoritmo recorre todo el proceso, desde el punto de entrada hasta su culminación, sin incluir ciclos o repeticiones.

Gráficamente se aprecia de la siguiente forma:



EJERCICIO

Se desea calcular el volumen de un paquete y para el efecto se dispone de las siguientes dimensiones: longitud, anchura y altura. Imprimir las dimensiones del paquete así como el volumen calculado.

Solución

a) Análisis

Entrada : Longitud, anchura y altura. Estas variables requieren ser leídas.
Proceso : Volumen = longitud x anchura x altura
Salida : Longitud, anchura, altura y volumen.

b) Variables a utilizar.

LON = longitud
AN = anchura
ALT = altura
VOL = volumen

c) Algoritmo en seudocódigo

```
Paso 1    INICIO
Paso 2      Leer LON, AN, ALT
Paso 3      VOL ← LON * AN * ALT
Paso 4      Imprimir LON, AN, ALT, VOL
Paso 5    FIN
```

EJERCICIO

Un ahorrador desea acumular $150.000= dentro de 3½ años. Por lo tanto coloca su dinero en una corporación de ahorro que le paga el 33% anual de interés compuesto. Cuánto debe depositar hoy para acumular tal cantidad al final del tiempo proyectado ?

Solución

a) Análisis

Entrada: Suma futura, Interés y Tiempo. Estas son variables que requieren ser leídas.
Proceso: Aplicar la siguiente fórmula:

$$Suma\ presente = Suma\ futura * \left(\frac{1}{1 + Interes}\right)^{Tiempo}$$

Salida: Suma futura, Interés, Tiempo, Suma presente.

b) Variables a utilizar

S = Suma futura
P = Suma de dinero actual ó suma presente
I = Interés
N = Tiempo ó número de períodos.

c) Algoritmo en seudocódigo

```
Paso 1    INICIO
Paso 2       Leer S, I, N
Paso 3       P ← S * (1/(1 + I))↑N
Paso 4       Imprimir S, I, N, P
Paso 5    FIN
```

EJERCICIO

Leer los valores A y B. Calcular Z según la siguiente ecuación.

$$Z = \frac{-Y}{\sqrt[5]{A^2 - B^2}} - \frac{1}{Y\sqrt[3]{A^2 - B^2}} \qquad \textit{donde: } Y = A - 2B^2$$

Solución

a) Análisis

Entrada: A, B
Proceso: Calcular Y = A - $2B^2$, para luego buscar Z.
También se puede resumir el ejercicio calculando D = A^2 - B^2
Salida : Z

b) Variables a utilizar

A, B = Valores a ser tratados
Y = Resultado parcial de A - $2B^2$
D = Reultado parcial de A^2 - B^2
Z = Valor resultante

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer A, B
Paso 3        Y ← A - 2 * B↑2
Paso 4        D ← A↑2 - B↑2
Paso 5        Z ← -Y / D↑(1/5) - 1/(Y*D↑(1/3))
Paso 6        Imprimir Z
Paso 7      FIN
```

En este ejercicio no se hizo necesario leer la variable Y, porque ella surge de efectuar el cálculo $A - 2B^2$

SINTAXIS FORTRAN

En Fortran las estructuras secuenciales, tanto simples como compuestas, van separadas unas de otras en filas diferentes.

Ejemplo:

```
A = 8
B = 41
C = 7
SUM = A + B + C
```

Ejemplo:

```
      READ(*,50)NRA
50    FORMAT(I5)
      ESFERA = 4 * 3.141592654 * NRA**2
      WRITE(*,60)ESFERA
60    FORMAT(// 20X, 'La superficie de la esfera es :',F9.2)
```

SINTAXIS PASCAL

Todas las proposiciones que componen la estructura secuencial van separadas por punto y coma y agrupadas por las proposiciones BEGIN y END.

Un punto a continuación del END indica el final físico del programa. No se utiliza el punto y coma después del BEGIN, ni antes del END.

Ejemplo:

```
BEGIN
        A:= 8;
        B:= 41;
        C:= 7;
        SUMA:= A + B + C
END;
```

El END de una sentencia compuesta debe terminar en punto y coma, si va separado de otra sentencia. No ocurre lo mismo si la instrucción siguiente es otro END, ya que éste es un delimitador igualmente válido.

Ejemplo:

```
BEGIN
    BEGIN
        BEGIN
            A:= 8;
            B:= 41;
            C:= 7
        END
    END;
    SUMA:= A+ B + C
END;
```

EJERCICIO

Dadas 3 notas de evaluaciones con un valor del 20% cada una y una nota de trabajos con un valor del 40%, calcular los porcentajes de cada nota, así como del trabajo final y hallar la nota definitiva de un estudiante.

Solución

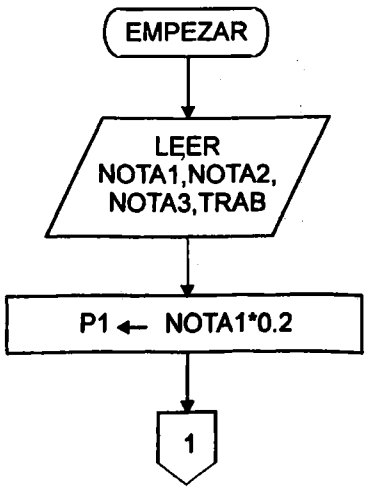
a) Análisis

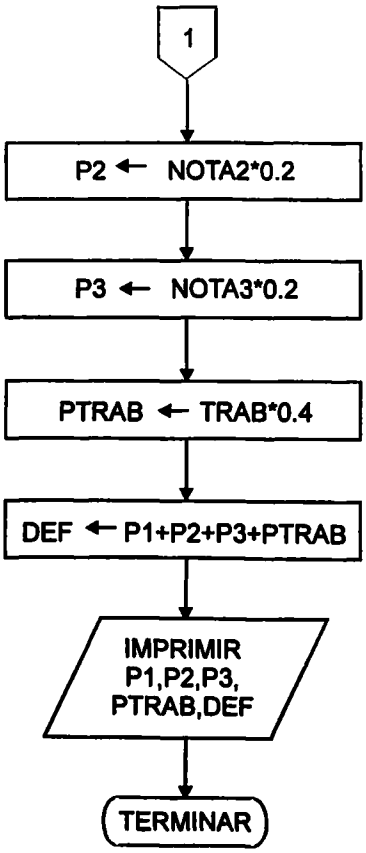
Entrada: Se debe leer cada una de las notas parciales, así como la nota del trabajo final.
Proceso: Se debe calcular el porcentaje para cada nota parcial y almacenarla en una variable, igual proceso se debe hacer con el trabajo final; posteriormente se calcula la nota definitiva como la sumatoria de los porcentajes parciales.
Salida : Se debe imprimir cada nota leída, los porcentajes calculados y la nota definitiva.

b) Variables a utilizar

NOTA1 = Nota primer parcial
NOTA2 = Nota segundo parcial
NOTA3 = Nota tercer parcial
TRAB = Trabajo final
P1 = Porcentaje primer parcial
P2 = Porcentaje segundo parcial
P3 = Porcentaje tercer parcial
PTRAB = Porcentaje del trabajo final
DEF = Nota definitiva

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER NOTA1, NOTA2, NOTA3, TRAB
Paso 3        P1 ← NOTA1 * 0.2
Paso 4        P2 ← NOTA2 * 0.2
Paso 5        P3 ← NOTA3 * 0.2
Paso 6        PTRAB ← TRAB * 0.4
Paso 7        DEF ← P1 + P2 + P3 + PTRAB
Paso 8        IMPRIMIR P1, P2, P3, PTRAB, DEF
Paso 9      FIN
```

e) Codificación FORTRAN

```
      PROGRAM NOTAS
```

```
      READ(*,1)PNOTA1,PNOTA2,PNOTA3,TRAB
1     FORMAT(4F2.2)
      P1 = PNOTA1 * 0.2
      P2 = PNOTA2 * 0.2
      P3 = PNOTA3 * 0.2
      PTRAB = TRAB * 0.4
      DEF = P1 + P2 + P3 + PTRAB
      WRITE(*,2)P1,P2,P3,PTRAB,DEF
2     FORMAT(5(3X,F2.2))
      STOP
      END
```

f) Codificación PASCAL

```
PROGRAM NOTAS;
USES PRINTER,CRT;
VAR
    NOTA1,NOTA2,NOTA3,TRAB,P1,P2,P3,PTRAB,DEF      :REAL;
BEGIN
  READ(NOTA1,NOTA2,NOTA3,TRAB);
  P1:= NOTA1 * 0.2;
  P2:= NOTA2 * 0.2;
  P3:= NOTA3 * 0.2;
  PTRAB:= TRAB * 0.4;
  DEF:= P1 + P2 + P3 + PTRAB;
  WRITE(Lst, '':5,P1:2:2, '':12,P2:2:2, '':20,P3:2:2, '':27,PTRAB:2:2, '':35,DEF:2:2)
END.
```

Observación

Al ser ejecutados los programas recién codificados por una persona ajena a quien los programó, se presenta el siguiente fenómeno: el computador se queda en espera de ser alimentado con los respectivos datos, pero a su vez el usuario no sabe lo que está sucediendo internamente en el computador, por lo tanto no hay entendimiento porque no hay comunicación entre la máquina y el usuario; para evitarlo es necesario incorporar dentro de la programación mensajes dirigidos al usuario, que le indiquen la forma como debe alimentar el sistema, así mismo los resultados producidos por éste deben ir acompañados de mensajes que le señalen a qué corresponde la salida producida por el computador. A este estilo de programación se le conoce como *PROGRAMACIÓN INTERACTIVA*.

A continuación se codifica nuevamente el ejercicio incluyendo diálogo interactivo.

e) Codificación FORTRAN

```
      PROGRAM NOTAS
C     Programa que calcula la nota definitiva de un estudiante
      WRITE(*,3)
3     FORMAT(/ 5X, 'ENTRE LA NOTA DE LA PRIMERA EVALUACIÓN')
      READ(*,5)PNOTA1
5     FORMAT(F2.2)
      WRITE(*,4)
4     FORMAT(/ 5X, 'ENTRE LA NOTA DE LA SEGUNDA EVALUACIÓN')
      READ(*,5)PNOTA2
      WRITE(*,7)
7     FORMAT(/ 5X, 'ENTRE LA NOTA DE LA TERCERA EVALUACIÓN')
      READ(*,5)PNOTA3
      WRITE(*,6)
6     FORMAT(/ 5X, 'ENTRE LA NOTA DEL TRABAJO FINAL')
      READ(*,5)TRAB
      P1 = PNOTA1 * 0.2
      P2 = PNOTA2 * 0.2
      P3 = PNOTA3 * 0.2
      PTRAB = TRAB * 0.4
      DEF = P1 + P2 + P3 + PTRAB
      WRITE(*,10)P1
10    FORMAT(/// 5X, 'EL PORCENTAJE DEL 1er. PARCIAL ES :',2X,F2.2)
      WRITE(*,11)P2
11    FORMAT(/ 5X, 'EL PORCENTAJE DEL 2do. PARCIAL ES :',2X,F2.2)
      WRITE(*,12)P3
12    FORMAT(/ 5X, 'EL PORCENTAJE DEL 3er. PARCIAL ES :',2X,F2.2)
      WRITE(*,13)PTRAB
13    FORMAT(/ 5X, 'EL PORCENTAJE DEL TRABAJO FINAL ES :',2X,F2.2)
      WRITE(*,14)DEF
14    FORMAT(/// 5X, 'LA NOTA DEFINITIVA ES :',2X,F2.2)
      STOP
      END
```

En este ejercicio se denominaron las variables como PNOTA1, PNOTA2, PNOTA3, con el fin de hacerlas reales, para que así pudieran almacenar valores reales; de haberlas llamado NOTA1, NOTA2, NOTA3, sólo podrían almacenar cantidades enteras, y no sería correcto puesto que las notas llevan componente entero y componente decimal.

f) Codificación PASCAL

```
PROGRAM NOTAS;
USES CRT;
```

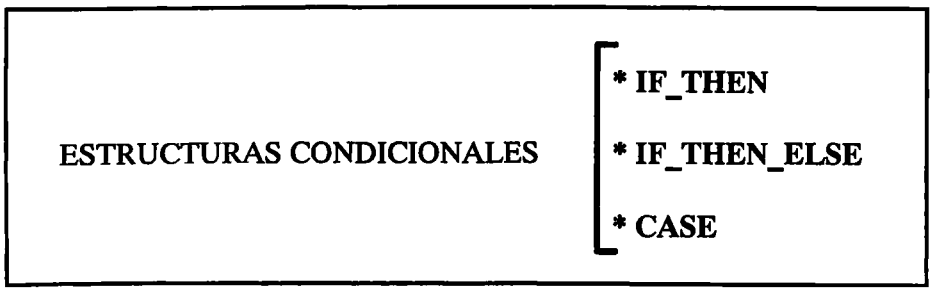
```
{Programa que calcula la nota definitiva de un estudiante}
VAR
    NOTA1,NOTA2,NOTA3,TRAB,P1,P2,P3,PTRAB,DEF      :REAL;
BEGIN
  WRITELN(‘’:5, ‘ENTRE LA NOTA DE LA PRIMERA EVALUACIÓN’);
  READ(NOTA1);
  WRITELN(‘’:5, ‘ENTRE LA NOTA DE LA SEGUNDA EVALUACIÓN’);
  READ(NOTA2);
  WRITELN(‘’:5, ‘ENTRE LA NOTA DE LA TERCERA EVALUACIÓN’);
  READ(NOTA3);
  WRITELN(‘’:5, ‘ENTRE LA NOTA DEL TRABAJO FINAL’);
  READ(TRAB);
  P1:= NOTA1 * 0.2;
  P2:= NOTA2 * 0.2;
  P3:= NOTA3 * 0.2;
  PTRAB:= TRAB * 0.4;
  DEF:= P1 + P2 + P3 + PTRAB;
  WRITELN;WRITELN;WRITELN;
  WRITELN(‘’:5, ‘EL PORCENTAJE DEL 1er. PARCIAL ES :’,P1:2:2);
  WRITELN(‘’:5, ‘EL PORCENTAJE DEL 2do. PARCIAL ES :’,P2:2:2);
  WRITELN(‘’:5, ‘EL PORCENTAJE DEL 3er. PARCIAL ES :’,P3:2:2);
  WRITELN(‘’:5, ‘EL PORCENTAJE DEL TRABAJO FINAL ES :’,PTRAB:2:2);
  WRITELN;WRITELN;
  WRITE(‘’:5, ‘LA NOTA DEFINITIVA ES :’, DEF:2:2)
END.
```

El mensaje que se encuentra encerrado entre llaves {} es un comentario que sirve como documentación del programa. En Fortran los comentarios se distinguen porque llevan codificada una C o un asterisco * en la columna 1.

2.3.2 ESTRUCTURAS CONDICIONALES

Es frecuente que en la mayoría de los algoritmos se tengan que tomar decisiones, lo que origina una bifurcación del flujo dependiendo de la condición que se esté analizando en ese momento. Si la condición es verdadera toma una ruta, en caso contrario toma la ruta alterna.

A este tipo de estructuras también se le conoce como **Estructuras de Decisión** y se pueden clasificar así:



2.3.2.1 Estructura IF -THEN (SI - ENTONCES).

Se emplea cuando se desea ejecutar una ó más instrucciones dependiendo de si cumple una determinada condición del problema, en caso contrario, no se ejecuta instrucción alguna y se continúa con el flujo del algoritmo.

La estructura general es:

Seudocódigo en inglés

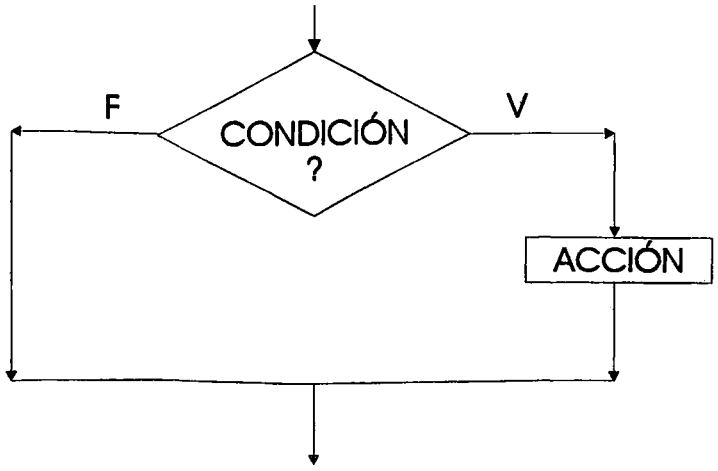
IF { Condición }
  THEN ACCIÓN
ENDIF

Seudocódigo en español

SI { Condición }
  ENTONCES ACCIÓN
FINSI

Como se observa, va acompañada de un indicador de final del rango del **IF** o del **SI**, para señalar cual es la acción ó acciones que se deben ejecutar cuando la condición sea verdadera.

Gráficamente se puede observar la estructura **IF_THEN (SI-ENTONCES)**, de la siguiente manera:



La estructura tiene una sola entrada y el flujo se vuelve a unir a partir de la condición analizada, para continuar con la salida unificada.

EJERCICIO

Leer dos valores en un registro y calcular la suma de los números leídos; sólo si la suma es negativa, imprimir el resultado.

Solución

a) Análisis

Entrada: Leer A, B
Proceso: Calcular SUMA = A+B
Averiguar si SUMA es menor que cero para imprimir su contenido, en caso contrario no se imprime.
Salida: Imprimir SUMA dependiendo de su valor.

b) Variables a utilizar

A, B = Valores a ser tratados.
SUMA = Valor resultante.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer A, B
Paso 3        SUMA ← A + B
Paso 4        SI SUMA < 0
                  ENTONCES IMPRIMIR SUMA
              FINSI
Paso 5      FIN
```

EJERCICIO

Leer en un registro los valores A y B. Calcular la suma, resta, multiplicación y división de los números leídos. Imprimir los valores leídos, así como todo lo calculado.
Nota: Si B= 0, no se debe efectuar la división A/B.

Solución

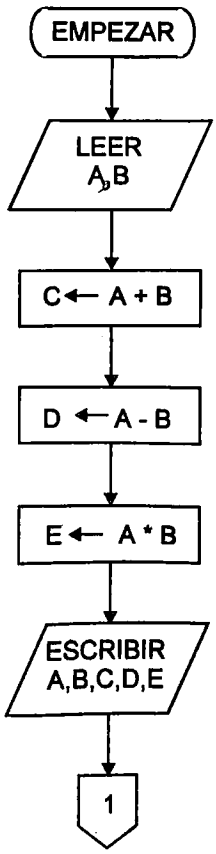
a) Análisis

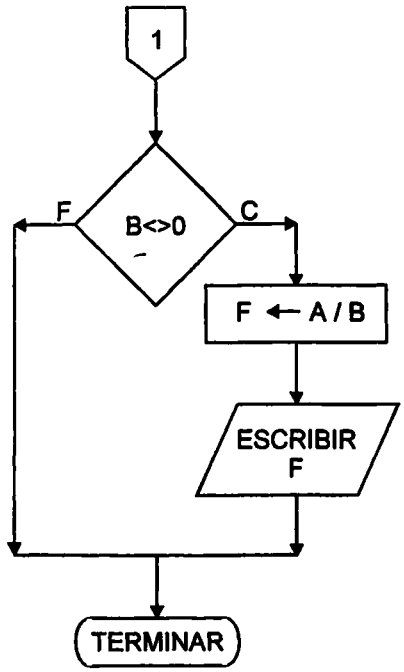
Entrada: Leer A, B
Proceso: Calcular
C = A + B
D = A - B
E = A * B
Si B <> 0, calcular F = A / B
Salida: A, B, C, D, E, F

b) Variables a utilizar

A, B = Variables a ser tratadas
C, D, E, F = Variables resultantes

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer A, B
Paso 3        C ← A + B
Paso 4        D ← A - B
Paso 5        E ← A * B
Paso 6        Imprimir A, B, C, D, E
Paso 7        SI B <> 0
                  ENTONCES F ← A / B
                      Imprimir F
              FINSI
Paso 8      FIN
```

Observación

Siempre que se formule un algoritmo, es fundamental probarlo con diversos datos para determinar su comportamiento. De aplicarse ésta técnica en forma permanente, se reducirá la probabilidad de errores y aumentará la confianza en las soluciones producidas.

Prueba de Escritorio

Consiste en reemplazar las variables que intervienen en el algoritmo con valores de diferente índole, efectuándoles seguimiento y observando su comportamiento. En esta forma se determina si el algoritmo funciona correctamente ó si por el contrario presenta errores de lógica. Esta prueba se debe hacer en repetidas ocasiones y con diversidad de datos, porque se puede presentar el caso de soluciones algorítmicas que se comportan eficientemente con ciertos datos, pero no con otros.

Las variables que intervienen en el proceso se colocan horizontalmente y debajo de ellas se ubican los correspondientes valores que van tomando de acuerdo al flujo del algoritmo.

Ejemplo de prueba de escritorio para el ejercicio anterior:

Si A = 10 y B = 5

Se tiene:	A	B	C	D	E	F
	10	5	15	5	50	2

Como B es diferente de 0, se efectúa la división y se tiene F = 2, para luego culminar el ejercicio.

Probando el mismo ejercicio con los siguientes datos:

Si A = 10 y B = 0

Se tiene:	A	B	C	D	E
	10	0	10	10	0

Como B es igual a 0, no se efectúa la división y termina el ejercicio.

2.3.2.2 Estructura IF-THEN-ELSE (SI-ENTONCES-DE LO CONTRARIO ó SI-ENTONCES-SINO).

En caso de que la condición analizada sea verdadera se efectúa una ó más instrucciones, de lo contrario se realiza otra acción o conjunto de acciones diferentes, unificándose los flujos de tal manera que al final de la estructura se enruten hacia el mismo punto. La estructura general es:

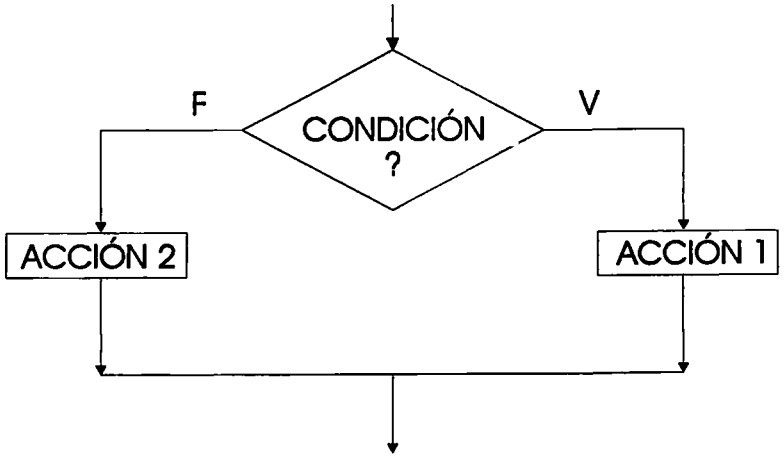
Seudocódigo en inglés

```
IF  { Condición }
  THEN  PASO 1
  ELSE  PASO 2
ENDIF
```

Seudocódigo en español

```
SI  { Condición }
  ENTONCES  PASO 1
  SINO  PASO 2
FINSI
```

Gráficamente se tiene:



SINTAXIS FORTRAN

IF aritmético

Forma General:

IF (A)N1,N2,N3

donde: A es una expresión aritmética válida o variable.
N1 cuando A<0
N2 cuando A=0
N3 cuando A>0

N1, N2, N3 deben ser valores enteros.

Ejemplo:

IF (Z - 0.5)10,15,7

Compara el valor de Z contra 0.5, si Z es menor que 0.5 transfiere el control a la instrucción identificada con el número 10; si Z es igual a 0.5 transfiere el control a la

instrucción referenciada con el número 15; si Z es mayor que 0.5 transfiere el control a la instrucción marcada con el número 7.

Ejemplo:

```
IF ( A)3,3,15
```

Cuando no se especifica contra que se está comparando, se asume que es contra cero; en este caso compara la variable A contra cero, si A es menor o igual a cero, trasfiere el control a la instrucción identificada con el número 3; si A es mayor que cero trasfiere el control a la instrucción marcada con el número 15.

IF lógico

Forma general:

IF (A)T

donde: A es una expresión lógica.
T es una instrucción ejecutable excepto un DO u otra instrucción IF lógico.

Ejemplo:

```
IF (A-B .LT. C)A=B
```

Compara si (A-B) es menor que la variable C, de ser cierto, asigna el contenido de la variable B a la variable A, si es falso continúa con la primera instrucción ejecutable después del IF lógico.

Obsérvese que en ambos casos, bien sea la expresión verdadera o falsa, se ejecuta la instrucción siguiente al IF lógico, salvo que la proposición ejecutable del IF lógico sea de transferencia de control.

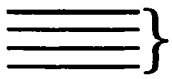
Ejemplo:

```
IF (A .EQ. 7)GOTO 9
   B=C**3
```

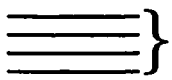
Si A es igual a 7, transfiere el control a la instrucción identificada con el número 9; si A no es igual a 7 eleva la variable C al cubo y el resultado lo almacena en la variable B.

Bloque IF
Forma general:

IF (Expresión lógica) THEN

 PROPOSICIONES VERDADERAS

ELSE

 PROPOSICIONES FALSAS

ENDIF

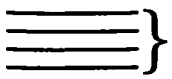
Ejemplo:

```
      IF (X-Y .GT. 0)THEN
         R=(X-Y)**(1/3)
         WRITE(*,1)X.Y.R
1        FORMAT(3(2X,F5.2))
      ELSE
         R=(X-Y)**3
         WRITE(*,2)X,Y,R
2        FORMAT(2X,F5.2,2(5X,F6.2))
      ENDIF
```

Si (X-Y) es mayor que cero entonces calcula R como la raíz cúbica de (X-Y) e imprime las variables X,Y,R con el formato 1, pero si (X-Y) es menor o igual a cero, se calcula R como el cubo de (X-Y) y se imprime con el formato 2.

Es posible formar el bloque IF sin la proposición ELSE, de la siguiente forma:

IF (Expresión lógica) THEN

 PROPOSICIONES VERDADERAS

ENDIF

Ejemplo:

```
      IF (X .LE. Y)THEN
         W=X**2+Y**2
         WRITE(*,5)W
5        FORMAT(5X,F7.3)
      ENDIF
```

Se pueden comparar dos o más expresiones relacionadas mediante el empleo de los operadores lógicos.

Ejemplo:

```
IF (X .GE. Y) .AND. (Y .EQ. Z) THEN
   A=X**2
   ELSE
   A=X**3
ENDIF
```

La expresión es verdadera cuando las dos condiciones son verdaderas.

SINTAXIS PASCAL

Forma general:

```
IF (Condición) THEN
            Proposición verdadera;
```

Si la condición es verdadera se ejecuta una proposición y se sale de la estructura IF con el punto y coma (;). Si la condición es falsa se sale del IF, sin ejecutar proposición alguna.

Forma general:

```
IF (Condición) THEN
      Proposición verdadera
ELSE
      Proposición falsa;
```

Si la condición es cierta se ejecuta la proposición verdadera y se sale del IF, si la condición es falsa se ejecuta la proposición falsa y se sale de la estructura IF.

Forma general:

```
IF (Condición) THEN
   BEGIN
      Proposiciones verdaderas;
      ____________________
      ____________________
      ____________________

   END
ELSE
```

```
BEGIN
   Proposiciones falsas;
   ═══════════════
   ═══════════════
   ═══════════════
END;
```

Cuando se debe ejecutar más de una sentencia ya sean verdaderas o falsas, éstas deben ser encerradas entre las proposiciones BEGIN y END y separadas por punto y coma (;).

Ejemplo:

```
   .
   .
BEGIN
   READ(A,B,C);
   IF A>B THEN
      MAYOR:=A
   ELSE
      MAYOR:=B;
   IF C>MAYOR THEN
      MAYOR:=C;
   GOTOXY(10,10);
   WRITE('El mayor de los 3 valores leídos es', MAYOR)
END;
   .
   .
```

Se leen 3 variables A,B,C y se compara A contra B, si A resulta ser mayor que B, se almacena el valor de A en la variable MAYOR, en caso contrario se almacena B en la variable MAYOR, en este punto termina la primera comparación; luego se pregunta si C es mayor que el contenido de la variable MAYOR, en caso de ser cierto, la variable C será la nueva MAYOR y se sale del IF; si no, se sale de la estructura IF. En ambos casos, una vez analizada la estructura IF, continúa con la impresión del mayor de los valores leídos.

EJERCICIO

Leer un valor diferente de cero y determinar si se trata de un número positivo o negativo.

Solución

a) Análisis

Entrada: Leer N
Proceso: Determinar si N es > ó < que cero

Salida : Producir el correspondiente mensaje.

b) Variables a utilizar

N = Variable a ser leída y analizada

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer N
Paso 3        SI  N > 0
                  ENTONCES Imprimir N "ES POSITIVO"
                  SINO Imprimir N "ES NEGATIVO"
              FINSI
Paso 4      FIN
```

d) Prueba de escritorio.

Si N = 7 IMPRIMIR 7 ES POSITIVO y termina el ejercicio.
Si N = -3 IMPRIMIR -3 ES NEGATIVO y termina el ejercicio.

e) Codificación FORTRAN

```
      PROGRAM NUMERO
      READ(*,*)N
      IF (N .GT. 0) THEN
         WRITE(*,2)N
2        FORMAT(I5, 'Es positivo')
      ELSE
         WRITE(*,3)N
3        FORMAT(I5, 'Es negativo')
      ENDIF
      STOP
      END
```

f) Codificación PASCAL

```
PROGRAM NUMERO;
USES CRT;
VAR
     N :INTEGER;
BEGIN
      READ(N);
      IF (N>0) THEN
           WRITE(N:5, 'Es positivo')
      ELSE
           WRITE(N:5, 'Es negativo')
END.
```

EJERCICIO

Leer en un registro los valores X, Y, Z. Calcular W, R, T, así:

Si $X <= Y$ _____ $W = X^2 + Y^2$
$X > Y$ _____ $W = X + Z^2Y$
$Y = Z$ _____ $R = X^2 + Y^2 + Z^2$
$Y <> Z$ _____ $R = X * Y * Z$
$W >= R$ _____ $T = W * R$
$W < R$ _____ $T = W + R$

Solución

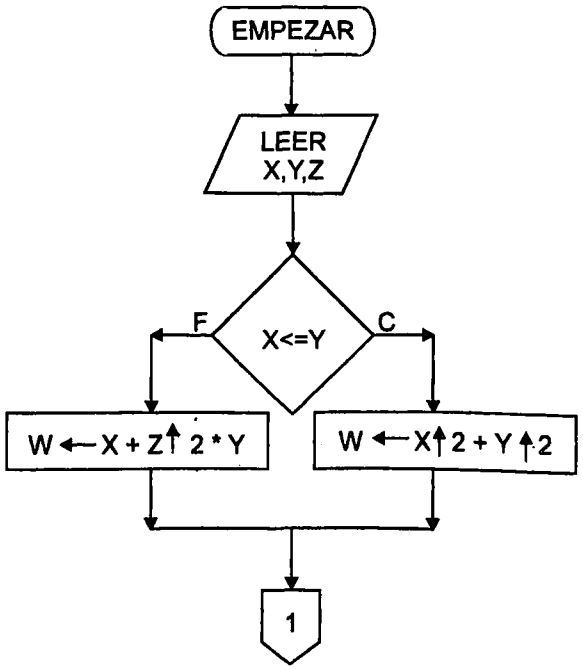
a) Análisis

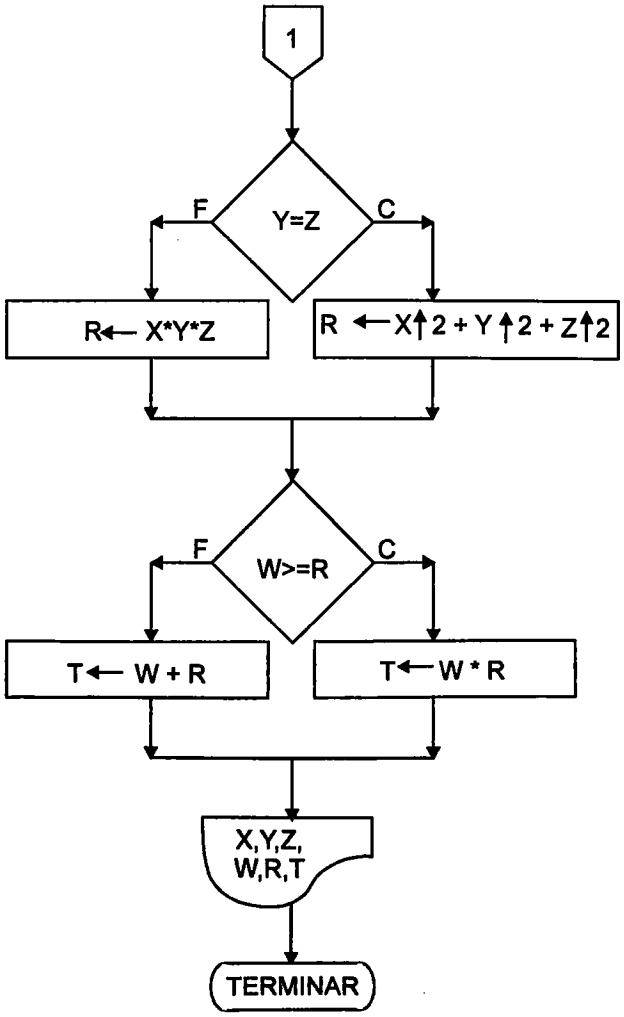
Entrada: leer X, Y, Z
Proceso: Se debe: Comparar X contra Y para calcular W.
Comparar Y contra Z para calcular R.
Comparar W contra R para calcular T.
Salida : Imprimir X, Y, Z, W, R, T

b) Variables a utilizar

X, Y, Z = Variables a ser leídas y analizadas
W, R, T = Variables resultantes.

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer X, Y, Z
Paso 3        SI X <= Y
                ENTONCES W ← X ↑ 2 + Y ↑ 2
                SINO    W ← X + Z ↑ 2 * Y
              FINSI
Paso 4        SI Y = Z
                ENTONCES R ← X ↑ 2 + Y ↑ 2 + Z ↑ 2
                SINO    R ← X * Y * Z
              FINSI
Paso 5        SI W >= R
                ENTONCES T ← W * R
```

```
                SINO    T ← W + R
              FINSI
Paso 6        Imprimir X, Y, Z, W, R, T
Paso 7      FIN
```

e) Codificación FORTRAN

```
      PROGRAM EQUIS
      READ(*,3)X,Y,Z
3     FORMAT(3F5.0)
      IF (X .LE. Y) THEN
         W=X**2 + Y**2
      ELSE
         W=X + Z**2 * Y
      ENDIF
      IF (Y .EQ. Z) THEN
         R=X**2 + Y**2 + Z**2
      ELSE
         R= X * Y * Z
      ENDIF
      IF (W .GE. R) THEN
         T=W * R
      ELSE
         T=W + R
      ENDIF
      WRITE(*,5)X,Y,Z,W,R,T
5     FORMAT(6(3X,F10.2))
      STOP
      END
```

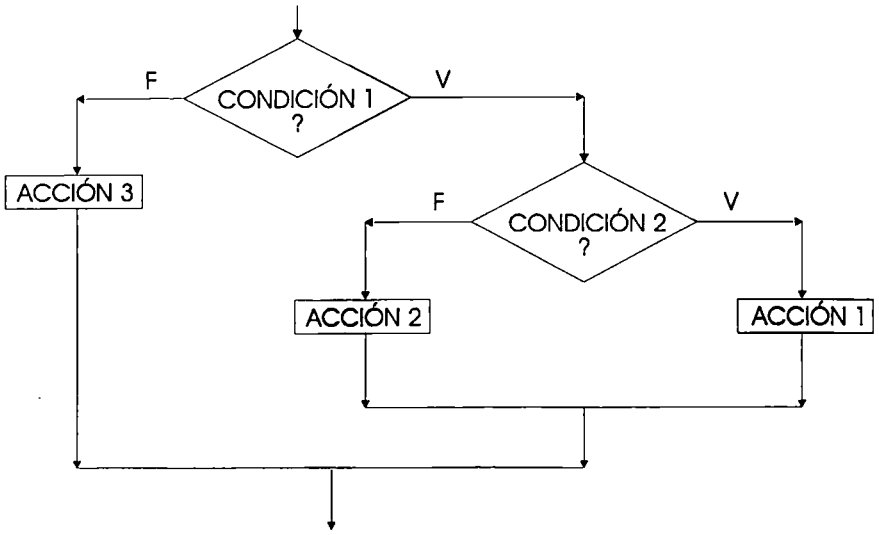
f) Codificación PASCAL

```
PROGRAM EQUIS;
USES CRT;
VAR
    X,Y,Z,W,R,T:REAL;
BEGIN
   READ(X,Y,Z);
   IF (X<=Y) THEN
      W:= X*X + Y*Y
   ELSE
      W:= X + Z*Z * Y;
   IF (Y=Z) THEN
      R:= X*X + Y*Y + Z*Z
   ELSE
      R:= X * Y * Z;
   IF (W>=R) THEN
      T:= W * R
   ELSE
      T:= W + R;
   WRITE(' ',X:10:2, ' ',Y;10;2, ' ',Z:10:2);
   WRITE(' ',W:10:2, ' ',R;10;2, ' ',T:10:2)
END.
```

2.3.2.2.1 Estructura IF anidada

En problemas de cualquier tamaño, las decisiones son necesarias y numerosas, conformándose con frecuencia estructuras que incluyen a su vez otras condiciones; a este tipo de estructuras se les conoce como *IF anidadas* o *cascada de decisiones.*

Gráficamente la estructura es así:



Si la condición 1 es verdadera, entonces se analiza la condición 2, si a su vez ésta es verdadera se ejecuta la acción 1, de lo contrario se ejecuta la acción 2; de no cumplirse la condición 1, se ejecuta la acción 3.

EJERCICIO

Leer dos valores en un registro e imprimir ambos números si son de diferente signo y distintos de cero.

Solución:

a) Análisis

Entrada: Leer dos valores A,B.
Proceso: Comparar A y B contra cero
Salida : Imprimir A y B, si son de diferente signo y distintos de cero

b) Variables a utilizar

A,B : Variables a ser leídas y analizadas

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer A,B
Paso 3        SI  A>0 y B<0
```

```
                    ENTONCES Imprimir A,B
                    SINO  SI A<0 y B>0
                              ENTONCES Imprimir A,B
                              SINO Imprimir "Los valores analizados son del mismo
                                                signo"
                          FINSI
                 FINSI
Paso 4      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

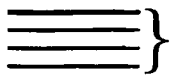
```
Paso 1      INICIO
Paso 2         Leer A,B
Paso 3         SI A<>0 y B<>0
                  ENTONCES C ← A * B
                            SI C<0
                               ENTONCES Imprimir A,B
                            FINSI
               FINSI
Paso 4      FIN
```

En esta solución se multiplicó el valor A por el valor B y el resultado se almacenó en la variable C, luego se comparó C contra cero. En caso de que el resultado de la comparación de menor que cero, es porque los valores A y B son de signo contrario, por lo tanto se imprimen; sino es porque son del mismo signo y termina el algoritmo sin imprimir.

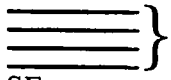
SINTAXIS FORTRAN DEL IF ANIDADO

Forma General:

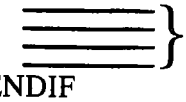
IF (Expresión lógica) THEN

 PROPOSICIONES VERDADERAS

ELSEIF (Expresión lógica) THEN

 PROPOSICIONES VERDADERAS DEL IF INTERNO

ELSE

 PROPOSICIONES FALSAS DEL IF INTERNO

ENDIF

Ejemplo:

```
      IF (NOTA .GE. 0) .AND. (NOTA .LT. 3) THEN
         WRITE(*,1)
1        FORMAT('Nota Reprobada')
      ELSEIF (NOTA .GE. 3) .AND. (NOTA .LT. 4) THEN
            WRITE(*,3)
3           FORMAT('Nota Buena')
          ELSEIF (NOTA .GE. 4) .AND. (NOTA .LE. 5) THEN
                WRITE(*,2)
2               FORMAT('Nota Excelente')
              ELSE WRITE(*,5)
5                   FORMAT('Error en nota, digite nuevamente')
      ENDIF
```

En este ejemplo se analiza si la nota de un estudiante es mayor o igual a cero y menor que tres, en caso afirmativo se imprime un mensaje indicando que la nota es reprobada; en caso que la condición estudiada no se cumpla, se analiza si la nota es mayor o igual a 3 y menor que 4, si la condición es cierta se imprime un mensaje señalando que la nota es buena, de no darse esta condición se averigua si la nota es mayor o igual a 4 y menor o igual a 5, si es cierto se imprime un mensaje indicando que la nota es excelente; si la condición no se cumple es porque se presentó un error en el dato. De todas formas, cada que se determine como es la nota, reprobada, buena o excelente, el control del ejercicio va al final del IF (ENDIF).

SINTAXIS PASCAL DEL IF ANIDADO

Forma General:

```
IF (Expresión lógica) THEN
      Proposición verdadera
ELSE IF (Expresión lógica) THEN
        Proposición verdadera del IF interno
      ELSE
        Proposición falsa del IF interno;
```

En este caso el anidamiento se cierra con el punto y coma (;).
Si el número de proposiciones a ejecutar son varias, es necesario encerralas entre BEGIN y END y separarlas con punto y coma, de la siguiente forma:

```
IF (Expresión lógica) THEN
   BEGIN
      Proposición A;
```

```
        Proposición B
    END
ELSE  IF (Expresión lógica) THEN
          BEGIN
             Proposición C;
             Proposición D
          END
        ELSE
           BEGIN
              Proposición E;
              Proposición F
           END;
```

La proposición END debe terminar en punto y coma (;) si va separada de otra sentencia; pero si le sigue otra proposición END, no es necesario el punto y coma (;) porque el END es un delimitador igualmente válido. No se utiliza punto y coma (;) en la proposición anterior al END, ni después del BEGIN.

El final físico de todo programa PASCAL se indica colocándole un punto (.) a la instrucción END.

Ejemplo:

```
IF (NOTA>=0) AND (NOTA<3) THEN
   WRITE('Nota Reprobada')
ELSE IF (NOTA>=3) AND (NOTA<4) THEN
         WRITE('Nota Buena')
       ELSE IF (NOTA>=4) AND (NOTA<=5) THEN
                WRITE('Nota Excelente')
             ELSE
                  WRITE('Error en nota, digite nuevamente');
```

EJERCICIO

Dadas las longitudes de un triángulo, determinar si se trata de triángulo equilátero (tres lados iguales), un triángulo isóseles (dos lados iguales, un lado diferente) o un triángulo escaleno (tres lados diferentes).

Solución

a) Análisis

Entrada: Se deben leer las tres longitudes del triángulo.

Proceso: Se deben comparar las longitudes entre sí, para poder determinar a qué tipo de triángulo corresponde.
Salida : Imprimir a qué tipo de triángulo corresponde las longitudes analizadas.

b) Variables a utilizar

LADO1, LADO2, LADO3 = Longitudes del triángulo a analizar.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER LADO1,LADO2,LADO3
Paso 3        SI (LADO1=LADO2) Y (LADO2=LADO3)
                 ENTONCES IMPRIMIR "Se trata de un triángulo equilátero"
                 SINO SI (LADO1=LADO2) O (LADO2=LADO3) O
                         (LADO1=LADO3)
                            ENTONCES IMPRIMIR "Se trata de un triángulo
                                               isóseles"
                            SINO IMPRIMIR "Se trata de un triángulo escaleno"
                          FINSI
              FINSI
Paso 4      FIN
```

EJERCICIO

Dados tres números aleatoriamente, hallar el mayor de ellos.

Solución

a) Análisis

Entrada: Se deben leer los tres valores.
Proceso: Se compara el primer valor contra el segundo, si resulta ser mayor, entonces se almacena en otra variable que contendrá el elemento mayor. Igual proceso se realiza entre el segundo y tercer valor.
Salida : Imprimir el elemento mayor.

b) Variables a utilizar

A,B,C = Valores a ser leídos y analizados
MAYOR = Contendrá el mayor de los tres valores analizados

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER A,B,C
Paso 3        SI  A>B
                  ENTONCES SI  A>C
                                  ENTONCES MAYOR ← A
                                  SINO        MAYOR ← C
                               FINSI
                  SINO  SI  B>C
                            ENTONCES  MAYOR ← B
                            SINO          MAYOR ← C
                          FINSI
              FINSI
Paso 4        IMPRIMIR A,B,C,MAYOR
Paso 5      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        LEER A,B,C
Paso 3        SI  A>B
                  ENTONCES MAYOR ← A
                  SINO         MAYOR ← B
              FINSI
Paso 4        SI  C>MAYOR
                  ENTONCES  MAYOR ← C
              FINSI
Paso 5        IMPRIMIR A,B,C,MAYOR
Paso 6      FIN
```

d) Codificación Pascal

```
PROGRAM  MAY;
USES CRT;
{DETERMINACIÓN DEL MAYOR DE TRES TÉRMINOS}
VAR
    A,B,C,MAYOR :INTEGER;
```

```
BEGIN
    WRITELN(':10, 'Teclee tres términos aleatoriamente');
    READ(A,B,C);
    IF A>B THEN
       MAYOR:= A
    ELSE
       MAYOR:= B;
    IF C>MAYOR THEN
       MAYOR:= C;
    WRITELN(':5,A, ',B, ',C, 'El mayor de los tres términos es :',MAYOR)
END.
```

EJERCICIO

Leer código, nombre y salario bruto de un trabajador. Calcular el salario neto según la siguiente tabla de retención en la fuente:

SALARIO BRUTO	RETENCIÓN EN LA FUENTE
0 - 41.000	0
41.001 - 60.000	1% del salario bruto
60.001 - 80.000	2% del salario bruto
80.001 - 100.000	4% del salario bruto
100.001 y más	7% del salario bruto

Solución

a) Análisis

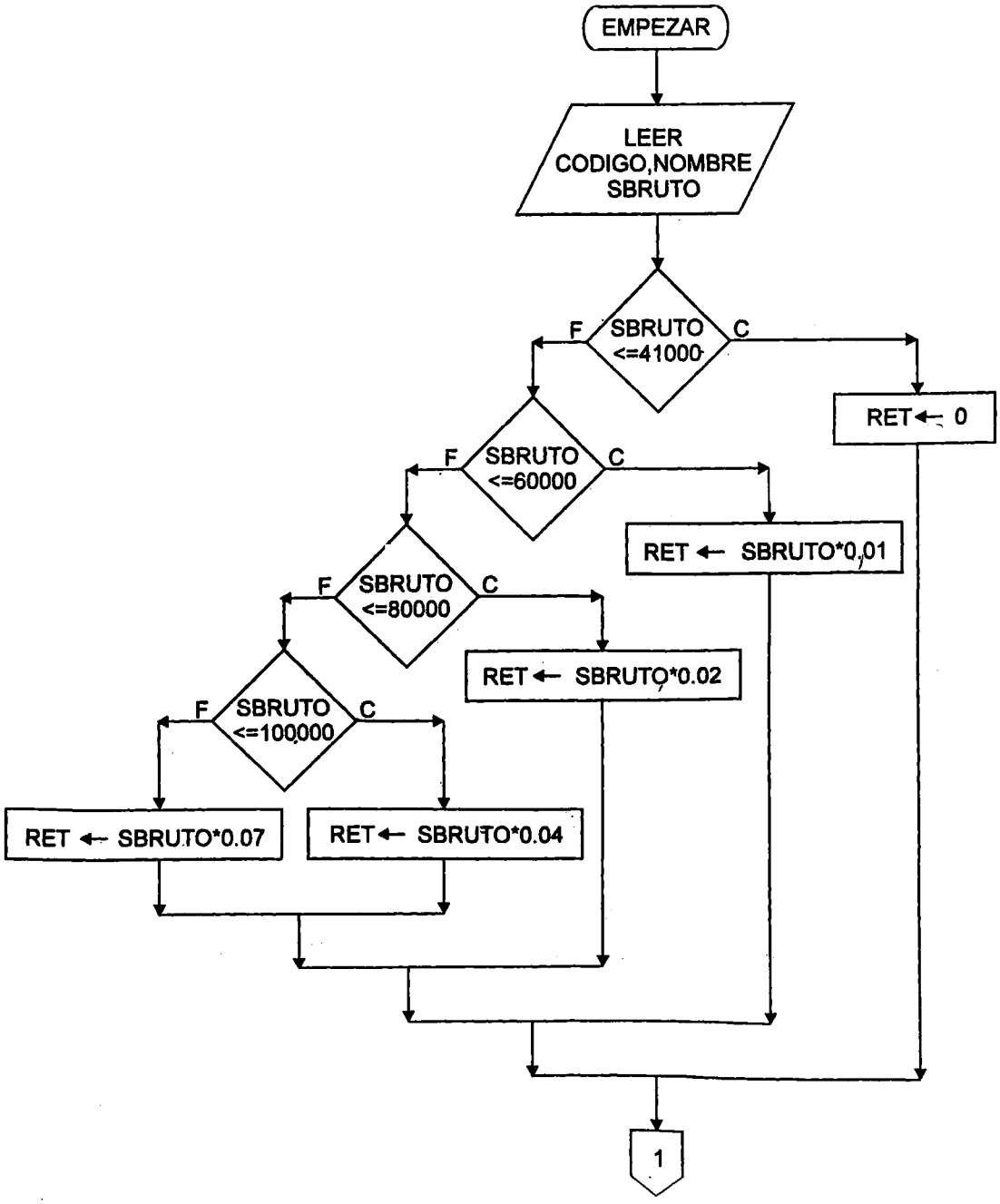
Entrada: Código del trabajador, Nombre del trabajador y salario bruto.
Proceso: Determinar a qué intervalo salarial corresponde el salario del trabajador para poder hallar la retención en la fuente, esto se logra mediante comparaciones sucesivas. Una vez calculada la retención en la fuente se procede al cálculo del salario neto.
Salida: Imprimir la información leída, así como la calculada.

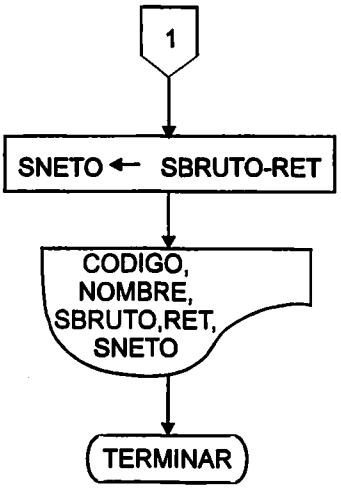
b) Variables a utilizar

CODIGO = Código del trabajador
NOMBRE = Nombre del trabajador
SBRUTO = Salario bruto

RET = Retención en la fuente
SNETO = Salario neto

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer CÓDIGO, NOMBRE, SBRUTO
Paso 3        SI SBRUTO <= 41.000
                 ENTONCES RET ← 0
                 SINO SI SBRUTO <= 60.000
                           ENTONCES RET ← SBRUTO * 0.01
                           SINO SI SBRUTO <= 80.000
                                     ENTONCES RET ← SBRUTO *0.02
                                     SINO SI SBRUTO <= 100.000
                                               ENTONCES RET ← SBRUTO *0.04
                                               SINO RET ← SBRUTO *0.07
                                          FINSI
                                FINSI
                      FINSI
              FINSI
Paso 4        SNETO ← SBRUTO - RET
Paso 5        Imprimir CODIGO, NOMBRE, SBRUTO, RET, SNETO
Paso 6      FIN
```

e) Prueba de escritorio

CODIGO	NOMBRE	SBRUTO	RET	SNETO
32745	JUAN PÉREZ	70000	70000x0.02=1400	70000-1400=68600

Otro ejemplo:

CODIGO	NOMBRE	SBRUTO	RET	SNETO
32745	MARÍA LOPEZ	110000	110000x0.07=7700	110000-7700=102300

f) Codificación FORTRAN

```
      PROGRAM NOMINA
      CHARACTER NOMBRE*30
      WRITE(*,3)
3     FORMAT(/// 20X, 'Digite código, nombre y salario bruto del trabajador:' /// )
      READ(*,5)KODIGO,NOMBRE,SBRUTO
5     FORMAT(I4,30A1,F7.0)
      IF (SBRUTO .LE. 41000) THEN
         RET=0
      ELSEIF (SBRUTO .LE. 60000) THEN
              RET=SBRUTO*0.01
           ELSEIF (SBRUTO .LE. 80000) THEN
                   RET=SBRUTO*0.02
                ELSEIF (SBRUTO .LE. 100000) THEN
                        RET=SBRUTO*0.04
                     ELSE RET=SBRUTO*0.07
      ENDIF
      SNETO=SBRUTO - RET
      WRITE(*,7)KODIGO,NOMBRE,SBRUTO,RET,SNETO
7     FORMAT(/// 20X, 'CODIGO',5X, 'NOMBRE TRABAJADOR',20X, 'SALARIO
     *BRUTO',5X,'RETENCIÓN FUENTE',5X, 'SALARIO NETO' /// 21X, I4,5X,
     *30A1,3(5X,F7.0))
      STOP
      END
```

g) Codificación PASCAL

```
PROGRAM NOMINA;
USES CRT;
VAR
    CODIGO              :INTEGER;
    SBRUTO,RET,SNETO    :REAL;
    NOMBRE              :STRING[30];
BEGIN
   WRITELN;WRITELN;WRITELN;
   WRITELN('  Teclee código, nombre y salario bruto del trabajador');
```

```
    READLN(CODIGO,NOMBRE,SBRUTO);
    IF SBRUTO <= 41000 THEN
       RET:=0
    ELSE IF SBRUTO <= 60000 THEN
            RET:=SBRUTO * 0.01
         ELSE IF SBRUTO <= 80000 THEN
                 RET:=SBRUTO * 0.02
              ELSE IF SBRUTO <= 100000 THEN
                      RET:=SBRUTO * 0.04
                   ELSE
                      RET:=SBRUTO * 0.07;
    SNETO:=SBRUTO - RET;
    WRITELN;WRITELN;WRITELN;
    WRITELN('CODIGO':10,'NOMBRE TRABAJADOR':40, 'SALARIO BRUTO':60,
            'RETENCION FUENTE':80, 'SALARIO NETO':100);
    WRITELN;WRITELN;
    WRITELN(' ':6,CODIGO:4, ' ':25,NOMBRE, ' ':50,SBRUTO:7:0, ' ':70,RET:7:0,
            ' ':90,SNETO:7:0)
END.
```

EJERCICIO

La información de un contribuyente consta de su identificación y valor del patrimonio gravable (VPG). Se requiere calcular el impuesto a pagar sobre el patrimonio (IPP) de la siguiente forma:

IPP = 0 si VPG <= $1000000
IPP = (VPG - $1000000)*0.01 si $1000000 < VPG <= $2000000
IPP = $50000+ (VPG - $2000000)*0.03 si $2000000 < VPG <= $3000000
IPP = $100000 + (VPG - $3000000)*0.05 si VPG > $3000000

Solución

a) Análisis

Entrada : Identificación del contribuyente, valor del patrimonio gravable.
Proceso : Determinar en que rango se encuentra el valor del patrimonio gravable, para así calcular el impuesto a pagar sobre el patrimonio.
Salida : Identificación del contribuyente, valor del patrimonio gravable, impuesto a pagar sobre el patrimonio.

b) Variables a utilizar

IC = Identificación del contribuyente
VPG = Valor del patrimonio gravable
IPP = Impuesto a pagar sobre el patrimonio

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer IC, VPG
Paso 3        SI VPG <= 1000000
                 ENTONCES IPP ← 0
                 SINO SI (VPG > 1000000)  y  (VPG <= 2000000)
                           ENTONCES IPP ← (VPG - 1000000)*0.01
                           SINO SI (VPG > 2000000) y (VPG <= 3000000)
                                     ENTONCES IPP←50000+(VPG-2000000)*0.03
                                     SINO IPP←100000 + (VPG - 3000000)*0.05
                                  FINSI
                        FINSI
              FINSI
Paso 4        Imprimir IC, VPG, IPP
Paso 5      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        Leer IC, VPG
Paso 3        SI VPG <= 1000000
                 ENTONCES IPP ← 0
                 SINO SI  VPG <= 2000000
                           ENTONCES IPP ← (VPG - 1000000)*0.01
                           SINO SI  VPG <= 3000000
                                     ENTONCES IPP←50000 + (VPG-2000000)*0.03
                                     SINO IPP←100000 + (VPG - 3000000)*0.05
                                  FINSI
                        FINSI
              FINSI
Paso 4        Imprimir IC, VPG, IPP
Paso 5      FIN
```

EJERCICIO

Se tienen 9 balines aparentemente de igual peso, pero uno de ellos pesa menos que los restantes; con sólo dos pesajes o comparaciones básicas, determinar cuál es el balín que pesa menos.

Solución:

a) Análisis

Entrada : Leer el peso de cada balín, representando cada peso con una variable.
Proceso : Formar tres grupos, cada uno de ellos compuesto de tres balines, luego comparar el grupo 1 contra el grupo 2,si son iguales, en el grupo 3 se encuentra el balín de menor peso (este es el primer pesaje o comparación básica); luego se toman 2 de los 3 balines que componen el grupo 3 y se comparan, si son iguales, el tercer balín es el de menor peso, si no son iguales es porque uno de ellos es menor, por lo tanto es el de menor peso (este es el segundo pesaje o comparación básica). Se debe seguir realizando este análisis con cada uno de los grupos de balines, porque en cualquiera de ellos se puede encontrar el balín de menor peso.
Salida : Cada que se determine el balín de menor peso, se debe imprimir la variable que lo representa.

b) Variables a utilizar

A,B,C,D,E,F,G,H,I = variables que representan cada uno de los balines.
G1 = grupo 1, compuesto por A,B,C
G2 = grupo 2, compuesto por D,E,F
G3 = grupo 3, compuesto por G,H,I

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer A,B,C,D,E,F,G,H,I
Paso 3        G1 ← A+B+C
Paso 4        G2 ← D+E+F
Paso 5        G3 ← G+H+I
Paso 6        SI G1 = G2
                 ENTONCES SI G = H
                                ENTONCES Imprimir I, "balín de menor peso"
                                SINO SI G < H
                                        ENTONCES Imprimir G, "balin de
                                                 menor peso"
```

```
                                    SINO Imprimir H,"balin de  menor
                                         peso"
                                FINSI
                            FINSI
                    SINO SI  G1 < G2
                            ENTONCES SI A = B
                                    ENTONCES Imprimir C,"balín de
                                             menor peso"
                                    SINO SI  A < B
                                            ENTONCES Imprimir A,
                                             "balin de menor peso"
                                            SINO Imprimir B, "balin
                                             de menor peso"
                                        FINSI
                                FINSI
                            SINO SI  D = E
                                    ENTONCES Imprimir F, "balín de menor
                                             peso"
                                    SINO SI D < E
                                            ENTONCES Imprimir D,"balín de
                                                     menor peso"
                                            SINO Imprimir E,"balín de menor
                                                 peso"
                                        FINSI
                                FINSI
                        FINSI
                FINSI
Paso 7          FIN
```

2.3.2.3 Estructura de Decisión múltiple CASE (EN CASO DE-HACER).

Permite simplificar una serie de **IF** anidados y es eficiente cuando la decisión gira alrededor de datos discretos, cuantificables o numerables; en cualquier otra situación no tiene efectos positivos.

Esta estructura evalúa una expresión, la cual puede tomar diferentes valores y dependiendo del asumido en un momento dado, ejecuta la acción correspondiente de las múltiples posibles que se encuentran agrupadas.

Es de aclarar que la estructura de decisión múltiple, no es indispensable, ya que puede ser sustituida por la estructura **IF**, sin embargo algunos lenguajes de programación la incluyen

y resulta ser de gran ayuda por su simpleza y legibilidad al elaborar y estudiar las soluciones algorítmicas.

La estructura general es:

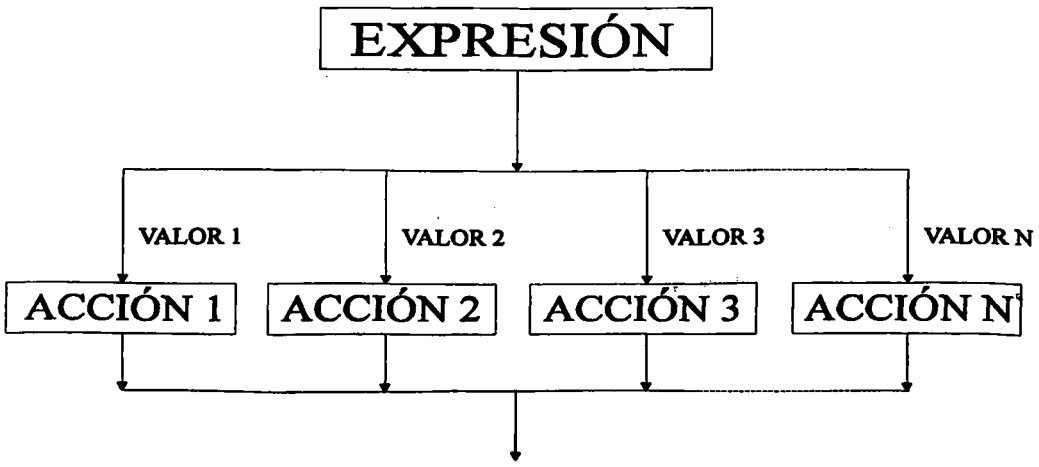
Seudocódigo en inglés

```
CASE  EXPRESIÓN OF
      V1: ACCIÓN 1
      V2: ACCIÓN 2
          . .
          . .
          . .
      VN: ACCIÓN N
END
```

Seudocódigo en español

```
EN CASO DE EXPRESIÓN HACER
      VALOR 1: ACCIÓN 1
      VALOR 2: ACCIÓN 2
                . .
                . .
                . .
      VALOR N: ACCIÓN N
FIN_CASO
```

Gráficamente se tiene:



EJERCICIO

Calcular el salario neto devengado por un trabajador de una compañía según la siguiente información:
CÓDIGO, NOMBRE, CATEGORÍA, NUMERO DE HORAS EXTRAS LABORADAS

Si:

CATEGORÍA	SALARIO BASE	VALOR HORA EXTRA	SEGURO SOCIAL	OTROS DESCUENTOS
1	40.000	400	1% sal. base	0
2	60.000	600	1.5% sal. base	1% sal. base

3	80.000	800	2% sal. base	2% sal. base
4	100.000	1.000	2.5% sal. base	3% sal. base
5	120.000	1.200	3% sal. base	3.5% sal. base
6	140.000	1.400	3.5% sal. base	4% sal. base
7	200.000	2.000	4% sal. base	5% sal. base

Solución

a) Análisis

Entrada : Código del trabajador, nombre del trabajador, categoría, número de horas extras.

Proceso : Con base en la categoría del trabajador se determina el resto de la información necesaria para efectuar el cálculo del salario neto. Las fórmulas a emplear son:
Valor horas extras = número de horas extras por valor respectivo de la categoría.
Seguro social = salario base por porcentaje respectivo de la categoría.
Descuento = salario base por porcentaje respectivo de la categoría.
Salario bruto = salario base + valor horas extras.
Total descuentos = seguro social + descuentos.
Salario Neto = salario bruto - total descuentos.

Salida : Imprimir código del trabajador, Nombre del trabajador, Salario bruto, Total descuentos, Salario Neto.

b) Variables a utilizar

CÓDIGO = Código del trabajador
NOMBRE = Nombre del trabajador
CATEGORÍA = Categoría
NHE = Número de horas extras
SAL = Salario base
VHE = Valor horas extras
SS = Seguro Social
DCTOS = Descuentos
SBRUTO = Salario bruto
TDCTOS = Total descuentos
SNETO = Salario Neto.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer CÓDIGO, NOMBRE, CATEGORÍA, NHE
Paso 3        EN CASO DE CATEGORÍA HACER
                1: SAL  ← 40000
                   VHE  ← NHE * 400
```

```
                SS      ← SAL * 0.01
                DCTOS   ← 0
              2: SAL    ← 60000
                VHE     ← NHE * 600
                SS      ← SAL * 0.015
                DCTOS   ← SAL * 0.01
              3: SAL    ← 80000
                VHE     ← NHE * 800
                SS      ← SAL * 0.02
                DCTOS   ← SAL * 0.02
              4: SAL    ← 100000
                VHE     ← NHE * 1000
                SS      ← SAL * 0.025
                DCTOS   ← SAL * 0.03
              5: SAL    ← 120000
                VHE     ← NHE * 1200
                SS      ← SAL * 0.03
                DCTOS   ← SAL * 0.035
              6: SAL    ← 140000
                VHE     ← NHE * 1400
                SS      ← SAL * 0.035
                DCTOS   ← SAL * 0.04
              7: SAL    ← 200000
                VHE     ← NHE * 2000
                SS      ← SAL * 0.04
                DCTOS   ← SAL * 0.05
            FIN_CASO
Paso 4      SBRUTO    ← SAL + VHE
Paso 5      TDCTOS    ← SS + DCTOS
Paso 6      SNETO     ← SBRUTO - TDCTOS
Paso 7      Imprimir CÓDIGO, NOMBRE, SBRUTO, TDCTOS, SNETO
Paso 8    FIN
```

SINTAXIS FORTRAN

En FORTRAN se le conoce como el GOTO computado o GOTO calculado, y permite efctuar la transferencia de control a una de varias instrucciones del programa, dependiendo del valor que tome una variable entera.

Forma general:
GOTO(N1,N2,N3,...,NN),J

donde: N1, N2, N3, ..., NN son etiquetas o identificadores de instrucción.
J es expresión entera, no suscrita y debe ser mayor o igual a uno y menor o igual al número de etiquetas o identificadores de instrucción.

Ejemplo:

```
GOTO(10,3,15,7),M
```

Si M es igual a 1, se trasfiere el control a la instrucción referenciada con el número 10; si M es igual a 3 se trasfiere el control a la instrucción etiquetada con el número 15 y así sucesivamente. Si el valor de M fuera negativo o mayor al número de etiquetas especificadas, la instrucción GOTO no es tenida en cuenta.

GOTO incondicional

Trasfiere el control a una posición ejecutable identificada por el número de etiqueta.

Forma general:

GOTO N

donde: N es un número entero positivo que señala la instrucción a donde es trasferido el control del programa.

Ejemplo:

```
GOTO 16
```

Trasfiere el control a la instrucción referenciada con el número 16.

Ejemplo:

```
      A=7.
      B=5.
      GOTO 77
99    C=A**2
77    D=B**2
      WRITE(*,3)A,B,C,D
3     FORMAT(4(5X,F5.0))
```

En este caso la instrucción 99 no se ejecuta debido a la transferencia de control ejercida por la instrucción GOTO 77, lo que acarrea un error más adelante, cuando se trata de imprimir la variable C sin haber sido definida, por la razón señalada anteriormente.

SINTAXIS PASCAL

Forma general:

```
CASE expresión OF
     1: proposición 1;
     2: proposición 2;
              .
              .
              .
     N: proposición N
END;
```

El valor de la expresión debe ser de tipo INTEGER, CHAR, BOOLEAN o ENUMERADO.

```
CASE expresión OF
     1: BEGIN
        ═══════ } CONJUNTO DE PROPOSICIONES;
        END;
     2: BEGIN
        ═══════ } CONJUNTO DE PROPOSICIONES;
        END;
END;
```

Cuando se deba ejecutar más de una instrucción por cada opción, ellas deben encerrarse entre las clausulas BEGIN y END.

Si el valor de la expresión que se está considerando es un caracter, se debe encerrar entre comillas; ejemplo:

```
CASE expresión OF
     'A': proposición 1;
     'B': proposición 2;
     'C': proposición 3
END;
```

Se pueden definir intervalos de valores por cada opción.

```
CASE expresión OF
    'A'..'D'  : proposición 1;
    'I'..'O'  : proposición 2
END;
```

Se puede definir la sentencia CASE complementada con la cláusula ELSE, aunque es opcional, es de gran ayuda.

Ejemplo:

```
CASE expresión OF
    'A', 'a', 'B', 'b'  : proposición 1;
    'C', 'D'            : proposición 2
    ELSE
      WRITELN('Error en datos')
END;
```

Si el valor de la expresión no está comprendido en la lista de constantes relacionadas, se ejecuta la clausula ELSE; de no existir tampoco ésta, el flujo de control se sale de la sentencia CASE y continúa con la siguiente instrucción.

Ejemplo:

```
CASE DIA OF
        1: WRITELN('Corresponde al día Lunes');
        2: WRITELN('Corresponde al día Martes');
        3: WRITELN('Corresponde al día Miércoles');
        4: WRITELN('Corresponde al día Jueves');
        5: WRITELN('Corresponde al día Viernes');
        6: WRITELN('Corresponde al día Sábado');
        7: WRITELN('Corresponde al día Domingo')
   ELSE
        WRITELN('Error en datos')
END;
```

CODIFICACIÓN FORTRAN

```
      PROGRAM NÓMINA1
      CHARACTER NOMBRE*30
      WRITE(*,1)
1     FORMAT(///10X, 'Teclee código del empleado:')
      READ(*,10)CODIGO
10    FORMAT(F6.0)
      WRITE(*,2)
```

```
2       FORMAT(//10X, 'Teclee nombre del empleado:')
        READ(*,3)NOMBRE
3       FORMAT(30A1)
        WRITE(*,8)
 8      FORMAT(//10X, 'Teclee categoría del empleado:')
        READ(*,4)KAT
4       FORMAT(I1)
        WRITE(*,7)
7       FORMAT(//10X, 'Teclee el número de horas extras trabajadas')
        READ(*,11)NHE
11      FORMAT(I2)
        GOTO(15,18,13,9,14,17,20),KAT
15        SAL=40000
          VHE=NHE*400
          SS=SAL*0.01
          DCTOS=0
          GOTO 100
18        SAL=60000
          VHE=NHE*600
          SS=SAL*0.015
          DCTOS=SAL*0.01
          GOTO 100
13        SAL=80000
          VHE=NHE*800
          SS=SAL*0.02
          DCTOS=SAL*0.02
          GOTO 100
9         SAL=100000
          VHE=NHE*1000
          SS=SAL*0.025
          DCTOS=SAL*0.03
          GOTO 100
14        SAL=120000
          VHE=NHE*1200
          SS=SAL*0.03
          DCTOS=SAL*0.035
          GOTO 100
17        SAL=140000
          VHE=NHE*1400
          SS=SAL*0.035
          DCTOS=SAL*0.04
          GOTO 100
20        SAL=200000
          VHE=NHE*2000
```

```
          SS=SAL*0.04
          DCTOS=SAL*0.05
100   SBRUTO=SAL+VHE
      TDCTOS=SS+DCTOS
      SNETO=SBRUTO-TDCTOS
      WRITE(*,21)CODIGO
21    FORMAT(//10X, 'El código del empleado es:',F6.0)
      WRITE(*,22)NOMBRE
22    FORMAT(//10X, 'El nombre del empleado es:',30Δ1)
      WRITE(*,30)SBRUTO
30    FORMAT(//10X, 'El salario bruto del empleado es:',F7.0)
      WRITE(*,35)TDCTOS
35    FORMAT(//10X, 'El total de descuentos del empleado es:',F6.0)
      WRITE(*,77)SNETO
77    FORMAT(//10X, 'El salario neto del empleado es:',F7.0)
      STOP
      END
```

Una vez especificados los cálculos que se deben realizar para cada categoría, es necesario efectuar una transferencia de control para salirse de ella, y se logra en este ejercicio a través de la instrucción GOTO 100.

CODIFICACIÓN PASCAL

```
PROGRAM XY;
USES CRT;
VAR
    CODIGO                          : INTEGER;
    SAL,VHE                         : LONGINT;
    NOMBRE                          : STRING[30];
    CATEGORIA,NHE                   : BYTE;
    SS,DCTOS,TDCTOS,SBRUTO,SNETO    : REAL;

BEGIN
      WRITELN;WRITELN;WRITELN;
      WRITELN(' ':10, 'Teclee código del empleado:');
      READ(CODIGO);
      WRITELN(' ':10, 'Teclee nombre del empleado:');
      READ(NOMBRE);
      WRITELN(' ':10, 'Teclee categoría del empleado:');
      READ(CATEGORIA);
      WRITELN(' ':10, 'Teclee número de horas extras trabajadas:');
      READ(NHE);
```

```
CASE CATEGORIA OF
  1: BEGIN
       SAL:=40000;
       VHE:=NHE*400;
       SS:=SAL*0.01;
       DCTOS:=0
     END;
  2: BEGIN
       SAL:=60000;
       VHE:=NHE*600;
       SS:=SAL*0.015;
       DCTOS:=SAL*0.01
     END;
  3: BEGIN
       SAL:=80000;
       VHE:=NHE*800;
       SS:=SAL*0.02;
       DCTOS:=SAL*0.02
     END;
  4: BEGIN
       SAL:=100000;
       VHE:=NHE*1000;
       SS:=SAL*0.025;
       DCTOS:=SAL*0.03
     END;
  5: BEGIN
       SAL:=120000;
       VHE:=NHE*1200;
       SS:=SAL*0.03;
       DCTOS:=SAL*0.035
     END;
  6: BEGIN
       SAL:=140000;
       VHE:=NHE*1400;
       SS:=SAL*0.035;
       DCTOS:=SAL*0.04
     END;
  7: BEGIN
       SAL:=200000;
       VHE:=NHE*2000;
       SS:=SAL*0.04;
       DCTOS:=SAL*0.05
     END
END;
```

```
        SBRUTO:=SAL+VHE;
        TDCTOS:=SS+DCTOS;
        SNETO:=SBRUTO-TDCTOS;
        WRITELN(' ':10, 'El código del empleado es:',CODIGO:6);
        WRITELN;WRITELN;
        WRITELN(' ':10, 'El nombre del empleado es:',NOMBRE);
        WRITELN;WRITELN;
        WRITELN(' ':10, 'El salario bruto del empleado es:',SBRUTO:7:0);
        WRITELN;WRITELN;
        WRITELN(' ':10, 'El total de descuentos es:',TDCTOS:7:0);
        WRITELN;WRITELN;
        WRITELN(' ':10, 'El salario neto del empleado es:',SNETO:7:0);
        WRITELN;WRITELN
END.
```

PROBLEMAS PROPUESTOS

1. Leer en un registro los valores A, B, y C. Calcular:
   $Z = (A+B)/C$ ____ si $C > 0$
   $Z = (A - B)^C$ ____ si $C < 0$
   $Z = 0$ _____ si $C = 0$

   Imprimir A, B, C y Z

2. La calificación de una asignatura está compuesta por la suma de 3 exámenes parciales que tienen la siguiente ponderación: 25%, 35% y 40%. Calcular la nota definitiva. Imprimir las notas parciales y la nota definitiva.
3. Leer código, nombre y salario bruto de un trabajador. Calcular la retención en la fuente como el 5% de su salario bruto si es superior a $60.000, en caso contrario la retención se calcula como el 2% de su salario bruto. Calcular el salario neto e imprimir tanto la información leída como la calculada.

4. Leer en un registro el valor de A y determinar:

   $Z = (A + 7)^2$ _____ si $1 \leq A \leq 3$
   $Z = \sqrt{3A - 5}$ _____ si $4 \leq A \leq 7$
   $Z = 7A / (A^3 - 7)$ _____ si $8 \leq A \leq 10$
   Imprimir tanto el valor de A como su correspondiente valor Z.

5. Leer código, nombre y salario de un trabajador e imprimir la información leída si el trabajador tiene un salario menor o igual a $50.000, en caso contrario, imprimir el código, nombre del trabajador y un mensaje que diga "gana más de $50.000".

6. Leer dos variables en un registro y calcular el cociente de dividir el primer valor por el segundo valor, imprimir el cociente hallado.
   Nota: si el segundo valor leído es cero, no efectúe el cálculo e imprima un mensaje que diga: "La división no se puede efectuar".

7. Leer en un registro el nombre, edad y sexo de una persona, e imprimir sólo si la persona es de sexo masculino y mayor de edad, el nombre de ella.
   Nota: se debe comparar la variable de sexo contra "M", en caso de ser iguales es porque el sexo es masculino, si no son iguales es porque el sexo de la persona es femenino. Se usa este artificio porque el computador no está en capacidad de definir por sí mismo, cuándo una persona es de sexo masculino o femenino, o si es alta o si es bonita.

8. Leer el nombre y edad de un estudiante. Si éste es menor de 18 años se debe asignar al edificio A, si tiene 18 años se debe asignar al edificio B, si tiene más de 18 años debe asignarse al edificio C. En todos los casos el nombre del estudiante debe aparecer en el listado general de ocupantes.

9. Señale en la siguiente codificación tres errores de sintaxis.

   A- Codificación FORTRAN;

```
      PROGRAM UNO;
      READ (*,*)A,B
      C:=A+B
      D:=A-B
      E:=A*B
      WRITE(*,1)A,B,C,D,E
10    FORMAT(5(2X;F4.2))
      IF (B .NE. 0) THEN
         F:=A/B
         WRITE(*,22)F
22       FORMAT(5X,I4.2)
      ENDIF
      STOP
      END
```

   B- Codificación PASCAL

```
PROGRAM UNO ;
USES CRT;
VAR
     A,B,C,D,E,F:REAL;
BEGIN
       READ(A,B);
```

```
        C:=A+B;
        D:=A-B;
        E:=A*B;
        WRITE(' ', A:4:2, ' ',B:4:2, ' ',C:4:2, ' ',D:4:2, ' ',E:4:2)
        IF (B< >0) THEN;
        BEGIN
              F:=A/B;
              WRITE(' ',F:4)
        END
END.
```

10.Considérese el siguiente segmento de algoritmo

```
A ← 7
B ← 13
SI A > B
   ENTONCES A ← A + 2
   SINO     B ← B + 1
FINSI
```

Determinar el valor de cada una de las variables después de la ejecución.

11.Considérese el siguiente segmento de algoritmo

```
A ← 7
B ← 13
SI A > B
   ENTONCES C ← 17
   SINO     D ← 17
FINSI
```

Después de ejecutado lo anterior, cuál de las siguientes afirmaciones es cierta ?

a) C = 17 y D = 17
b) C = 17 y D queda indefinida
c) D = 17 y C queda indefinida
d) C y D quedan indefinidas

12.Considérese el siguiente segmento de algoritmo

```
X ← 7
Y ← 9
Z ← 17
SI X > Y
   ENTONCES SI Y > Z
```

```
                    ENTONCES IMPRIMIR X
                    SINO        IMPRIMIR Y
                FINSI
    SINO IMPRIMIR Z
FINSI
```

Qué resultados se imprimen cuando se ejecuta el algoritmo ?

13.Considérese el siguiente segmento de algoritmo

```
X ← -1
Y ← -3
SI X >= Y
   ENTONCES IMPRIMIR X
   SINO SI Y <= 0
            ENTONCES IMPRIMIR Y
            SINO IMPRIMIR Z
        FINSI
FINSI
```

Qué resultado se imprime cuando se ejecuta el algoritmo ?

14.Determinar cuál es el valor final al ser ejecutadas las siguientes instrucciones ?

```
EN CASO DE NOTA = 3 HACER
    1: IMPRIMIR "nota insuficiente"
    2: IMPRIMIR "nota reprobada"
    3: IMPRIMIR "nota aprobatoria"
    4: IMPRIMIR "nota buena"
    5: IMPRIMIR "nota excelente"
FIN_CASO
```

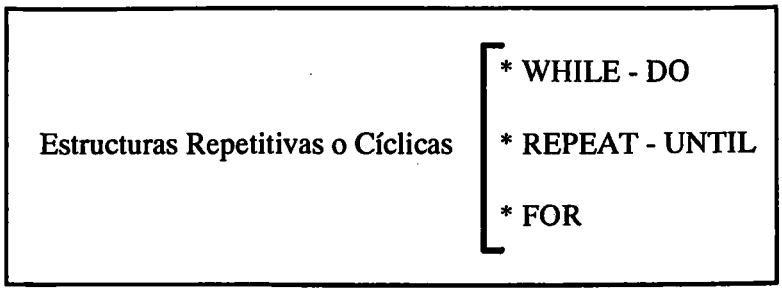
15.Escribir una estructura EN_CASO_DE_HACER equivalente a la siguiente porción algorítmica.

```
SI CALIF = 2
   ENTONCES IMPRIMIR "el trabajo es deficiente"
   SINO SI CALIF = 3
            ENTONCES IMPRIMIR "El trabajo es regular"
            SINO SI CALIF = 4
                     ENTONCES IMPRIMIR "El trabajo es bueno"
                     SINO IMPRIMIR "El trabajo es excelente"
                 FINSI
        FINSI
FINSI
```

2.3.3 ESTRUCTURAS REPETITIVAS O CÍCLICAS

Los algoritmos estudiados hasta el momento presentan una estructura simple, compuesta sencillamente de una o varias entradas, seguida de algunos procesos y concluyendo con una o varias salidas, pero no todos los problemas tienen soluciones de este tipo, algunos de ellos deben repetir una o más actividades (instrucciones) hasta que ciertas condiciones sean satisfechas, formando lo que se conoce como **ciclos o procesos repetitivos.**

Las estructuras repetitivas o cíclicas se pueden clasificar así:



2.3.3.1 Estructura WHILE-DO (Mientras-Hacer).

Simboliza un conjunto de actividades que se desarrollan mientras una condición sea verdadera. Cuando la condición se torna falsa, se interrumpe dicha ejecución y continúa la secuencia normal en el algoritmo o programa.

La estructura general es:

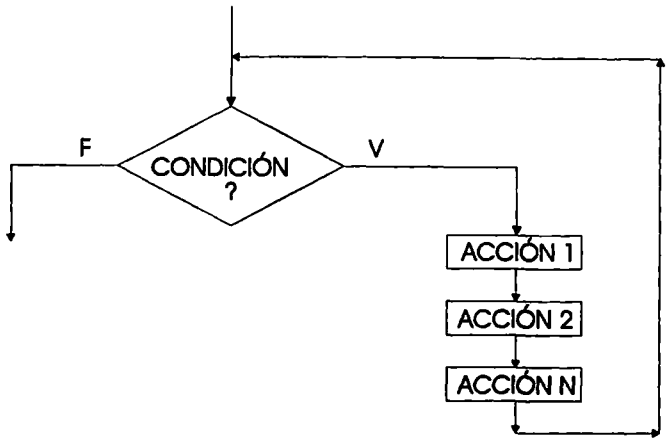
Seudocódigo en inglés

```
WHILE {Condición} DO
   ACCIÓN 1
   ACCIÓN 2
      .
      .
      .
   ACCIÓN N
END
```

Seudocódigo en Español

```
MIENTRAS {Condición} HACER
   ACCIÓN 1
   ACCIÓN 2
      .
      .
      .
   ACCIÓN N
FIN_MIENTRAS
```

Su representación gráfica es:



Dependiendo de la condición analizada, es posible que la acción ó conjunto de acciones jamás se ejecuten.

Dentro del rango de acciones debe haber por lo menos un parámetro que permita que la condición cambie su valor lógico, es decir, que en un momento dado pase de cierto a falso y continúe el flujo del algoritmo, de no ser así se caería en un ciclo infinito.

SINTAXIS PASCAL

```
WHILE condición DO
BEGIN
        ACCIÓN 1;
        ACCIÓN 2;
              .
              .
              .
        ACCIÓN N
END;
```

Todas las acciones que componen el cuerpo de la estructura WHILE van encerradas entre las proposiciones BEGIN y END y separadas entre sí por punto y coma (;).

Cuando el cuerpo de la estructura WHILE está compuesto de sólo una sentencia, no es necesario agruparlo entre BEGIN y END. En este caso el punto y coma colocado al final de la sentencia indica el final físico de la estructura WHILE.

Ejemplo:

```
CON:=1;
WHILE (CON<=10) DO
BEGIN
        A:=1;
        B:=2;
        C:=3;
        SUMA:=A+B+C;
        CON:=CON+1
END;
```

EJERCICIO

Elaborar un algoritmo para que calcule e imprima una tabla de multiplicar.

Solución

a) Análisis

Entrada: Leer el multiplicando o valor de la tabla que se desea calcular.
Proceso: Se inicializa la variable que actúa como multiplicador en 1y se va incrementando de unidad en unidad en la medida en que se realizan los ciclos hasta llegar al límite, en este caso 10. Por cada ciclo se efectúa la multiplicación del multiplicando por el multiplicador.
Salida : Imprimir por cada ciclo, el multiplicando, el multiplicador y el resultado.

b) Variables a utilizar

MDO = Multiplicando
MDOR = Multiplicador
RES = Resultado

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Leer MDO
Paso 3        MDOR ← 1
Paso 4        MIENTRAS MDOR <= 10 HACER
                  RES ← MDO * MDOR
                  IMPRIMIR MDO, MDOR, RES
                  MDOR ← MDOR + 1
              FIN_MIENTRAS
Paso 5      FIN
```

d) Codificación PASCAL

```
PROGRAM TABLA;
{CÁLCULO DE LA TABLA DE MULTIPLICAR}
VAR
    MDO,MDOR,RES : INTEGER;
BEGIN
      WRITELN('Teclee el multiplicando deseado');
      READ(MDO);
      MDOR:= 1;
      WHILE MDOR<=10 DO
      BEGIN
            RES:=MDO*MDOR;
            WRITELN(' ', MDO, '*', MDOR, '=', RES);
            MDOR:=MDOR+1
      END
END.
```

e) Prueba de escritorio

Si la variable leída MDO representa un valor de 5, se tiene:

MDO		MDOR		RES
5	x	1	=	5
	x	2	=	10
	x	3	=	15
		.		.
		.		.
		.		.
		10	=	50

EJERCICIO

Sea la ecuación $y = x^3 - 3x^2 - x + 5$
Calcular Y para cada valor de X que oscila entre -5 y 5 y que sufre incrementos de 0.5 cada vez. Imprimir X y su correspondiente valor Y.

Solución

a) Análisis:
Entrada : No requiere lectura de variables, pero se debe inicializar la variable X en -5.

Proceso : Tan pronto se calcula el valor Y correspondiente a X, se debe proceder a incrementar ésta última en 0.5 por cada ciclo, sin que llegue a superar el valor de 5. Para cada valor de X se evalúa la ecuación.
Salida : Imprimir las variables X, Y

b) Variables a utilizar

X = Valor a ser considerado en la ecuación
Y = Valor resultante de la ecuación

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        X ← -5
Paso 3        MIENTRAS X <= 5 HACER
                  Y ← X ↑ 3 - 3 * X ↑ 2 - X + 5
                  IMPRIMIR X, Y
                  X ← X + 0.5
              FIN_MIENTRAS
Paso 4      FIN
```

2.3.3.1.1 Acumulador

Es una variable que tiene como función acumular o totalizar los cálculos intermedios que se realizan en un proceso. Esta variable se debe inicializar en cero para que no altere los resultados tanto parciales como definitivos.

EJERCICIO

Calcular la suma de los 100 primeros números naturales.

Solución

a) Análisis:

Entrada : No requiere lectura de variables, pero se debe inicializar una variable en el primer número natural ó sea en 1, así mismo se debe inicializar un acumulador en cero.

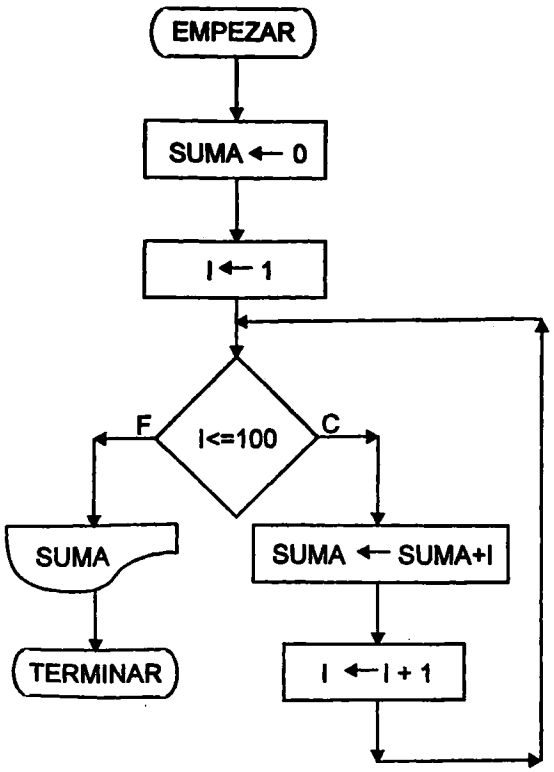
Proceso : Incrementar la variable que haga las veces de número natural en una unidad por cada ciclo, mientras sea inferior a 100, e igualmente aumentar la variable acumuladora en el nuevo número natural.

Salida : Imprimir la variable acumuladora, pero sólo al culminar el proceso de sumar los 100 primeros números naturales.

b) Variables a utilizar

I = Número natural
SUMA= Acumulador

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

Paso 1 **INICIO**
Paso 2 SUMA ← 0
Paso 3 I ← 1

```
Paso 4          MIENTRAS I <= 100 HACER
                    SUMA← SUMA + I
                    I ← I + 1
                FIN_MIENTRAS
Paso 5          IMPRIMIR SUMA
Paso 6      FIN
```

Las instrucciones o actividades que se encuentran dentro del ciclo **MIENTRAS-HACER** se ejecutan tantas veces como la condición lo permita.

Analizar el siguiente algoritmo

```
Paso 1      INICIO
Paso 2          SUMA ← 0
Paso 3          I ← 1
Paso 4          MIENTRAS I < 100 HACER
                    SUMA← SUMA + I
                    I ← I +1
                    IMPRIMIR SUMA
Paso 5          FIN_MIENTRAS
Paso 6      FIN
```

Pregunta

- ¿Cuántos números naturales son procesados?
- ¿Qué valores son impresos?

EJERCICIO

Calcular e imprimir la suma de los números pares comprendidos entre 1 y 100.

Solución

a) Análisis

Entrada : Se debe inicializar el acumulador de números pares en cero. No obstante ser el primer número par 2, se debe inicializar una variable que haga sus veces en 1, por restricciones del enunciado del problema.

Proceso : Se debe aplicar un artificio de tal forma que genere todos los pares comprendidos entre 1 y 100, incluyendo el primero que es 2, e igualmente incrementar la variable que haga las veces de número par en 2 mientras sea inferior a 100.

Salida : Imprimir la variable acumuladora, pero solo al culminar el proceso de sumar todos los números pares.

b) Variables a utilizar

J = Número par
ACUM = Acumulador

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2          ACUM ← 0
Paso 3          J ← 1
Paso 4          MIENTRAS J < 100 HACER
                    ACUM ← ACUM + (J + 1)
                    J ← J + 2
                FIN_MIENTRAS
Paso 5          IMPRIMIR ACUM
Paso 6      FIN
```

d) Prueba de escritorio

	ACUM		J
0			1
0+(1+1)	= 0+2	= 2	3
2+(3+1)	= 2+4	= 6	5
6+(5+1)	= 6+6	= 12	7
12+(7+1)	= 12+8	= 20	9
20+(9+1)	= 20+10	= 30	.
	.		.
	.		.
	.		.
	.		.

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        ACUM ← 0
Paso 3        J ← 1
Paso 4        MIENTRAS (2*J-2)<=100 HACER
                  ACUM ← ACUM + (2*J-2)
                  J ← J+1
              FIN_MIENTRAS
Paso 5        IMPRIMIR ACUM
Paso 6      FIN
```

e) Codificación PASCAL

```
PROGRAM SUMA;
USES CRT;
VAR
    J,ACUM : INTEGER;
{CÁLCULO DE LA SUMA DE LOS NÚMEROS PARES
COMPRENDIDOS ENTRE 1 Y 100}
BEGIN
      ACUM:=0;
      J:=1;
      WHILE ((2*J-2)<100) DO
      BEGIN
            ACUM:=ACUM+(2*J-2);
            J:=J+1;
      END;
      WRITE(' ':10, 'Los números pares comprendidos entre 1 y 100 suman:');
      WRITE(ACUM:8)
END.
```

Pregunta

Es frecuente oír en los estudiantes de lógica computacional, la siguiente pregunta:
Al abordar un problema, ¿cuándo se debe leer y cuándo no ?

Respuesta

Se debe leer cuando los datos a ser tratados se desconocen, por lo tanto éstos son reemplazados por variables. Cuando los datos son conocidos se inicializan directamente las variables en sus respectivos valores o datos.

EJERCICIO

Calcular $SUM = \frac{1}{1} + \frac{1}{3} + \frac{1}{5} + \cdots + \frac{1}{55}$

Solución

a) Análisis

Entrada : Cuando se trata de una sumatoria, se debe inicializar una variable acumuladora en cero.
Observando el denominador de cada término, tiene un comportamiento impar, por lo tanto, se debe inicializar una variable en 1 para que maneje el denominador. Con el numerador no hay problema por ser constante.

Proceso : Generar cada término de la serie mediante incrementos de dos unidades que debe sufrir el denominador, siempre que sea inferior al límite preestablecido que en este caso es 55, e irlo acumulando.

Salida : Imprimir el total acumulado.

b) Variables a utilizar

K = Denominador del término
SUM = Sumatoria de términos

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUM ← 0
Paso 3        K ← 1
Paso 4        MIENTRAS K <= 55 HACER
                  SUM ← SUM + 1 / K
                  K ← K + 2
              FIN_MIENTRAS
Paso 5        IMPRIMIR SUM
Paso 6      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
```

```
Paso 2          SUM ← 0
Paso 3          K ← 1
Paso 4          MIENTRAS K<=55 HACER
                    SUM ← SUM + (1/(2*K-1))
                    K ← K+1
                FIN_MIENTRAS
Paso 5          IMPRIMIR SUM
Paso 6      FIN
```

PROBLEMAS PROPUESTOS

1- Imprimir los 100 primeros números naturales.

2- Producir una tabla de enteros y cuadrados para todos los impares comprendidos entre el 3 y 75 inclusive.

3- Producir una tabla de raíces cuadradas para todos los números impares comprendidos entre 0 y 50.

4- Calcular R = $\frac{1}{5}+\frac{1}{10}+\frac{1}{15}+\frac{1}{20}+\cdots+\frac{1}{100}$

5- Considérese la ecuación $y = x^3 - 16x^2 + 13x - 7$. Diseñar un algoritmo que calcule y para los valores de x que oscilan entre 4 y -4 y que tiene decrementos de 0.2. Imprimir para cada x su correspondiente valor y.

2.3.3.1.2 Lo usual en los procesos computacionales es elaborar soluciones que consideren ciertos volúmenes de información, pero se presenta el inconveniente que el computador no sabría determinar cuando debe culminar de leer un lote de registros ó cuando finalizar la repetición de un cálculo o proceso, salvo que se le indique la terminación del proceso repetitivo y esto se logra mediante el empleo de una variable contadora de ciclos. En la medida en que se incrementen o decrementen los ciclos, así mismo se comporta la variable contadora, hasta llegar a un tope o límite previamente establecido. A este proceso se le conoce con el nombre de **CONTADOR.**

Observando los enunciados de los ejercicios desarrollados hasta el momento, presentan la siguiente formulación: "Se tiene un registro con la siguiente información ...", ó "leer un registro con la siguiente información...", ó "leer en un registro..."; lo cierto es que siempre se trataba de un registro, ahora bien, una de las grandes ayudas que brindan los computadores es el manejo eficiente de grandes volúmenes de información en tiempos mínimos, resultando utópico pensar en resolver un problema en forma computacional para

unos datos individuales o únicos, por lo tanto, lo visto hasta el momento tiene un carácter netamente académico-docente, más no práctico.

De acuerdo a lo anterior y con el soporte del conocimiento de los conceptos de **ACUMULADOR** y **CONTADOR**, se realizarán en adelante ejercicios con un mayor grado de eficiencia y eficacia.

EJERCICIO

Leer 10 valores enteros positivos, uno por registro y obtener el cuadrado, el cubo, raíz cuadrada y raíz cúbica. Imprimir cada valor leído, así como lo hallado.

Solución

a) Análisis

Entrada : Leer un dato por registro.
Inicializar una variable contadora de ciclos en uno.

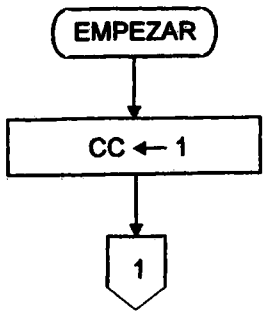
Proceso : Mientras el contador de ciclos sea menor que 10 se debe leer un dato y calcular el cuadrado, el cubo, la raíz cuadrada y raíz cúbica del dato leído.
Incrementar el contador de ciclos en 1.

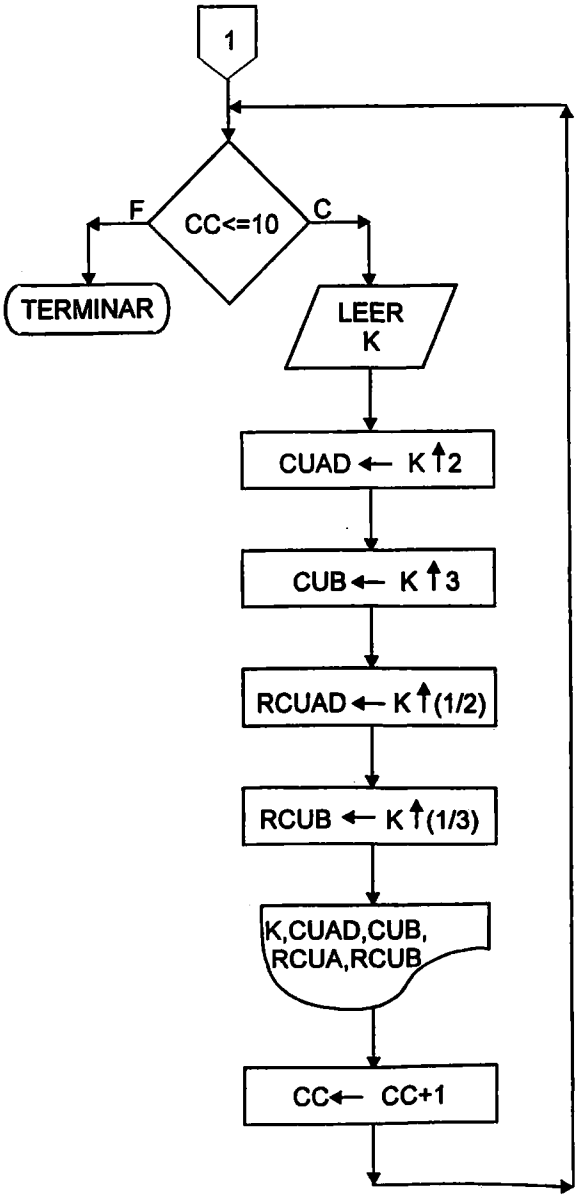
Salida : Por cada dato leído imprimir lo calculado.

b) Variables a utilizar:

CC = contador de ciclos
K = dato a ser leído
CUAD = cuadrado
CUBO = cubo
RCUA = raíz cuadrada
RCUB = raíz cúbica

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CC ← 1
Paso 3        MIENTRAS CC <= 10 HACER
                Leer K
```

```
                    CUAD ← K ↑ 2
                    CUB  ← K ↑ 3
                    RCUA ← K ↑ (1/2)
                    RCUB       ← K ↑ (1/3)
                    IMPRIMIR K, CUAD, CUB, RCUA, RCUB
                    CC ← CC + 1
                FIN_MIENTRAS
Paso 4      FIN
```

La variable contadora de ciclos puede estar inicializada indistintamente en cero o en uno, pero se debe tener especial cuidado al controlar el límite con el fin no incurrir en problemas como realizar menos procesos de los deseados, o por el contrario, sobrepasar los límites establecidos.

Lo anterior se puede observar con un sencillo ejemplo de una nómina de 100 empleados. Por lógica, el cálculo del pago para cada uno de los empleados se realiza en forma individual y es así como se calcula la liquidación respectiva y se edita su correspondiente pago o cheque; si por error en el control del límite, éste sólo realiza 99 ciclos, dejaría de calcular el pago para un trabajador, o si por el contrario realiza 101 ciclos, el flujo del algoritmo se quedaría en espera de ser alimentado con la información correspondiente al trabajador 101, el cual no existe.

EJERCICIO

Presentación del ejercicio inmediatamente anterior trabajado con el contador de ciclos inicializado en CERO.

Algoritmo

```
Paso 1      INICIO
Paso 2        CC ← 0
Paso 3        MIENTRAS CC < 10 HACER
                  Leer K
                  CUAD ← K ↑ 2
                  CUB  ← K ↑ 3
                  RCUA       ← K ↑ (1/2)
                  RCUB       ← K ↑ (1/3)
                  IMPRIMIR K, CUAD, CUB, RCUA, RCUB
                  CC ← CC + 1
              FIN_MIENTRAS
Paso 4      FIN
```

EJERCICIO

Conocida la edad de cada uno de los 50 estudiantes de un grupo, determinar cuántos estudiantes son mayores de 30 años y cuántos tienen 30 años o menos.

Solución

a) Análisis

Entrada : Se requiere de un contador de procesos (ciclos) inicializado en cero.
Un contador de edades mayores de 30 años inicializado en cero.
Un contador de edades menores o iguales a 30 años inicializado en cero.

Proceso : Mientras el contador de ciclos sea inferior a 50, debe leer una edad y compararla contra 30 para determinar si se trata de una edad mayor de 30 años, de ser afirmativo, se incrementa en 1 el contador de mayores de 30 años, en caso contrario, se incrementa en 1 el contador de menores o iguales a 30 años. Una vez hecho esto, se debe incrementar el contador de procesos en 1.

Salida : Imprimir el contador de mayores de 30 años y el contador de menores o iguales a 30 años.

b) Variables a utilizar

CP = Contador de procesos
CPME = Contador de personas menores o iguales a 30 años
CPMA = Contador de personas mayores de 30 años
EDAD = Edad de una persona.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CP ← 0
Paso 3        CPME ← 0
Paso 4        CPMA ← 0
Paso 5        MIENTRAS CP < 50 HACER
                  Leer EDAD
                  SI EDAD <= 30
                     ENTONCES CPME ← CPME + 1
                     SINO CPMA ← CPMA + 1
                  FINSI
                  CP ← CP + 1
              FIN_MIENTRAS
Paso 6        IMPRIMIR CPMA, CPME
Paso 7      FIN
```

EJERCICIO

Con base en la siguiente tabla, determinar el puntaje para cada uno de los 100 deportistas que participan en un evento de natación, con el propósito de conocer sus posibilidades de éxito.

	PARÁMETRO	PUNTAJE
Estado físico	• EXCELENTE	10
	• ACEPTABLE	7
	• REGULAR	5
Disciplina	• EXCELENTE	10
	• ACEPTABLE	7
	• REGULAR	3
Dirección técnica	• EXCELENTE	10
	• ACEPTABLE	7
	• REGULAR	5
Edad	• >= 30	5
	• >=15 y <30	10
	• <15	5

Solución

a) Análisis

Entrada: Se debe leer el código, estado físico, disciplina, dirección técnica y edad por cada deportista.
Se debe inicializar el contador de proceso en cero o uno.
Se debe inicializar el acumulador de puntajes en cero.

Proceso: Como el computador no está en capacidad de cualificar sino de cuantificar, se hace necesario asignar un código a excelente, otro a aceptable y otro a regular, así:

Estado físico	• EXCELENTE	Código 1
	• ACEPTABLE	Código 2
	• REGULAR	Código 3
Disciplina	• EXCELENTE	Código 1
	• ACEPTABLE	Código 2
	• REGULAR	Código 3

y así sucesivamente, de tal forma que cuando se desea analizar la disciplina se

compara contra 2,si es igual corresponde a disciplina aceptable y por lo tanto se le asigna 7 puntos, si al compararse la disciplina contra 2 resulta menor, corresponde a disciplina excelente y se le asigna 10 puntos, de lo contrario la disciplina es regular y se le acumula 5 puntos. Igual análisis se hace con cada uno de los parámetros, excepto con la edad que se puede hacer la comparación directamente.

Salida : Imprimir el código del deportista y el puntaje.

b) Variables a utilizar

CON = Contador de procesos
PUN = Acumulador de puntajes
COD = Código del deportista
EF = Estado físico
DIS = Disciplina
DT = Dirección técnica
EDAD = Edad

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CON ← 0
Paso 3        MIENTRAS CON<100 HACER
                PUN ← 0
                LEER COD,EF,DIS,DT,EDAD
                SI EF=2
                  ENTONCES PUN ← PUN + 7
                  SINO SI EF<2
                            ENTONCES PUN ← PUN + 10
                            SINO     PUN ← PUN + 5
                       FINSI
                FINSI
                SI DIS=2
                  ENTONCES PUN ← PUN + 7
                  SINO SI DIS<2
                            ENONCES PUN ← PUN + 10
                            SINO     PUN ← PUN + 3
                       FINSI
                FINSI
                SI DT=1
                  ENTONCES PUN ← PUN + 10
                  SINO SI DT=2
                            ENTONCES PUN ← PUN + 7
                            SINO     PUN ← PUN + 5
                       FINSI
```

```
                    FINSI
                    SI (15<=EDAD<30)
                       ENTONCES PUN ← PUN + 10
                       SINO     PUN ← PUN + 5
                    FINSI
                    IMPRIMIR COD,PUN
                    CON ← CON + 1
                 FIN_MIENTRAS
Paso 4        FIN
```

d) Codificación Fortran

```
      PROGRAM DEPORT
C     Cálculo del puntaje de los 100 deportistas
      WRITE(*,3)
   3  FORMAT(// 10X, 'Teclee código, estado físico, disciplina, dirección técnica y edad
     *de cada uno de los 100 deportistas')
      CON= 0
 200  IF (CON .GE. 100)GOTO 50
        IPUN= 0
        READ(*,5)ICOD,IEF,IDIS,IDT,IEDAD
   5    FORMAT(2X,I3,4(2X,I2))
        IF (IEF .EQ. 2) THEN
            IPUN= IPUN + 7
          ELSEIF (IEF .LT. 2) THEN
            IPUN= IPUN + 10
          ELSE
            IPUN= IPUN + 5
        ENDIF
        IF (IDIS .EQ. 2) THEN
          IPUN= IPUN + 7
        ELSEIF (IDIS .LT. 2) THEN
          IPUN= IPUN + 10
        ELSE
          IPUN= IPUN + 3
        ENDIF
        IF (IDT - 2)10,12,14
  10      IPUN= IPUN + 10
          GOTO 100
  12      IPUN= IPUN + 7
          GOTO 100
  14      IPUN= IPUN + 5
 100    IF ((IEDAD .GE. 15) .AND. (IEDAD .LT. 30)) THEN
```

```
            IPUN= IPUN + 10
          ELSE
            IPUN= IPUN + 5
        ENDIF
        WRITE(*,17)ICOD,IPUN
17      FORMAT(// 20X, 'El deportista código', 2X,I4,2X, 'obtuvo un puntaje de
     *  :',2X,I3)
        CON= CON + 1
        GOTO 200
50    STOP
      END
```

Este ejercicio se ha codificado con el IF lógico y el IF aritmético, permitiéndole al estudiante hacer un paralelo entre éstos, mediante la visualización de su forma de operación

EJERCICIO

Calcular e imprimir el número de términos necesarios para que el valor de la siguiente sumatoria se aproxime a 3000 sin que la exceda.

$$\sum_{J=1}^{?} \frac{J^2+1}{J}$$

Solución

a)Análisis

Entrada: Se inicializa la sumatoria de términos en cero y el valor inicial del término J en cero.
Este ejercicio no requiere lectura de datos.

Proceso: Mientras la sumatoria sea menor que 3000 se calcula $(J^2+1)/(J)$ y se almacena en una variable, para posteriormente acumularla. Si en un momento determinado se sobrepasa la sumatoria, se debe retroceder un ciclo, ya que es en ese ciclo donde se encuentra el término que hace que la sumatoria se aproxime lo más cerca a 3000.

Salida : Imprimir el término J y la sumatoria.

b) Variables a utilizar

SUM = Sumatoria
J = Término de la sumatoria
X = Almacena el resultado de evaluar $(J^2+1)/J$

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        J ← 0
Paso 3        SUM ← 0
Paso 4        MIENTRAS SUM < 3000 HACER
                  J←J+1
                  X←(J↑2 + 1)/J
                  SUM←SUM + X
              FIN_MIENTRAS
Paso 5        SI SUM=3000
                  ENTONCES IMPRIMIR J,SUM
                  SINO SI  SUM>3000
                           ENTONCES J← J-1
                                    SUM← SUM-X
                                    IMPRIMIR J,SUM
                       FINSI
              FINSI
Paso 6      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        J ← 0
Paso 3        SUM ← 0
Paso 4        MIENTRAS SUM < 3000 HACER
                  J←J+1
                  X←(J↑2 + 1)/J
                  SUM←SUM + X
              FIN_MIENTRAS
Paso 5        SI SUM>3000
                  ENTONCES  J← J-1
                            SUM← SUM-X
              FINSI
Paso 6        IMPRIMIR J,SUM
Paso 7      FIN
```

EJERCICIO PROPUESTO

Analizar el siguiente algoritmo mediante una prueba de escritorio y determinar lo que calcula, así como el último valor que se imprime en la variable IMP.

Variable a utilizar

IMP = Número impar

Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        IMP ← 1
Paso 3        MIENTRAS IMP < 50 HACER
                 IMPRIMIR IMP
                 IMP ← IMP + 2
              FIN_MIENTRAS
Paso 4      FIN
```

Pregunta
¿Cuál es la diferencia entre un CONTADOR y un ACUMULADOR?

Respuesta
La variable que hace las veces de CONTADOR, sufre modificaciones por cada ciclo en valores constantes, en cambio, la variable ACUMULADORA se ve afectada en cantidades variables por cada iteración.

2.3.3.2 Estructura REPEAT-UNTIL (REPETIR HASTA_QUE).

Esta estructura se ejecuta cíclicamente en tanto el valor de la condición sea falso, es allí donde cesa la repetición y continúa la marcha normal del algoritmo. Esta estructura es exactamente contraria a la estructura **WHILE-DO**.

La estructura general es:

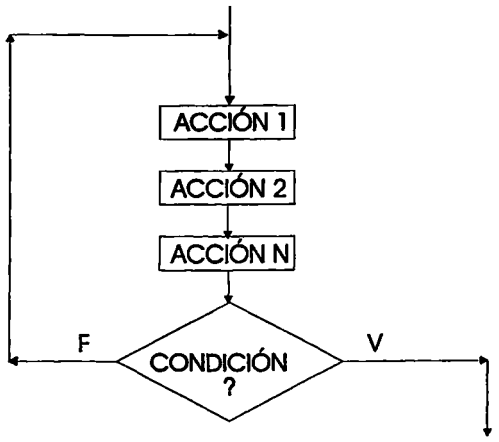
Seudocódigo en inglés

```
REPEAT
   ACCIÓN 1
   ACCIÓN 2
      .
      .
      .
   ACCIÓN N
UNTIL {CONDICIÓN}
```

Seudocódigo en español

```
REPETIR
   ACCIÓN 1
   ACCIÓN 2
      .
      .
      .
   ACCIÓN N
HASTA_QUE {CONDICIÓN}
```

Su representación gráfica es:



Esta estructura se ejecuta al menos una vez.

Observaciones:

- Debe haber por lo menos un parámetro que permita que la condición pase de falso a verdadero.

- Todo lo que es capaz de hacer el **REPEAT** lo puede hacer el **WHILE** y con mayor eficiencia, ya que por el hecho de examinar antes de ejècutar evita situaciones anómalas; es en estos casos cuando se requiere del **WHILE-DO**, siendo esta estructura más amplia o general.

- En la primera pasada o ciclo del **REPEAT** se pueden establecer las condiciones iniciales; en el **WHILE-DO** se deben establecer las condiciones iniciales antes de entrar en él, por ejemplo tener dos bloques de lectura, uno para arrancar y el otro para permanecer en el cuerpo del **WHILE-DO**.

- La utilización de la estructura **REPEAT** es recomendable cuando se conoce previamente el número de repeticiones a efectuar; así mismo es muy empleada para validación de datos.

EJERCICIO

Asumiendo que no se conoce el concepto de división aritmética, determinar si un número entero positivo es par o impar.

Solución

a) Análisis

Entrada: Leer el valor a ser analizado

Proceso: Mientras el valor a ser analizado sea mayor que 1, se le resta sucesivamente 2, hasta que sea igual a cero o uno, determinando así si es par o impar respectivamente.

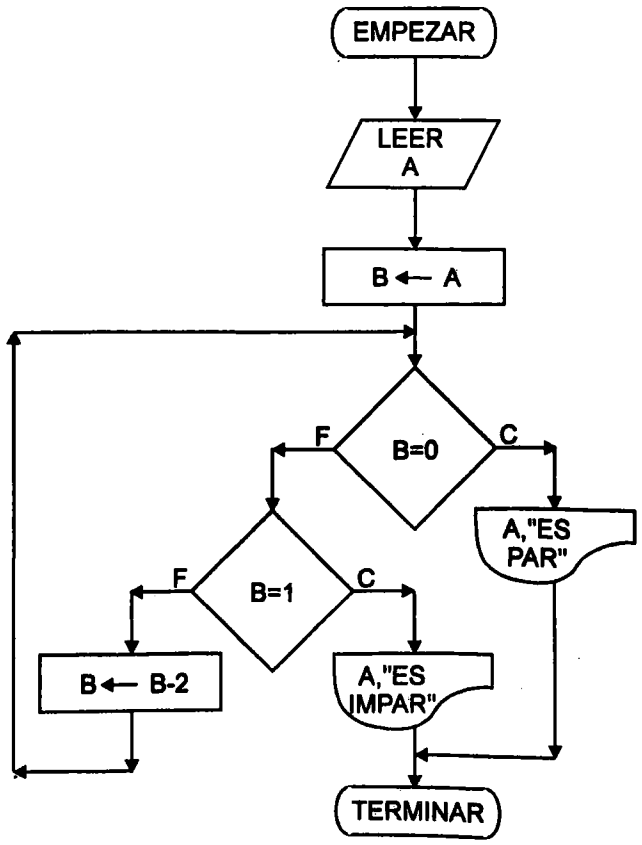
Para no distorsionar el valor original por las restas sucesivas, es conveniente almacenarlo en una segunda variable, sobre la cual se efectuará el trabajo.

Salida : Imprimir el valor analizado y un mensaje indicando si es par o impar.

b) Variables a utilizar

A = Valor a ser analizado
B = Variable temporal que almacena el valor A

c) Algoritmo en diagrama de flujo

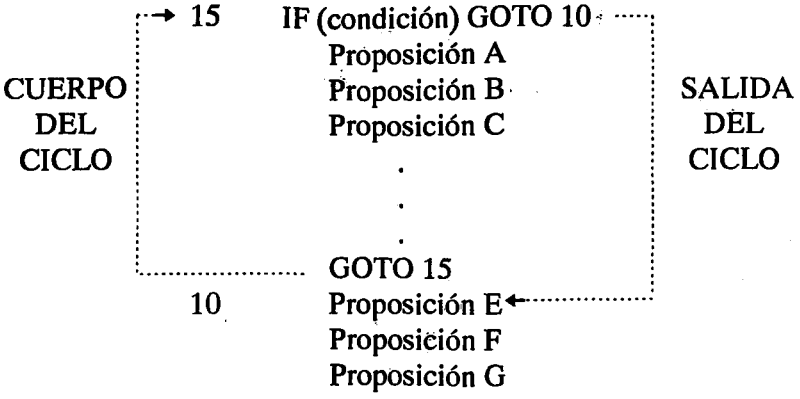


d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER A
Paso 3        B ← A
Paso 4        SI B = O
                 ENTONCES IMPRIMIR A, "Es par"
                 SINO SI B = 1
                          ENTONCES IMPRIMIR A, "Es impar"
                       FINSI
              FINSI
Paso 5        SI B > 1
                 ENTONCES REPETIR
                                B ← B - 2
                                SI B = 0
                                   ENTONCES IMPRIMIR A, "Es par"
                                   SINO SI B = 1
                                            ENTONCES IMPRIMIR
                                                     A, "Es impar"
                                         FINSI
                                SI
                             HASTA_QUE (B = 0) O (B = 1)
              FINSI
Paso 6      FIN
```

SINTAXIS FORTRAN

Las estructuras de repetición WHILE y REPEAT no fueron incluídas en la versión estándar del FORTRAN 77, por lo tanto es necesario utilizar la proposición de transferencia GOTO N para poder simular la forma como actúan.



En el anterior ejemplo se utilizaron dos proposiciones de transferencia de control, el GOTO 15 que permite efectuar ciclos repetitivos mientras la condición no se cumpla o sea falsa, cuando la condición se convierte en verdadera se rompe el ciclo, transfiriendo el control a la instrucción que está referenciada con la etiqueta 10, para continuar con la ejecución del programa. Es indispensable que exista dentro del cuerpo del ciclo una proposición tal que afecte la condición de la proposición IF, para que en un momento dado deje de ser falsa y no realice más ciclos.

No es posible transferir el control mediante la proposición GOTO N a una instrucción que no esté debidamente etiquetada, obedeciendo la definición de la etiqueta prácticamente al criterio del programador.

La proposición de transferencia de control GOTO N se encuentra presente en la gran mayoría de los leguajes de alto nivel, incluyendo el FORTRAN y el PASCAL, pero su uso debe limitarse a los casos estrictamente necesarios; no obstante, la dependencia en FORTRAN estándar de esta proposición es notoria, pero no sucede lo mismo con el PASCAL, por lo tanto su utilización en la presente obra será ninguna cuando se programe en este último lenguaje.

El tratar de limitar el empleo del GOTO N, se justifica en el hecho de que su uso frecuente origina desorden en la programación y dificulta el seguimiento de programas cuando de localizar errores o estudiar su lógica se trata.

Ejemplo:

```
        CON=1
10      IF (CON .GT. 10) GOTO 13
           A=1
           B=2
           C=3
           SUMA=A+B+C
           CON=CON+1
           GOTO 10
13      ENDIF
```

SINTAXIS PASCAL

```
REPEAT
   ACCION 1;
   ACCION 2;
      .
      .
   ACCION N
UNTIL CONDICION;
```

Todas las acciones que componen el cuerpo de la estructura **REPEAT** van separadas entre sí por punto y coma y no requiere que se agrupen entre BEGIN y END porque las palabras **REPEAT** y **UNTIL** sirven de delimitadores.

Ejemplo

```
CONTADOR:= 0;
REPEAT
      CONTADOR:= CONTADOR + 1;
      WRITELN(CONTADOR:3)
UNTIL CONTADOR = 10;
```

Ejemplo

```
CONTADOR:= 1;
REPEAT
      READLN(A);
      CONTADOR:= CONTADOR + 1
UNTIL CONTADOR>10;
WRITELN('Se leyeron 10 valores A, uno por cada registro');
```

EJERCICIO

Generar la siguiente sucesión: 1, 8, 15, 22, ... 50. Hallar la suma de sus términos.

Solución

a) Análisis

Entrada: El ejercicio no requiere lectura de datos porque se conocen. Se debe inicializar la sumatoria de términos en cero, y una variable que haga las veces del término en uno, puesto que la serie se inicia en uno.

Proceso: Observando la serie, ésta tiene un crecimiento constante de 7 unidades. Se debe realizar tantos ciclos como términos hasta llegar al término 50 inclusive y cada que se genere un término debe ser acumulado.

Salida : Se debe imprimir la serie y el acumulado de términos.

b) Variables a utilizar

J = Término de la serie.
SUM = Sumatoria de términos.

c) Algoritmo en seudocódigo

Paso 1 **INICIO**

```
Paso 2          J ← 1
Paso 3          SUM ← 0
Paso 4          REPETIR
                    IMPRIMIR J
                    SUM ← SUM + J
                    J ← J + 7
                HASTA_QUE J > 50
Paso 5          IMPRIMIR SUM
Paso 6      FIN
```

d) Codificación FORTRAN

```
      PROGRAM SUCESION
C     GENERA UNA SUCESION DE TERMINOS
      J = 1
      SUM = 0
 50   IF (J .GT. 50) GOTO 100
      WRITE(*,1)J
  1   FORMAT(2X,I3)
      SUM = SUM + J
      J = J + 7
      GOTO 50
100   WRITE(*,2)SUM
  2   FORMAT(10X, 'La suma de los términos es:',F4.0)
      STOP
      END
```

e) Codificación PASCAL

```
PROGRAM SUCESION;
{Genera una serie de términos}
USES CRT;
VAR
    J,SUM :INTEGER;
BEGIN
      J:= 1;
      SUM:= 0;
      REPEAT
          WRITE(' ',J:3);
          SUM:= SUM + J;
          J:= J + 7
      UNTIL J > 50;
```

```
        WRITELN(' ',:10, 'La suma de los términos es :', SUM:4)
END.
```

EJERCICIO

Encontrar el promedio de edad de los 2000 estudiantes de una universidad, mayores de 21 años y el promedio del resto. Por cada estudiante se dispone de un registro que contiene el código del estudiante y el año de nacimiento.

Solución

a) Análisis

Entrada: Se requiere inicializar un contador de ciclos en cero ó en uno.
Un contador de edades mayores de 21 años en cero.
Un contador de edades menores de 21 años en cero.
Un acumulador de edades mayores de 21 años en cero.
Un acumulador de edades menores de 21 años en cero.
Se debe leer por cada registro el código y el año de nacimiento de un estudiante.

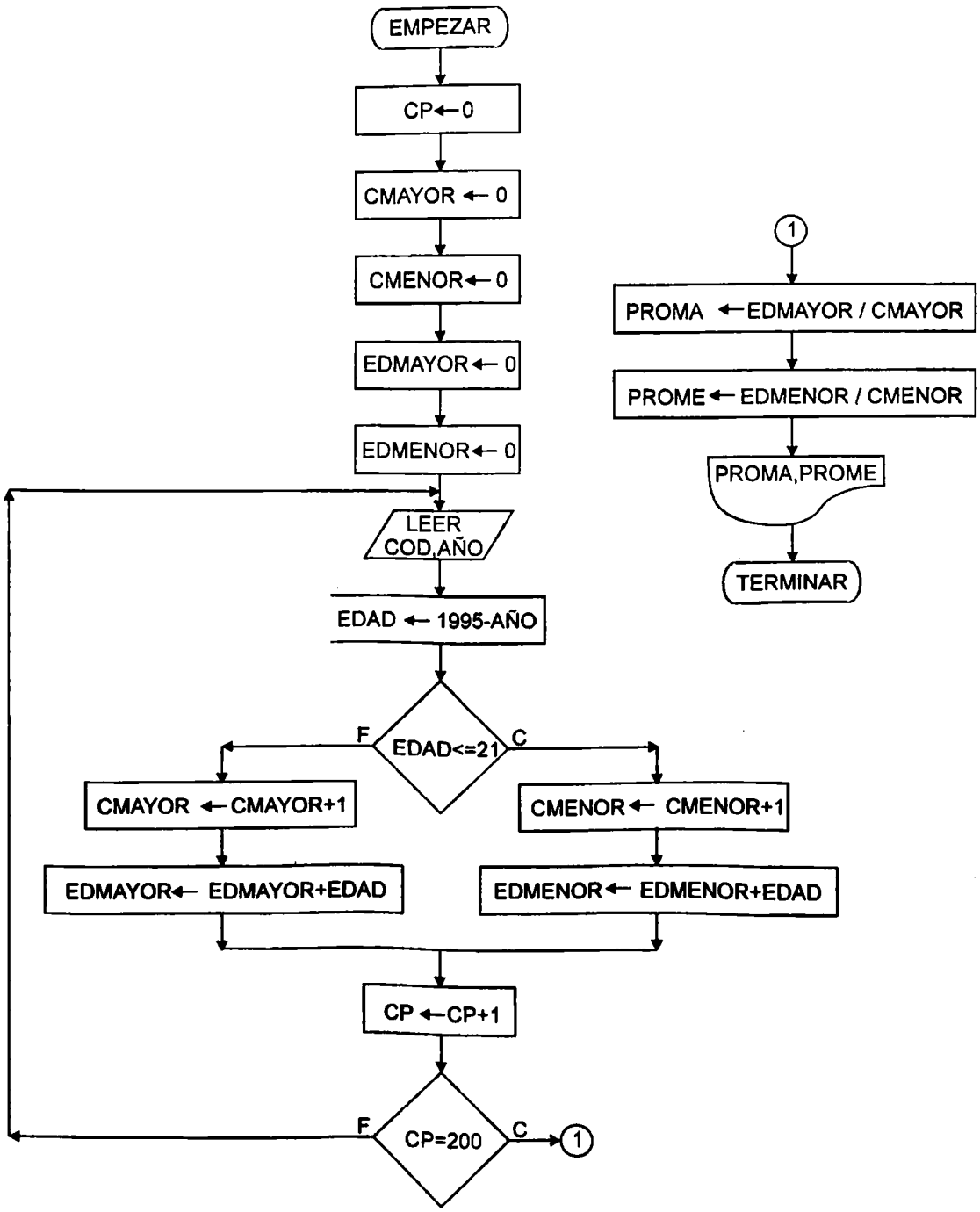
Proceso: Se van a realizar tantos ciclos hasta que la condición de terminación lo permita (2000 ciclos, uno por cada estudiante analizado).
Cada que se analice la edad de un estudiante se incrementará el contador de edades respectivo y así mismo se acumulará la edad correspondiente. Una vez realizados todos los ciclos, se procede a hallar el promedio de estudiantes mayores de 21 años y el promedio de estudiantes menores de 21 años.

Salida : Imprimir el promedio de estudiantes mayores de 21 años y el promedio de estudiantes menores de 21 años.

b) Variables a utilizar

CP = Contador de procesos.
CMAYOR = Contador de edades mayores de 21 años.
CMENOR = Contador de edades menores de 21 años.
EDMAYOR = Acumulador de edades mayores de 21 años.
EDMENOR = Acumulador de edades menores de 21 años.
PROMA = Promedio de edades mayores de 21 años.
PROME = Promedio de edades menores de 21 años.
COD = Código del estudiante.
AÑO = Año de nacimiento.
EDAD = Edad del estudiante.

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CP ← 0
Paso 3        CMAYOR ← 0
Paso 4        CMENOR ← 0
Paso 5        EDMAYOR ← 0
Paso 6        EDMENOR ← 0
Paso 7        REPETIR
                 LEER COD,AÑO
                 EDAD ← 1995 - AÑO
                 SI EDAD <= 21
                    ENTONCES  CMENOR ← CMENOR + 1
                              EDMENOR ← EDMENOR + EDAD
                    SINO   CMAYOR ← CMAYOR + 1
                           EDMAYOR ← EDMAYOR + EDAD
                 FIN_SI
                 CP ← CP + 1
              HASTA_QUE  CP = 2000
Paso 8        PROMA ← EDMAYOR / CMAYOR
Paso 9        PROME ← EDMENOR / CMENOR
Paso 10       IMPRIMIR PROMA, PROME
Paso 11     FIN
```

e) Codificación PASCAL

```
PROGRAM PROMEDIOS;
USES CRT;
{Cálculo del promedio de edades de los 2000 estudiantes de una universidad}
VAR
     CP,CMAYOR,CMENOR,EDMAYOR,EDMENOR,COD,ANO,EDAD  :INTEGER;
     PROMA,PROME                                    :REAL;
BEGIN
       CP:= 0;
       CMAYOR:= 0;
       CMENOR:= 0;
       EDMAYOR:= 0;
       EDMENOR:= 0;
       REPEAT
          WRITELN(' ':10, 'Teclee código del estudiante y año de nacimiento :');
          READLN(COD,ANO);
```

```
            EDAD:= 1995 - ANO;
            IF EDAD <= 21 THEN
               BEGIN
                  CMENOR:= CMENOR + 1;
                  EDMENOR:= EDMENOR + EDAD
               END
            ELSE
               BEGIN
                  CMAYOR:= CMAYOR + 1;
                  EDMAYOR:= EDMAYOR + EDAD
               END;
            CP:= CP + 1;
         UNTIL CP = 2000;
         PROMA:= EDMAYOR / CMAYOR;
         PROME:= EDMENOR / CMENOR;
         WRITELN(' ':10, 'El promedio de edades de estudiantes mayores de 21 años es :',
                  PROMA:3:2);
         WRITELN(' ':10, 'El promedio de edades de estudiantes menores de 21 años es :',
                  PROME:3:2)
END.
```

EJERCICIO

Leer 20 registros, cada uno conteniendo una EDAD. Determinar:

- Número de personas que tienen 20 años o menos.
- Número de personas que son mayores de 20 años y menores ó iguales a 40 años.
- Número de personas que son mayores de 40 años y menores o iguales a 60 años.
- Número de personas mayores de 60 años.

Solución

a) Análisis

Entrada: Se requiere de cinco contadores que se inicializan todos en cero y son:

- Contador de registros
- Contador de menores o iguales a 20 años
- Contador de mayores de 20 años y menores o iguales a 40 años.
- Contador de mayores de 20 años y menores o iguales a 60 años.
- Contador de mayores de 60 años.

Se debe leer una edad por registro.

Proceso: Determinar en qué intervalo se encuentra la edad leída para incrementar el

correspondiente contador, así como el contador de registros sin que sobrepase de veinte (20).

Salida : Imprimir los diferentes contadores de edades.

b) Variables a utilizar

NR = Número de registros
NP20 = Número de personas menores o iguales a 20 años
NP40 = Número de personas mayores de 20 años y menores o iguales a 40 años.
NP60 = Número de personas mayores de 40 años y menores o iguales a 60 años.
NPMA = Número de personas mayores de 60 años
EDAD = Edad de la persona.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        NR ← 0
Paso 3        NP20 ← 0
Paso 4        NP40 ← 0
Paso 5        NP60 ← 0
Paso 6        NPMA ← 0
Paso 7        REPETIR
                LEER EDAD
                SI EDAD <= 20
                  ENTONCES NP20 ← NP20 + 1
                  SINO SI EDAD <= 40
                            ENTONCES NP40 ← NP40 + 1
                            SINO SI EDAD <= 60
                                      ENTONCES NP60 ← NP60 + 1
                                      SINO      NPMA ← NPMA + 1
                                    FINSI
                          FINSI
                FINSI
                NR ← NR + 1
              HASTA_QUE NR = 20
Paso 8        IMPRIMIR NP20, NP40, NP60, NPMA
Paso 9      FIN
```

EJERCICIO

Leer un número entero y calcular su factorial.
Recuerde que 0! = 1

3! = 1x2x3 = 6
N! = 1x2x.... xN

Solución

a) Análisis

Entrada: Leer la variable a la cual se le calculará el factorial.
Inicializar una variable en 1, que se comportará como el multiplicador.
Inicializar una variable acumuladora del multiplicador en 1.
Proceso: Es necesario controlar que el valor leído no sea negativo, en caso de presentarse, se debe producir un mensaje de inconsistencia .
Se debe efectuar la multiplicación de la variable acumuladora por el multiplicador y el resultado almacenarlo en ella misma, e incrementar el multiplicador en 1 hasta que sea mayor que la variable leída, que en este caso hace las veces de límite.
Salida : Imprimir el valor leído, así como el factorial calculado.

b) Variables a utilizar

N = Valor al cual se le calculará el factorial
FACN = Factorial de N (o productoria)
J = Multiplicador

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        FACN ← 1
Paso 3        J ← 1
Paso 4        LEER N
Paso 5        SI N < 0
                 ENTONCES IMPRIMIR " Error en Datos"
                 SINO REPETIR
                          FACN ← FACN * J
                          J ← J + 1
                      HASTA_QUE J > N
                      IMPRIMIR N, FACN
              FINSI
Paso 6      FIN
```

d) Prueba de escritorio

Si N = 4 se tiene:

N	FACN	J
4	1	1
	1 x 1 = 1	1 + 1 = 2
	1 x 2 = 2	2 + 1 = 3
	2 x 3 = 6	3 + 1 = 4
	6 x 4 = 24	4 + 1 = 5

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        FAC ← 1
Paso 4        J ← 1
Paso 5        SI N < 0
                 ENTONCES IMPRIMIR N " No tiene factorial "
                 SINO SI N = 0
                           ENTONCES IMPRIMIR N,FAC
                           SINO REPETIR
                                     FAC ← FAC * J
                                     J ← J + 1
                                HASTA_QUE J > N
                                IMPRIMIR N, FAC
                      FINSI
              FINSI
Paso 6      FIN
```

TERCERA VERSIÓN DEL ALGORITMO:

Se puede realizar el ejercicio procediendo regresivamente, inicializando la multiplicatoria en N y el multiplicador se va rebajando en una unidad por cada ciclo sin dejarlo bajar de 1, ya que vuelve cero la productoria.

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        FAC ← N
Paso 4        J ← N - 1
Paso 5        SI N < 0
                 ENTONCES IMPRIMIR N, " No tiene factorial "
                 SINO SI (N = 0) O (N = 1)
                           ENTONCES FAC ← 1
                                     IMPRIMIR N,FAC
```

```
                    SINO REPETIR
                            FAC ← FAC * J
                            J ← J - 1
                        HASTA_QUE J = 1
                        IMPRIMIR N, FAC
                FINSI
        FINSI
Paso 6  FIN
```

Observación

Çuando se requiere inicializar variables con propósito de sumatoria, éstas deben partir de cero para que no se presenten distorsiones en los resultados. Si los fines son de productoria, las variables deben comenzar en 1 para que el multiplicador no convierta la función en cero, como sucedería si fueran empezadas en cero.

d) Codificación FORTRAN

```
      PROGRAM FACTO
C     CALCULA EL FACTORIAL DE UN NUMERO DADO
      FACN = 1
      J = 1
      READ(*,1)N
    1 FORMAT(I2)
      IF (N .LT. 0) THEN
         WRITE(*,2)
    2    FORMAT(5X, 'Error en datos')
   50 ELSEIF (J .GT. N) GOTO 100
         FACN = FACN*J
         J = J + 1
         GOTO 50
  100    WRITE(*,200)N,FACN
  200    FORMAT(10X, 'El factorial de',I2, 'es:',5X,F10.0)
      ENDIF
      STOP
      END
```

EJERCICIO

Calcular: $SUM = \frac{1!}{2} + \frac{2!}{4} + \frac{3!}{6} + \frac{4!}{8} + \cdots \frac{N!}{2N}$

Solución

a) Análisis

Entrada: Leer N ó límite de la serie
Inicializar el numerador en 1, sumatoria de términos en cero y multiplicador del factorial en 1.

Proceso: Tan pronto se calcula el numerador de cada término de la serie se halla el denominador como 2 veces el numerador. Conformado cada fraccionario, se va acumulando.
Se debe validar la variable N, hasta tanto se tenga la certeza de que es un dato bueno, para así evitar que el compilador sea alimentado con datos erróneos.
La variable N hace las veces de límite.

Salida : Imprimir el acumulado

b) Variables a utilizar

FACT = Multiplicador del factorial
SUM = Sumatoria de términos
N = Límite de términos
I = Numerador
DOBLE = Denominador

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        FACT ← 1
Paso 3        SUM ← 0
Paso 4        I ← 1
Paso 5        REPETIR
                 LEER N
                 SI N < 0
                    ENTONCES IMPRIMIR "Error en datos, teclee de nuevo."
                 FINSI
              HASTA_QUE N > 0
Paso 6        REPETIR
                 FACT ← FACT * I
                 DOBLE ← 2 * I
                 SUM ← SUM + FACT / DOBLE
                 I ← I + 1
              HASTA_QUE I > N
Paso 7        IMPRIMIR SUM
Paso 8      FIN
```

2.3.3.3 Estructura FOR (PARA).

Cuando se conoce de antemano el número de veces que debe ejecutarse un proceso, se puede realizar automáticamente mediante la utilización de una variable a la cual se le asignan condiciones iniciales, condiciones finales y los incrementos que sufre ésta. Cuando el incremento es negativo, el ciclo no se ejecuta si el valor inicial es menor que el límite, o también, cuando siendo el incremento positivo y el valor inicial es mayor que el valor límite.

La estructura general es:

Seudocódigo en inglés

```
FOR V: VI TO VF,IN DO
        ACCIÓN 1
        ACCIÓN 2
            .
            .
            .
        ACCIÓN N
END
```

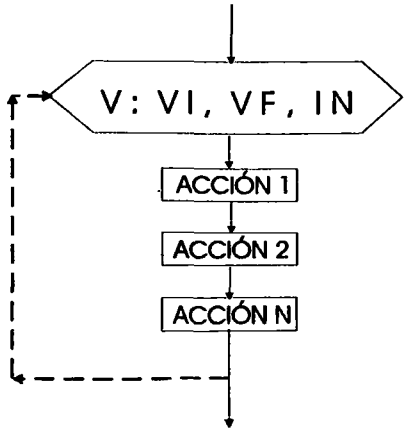
Seudocódigo en español

```
PARA V: DE VI HASTA VF, IN HACER
        ACCIÓN 1
        ACCIÓN 2
            .
            .
            .
        ACCIÓN N
FIN_PARA
```

Siendo:

V = Variable controladora del ciclo
VI = Valor inicial
VF = Valor final
IN = Incrementos

Su representación gráfica es:



Cuando no se indica el valor del incremento, se asume la unidad.

Esta estructura es sumamente flexible y práctica, por lo tanto al utilizarse simplifica bastante la solución de los problemas, porque no requiere inicializar variables controladoras de ciclos, ni incrementarlas o compararlas contra límites preestablecidos ya que lo hace en forma automática.

EJERCICIO

Se desea calcular la cantidad a pagar de matrícula para cada uno de los 2000 estudiantes de una universidad, en la siguiente forma:

Total a pagar es igual a costos fijos más costos variables.

Los costos fijos son de $1000 y los costos variables se calculan sumando el 12% del patrimonio y el 17% de la renta.

Por cada estudiante se tiene un registro con la siguiente información: código del estudiante, nombre del estudiante, renta y patrimonio.

Solución

a) Análisis

Entrada: Leer código del estudiante, nombre, renta y patrimonio. Hacer costos fijos igual a 1000.

Proceso: Aplicar las siguientes fórmulas:
Costo variable = Renta x 12% y Patrimonio x 17%
Costo total = Costo fijo + Costo variable.

Salida : Imprimir código del estudiante, nombre, renta, patrimonio, costos fijos, costos variables y costo total.

b) Variables a utilizar

COD = Código del estudiante
NOM = Nombre
RENTA = Renta
PAT = Patrimonio
CFIJO = Costo fijo
CVAR = Costo variable
TPAG = Total a pagar
I = Variable índice de la estructura FOR (PARA)

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CFIJO ← 1000
```

```
Paso 3          PARA I DE 1 HASTA 2000 HACER
                    LEER COD, NOM, RENTA, PAT
                    CVAR ← RENTA * 0.12 + PAT * 0.17
                    TPAG ← CFIJO + CVAR
                    IMPRIMIR COD, NOM, RENTA, PAT, CFIJO, CVAR, TPAG
                FIN_PARA
Paso 4      FIN
```

SINTAXIS FORTRAN

DO n I=M1,M2,M3

n = Debe ser un número entero e identifica la última instrucción a ser ejecutada en el rango del DO.
I = Debe ser variable entera. Se le conoce como la variable índice de la estructura DO y es la que lleva el control de los ciclos que se realizan.
M1 = Es el valor inicial que toma la variable I.
M2 = Es el valor final que toma la variable I.
M3 = Es el incremento por cada ciclo que sufre la variable I. Cuando no se especifica, se asume incrementos de una unidad.
M1, M2, M3 = Deben ser constantes o variables enteras no suscritas.

La última instrucción del rango de un DO debe ser una instrucción CONTINUE con etiqueta o cualquier otra instrucción ejecutable, excepto instrucciones que sean de transferencia de control como GOTO, IF, DO, PAUSE, STOP, RETURN.

Ejemplo

```
      DO 4  I = 50, 500, 5
        J=I/5
4     CONTINUE
```

La variable I tiene un valor inicial de 50, un valor final de 500 y por cada ciclo sufre incrementos de 5, por lo tanto los valores que toma I son 50, 55, 60, 65 ... 500.

En la mayoría de los casos, el número de ciclos ejecutados en una instrucción DO es muy legible.

Ejemplo DO 50 I = 1, 100

El DO realiza ciclos hasta que el valor final sea menor que el valor inicial, generando así 100 ciclos.

Ejemplo

```
      DO 70  K = 3, 100, 3
70        RAIZ = K**(1/2)
```

Este ejemplo es válido ya que la última instrucción del DO es una instrucción ejecutable y forma parte del rango del DO. Aquí se está calculando la raíz cuadrada de los múltiplos de 3 hasta 100.

Ejemplo

```
      DO 7  J = -3, 20, 4
         M = J**2
         WRITE(*,*)M
7     CONTINUE
```

El ciclo se realiza mientras J sea menor o igual a 20, en consecuencia J tomará los siguientes valores: -3, 1, 5, 9, 13, 17.
Si el valor final del ciclo de un DO es mayor que el valor inicial y el incremento es negativo, el ciclo no es ejecutado.

Ejemplo `DO 100  L = 1, 50, -2`

No es permitida ninguna instrucción en el rango del DO, que cambie cualquiera de sus parámetros.

Ejemplo

```
      DO 7  NUM = 1, 100
         ICUAD = NUM**2
         NUM = NUM - 2
7     CONTINUE
```

En este caso está siendo alterada la variable índice del DO por restas de 2 (NUM=NUM-2).
Los parámetros del DO pueden ser variables enteras, siempre y cuando hayan sido definidas previamente.

Ejemplo

```
      SUM = 0
      K = 3
      DO 2000  NUM = K, 50
2000      SUM = SUM + NUM
```

No es permitido transferir el control de fuera del rango del DO a una instrucción que forme parte del mismo.

Ejemplo

```
      SUM = 0
      GOTO 3
      DO 77  I = 1, 20, 5
         KUAD = I**2
3        SUM = SUM + KUAD
77    CONTINUE
```

El número de ciclos que realiza una instrucción DO se calcula de la siguiente forma:

$$Numero\ de\ ciclos = \frac{Valor\ final - Valor\ inicial + Incremento}{Incremento}$$

Ejemplo DO 15 K = 3, 22, 2

$$Numero\ de\ ciclos = \frac{22 - 3 + 2}{2} = 10.5 = 10$$

Se debe tomar la parte entera de la división. Si el resultado de aplicar la fórmula da un valor negativo, el ciclo no se ejecuta.

SINTAXIS PASCAL

```
FOR I:= VI TO VF DO
      BEGIN
         {Cuerpo del ciclo;
      END;
```

I = Identificador índice de la estructura FOR. Debe ser un dato estándar excepto real.
VI = Valor inicial que toma el identificador I.
VF = Valor final que toma el identificador I.

Los incrementos o decrementos de la estructura FOR siempre son de una unidad. Cuando el valor inicial es menor o igual al valor final se debe utilizar la partícula TO

Ejemplo

```
      SUMA:= 0;
      FOR I:= 10 TO 30 DO
```

```
BEGIN
    SUMA:= SUMA + I;
    WRITELN(' ':2, SUMA:4);
END;
```

Cuando el valor inicial es mayor que el valor final se debe utilizar la partícula DOWNTO.

Ejemplo

```
SUMA:= 0;
FOR I:= 30 DOWNTO 10 DO
   BEGIN
      SUMA:= SUMA + I;
      WRITELN(' ':2, SUMA:4);
   END;
```

Si el cuerpo de la estructura FOR está conformado por sólo una proposición, no es necesario agruparla entre BEGIN y END; en este caso la delimitación del ciclo se senala con el punto y coma.

Ejemplo

```
SUMA:= 0;
FOR J:= 10 TO 100 DO
   SUMA:= SUMA + J;
```

EJERCICIO

Leer por registro, el código de un empleado, el salario básico por hora y el número de horas trabajadas en la semana. Calcular el salario neto para una nómina de 100 empleados teniendo en cuenta que si el número de horas trabajadas durante la semana es mayor de 48, las horas demás, se consideran horas extras y tienen un recargo del 35%. Imprimir código y salario neto para cada empleado.

Solución

a) Análisis

Entrada: Se aplica una estructura **PARA (FOR)**, en donde su valor inicial es 1, su valor final es el total de empleados, ó sea 100 y formando parte de la estructura, se tiene el cálculo del salario bruto igual al salario básico hora por el número de horas trabajadas.

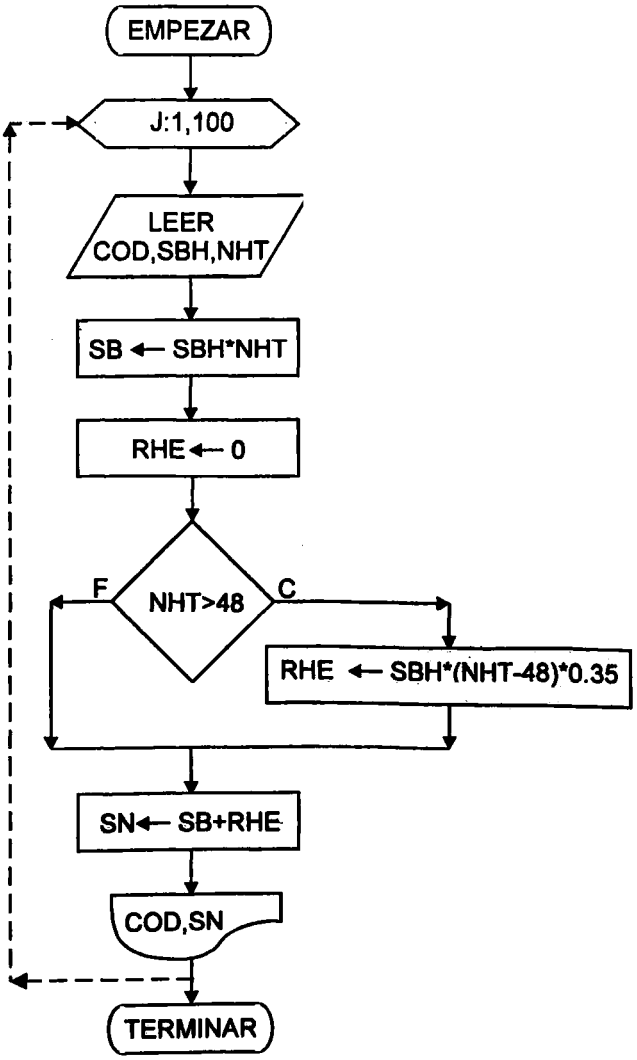
Proceso: Comparar número de horas trabajadas contra 48,si es mayor, se calcula el recargo correspondiente a horas extras como salario básico hora x (número de horas trabajadas - 48) * 0.35.

Calcular el salario neto = salario bruto + recargo de horas extras.
Salida : Código del empleado, salario neto.

b) Variables a utilizar

COD = Código del empleado
SBH = Salario básico hora
NHT = Número de horas trabajadas en la semana
SB = Salario bruto
RHE = Recargo por horas extras
SN = Salario neto
J = Variable índice de la estructura **FOR (PARA)**

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA J DE 1 HASTA 100 HACER
                LEER COD, SBH, NHT
                SB ← SBH * NHT
                RHE ← 0
                SI NHT > 48
                  ENTONCES RHE ← SBH * (NHT-48) * 0.35
                FINSI
                SN ← SB + RHE
                IMPRIMIR COD, SN
              FIN_PARA
Paso 3      FIN
```

En el presente ejercicio se inicializó la variable RHE (recargo por horas extras) en cero, para que no afecte el salario neto por indefinición cuando al comparar NHT (número de horas trabajadas) contra 48 resulte ser menor o igual.

e) Codificación FORTRAN

```
      PROGRAM NOMINA
C     CALCULO DE LA NOMINA DE 100 TRABAJADORES
      WRITE(*,1)
1     FORMAT(// 10X, 'Por favor digite el código del trabajador, salario básico hora y
     *número de horas trabajadas en la semana')
      DO 5 J = 1, 100
        READ(*,3)COD,SBH,NHT
3       FORMAT(5X,F6.0,5X,F5.0,5X,I2)
        SB = SBH*NHT
        RHE = 0
        IF (NHT .GT. 48) THEN
          RHE = SBH*(NHT - 48)*0.35
        ENDIF
        SN = SB + RHE
        WRITE(*,7)COD,SN
7       FORMAT(// 10X, 'El trabajador código :',2X,F6.0,5X, 'Gana un salario neto de
     *$',F7.0)
5     CONTINUE
      STOP
      END
```

f) Codificación PASCAL

```
PROGRAM NOMINA;
{CALCULO DE LA NOMINA DE 100 TRABAJADORES}
USES CRT;
VAR
    J,COD,NHT        :INTEGER;
    SB,SBH,RHE,SN    :REAL;
BEGIN
   WRITELN(' ':10, 'Por favor digite el código del trabajador, salario básico hora y
            número de horas trabajadas en la semana');
   FOR J:= 1 TO 100 DO
      BEGIN
         READLN(COD,SBH,NHT);
         SB:= SBH*NHT;
         RHE:= 0;
         IF (NHT > 48) THEN
            RHE:= SBH*(NHT - 48)*0.35;
         SN:= SB + RHE;
         WRITELN(' ':10, 'El trabajador código :', COD:6,' ':45, 'Gana un salario neto
                  de $',SN:7:0)
      END
END.
```

Observación

El límite máximo o tope de un conjunto de datos puede ser reemplazado por una variable. En este caso se compara el contador de registros o contador de ciclos contra la variable límite para determinar el final de un proceso ó ciclo repetitivo; de emplearse la estructura **PARA**, no es necesario hacer la comparación anterior porque la estructura lo controla automáticamente. La variable límite debe ser definida previamente, siendo lo más lógico, mediante lectura.

EJERCICIO

Una editorial estima los precios de sus libros de acuerdo al siguiente criterio: El precio básico de un libro es de $100 más 0.70 centavos por página. Si el número de páginas es superior a 250, el precio del libro se incrementará en $300, pero si el número de páginas excede de 500, el precio del libro se aumentará en otros $1000,00.

Se dispone de un volumen de M libros y por cada registro se conoce el código del libro y el número de páginas. Calcular e imprimir el valor de cada libro identificándolo con su código.

Solución

a) Análisis:

Entrada: Leer el límite de datos.
Leer por cada registro, el código del libro y el número de páginas.
Proceso: Calcular el precio básico del libro como $100,00 + 0.70 por número de páginas.
Dependiendo del número de páginas, se tiene un recargo, así:
Si el número de páginas es mayor de 250 y menor o igual a 500 el recargo será de $300, pero si el número de páginas excede de 500 sufre un incremento de $1300.
Salida : Imprimir código y precio de cada libro.

b) Variables a utilizar

M = Límite de datos a analizar
J = Índice de la estructura PARA
CÓDIGO = Código del libro
NPAG = Número de páginas
PBAS = Precio básico del libro
PLIB = Precio del libro

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER M
Paso 3        PARA J DE 1 HASTA M HACER
                  LEER CÓDIGO, NPAG
                  PBAS ← 100 + 0.7 * NPAG
                  SI  NPAG <= 250
                     ENTONCES PLIB ← PBAS
                     SINO  SI  NPAG <= 500
                                 ENTONCES PLIB ← PBAS + 300
                                 SINO  PLIB ← PBAS + 1300
                           FINSI
                  FINSI
                  IMPRIMIR CÓDIGO, NPAG, PLIB
              FIN_PARA
Paso 4      FIN
```

EJERCICIO

Calcular la combinatoria de N en M, según la siguiente fórmula:

$$\begin{bmatrix} N \\ M \end{bmatrix} = \frac{N!}{M!(N-M)!}$$

Restricción: N debe ser positivo y mayor que M.

Solución

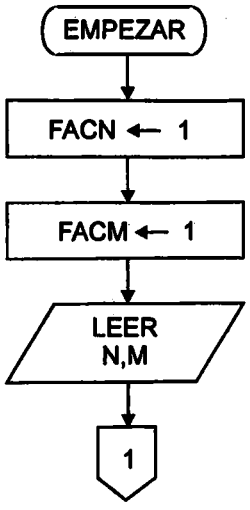
a) Análisis

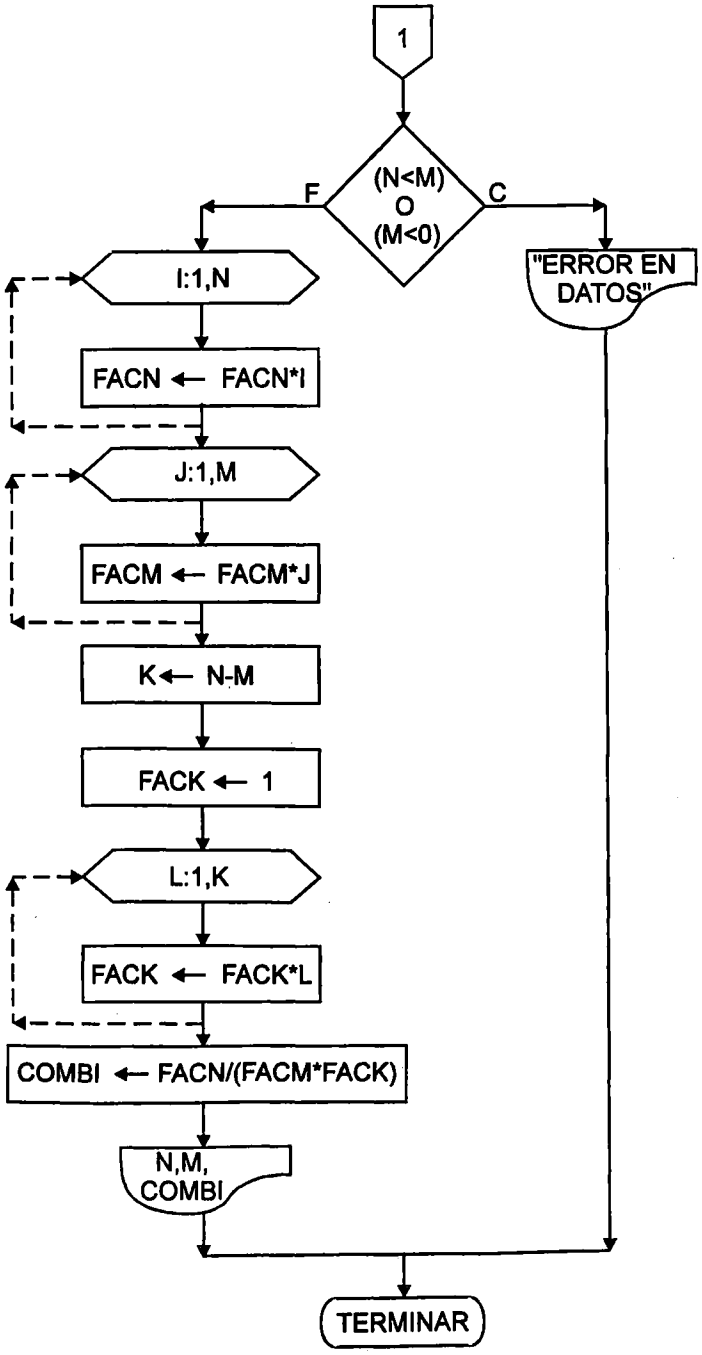
Entrada: Leer N, M
Inicializar variables multiplicatorias en 1 para ser empleadas en el cálculo de los factoriales.
Proceso: Chequear que N sea mayor que M
Calcular la fórmula por partes, primero el factorial de N, luego el factorial de M y por último el factorial de la diferencia de N con M.
Salida : Imprimir N, M y la combinatoria

b) Variables a utilizar

N, M, K = Valores a ser factorizados
FACN = Factorial de N
FACM = Factorial de M
FACK = Factorial de K
COMBI = Combinatoria
I, J, L = Variables índices de los **FOR (PARA)**

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        FACN ← 1
Paso 3        FACM ← 1
```

```
Paso 4          LEER N, M
Paso 5          SI (N < M) O (M < 0)
                    ENTONCES IMPRIMIR "ERROR EN DATOS"
                    SINO PARA I DE 1 HASTA N HACER
                            FACN ← FACN * I
                         FIN_PARA
                         PARA J DE 1 HASTA M HACER
                            FACM ← FACM * J
                         FIN_PARA
                         K ← N - M
                         FACK ← 1
                         PARA L DE 1 HASTA K HACER
                            FACK ← FACK * L
                         FIN_PARA
                         COMBI ← FACN / (FACM * FACK)
                         IMPRIMIR N, M, COMBI
                FINSI
Paso 6          FIN
```

e) Codificación FORTRAN

```
      PROGRAM COMBIN
C     CALCULO DE LA COMBINATORIA DE N EN M
      FAC = 1
      FACM = 1
      WRITE(*,1)
1     FORMAT(// 10X, 'Por favor teclee el valor de N y M')
      READ(*,3)N,M
3     FORMAT(2I2)
      IF ((N .LT. M) .OR. (M .LT. 0)) THEN
         WRITE(*,10)
10       FORMAT(// 10X, 'Error en datos')
      ELSE
         DO 7  I = 1,N
7           FACM = FACM*I
         DO 9  J= 1,M
9           FACM= FACM*J
         K = N - M
         FACK = 1
         DO 5  L = 1,K
5           FACK = FACK*L
         COMBI = FACN/(FACM*FACK)
         WRITE(*,20)N,M,COMBI
```

```
20      FORMAT(//10X, 'La combinatoria de :', 2X,I2, 'en',2X,I2, 'es igual a',2X,F8.0)
      ENDIF
      STOP
      END
```

EJERCICIO

En una empresa manufacturera se calcula el jornal para cada uno de sus M obreros de acuerdo al salario básico hora por el número de horas trabajadas, más una bonificación de $100 por cada unidad extra producida sobre un mínimo de producción diaria.

Cuando la producción es inferior a la producción mínima establecida, el jornal es equivalente al salario básico hora multiplicado por el número de horas trabajadas.

Por cada obrero se lee: código del obrero, nombre, salario básico hora, número de horas trabajadas, producción diaria, y producción mínima por día.

Se debe producir un informe que contenga la misma información leída, además del salario neto por día para cada obrero. Adicionalmente la empresa requiere saber el total de unidades extras producidas, así como el total pagado por nómina cada día.

Solución

a) Análisis

Entrada: Se requiere de dos acumuladores, uno para el número total de unidades extras producidas y el otro para el total de nómina pagada, ambos inicializados en cero. Se debe leer el límite de datos a procesar, así como el código del obrero, nombre, salario básico hora, número de horas trabajadas, producción diaria y producción mínima diaria por cada uno de los obreros.

Proceso: Emplear las siguientes fórmulas:

Salario básico diario = salario básico hora x número de horas trabajadas.

Número de unidades extras producidas = producción diaria - producción mínima diaria.

Salario neto diario = salario básico diario + número de unidades extras producidas x 100.

Número total de unidades extras producidas = Número total de unidades extras producidas + número de unidades extras producidas.

Total nómina pagada = total nómina pagada + salario neto diario.

Salida : Imprimir código del obrero, nombre, salario básico hora, producción mínima diaria, producción diaria, salario neto diario, número total de unidades extras producidas y total nómina pagada.

b) Variables a utilizar

COD = Código del obrero
NOM = Nombre
SBH = Salario básico hora
NHT = Número horas trabajadas
PMD = Producción mínima diaria
PD = Producción diaria
NUEP = Número de unidades extras producidas
NTUEP = Número total de unidades extras producidas
SND = Salario neto diario
TNP = Total nómina pagada
SBD = Salario básico diario
M = Límite de datos a ser procesados
K = Variable índice de la estructura PARA (FOR)

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        TNP ← 0
Paso 3        NTUEP ← 0
Paso 4        LEER M
Paso 5        PARA K DE 1 HASTA M HACER
                LEER COD, NOM, SBH, NHT, PMD, PD
                SBD ← SBH * NTH
                SI PD > PMD
                  ENTONCES NUEP ← PD - PMD
                           NTUEP ← NTUEP + NUEP
                           SND ← SBD + NUEP * 100
                  SINO SND ← SBD
                FINSI
                IMPRIMIR COD, NOM, SBH, PMD, PD, SND
                TNP ← TNP + SND
              FIN_PARA
Paso 6        IMPRIMIR NTUEP, TNP
Paso 7      FIN
```

CONCLUSIÓN

Estudiadas las estructuras repetitivas o cíclicas se puede llegar a la conclusión:

- La estructura REPEAT_UNTIL se podrá utilizar cuando el ciclo se deba realizar al menos una vez; es de gran aplicación cuando de validar datos se trata.
- La estructura FOR, DO sólo se puede emplear cuando se conoce previamente el número de iteraciones a desarrollar y la variable índice sea de tipo ordinal. Tanto en FORTRAN como en PASCAL, la variable índice de la estructura FOR ó DO queda indefinida, una vez ha concluido el número de ciclos o iteraciones.
- Para los demás casos se debe emplear la estructura WHILE_DO.

PROBLEMAS PROPUESTOS

1- Qué valores se imprimen al ejecutarse la siguiente fracción de algoritmo.

```
INICIO
    SUMA ← 0
    MIENTRAS J <= 18 HACER
       SUMA ← SUMA + J
       I ← I + 2
    FIN_MIENTRAS
    IMPRIMIR J, SUMA
FIN
```

2- Dados los siguientes datos de entrada 5, 10, 15, 0, 20, determinar que valores se imprimen según el siguiente algoritmo.

```
INICIO
    SUMA ← 0
    LEER X
    MIENTRAS X > 0 HACER
       SUMA ← SUMA + X
       LEER X
    FIN_MIENTRAS
    IMPRIMIR SUMA
FIN
```

3- Qué valores se imprimen al ejecutarse el siguiente algoritmo.

```
INICIO
    I ← 1
    SUMA ← 0
    MIENTRAS I < 5 HACER
       CUA ← I ↑ 2
       SUMA ← SUMA + CUA
       I ← I + 2
```

```
    FIN_MIENTRAS
    IMPRIMIR SUMA
FIN
```

4- Cuántos ciclos realiza la siguiente estructura REPETIR.

```
INICIO
    A ← 7
    B ← 5
    REPETIR
       A ← A + 1
    HASTA_QUE A > B
FIN
```

5- Cuál es el valor final de la VARIABLE SUMA ?

```
INICIO
    SUMA ← 0
    PARA I DE 1 HASTA 5, 2 HACER
       L ← 2 * I
       PARA J DE 1 HASTA L HACER
          SUMA ← SUMA + J
       FIN_PARA
    FIN_PARA
    IMPRIMIR SUMA
FIN
```

6- Qué valores se imprimen en las variables X, Y ?

```
INICIO
    PARA X DE 1 HASTA 3 HACER
       SI  X <= 1
          ENTONCES Y ← X - 1
          SINO SI X <= 2
                  ENTONCES Y ← X*X
                  SINO Y ← X ↑ (1/2)
               FINSI
       FINSI
    FIN_PARA
    IMPRIMIR X,Y
FIN
```

7- Cuántos ciclos realiza la estructura REPETIR ? Qué valor se imprime en la variable SUM ?

```
INICIO
    SUM ← 0
    A ← 1
    B ← 2
    REPETIR
       C ← A * B
       SUM ← SUM + C
    HASTA_QUE A > 5
    IMPRIMIR SUM
FIN
```

8- Realizar prueba de escritorio al siguiente algoritmo y determinar qué produce:

```
INICIO
    C ← 0
    IP ← 1
    I ← 1
    REPETIR
       C ← C + IP
       IMPRIMIR I,C
       IP ← IP + 2
       I ← I + 1
    HASTA_QUE I > 20
FIN
```

9- Realizar prueba de escritorio al siguiente algoritmo y determinar qué produce:

```
INICIO
    TN ← 0
    TP ← 0
    TC ← 0
    CT ← 0
    MIENTRAS CT <= 10 HACER
       LEER T
       SI T < 0
          ENTONCES TN ← TN + 1
          SINO SI T = 0
                    ENTONCES TC ← TC + 1
                    SINO  TP ← TP + 1
                 FIN_SI
       FIN_SI
       CT ← CT + 1
```

```
    FIN_MIENTRAS
    IMPRIMIR TN, TP, TC
FIN
```

10-Cuántos ciclos realiza el siguiente algoritmo. Sustente la respuesta.

```
INICIO
    SUMA ← 0
    CONTADOR ← 10
    MIENTRAS CONTADOR <> 20 HACER
       LEER NUMERO
       SUMA ← SUMA + NUMERO
       CONTADOR ← CONTADOR + 3
    FIN_MIENTRAS
FIN
```

11-Realizar prueba de escritorio al siguiente algoritmo y determinar qué produce.

```
INICIO
    J ← 0
    REPETIR
       I ← (J*2) + 1
       R ← I ↑ (1/2)
       IMPRIMIR J,R
       J ← J + 1
    HASTA_QUE J >=50
FIN
```

12-Determinar qué produce el siguiente algoritmo mediante la elaboración de una prueba de escritorio.

```
INICIO
    LEER A,N
    P ← 0
    SI A<> 0  Y  N > 0
       ENTONCES C ← 1
                  REPETIR
                     P ← P + A
                     C ← C + 1
                  HASTA_QUE C > N
    FINSI
    IMPRIMIR A,N,P
FIN
```

13-De 300 matrimonios encuestados, se desea saber cuántos tienen hijos y cuántos no; además se desea conocer el promedio del número de hijos. Por cada registro leido se dispone de la siguiente información: cédula y número de hijos.

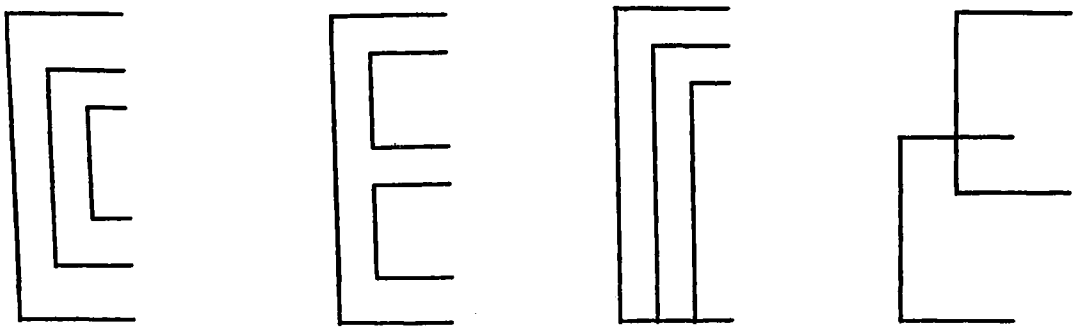
14-Leer un lote de N registros, cada uno de ellos conteniendo la siguiente información: sexo, año de nacimiento y estado civil, se desea saber:

- Número de hombres solteros que pueden votar.
- Número de hombres casados que pueden votar.
- Número de mujeres solteras que pueden votar.
- Número de mujeres casadas que pueden votar.
- Número total de personas solteras que pueden votar.
- Número total de personas casadas que pueden votar.
- Total de personas que pueden votar.

15-En cierto peaje se desea conocer cuántos automóviles y cuántos buses pasaron en determinado día, así mismo se quiere saber cuál fue el promedio de personas que viajaron en automóvil y el promedio de personas que viajaron en bus. Encontrar lo anterior teniendo en cuenta que por cada uno de los N registros se lee el tipo de vehículo (0=automóvil, 1=bus) y el número de ocupantes.

2.3.4 ESTRUCTURAS REPETITIVAS ANIDADAS

Así como se pueden anidar estructuras de decisión, también es posible efectuar anidamientos con estructuras repetitivas, teniendo especial cuidado para que la estructura interna quede totalmente incluida en la estructura externa, sin que se presente sobreposición o traslapamiento de ellas.



Las gráficas a,b,c corresponden a anidamientos efectuados correctamente, pero no sucede lo mismo con el gráfico d, porque en él se presenta traslape de estructuras.

Se pueden realizar tantas combinaciones de estructuras repetitivas como se desee, es decir, se pueden realizar ciclos FOR - FOR, WHILE - FOR, REPEAT - WHILE, FOR- REPEAT, REPEAT - REPEAT, WHILE - WHILE, etc. Igualmente no existe límite al número de anidamiento de estructuras, ya que es determinado por el ejercicio que se esté resolviendo. Se debe tener presente al ejecutarse una estructura repetitiva anidada, que por cada valor que toma la variable índice de la estructura externa se ejecuta totalmente la estructura interna.

Ejemplo

```
PARA J DE 1 HASTA 3 HACER
  PARA L DE 1 HASTA 4 HACER
    CUAD ← L ↑ 2
    IMPRIMIR CUAD
  FIN_PARA
FIN_PARA
```

Prueba de escritorio

J	L	CUAD
1	1	$1^2 = 1$
	2	$2^2 = 4$
	3	$3^2 = 9$
	4	$4^2 = 16$
	5	
2	1	$1^2 = 1$
	2	$2^2 = 4$
	3	$3^2 = 9$
	4	$4^2 = 16$
	5	
3	1	$1^2 = 1$
	2	$2^2 = 4$
	3	$3^2 = 9$
	4	$4^2 = 16$
	5	
4		

Observando la prueba de escritorio se tiene que cuando J vale 1 se ejecuta totalmente la estructura interna L, que está variando de 1 a 4, luego J se incrementa a 2 y se activa de nuevo la estructura interna L y así sucesivamente hasta culminar la estructura

externa J, es decir, hasta que J llegue a 4, superando el valor final y abandonando el anidamiento.

Ejemplo

```
REPETIR
  Leer J
  PARA L DE 1 HASTA 4 HACER
    CUAD ← L ↑ 2
    IMPRIMIR CUAD
  FIN_PARA
HASTA_QUE J <= 0
```

Prueba de Escritorio

Si J = 3	L	CUAD
	1	$1^2 = 1$
	2	$2^2 = 4$
	3	$3^2 = 9$
	4	$4^2 = 16$

Se sale de la estructura interna y lee un nuevo valor de J. Así:

Si J = 7	L	CUAD
	1	$1^2 = 1$
	2	$2^2 = 4$
	3	$3^2 = 9$
	4	$4^2 = 16$

El proceso termina cuando lee un valor J <= 0

EJERCICIO

Para determinar el pago mensual un almacén de electrodomésticos tiene en cuenta el monto del crédito y el plazo, aplicando la siguiente fórmula:

$$PAGO\ MENSUAL = \frac{DEUDA\ TOTAL}{PLAZO} + INTERESES$$

donde interés = 3% del saldo.

Por cada uno de los 70 clientes, se dispone de la siguiente información: código del cliente, nombre, deuda total y plazo. Determinar para cada cliente, los intereses pagados, el pago mensual y el saldo, para cada mes según el plazo concedido.

Solución

a) Análisis:

Entrada: Leer código, nombre, deuda total y plazo.
Inicializar los acumuladores de total de intereses y total cuota en cero.
Proceso: Se utilizarán dos estructuras PARA anidadas. Una para leer la información de cada uno de los clientes y la otra estructura interna para calcular e imprimir la correspondiente tabla de pagos mensuales para cada cliente.
Salida : Imprimir código, nombre, deuda total, plazo, pago mensual, saldo, total de intereses.

b) Variables a utilizar

NOM = Nombre
COD = Código
DTOTAL = Deuda total
TINT = Total intereses
PLAZO = Plazo concedido
TCUOTA = Total cuota
CUOTA = Cuota mensual
SALDO = Saldo
INT = Interés
PAGOME= Pago mensual
I, J = Variables índice de las estructuras FOR (PARA)

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA I DE 1 HASTA 70 HACER
                LEER COD, NOM, DTOTAL, PLAZO
                IMPRIMIR COD, NOM, DTOTAL, PLAZO
                TINT ← 0
                TCUOTA ← 0
                CUOTA ← DTOTAL / PLAZO
                PARA J DE 1 HASTA PLAZO HACER
                  TCUOTA ← TCUOTA + CUOTA
                  SALDO ← DTOTAL - TCUOTA
                  INT ← SALDO * 0.03
```

```
                    PAGOME ← CUOTA + INT
                    TINT ← TINT + INT
                    IMPRIMIR PAGOME, SALDO, TINT
                FIN_PARA
            FIN_PARA
Paso 3      FIN
```

EJERCICIO

Hallar el factorial de un número M suponiendo que no se conoce la multiplicación. El factorial de un número N es igual a la suma de N veces el factorial de (N - 1).

```
1! = 1                  = 1
2! = 1 + 1              = 2
3! = 2 + 2 + 2          = 6
4! = 6 + 6 + 6 + 6      = 24
5! = 24+24+24+24+24     = 120
```

Solución

a) Análisis

Entrada : Se debe leer M y validarlo para tener la certeza de que es mayor que cero, igualmente se debe inicializar la sumatoria en cero y el factorial en uno.
Proceso : Se utilizará un anidamiento de estructuras PARA, el ciclo más interno calculará la sumatoria de N veces el factorial anterior.
Salida : Imprimir M y su factorial.

b) Variables a utilizar

M = Valor al cual se le calculará el factorial.
FAC = Factorial.
SUM = Sumatoria.
I, J = Variable índice del PARA.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        REPETIR
                  LEER M
                  SI M < 0
                     ENTONCES IMPRIMIR "Error en dato, digite nuevamente"
```

```
                    FINSI
                HASTA_QUE M > 0
Paso 3          FAC ← 1
Paso 4          PARA I DE 1 HASTA M HACER
                    SUM ← 0
                    PARA J DE 1 HASTA I HACER
                        SUM ← SUM + FAC
                    FIN_PARA
                    FAC ← SUM
                FIN_PARA
Paso 5          IMPRIMIR M,FAC
Paso 6      FIN
```

d) Codificación PASCAL

```
PROGRAM FACTO;
{CALCULO DEL FACTORIAL DE M SUPONIENDO QUE NO SE CONOCE LA
MULTIPLICACION}
USES CRT;
VAR
    I,J,M          :INTEGER;
    SUM,FAC        :REAL;

BEGIN
   WRITELN(' ':5, 'Digite un valor entero positivo :');
   REPEAT
      READ(M);
      IF M < 0 THEN
         WRITELN(' ':5, 'Error en dato, digite nuevamente :');
   UNTIL M >= 0;
   FAC:= 1;
   FOR I:= 1 TO M DO
      BEGIN
          SUM:= 0;
          FOR J:= 1 TO I DO
             SUM:= SUM + FAC;
          FAC:= SUM;
      END;
   WRITELN(''':10, 'El factorial de :',M:3, 'es =',FAC:10:0)
END.
```

EJERCICIO

El seno de un ángulo en radianes puede calcularse usando N términos de la siguiente serie:

$$SenX = \sum_{N=1}^{M} (-1)^{N+1} . \frac{X^{(2N-1)}}{(2N-1)!}$$

Calcular el seno de X para un lote de 10 registros, en donde por cada registro se lee un valor X y un valor M.

En la expresión X debe convertirse a radianes.

Solución

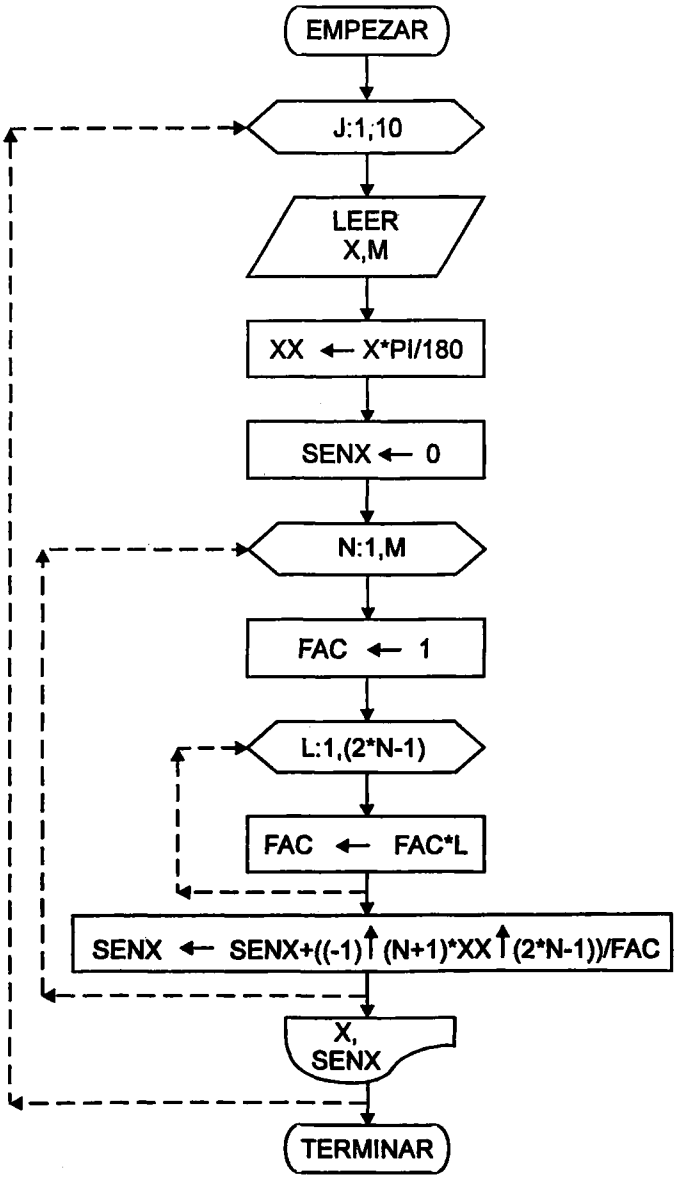
a) Análisis

Entrada: Leer las variables X, M en 10 registros, uno por cada ciclo.
Inicializar la sumatoria en cero y el factorial en uno.
Proceso: Se utiliza un anidamiento de 3 estructuras PARA, la más externa controlará el número de registros, el ciclo intermedio calculará la sumatoria y el ciclo más interno calculará el factorial de (2N - 1).
Se debe convertir el ángulo X a radianes.
Salida : Imprimir X y el seno de X.

b) Variables a utilizar

X = Ángulo a ser leido.
XX = Ángulo expresado en radianes.
N = Inicio de la sumatoria, variable índice de la estructura PARA.
M = Límite de la sumatoria.
SENX = Seno de X.
FAC = Factorial de (2N - 1)
J,L = Variable índice de la estructura PARA

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

Paso 1 **INICIO**
Paso 2 **PARA J DE 1 HASTA 10 HACER**
LEER X,M
XX ← X*π / 180
SENX ← 0

```
                    PARA N DE 1 HASTA M HACER
                       FAC ← 1
                       PARA L DE 1 HASTA (2*N - 1) HACER
                           FAC ← FAC*L
                       FIN_PARA
                       SENX ← SENX + ((-1)↑(N+1) * XX↑(2*N-1)) / FAC
                    FIN_PARA
                    IMPRIMIR X,SENX
                FIN_PARA
Paso 3      FIN
```

d) Codificación FORTRAN

```
      PROGRAM SENO
C     CALCULO DEL SENO DE UN ANGULO EXPRESADO EN RADIANES
      WRITE(*,1)
1     FORMAT(// 10X, 'Digite el ángulo y el límite de la sumatoria :')
      DO 5  J=1,10
         READ(*,3)X,M
3        FORMAT(F4.0,I3)
         XX = X*3.14159 / 180
         SENX = 0
         DO 9  N = 1, M
            FAC = 1
         DO 7  L = 1, (2*N - 1)
7           FAC = FAC*L
9        SENX = SENX + ((-1)**(N + 1) * XX**(2*N - 1)) / FAC
5     WRITE(*,10)X,SENX
10    FORMAT(// 10X, 'El seno de :',F4.0, 'es =',F8.5)
      STOP
      END
```

EJERCICIO

Calcular e imprimir para X = 5, 10, 15, 20 ... 360 grados, el seno de X mediante la expresión:

$$SenX = \frac{X^1}{1!} - \frac{X^3}{3!} + \frac{X^5}{5!} - \frac{X^7}{7!} + \frac{X^9}{9!} - \cdots \frac{X^{41}}{41!}$$

En la expresión X debe convertirse a radianes.

Solución

a) Análisis

Entrada: no requiere lectura de datos, ya que éstos se van generando.
Inicializar la sumatoria de términos en cero, la variable que haga las veces de potencia en uno, la que se incrementará en dos unidades por cada término, y la variable que manejará el signo en uno, la cual se multiplicará por (-1) en cada término.
Se debe convertir los grados a radianes.

Proceso: Se utilizará el anidamiento de 3 estructuras repetitivas, la más externa controlará los grados (5, 10, 15... 360), la estructura intermedia calculará cada término de la serie y la estructura interna calculará los factoriales.

Salida : imprimir X y su correspondiente seno.

b) Variables a utilizar

IX = Variable índice que representa los grados.
X = Grados expresados en radianes.
SENX = Sumatoria de términos de la serie.
SIG = Signo del término.
L = Potencia.
FAC = Factorial.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA IX DE 5 HASTA 360. 5 HACER
                X ← IX * 3.14159 / 180
                SENX ← 0
                SIG ← 1
                L ← 1
                MIENTRAS L <= 41 HACER
                  FAC ← 1
                  PARA J DE 1 HASTA L HACER
                    FAC ← FAC * J
                  FIN_PARA
                  SENX ← SENX + SIG * X↑L / FAC
                  L ← L + 2
                  SIG ← SIG *(-1)
                FIN_MIENTRAS
                IMPRIMIR IX,SENX
              FIN_PARA
Paso 3      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO

```
Paso 1      INICIO
Paso 2        PARA IX DE 5 HASTA 360, 5 HACER
                  X ← IX * 3.14159 / 180
                  SIG ← 1
                  SENX ← 0
                  PARA L DE 1 HASTA 41, 2 HACER
                     FAC ← 1
                     PARA J DE 1 HASTA L HACER
                        FAC ← FAC * J
                     FIN_PARA
                     SENX ← SENX + SIG * X↑L / FAC
                     SIG ← SIG *(-1)
                  FIN_PARA
                  IMPRIMIR IX,SENX
               FIN_PARA
Paso 3      FIN
```

EJERCICIO

Calcular el valor de Y según la siguiente serie:

$$Y = \frac{X^0}{0!} + \frac{X^2}{3!} - \frac{X^5}{7!} - \frac{X^9}{12!} + \frac{X^{14}}{18!} + \frac{X^{20}}{25!} \cdots \textit{HASTA 20 TERMINOS}$$

Para cada valor de X que oscile entre 1 y 10, imprimir el valor de X y su correspondiente valor de Y.

Solución

a) Análisis

Entrada: Inicializar el primer término de la serie en 1, el exponente en 2, el denominador en 3, la diferencia entre exponentes en 3, la diferencia entre denominadores en 4.

Proceso: Se genera la serie a partir del segundo término puesto que el primero es igual a uno. Se debe observar el comportamiento de los exponentes de la serie así como de los denominadores, porque la diferencia entre ellos es cada vez más creciente. Para definir la nueva potencia, se debe proceder a hallar la diferencia entre potencias de los dos términos anteriores e incrementarla en una unidad, para luego aumentarle ese resultado a la última potencia encontrada. El mismo procedimiento se realiza para calcular el próximo denominador. El signo del término se obtiene elevando la variable X al exponente respectivo.

Salida : Imprimir X y su correspondiente valor Y.

b) Variables a utilizar

EXP = Exponente
DEN = Denominador
DEXP = Diferencia entre exponentes
DDEN = Diferencia entre denominadores
FACDEN = Factorial del denominador.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA X DE 1 HASTA 10 HACER
                  Y ← 1
                  EXP ← 2
                  DEN ← 3
                  DEXP ← 3
                  DDEN ← 4
                  PARA N DE 2 HASTA 20 HACER
                     FACDEN ← 1
                     PARA I DE 1 HASTA DEN HACER
                        FACDEN ← FACDEN * I
                     FIN_PARA
                     Y ← Y + ((-1) * X )↑ EXP / FACDEN
                     EXP ← EXP + DEXP
                     DEN ← DEN + DDEN
                     DEXP ← DEXP + 1
                     DDEN ← DDEN + 1
                  FIN_PARA
                  IMPRIMIR X, Y
               FIN_PARA
Paso 3      FIN
```

EJERCICIO

Leer un conjunto N de valores X y encontrar su correspondiente valor Y, según las siguientes condiciones:

$$Y = X^2 - 7 \quad si \ldots 3 < X < 7$$

$$Y = \frac{(X-1)!}{(2X)!} \quad si \ldots 7 \leq X < 15$$

$$Y = \sqrt{X} \quad si\ X\ es\ diferente\ a\ las\ condiciones\ anteriores$$

Solución

a) Análisis:

Entrada: Leer el límite del conjunto de datos N.
Leer el valor de X, uno por registro.
Como el ejercicio debe calcular el factorial en 2 ocasiones, debe inicializar las variables multiplicadoras en 1.

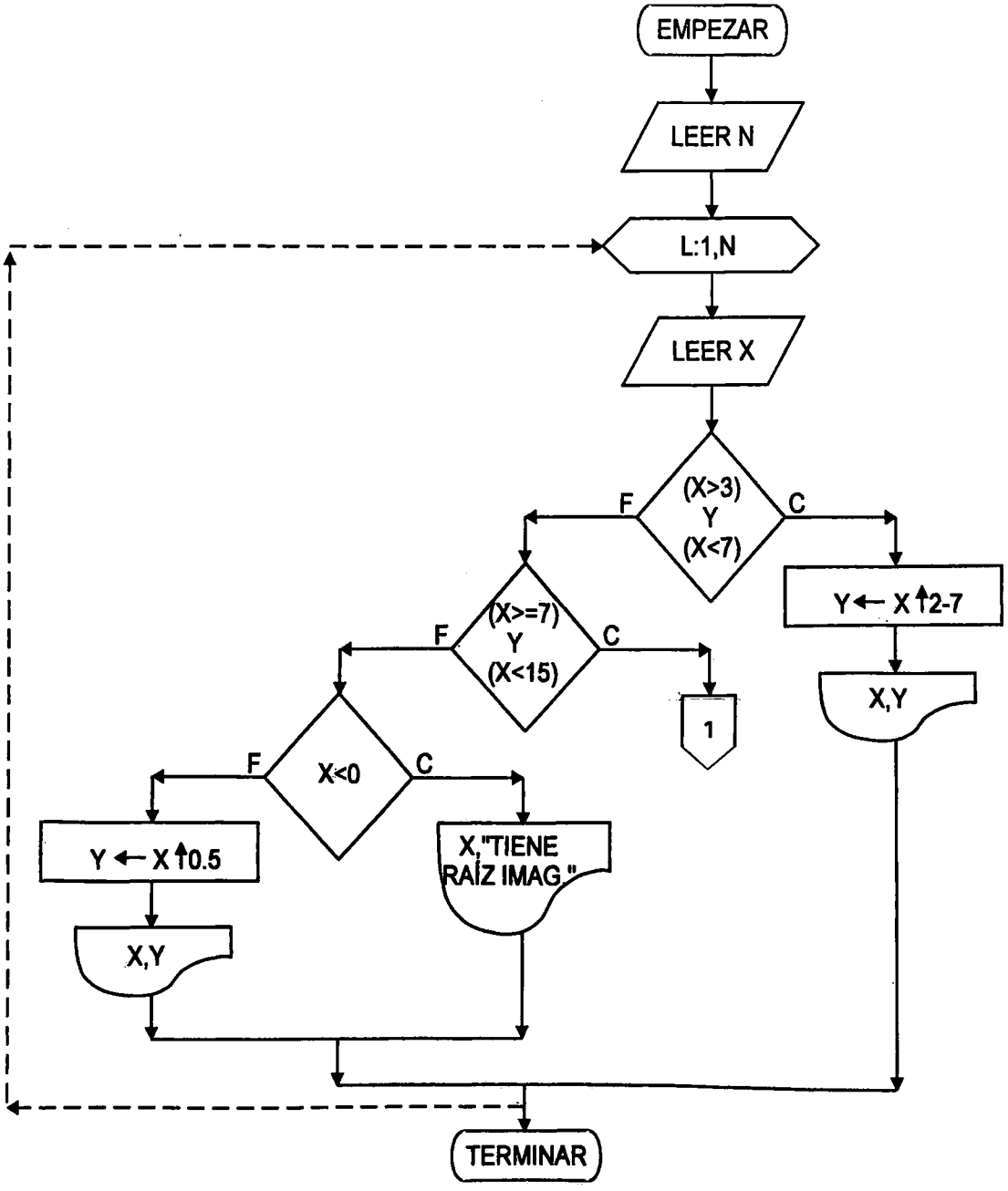
Proceso: Se utiliza una estructura **PARA** externa cuyo límite máximo es N. Por cada ciclo debe leer un valor X y calcular su correspondiente valor Y de acuerdo a las especificaciones del enunciado.

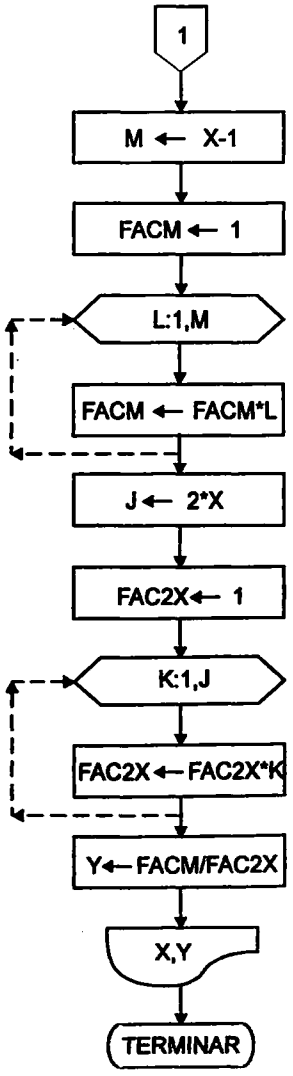
Salida : Imprimir X, Y

b) Variables a utilizar

N = Límite del conjunto de valores
X = Valor a ser analizado, debe ser entero
FACM = Factorial del numerador
FAC2X = Factorial del denominador
L, K = Valores índices de las estructuras PARA

c) Algoritmo en diagrama de flujo


EMPEZAR
LEER N
L:1,N
LEER X
(X>3) Y (X<7)
F
C
Y ← X ↑2-7
X,Y
(X>=7) Y (X<15)
F
C
1
X<0
F
C
Y ← X ↑0.5
X,Y
X,"TIENE RAÍZ IMAG."
TERMINAR



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        PARA I DE 1 HASTA N HACER
                  LEER X
                  SI (X>3) Y (X<7)
                     ENTONCES Y ← X ↑ 2 - 7
                        IMPRIMIR X,Y
                     SINO SI (X>=7) Y (X<15)
                              ENTONCES M ← X - 1
```

```
                                    FACM ← 1
                                    PARA L DE 1 HASTA M HACER
                                       FACM ← FACM * L
                                    FIN_PARA
                                    J ← 2 * X
                                    FAC2X ← 1
                                    PARA K DE 1 HASTA J HACER
                                       FAC2X ← FAC2X * K
                                    FIN_PARA
                                    Y ← FACM / FAC2X
                                    IMPRIMIR X,Y
                                SINO SI X < 0
                                        ENTONCES
                                           IMPRIMIR X "tiene raíz imaginaria"
                                        SINO  Y ← X↑0.5
                                              IMPRIMIR X,Y
                                     FINSI
                             FINSI
                      FINSI
                   FIN_PARA
Paso 4        FIN
```

2.4 Existen tres formas para poder determinar la culminación de un ejercicio o proceso:

a) Cuando se da como límite una **CONSTANTE.**

Ejemplo "Se tiene una nómina de 100 empleados..."
En este caso el límite es proporcionado por el mismo enunciado del ejercicio.

Ejemplo "Leer 1000 registros con la siguiente información ..."
El límite en este caso es 1000.

b) Cuando se da como límite una **VARIABLE.**

Ejemplo "Se tienen M estudiantes en una universidad ..."
El límite es M, por lo tanto esta variable debe ser leida previamente, antes de proceder a utilizar cualquier estructura de repetición.

c) Cuando no se determina el límite.

Ejemplo "Se tiene un gran volumen de datos con la siguiente información ..."

En este caso es imposible determinar el número de registros que se deben procesar y para poder definírselo al computador, se utiliza otro registro no procesable que se coloca al final del volumen de la información (ARCHIVO) que se desea procesar. A este registro se le da el nombre de **Registro DUMMY o Registro CENTINELA.**

Generalmente se utiliza como marca de fin de archivo el /*, en otras ocasiones se emplea una comparación contra nueves (99999.) ó contra ceros (00000.), sin que se llegue a pensar que se están procesando 99999 registros o 00000 registros, porque se debe recordar, que ésta es una marca de fin de lote o fin de archivo y no es procesable. El criterio utilizado lo escoge el programador de acuerdo a las circunstancias presentadas en cada situación.

Por ejemplo, la mejor señal de que un cuaderno de notas se ha acabado, es cuando se han agotado todas sus hojas y se llega a la cubierta o pasta, siendo ésta un indicativo de finalización del cuaderno de notas.

EJERCICIO

Leer una cantidad no determinada de nombres, imprimir los nombres.

Solución

a) Análisis

Entrada: leer los nombres, uno por cada ciclo.
Proceso: utilizar como condición de terminación de ciclos, la comparación del nombre contra la marca /*, y se procesarán tantos nombres mientras sean diferentes de la marca.
Salida : imprimir los nombres.

b) Variables a utilizar

NOM = Nombre.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER NOM
Paso 3        MIENTRAS NOM <> /* HACER
                IMPRIMIR NOM
                LEER NOM
```

```
            FIN_MIENTRAS
Paso 4      FIN
```

Cuando se trabaja con registros centinelas, la estructura repetitiva que más se ajusta es el **MIENTRAS (WHILE_DO)**, puesto que ella evalúa antes de procesar, permitiendo controlar que el registro **DUMMY** sólo sea tenido en cuenta para finalizar el ciclo repetitivo, sin llegar a ser procesado.

Observando la solución anterior, se emplea en ella dos instrucciones de lectura, una externa a la estructura de repetición puesto que es requerida como parte del condicionamiento de la misma y la otra interna a la estructura para poder realizar las lecturas de los diferentes nombres.

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        REPETIR
                LEER NOM
                SI NOM <> /*
                  ENTONCES IMPRIMIR NOM
                FINSI
              HASTA_QUE NOM = /*
Paso 4      FIN
```

De trabajar la estructura **REPETIR (REPEAT_UNTIL)** con registros centinelas, se hace necesario controlarlos con estructuras **IF**, debido a que el REPEAT_UNTIL mínimo realiza un ciclo y es allí donde podría procesarse la marca de final de datos, lo cual no es correcto.

No es posible utilizar la estructura **PARA** en el manejo de registros DUMMY o CENTINELA, ya que ella exige conocer previamente el límite de ciclos a realizar.

EJERCICIO

Encontrar la calificación promedio de un gran número, aunque desconocido, de exámenes. La escala de calificaciones es de 0 a 5.

Solución

a) Análisis

Entrada: Se necesita iniciar el contador de calificaciones y la sumatoria de calificaciones en cero, leer calificaciones.

Proceso: Para poder calcular el promedio, se hace necesario conocer la sumatoria de las calificaciones así como el número de calificaciones sumadas, en este caso desconocido. Para obviar este inconveniente se utiliza un registro DUMMY o CENTINELA que controla el total de calificaciones a procesar,consistente en leer una calificación y compararla contra 5, mientras sea mayor o igual a cero y menor o igual a 5 se procesa; cuando encuentra una nota superior a 5, o inferior a cero deja de leer más datos y calcula el promedio. Como en este ejercicio la nota oscila entre 0 y 5, no sería normal encontrar una calificación inferior a cero o superior a 5, por lo tanto, se emplea esta condición como CENTINELA.

Salida : Imprimir el promedio

b) Variables a utilizar

SUMA = Sumatoria de calificaciones
CONT = Contador de calificaciones
CALIF = Calificación
PROM = Promedio

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        CONT ← 0
Paso 4        LEER CALIF
Paso 5        MIENTRAS (CALIF >=0) Y (CALIF <= 5) HACER
                 SUMA ← SUMA + CALIF
                 CONT ← CONT + 1
                 LEER CALIF
              FIN_MIENTRAS
Paso 6        PROM ← SUMA / CONT
Paso 7        IMPRIMIR PROM
Paso 8      FIN
```

d) Prueba de escritorio

Se desean procesar las siguientes notas: 3.0, 4.0, y 2.0. Se utilizará como registro centinela una nota fuera del rango 0-5 que podría ser 6.0,o cualquiera otra fuera del rango permitido.

CALIF	SUMA	CONT	PROM
3.0	0	0	
4.0	0+3.0 = 3.0	0+1 = 1	9.0/3 = 3
2.0	3.0+4.0 = 7.0	1+1 = 2	
6.0	7.0+2.0 = 9.0	2+1 = 3	

La última nota leída no se procesa, puesto que ella sólo sirve de controladora de final de datos.

EJERCICIO

El radio de una circunferencia se calcula según la siguiente fórmula:

$$R = \sqrt{(X - CX)^2 + (Y - CY)^2}$$

Siendo el punto (X,Y) el centro de una circunferencia y (CX,CY) un punto sobre la circunferencia.
Calcular además, el área y la longitud de la circunferencia, así como la suma de las áreas y la suma de las longitudes de todos los registros procesados.

Fórmulas a emplear: $AREA = \pi R^2$
$LONGITUD = 2\pi R$

Solución

a) Análisis:

Entrada: Inicializar sumatoria de las áreas y sumatoria de las longitudes en cero.
Reemplazar la constante PI (π) por 3.14159
Leer los puntos X, Y, CX, CY.

Proceso: Se calcula el radio y se leen los puntos X, Y, CX, CY tantas veces mientras el Radio sea mayor que cero, sirviendo en este caso de registro centinela, pues determina la cantidad de datos a procesar.

Salida : Imprimir los puntos X, Y, CX, CY, el área y longitud de la circunferencia, la sumatoria de las áreas y la sumatoria de las longitudes.

b) Variables a utilizar

X, Y = Centro de la circunferencia
CX, CY = Punto sobre la circunferencia
RADIO = Radio
AR = Área de la circunferencia

SUMAR = Sumatoria de las áreas
LON = Longitud de la circunferencia
SUMLON = Sumatoria de longitudes
PI = Número π = 3.14159

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMAR ← 0
Paso 3        SUMLON ← 0
Paso 4        PI ← 3.14159
Paso 5        LEER X, Y, CX, CY
Paso 6        RADIO ← ((X-CX)↑2 + (Y-CY)↑2)↑0.5
Paso 7        MIENTRAS RADIO > 0 HACER
                AR ← PI * RADIO ↑ 2
                LON ← 2 * PI * RADIO
                SUMAR ← SUMAR + AR
                SUMLON ← SUMLON + LON
                IMPRIMIR X, Y, CX, CY, AR, LON
                LEER X, Y, CX, CY
                RADIO ← ((X-CX)↑2 + (Y-CY)↑2)↑0.5
              FIN_MIENTRAS
Paso 8        IMPRIMIR SUMAR, SUMLON
Paso 9      FIN
```

EJERCICIO

Encontrar el promedio de estaturas de un grupo de estudiantes universitarios.

Solución

a) Análisis

Entrada: Leer una estatura por registro.
Inicializar sumatoria de estaturas y contador de estaturas en cero.
Proceso: Emplear la siguiente fórmula:

$$\text{Promedio} = \frac{\textit{Sumatoria de estaturas}}{\textit{Numero de estaturas sumadas}}$$

Se toma como condición de terminación, el evento de encontrar una estatura

menor o igual a cero. Mientras la estatura sea mayor que cero se debe acumular e incrementar el contador de estaturas sumadas en 1.

Salida : Imprimir promedio de estaturas.

b) Variables a utilizar

CE = Contador de estaturas
SE = Sumatoria de estaturas
EST = Estatura
PROM = Promedio de estaturas

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CE ← 0
Paso 3        SE ← 0
Paso 4        LEER EST
Paso 5        MIENTRAS EST > 0 HACER
                  CE ← CE + 1
                  SE ← SE + EST
                  LEER EST
              FIN_MIENTRAS
Paso 6        PROM ← SE / CE
Paso 7        IMPRIMIR PROM
Paso 8      FIN
```

Observación

En algunos ejercicios es necesario recurrir al cambio de variables (sustitución de variables), con el propósito de NO proliferar en la generación de variables o para rescatar el valor original de una variable que es objeto de múltiples modificaciones a través del almacenamiento en una segunda variable, convirtiéndola en variable de trabajo.

PROBLEMAS RESUELTOS

EJERCICIO

La siguiente serie 0, 1, 1, 2, 3, 5, 8, 13 .... K , se conoce como la sucesión de FIBONACCI; elaborar un algoritmo para calcular K términos de la serie.

Solución

a) Análisis

Entrada: Sólo requiere la lectura del límite de la sucesión.
Se inicializa el primer término en cero y el segundo en 1.

Proceso: Analizando la serie se puede observar que el tercer término lo genera la suma de los dos términos anteriores, así mismo, el cuarto término es generado por la suma de los dos términos que le anteceden y así sucesivamente.
Para no proliferar en la creación de variables por cada término calculado, se hace un intercambio de variables, es decir, inicialmente se suman las variables Ay B y el contenido se almacena en la variable C, se imprime C y luego el contenido de B se traslada a la variable A, el contenido de C se traslada a la variable B y se efectúa nuevamente el cálculo de A+B, tantas veces como términos se deseen generar y trabajando siempre con las mismas 3 variables A, B, C.

Salida : Imprimir la serie

b) Variables a utilizar

A, B = Valores iniciales de la serie
C = Término generado
K = Límite de términos de la serie.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER K
Paso 3        A ← 0
Paso 4        B ← 1
Paso 5        IMPRIMIR A, B
Paso 6        PARA M DE 3 HASTA K HACER
                C ← A + B
                IMPRIMIR C
                A ← B
                B ← C
              FIN_PARA
Paso 7      FIN
```

d) Prueba de escritorio

A	B	C	K	M
0	1	1	7	3
1	1	2		4
1	2	3		5
2	3	5		6
3	5	8		7
5	8			8

En este ejercicio se inicializó la estructura de repetición en 3, porque ya se había generado los dos primeros términos de la serie.

SEGUNDA VERSIÓN DEL ALGORITMO

Se inicializa el contador de términos en 1 y por cada ciclo se imprime únicamente la variable A.

```
Paso 1      INICIO
Paso 2        LEER K
Paso 3        A ← 0
Paso 4        B ← 1
Paso 5        PARA M DE 1 HASTA K HACER
                  C ← A + B
                  IMPRIMIR A
                  A ← B
                  B ← C
              FIN_PARA
Paso 6      FIN
```

EJERCICIO

Leer en un registro los valores A y B, y ordenarlos ascendentemente.

Solución

a) Análisis

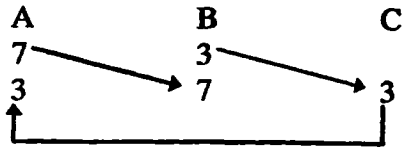
Entrada: Leer A y B
Proceso: Se compara A contra B, si A es mayor que B se hace un intercambio de variables mediante la utilización de una variable temporal.
Salida : Imprimir los valores ordenados.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER A, B
Paso 3        SI  A > B
                  ENTONCES C ← B
                           B ← A
                           A ← C
              FINSI
Paso 4        IMPRIMIR A, B "ORDENADOS ASCENDENTEMENTE"
Paso 5      FIN
```

d) Prueba de escritorio

Si A = 7 y B = 3 entonces:



EJERCICIO

Leer en un registro los valores A, B, C y ordenarlos ascendentemente.

Solución

a) Análisis:

Entrada: Leer A, B, C
Proceso: Se utiliza el mismo procedimiento del ejercicio anterior.
Salida : Imprimir los valores ordenados.

b) Variables a utilizar

A, B, C = Variables a ser ordenadas
D, E = Variables temporales

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER A, B, C
Paso 3        REPETIR
                 SI A > B
                    ENTONCES D ← B
                              B ← A
                              A ← D
                 FINSI
                 SI B > C
                    ENTONCES E ← C
                              C ← B
                              B ← E
                 FINSI
              HASTA_QUE (A <= B) Y (B <= C)
Paso 4        IMPRIMIR A, B, C "ORDENADOS ASCENDENTEMENTE"
Paso 5      FIN
```

EJERCICIO PROPUESTO

Probar el algoritmo anterior mediante una prueba de escritorio.

EJERCICIO

Calcular : $W = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{7} \cdots \frac{1}{61}$

Solución

a) Análisis

Entrada: No requiere lectura de datos porque cada término se autogenera.
Inicializar la sumatoria de términos en cero y el controlador de términos I en 1.
Proceso: Aplicar la siguiente fórmula:

$$W = \frac{(-1)^{(I-1)}}{(2(I-1)+1)}$$

donde $(-1)^{(I-1)}$ genera el signo del término y (2 (I-1) + 1) genera el denominador.

Salida : Imprimir el acumulado W

b) Variables a utilizar
W = Sumatoria de términos
I = Límite de términos

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2
Paso 3        I ← 1
Paso 4        REPETIR
                 W ←    (-1)↑(I-1)/(2*(I-1)+1)
                 I ← I + 1
              HASTA_QUE I > 31
Paso 5        IMPRIMIR W
Paso 6      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        W ← 0
Paso 3        PARA I DE 1 HASTA 31 HACER
                 W ← W+(-1)↑(I-1)/(2*I-1)
              FIN_PARA
Paso 4        IMPRIMIR W
Paso 5      FIN
```

TERCERA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        N ← -1 {Controla el signo del término}
Paso 3        W ← 0
Paso 4        PARA I DE 1 HASTA 61,2 HACER
                 {I sufre incrementos de 2 por cada ciclo}
                 N ← N * (-1)
                 W ← W + N / I
              FIN_PARA
```

Paso 5 **IMPRIMIR** W
Paso 6 **FIN**

EJERCICIO PROPUESTO

Realizar las correspondientes pruebas de escritorio de los ejercicios anteriores.

EJERCICIO

Calcular : $T = \frac{1!}{2} + \frac{3!}{4} + \frac{5!}{6} + \frac{7!}{8} + \cdots + \frac{47!}{48}$

Solución

a) Análisis

Entrada: Inicializar el acumulador de términos en cero y la variable manejadora del factorial en 1.
Proceso: Se emplea un anidamiento de 2 estructuras FOR,siendo la interna para calcular el correspondiente factorial y la externa para calcular totalmente cada término y acumularlo.
Salida : Imprimir el acumulado de términos.

b) Variables a utilizar

SUMA = Acumulador de términos
FAC = Multiplicador del factorial
T = Término de la serie
L = Numerador
I,J = Variables índice de la estructura **PARA**

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        PARA I DE 1 HASTA 24 HACER
                  L ← 2 * I - 1
                  FAC ← 1
                  PARA J DE 1 HASTA L HACER
                      FAC ← FAC * J
                  FIN_PARA
```

```
                T ← FAC / (2 * I)
                SUMA ← SUMA + T
            FIN_PARA
Paso 4      IMPRIMIR SUMA
Paso 5    FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

Solución

a) Análisis

Entrada: No requiere lectura de datos, puesto que los términos se autogeneran.
Se debe inicializar: una variable para manejar el factorial en 1, una variable acumuladora de términos en 0 y una variable para manejar el denominador en 2.

Proceso: Se genera la serie a partir del segundo término.
Emplear las siguientes fórmulas:
FAC = FAC * (I-1) * I, siendo I la variable índice de la estructura **PARA.**
DOBLE = DOBLE + 2
SUMA = SUMA + FAC / DOBLE
T = T + SUMA

Salida : Imprimir el acumulado T

b) Variables a utilizar

FAC = Factorial
DOBLE = Denominador
SUMA = Acumulador de términos
T = Primer término

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        DOBLE ← 2
Paso 3        T ← 1 / DOBLE
Paso 4        FAC ← 1
Paso 5        SUMA ← 0
Paso 6        PARA I DE 3 HASTA 47,2 HACER
                FAC ← FAC * (I - 1) * I
                DOBLE ← DOBLE + 2
                SUMA ← SUMA + FAC / DOBLE
              FIN_PARA
```

```
Paso 7          T ← T + SUMA
Paso 8          IMPRIMIR T
Paso 9      FIN
```

TERCERA VERSIÓN DEL ALGORITMO:.

```
Paso 1      INICIO
Paso 2          SUMA ← 0
Paso 3          DEN ← 2
Paso 4          PARA I DE 1 HASTA 47,2 HACER
                    FAC ← 1
                    PARA J DE 1  HASTA I HACER
                        FAC ← FAC*J
                    FIN_PARA
                    SUMA ← SUMA + FAC / DEN
                    DEN ← DEN + 2
                FIN_PARA
Paso 5          IMPRIMIR SUMA
Paso 6      FIN
```

CUARTA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2          SUMA ← 0
Paso 3          PARA I DE 1 HASTA 47,2 HACER
                    FAC ← 1
                    PARA J DE 1  HASTA I HACER
                        FAC ← FAC*J
                    FIN_PARA
                    SUMA ← SUMA + FAC / (I + 1)
                FIN_PARA
Paso 4          IMPRIMIR SUMA
Paso 5      FIN
```

EJERCICIO

Calcular: $$Y = 1 + \frac{X^2}{2!} + \frac{X^4}{4!} + \frac{X^6}{6!} + \cdots + \frac{X^{30}}{30!}$$

Solución

a) Análisis

Entrada: leer X.
Almacenar el primer término en la variable Y, la que hará las veces de sumatoria, luego se genera la serie a partir del segundo término.
Inicializar el factorial en 1.
Proceso: se aplicará un anidamiento de dos estructuras PARA, el más externo controlará los términos de la serie y el ciclo más interno calculará el factorial.
Salida : imprimir X,Y.

b) Variables a utilizar

X = Valor a ser leido.
Y = Sumatoria de términos.
FAC = Factorial.
I,J = Variable índice de la estructura PARA.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        Y ← 1
Paso 3        LEER X
Paso 4        PARA J DE 2 HASTA 30,2 HACER
                  FAC ← 1
                  PARA I DE 1 HASTA J HACER
                      FAC ← FAC*I
                  FIN_PARA
                  Y ← Y + X↑J / FAC
              FIN_PARA
Paso 5        IMPRIMIR X,Y
Paso 6      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        Y ← 1
Paso 3        N ← 2
Paso 4        LEER X
Paso 5        MIENTRAS N <= 30 HACER
```

```
                    FAC ← 1
                    PARA J DE 1 HASTA N HACER
                        FAC ← FAC*J
                    FIN_PARA
                    Y ← Y + X↑N / FAC
                    N ← N + 2
                FIN_MIENTRAS
Paso 6          IMPRIMIR X.Y
Paso 7      FIN
```

TERCERA VERSIÓN DEL ALGORITMO

```
Paso 1      INICIO
Paso 2          Y ← 1
Paso 3          N ← 2
Paso 4          FAC ← 1
Paso 5          LEER X
Paso 6          MIENTRAS N <= 30 HACER
                    FAC ← FAC*(N - 1)*N
                    Y ← Y + X↑N / FAC
                    N ← N + 2
                FIN_MIENTRAS
Paso 7          IMPRIMIR X,Y
Paso 8      FIN
```

EJERCICIO

La compañía Colombiana S.A. vende 6 productos diferentes y para la venta de éstos tiene N vendedores, en donde cada vendedor está encargado de la venta de la misma clase de artículo, pero un artículo puede ser vendido por varios vendedores.

Por cada venta se dispone de la siguiente información:
Código del vendedor, código del artículo, cantidad de unidades vendidas, precio de etiqueta del artículo y precio de venta real.

Calcular e imprimir lo siguiente:
Código del vendedor, código del artículo, precio de etiqueta, precio de venta real y comisión.

La comisión se calcula de acuerdo a la siguiente tabla:

CÓDIGO DEL ARTICULO	COMISIÓN
1	2% del precio total de la venta real
2	1% de la diferencia, del precio total de lo etiquetado y el precio total de la venta real.
3	5% del precio total de la venta real, más el 10% de la diferencia del precio total de lo etiquetado y el precio total de la venta real.
4	$100,00 por unidad vendida.
5	$50 por unidad vendida, más 7% del precio total de la venta real.
6	$30 por unidad vendida,más 1% del precio de etiqueta del artículo.

Solución

a) Análisis

Entrada: Leer el límite de vendedores.
Leer el código del vendedor, código del artículo, cantidad de unidades vendidas, precio de etiqueta del artículo y precio de venta real para cada venta.

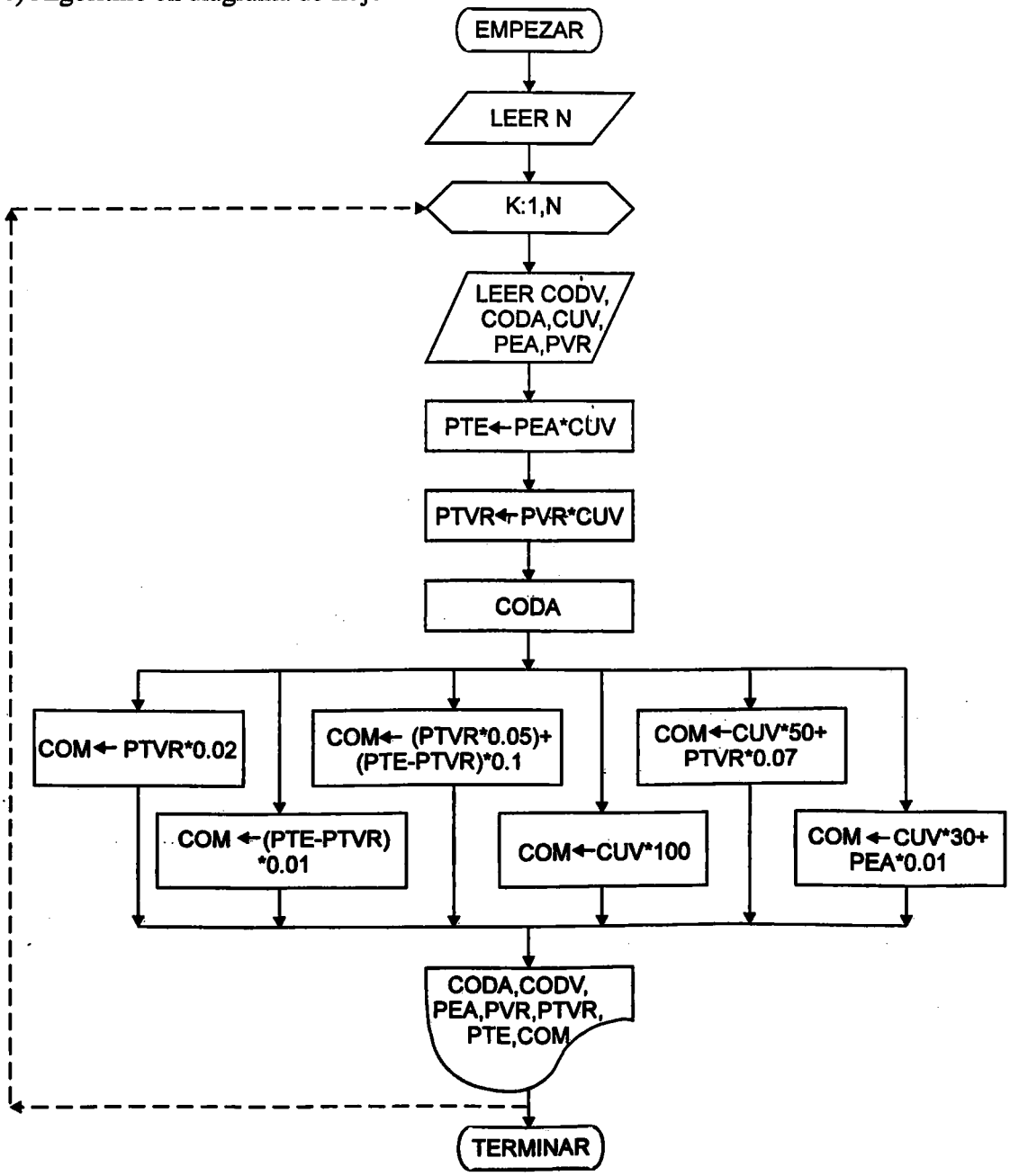
Proceso: Calcular:
El precio total de la venta real = precio de venta real x cantidad de unidades vendidas.
El precio total de lo etiquetado = precio de etiqueta por x cantidad de unidades vendidas.
Se utiliza una estructura **EN CASO DE ... HACER** y dependiendo del código del artículo, se calcula la correspondiente comisión.

Salida : Código del vendedor, código del artículo, precio de etiqueta, precio de venta real, precio total de lo etiquetado, precio total de venta real y comisión.

b) Variables a utilizar:

N = Límite de datos a ser procesados
CODV = Código del vendedor
CODA = Código del artículo
CUV = Cantidad de unidades vendidas
PEA = Precio de etiqueta del artículo
PVR = Precio de venta real
PTE = Precio total de lo etiquetado
PTVR = Precio total de venta real
COM = Comisión
K = Índice de la estructura PARA

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

Paso 1

```
INICIO
   LEER N
   PARA K DE 1 HASTA N HACER
      LEER CODV, CODA, CUV, PEA, PVR
```

```
                    PTE ← PEA * CUV
                    PTVR ← PVR * CUV
                    EN CASO DE CODA HACER
                       1: COM ← PTVR * 0.02
                       2: COM ← (PTE - PTVR) * 0.01
                       3: COM ← (PTVR*0.05)+(PTE-PTVR)*0.1
                       4: COM ← CUV * 100
                       5: COM ← CUV * 50 + PTVR * 0.07
                       6: COM ← CUV * 30 + PEA * 0.01
                    FIN_CASO
                    IMPRIMIR CODA, CODV, PEA, PVR, PTE, PTVR,COM
                 FIN_PARA
Paso 4        FIN
```

EJERCICIO

Una compañía de seguros desea conocer el valor del cheque que debe elaborar cada mes por concepto de daños de vehículos. El taller de reparación envía a la compañía cada una de las N facturas, con la siguiente información: placa del vehículo, tipo del daño y costo. El valor que reconoce la empresa de seguros puede ser de tres tipos:

Tipo 1: reconoce hasta 1'000.000
Tipo 2: reconoce hasta 500.000
Tipo 3: reconoce hasta 300.000

Si el valor del daño reportado por el taller de reparaciones es superior al valor aceptado por la compañía de seguros, ésta solamente reconocerá lo previamente establecido.
Calcular el total del cheque que la empresa de seguros debe pagar al final del mes, así como el total del cheque en caso de que no se tenga en cuenta las anteriores restricciones.

Solución

a) Análisis

Entrada: Leer el límite de facturas a procesar.
Leer por cada registro: placa del vehículo, tipo de daño y costo.
Inicializar en cero las variables acumuladoras del costo del cheque sin restricciones y del costo del cheque con restricciones.

Proceso: Dependiendo del tipo de daño y del costo se determina el valor reconocido por la compañía de seguros, para luego efectuar los correspondientes acumulados.

Salida : Imprimir el total del cheque con restricciones y el total del cheque sin restricciones.

b) Variables a utilizar

N = Límite de facturas a procesar
PLACA = Placa del vehículo
TD = Tipo de daño
CD = Costo del daño
VR = Valor reconocido
TCSR = Total cheque sin restricciones
TCCR = Total cheque con restricciones

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        TCSR ← 0
Paso 3        TCCR ← 0
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                LEER PLACA, TD, CD
                TCSR ← TCSR + CD
                SI TD = 1
                  ENTONCES SI CD >= 1'000.000
                                ENTONCES VR ← 1'000.000
                                SINO    VR ← CD
                             FINSI
                  SINO SI TD = 2
                          ENTONCES SI CD >= 500.000
                                        ENTONCES VR ← 500.000
                                        SINO    VR ← CD
                                     FINSI
                          SINO SI CD >= 300.000
                                    ENTONCES VR ← 300.000
                                    SINO    VR ← CD
                                  FINSI
                       FINSI
                FINSI
                TCCR ← TCCR + VR
              FIN_PARA
Paso 6        IMPRIMIR TCCR, TCSR
Paso 7      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        TCSR ← 0
Paso 3        TCCR ← 0
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                 LEER PLACA, TD, CD
                 TCSR ← TCSR + CD
                 EN CASO DE TD HACER
                    1: SI CD >= 1'000.000
                          ENTONCES VR ← 1'000.000
                          SINO    VR ← CD
                       FINSI
                    2: SI CD >= 500.000
                          ENTONCES  VR ← 500.000
                          SINO     VR ← CD
                       FINSI
                    3: SI CD >= 300.000
                          ENTONCES VR ← 300.000
                          SINO    VR ← CD
                       FINSI
                 FIN_CASO
                 TCCR ← TCCR + VR
              FIN_PARA
Paso 6        IMPRIMIR TCCR, TCSR
Paso 7      FIN
```

TERCERA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        TCSR ← 0
Paso 3        TCCR ← 0
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                 LEER PLACA, TD, CD
                 TCSR ← TCSR + CD
                 SI TD = 1
                    ENTONCES SI CD >= 1'000.000
                                ENTONCES VR ← 1'000.000
                                SINO    VR ← CD
                             FINSI
                 FINSI
                 SI TD = 2
```

```
                    ENTONCES SI CD >= 500.000
                                ENTONCES VR ← 500.000
                                SINO    VR ← CD
                              FINSI
                FINSI
                SI TD = 3
                  ENTONCES SI CD >= 300.000
                                ENTONCES VR ← 300.000
                                SINO    VR ← CD
                              FINSI
                FINSI
                TCCR ← TCCR + VR
              FIN_PARA
              IMPRIMIR TCCR, TCSR
Paso 6      FIN
```

EJERCICIO

La compañía Caldas S.A. desea obtener el total de ventas realizadas durante 1995. La compañía ofrece cuatro formas de crédito diferentes al mismo tiempo que puede vender artículos de contado. La información leída por cada venta es la siguiente: tipo de pago, cantidad de artículos vendidos y precio unitario.

Si el tipo de pago = 1, la forma de pago es a 10 días
= 2, la forma de pago es a 20 días
= 3, la forma de pago es a 30 días
= 4, la forma de pago es a 60 días
= 5, la forma de pago es de contado
= 6, no debe procesarse por ser el final de datos.

Calcular:

1- Ventas totales realizadas por la compañía
2- Total de ventas realizadas por cada una de las cuatro formas de pago a crédito.
3- Total ventas de contado

Solución

a) Análisis

Entrada: Leer tipo de pago, cantidad de artículos vendidos y precio unitario.
Inicializar en cero cada una de las cuatro formas de pago a crédito, así como las ventas totales y las ventas de contado.

Proceso: Como no se conoce el total de datos a procesar se utiliza un registro centinela, en este caso, cuando el tipo de pago sea menor que 1 ó mayor que 5 concluye la lectura de datos.
Se calcula la venta del artículo como precio unitario por cantidad.
Total ventas = total de ventas + venta de cada artículo.
Se utiliza una estructura **EN CASO DE ... HACER** y dependiendo del tipo de venta se incrementa el correspondiente acumulador.

Salida : Ventas a 10 días, ventas a 20 días, ventas a 30 días, ventas a 60 días, ventas de contado, ventas totales.

b) Variables a utilizar

VEN10 = Ventas a 10 días
VEN20 = Ventas a 20 días
VEN30 = Ventas a 30 días
VEN60 = Ventas a 60 días
VENCON = Ventas de contado
VENTOT = Ventas totales
TIPO = Tipo de pago
CAN = Cantidad de artículos
PU = Precio unitario
VENTA = Venta realizada

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        VEN10 ← 0
Paso 3        VEN20 ← 0
Paso 4        VEN30 ← 0
Paso 5        VEN60 ← 0
Paso 6        VENCON ← 0
Paso 7        VENTOT ← 0
Paso 8        LEER TIPO, CAN, PU
Paso 9        MIENTRAS (TIPO >= 1) Y (TIPO <=5) HACER
                  VENTA ← PU * CAN
                  VENTOT ← VENTOT + VENTA
                  EN CASO DE TIPO HACER
                      1: VEN10 ← VEN10 + VENTA
                      2: VEN20 ← VEN20 + VENTA
                      3: VEN30 ← VEN30 + VENTA
                      4: VEN60 ← VEN60 + VENTA
                      5: VENCON ← VENCON + VENTA
```

```
                FIN_CASO
                LEER TIPO, CAN, PU
            FIN_MIENTRAS
Paso 10     IMPRIMIR VEN10, VEN20, VEN30, VEN60, VENTOT, VENCON
Paso 11   FIN
```

EJERCICIO

Se tiene un inventario de 50 artículos y por cada uno de ellos se dispone de la siguiente información: código del artículo, cantidad mínima, cantidad máxima, saldo y costo unitario. Se desea un reporte de aquellos artículos que tienen poca o ninguna existencia en el almacén calculando el número de unidades a ordenar y el costo de la orden para ese artículo. Al final del informe se desea saber cuál es el costo total de todos los pedidos.

Solución

a) Análisis

Entrada: Inicializar el contador de registros y el costo total de todos los pedidos en cero.
Leer código, cantidad mínima, cantidad máxima, saldo y costo unitario.
Proceso: Si el saldo es menor que la cantidad mínima, se determina el pedido a realizar.
El pedido = cantidad máxima - saldo
El costo del pedido = pedido x costo unitario
El costo total de los pedidos = sumatoria de los pedidos.
Salida : Imprimir código,cantidad mínima,cantidad máxima, saldo,costo unitario, pedido, costo del pedido y costo total de todos los pedidos.

b) Variables a utilizar:

CR = Contador de registros
COD = Código
CMIN = Cantidad mínima
CMAX = Cantidad máxima
SALDO = Saldo
CUNIT = Costo unitario
PED = Pedido a efectuar
COSPED = Costo del pedido
CTP = Costo total de los pedidos.

c) Algoritmo en seudocódigo

Paso 1 **INICIO**

```
Paso 2          CR ← 1
Paso 3          CTP ← 0
Paso 4          REPETIR
                    LEER COD, CMIN, CMAX, SALDO, CUNIT
                    SI SALDO >= CMIN
                       ENTONCES PED ← 0
                       SINO PED ← CMAX - SALDO
                    FINSI
                    COSPED ← PED * CUNIT
                    IMPRIMIR COD,CMIN,CMAX,SALDO,CUNIT,PED, COSPED
                    CTP ← CTP + COSPED
                    CR ← CR + 1
                HASTA_QUE CR > 50
Paso 5          IMPRIMIR CTP
Paso 6      FIN
```

Otra solución al mismo ejercicio empleando la estructura **PARA**, segunda versión del algoritmo

```
Paso 1      INICIO
Paso 2          CTP ← 0
Paso 3          PARA I DE 1 HASTA 50 HACER
                    LEER COD, CMIN, CMAX, SALDO, CUNIT
                    SI SALDO >= CMIN
                       ENTONCES PED ← 0
                       SINO PED ← CMAX - SALDO
                    FINSI
                    COSPED ← PED * CUNIT
                    IMPRIMIR COD,CMIN,CMAX,SALDO,CUNIT,PED,COSPED
                    CTP ← CTP + COSPED
                FIN_PARA
Paso 4          IMPRIMIR CTP
Paso 5      FIN
```

EJERCICIO

Se tiene un grupo de N estudiantes y por cada uno de ellos se dispone de la siguiente información: código del estudiante, nombre, primer parcial, segundo parcial, tercer parcial. La nota definitiva se calcula como el promedio de los tres parciales. Imprimir por cada estudiante, su código, nombre, cada una de las notas parciales y su calificación definitiva.

Calcular e imprimir además: el total de estudiantes reprobados, total de estudiantes cuya nota definitiva se encuentre entre 3 y 4, y el total de estudiantes cuya nota definitiva sea superior a 4. La escala de calificaciones es de 0 a 5.

Solución

a) Análisis

Entrada: Inicializar: el contador de estudiantes reprobados en cero, el contador de notas entre 3 y 4 en cero, el contador de notas mayores a 4 en cero.
Leer el límite de estudiantes N.

Proceso: Calcular la nota definitiva como (primer parcial + segundo parcial + tercer parcial) / 3, para luego compararla contra 3 y así incrementar el contador de reprobados si es menor que 3, igual proceso se hace con las notas que se encuentran entre 3 y 4 y las superiores a 4.

Salida : Imprimir código, nombre, primer parcial, segundo parcial, tercer parcial, calificación definitiva, contador de reprobados, contador de notas entre 3 y 4 y contador de notas mayores de 4.

b) Variables a utilizar:

CREP = Contador de reprobados
CN3Y4 = Contador de notas entre 3 y 4
CNMA4 = Contador de notas mayores de 4
N = Número total de estudiantes
COD = Código
NOM = Nombre
PP = Primer parcial
SP = Segundo parcial
TP = Tercer parcial
CDEF = Calificación definitiva

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CREP ← 0
Paso 3        CN3Y4 ← 0
Paso 4        CNMA4 ← 0
Paso 5        LEER N
Paso 6        PARA I DE 1 HASTA N HACER
                LEER COD, NOM, PP, SP, TP
                CDEF ← (PP+SP+TP)/3
                IMPRIMIR COD, NOM, PP, SP, TP, CDEF
```

```
                    SI CDEF < 3
                       ENTONCES CREP ← CREP + 1
                       SINO SI CDEF <= 4
                                ENTONCES CN3Y4 ← CN3Y4 + 1
                                SINO CNMA4 ← CNMA4 + 1
                            FINSI
                    FINSI
                FIN_PARA
Paso 7          IMPRIMIR CREP, CN3Y4, CNMA4
Paso 8      FIN
```

EJERCICIO

Se tiene una serie de registros cada uno con la siguiente terna de valores, X, Y, Z. El proceso termina cuando Z=000.

Si

$X \leq Y$ _____ $T = ZX + Y$
$X > Y$ _____ $T = X - Y$
$Y < Z$ _____ $T = X + Y + Z$
$Y \geq Z$ _____ $T = 3X + 2Y - Z$

Hallar e imprimir: $\sum T;\ P = \frac{\sum X}{\sum Y};\ Q = \sum \frac{X}{Z}$

Solución

a) Análisis

Entrada: Inicializar en cero la sumatoria de X, sumatoria de Y, sumatoria de T y sumatoria de X/Z
Leer los valores X, Y, Z.

Proceso: Se utiliza como control de parada, la comparación de Z contra 000, en caso de ser diferente, se hacen los respectivos cálculos que se originan de las comparaciones de X contra Y y de Y contra Z. Cabe anotar que por cada ciclo ejecutado, se calculan dos valores de T, los cuales deben ser almacenados.

Salida : Imprimir la sumatoria de T, P y Q.

b) Variables a utilizar

X, Y, Z= Valores a ser leídos
SX = Sumatoria de X

SY = Sumatoria de Y
ST = Sumatoria de T
SXZ = Sumatoria de X/Z

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SX ← 0
Paso 3        SY ← 0
Paso 4        ST ← 0
Paso 5        SXZ ← 0
Paso 6        LEER X, Y, Z
Paso 7        MIENTRAS Z <> 000 HACER
                 SI  X > Y
                    ENTONCES T ← X - Y
                    SINO T ← Z * X + Y
                 FINSI
                 SX ← SX + X
                 SY ← SY + Y
                 SXZ ← SXZ + X/Z
                 ST ← ST + T
                 SI Y >= Z
                    ENTONCES T ← 3 * X + 2 * Y - Z
                    SINO T← X + Y + Z
                 FINSI
                 ST ← ST + T
                 LEER X, Y, Z
Paso 8        FIN_MIENTRAS
Paso 9        P ← SX / SY
Paso 10       Q ← SXZ
Paso 11       IMPRIMIR ST, P, Q
Paso 12     FIN
```

EJERCICIO

Calcular e imprimir para X = 5, 10, 15, 20, ... , 360 grados, el seno de X mediante la expresión:

$$SenX = \frac{X^1}{1!} - \frac{X^3}{3!} + \frac{X^5}{5!} - \frac{X^9}{9!} + \frac{X^{15}}{15!} \cdots \frac{X^{33}}{33!}$$

En la expresión X debe darse en radianes.

Solución

a) Análisis

Entrada:No requiere lectura de datos, porque éstos se van autogenerando.
Se debe convertir los grados a radianes.
Se inicializa la sumatoria en el primer término (X^1 / 1!) y se genera la serie a partir del segundo término. La variable que haga las veces de potencia se inicializa en 3, y la variable que maneja el signo de cada término en -1.
Se debe inicializar una variable que partiendo del segundo término de la serie tome un valor de 2 y a partir de allí genere las demás potencias, las cuales tienen un comportamiento creciente.

Proceso:Se debe utilizar el anidamiento de 3 estructuras repetitivas, la más externa controlará los grados (5, 10, 15 ... 360), la estructura intermedia calculará el límite de términos de la serie y la estructura interna calculará los distintos factoriales.

Salida : Imprimir X y su correspondiente seno.

b) Variables a utilizar

IX = Variable índice que representa los grados.
X = Grados expresados en radianes.
SEN = Sumatoria de términos de la serie.
SIG = Signo del término.
POT = Potencia.
DIF = Diferencia entre potencias
FAC = Factorial.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA IX DE 5 HASTA 360, 5 HACER
                X ← IX * 3.141516 / 180
                SIG ← -1
                SEN ← X↑1/1
                POT ← 3
                DIF ← 2
                MIENTRAS (POT <= 33) HACER
                  FAC ← 1
                  PARA J DE 1 HASTA POT HACER
                    FAC ← FAC * J
```

```
                    FIN_PARA
                    SEN ← SEN+ SIG *( X↑POT / FAC)
                    POT ← POT + DIF
                    SIG ← SIG *(-1)
                    DIF ← DIF + 2
                FIN_MIENTRAS
                IMPRIMIR IX,SEN
            FIN_PARA
Paso 3      FIN
```

EJERCICIO

Elaborar un algoritmo para que imprima el siguiente triángulo.

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
```

Solución

a) Análisis

Entrada: Leer el límite del triángulo.
Inicializar filas en 1 y columnas en 1.
Proceso: Se utilizan dos estructuras repetitivas;la externa controla el límite del triángulo, la interna genera cada elemento del triángulo.
Salida : Imprimir el triángulo generado.

b) Variables a utilizar

X = Columna
Y = Fila
M = Límite del ejercicio
N = Valores a imprimir por filas
X,Y = Posición a ser escrito cada valor

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        IMPRIMIR "Teclee el límite del triángulo :"
Paso 3        LEER M
Paso 4        PARA J DE 1 HASTA M  HACER
                  PARA N DE 1 HASTA J HACER
                     IMPRIMIR N
                  FIN_PARA
              FIN_PARA
Paso 5      FIN
```

d) Codificación PASCAL

```
PROGRAM TRIANGULO;
{GENERACION DE UN TRIANGULO}
USES CRT;
VAR
    X,Y,N,M,J  :INTEGER;
BEGIN
   Y:= 1;
   WRITELN('Teclee el límite del triángulo :');
   READLN(M);
   FOR J:= 1 TO M DO
      BEGIN
         X:= 1;
         FOR N:= 1 TO J DO
            BEGIN
               GOTOXY(X,Y);
               WRITELN(N);
               X:= X + 2;
            END;
         WRITELN;
         Y:= Y + 1
      END
END.
```

EJERCICIO

Calcular e imprimir el siguiente triángulo

```
 1
 2   3
 3   4   5
 4   5   6   7
 5   6   7   8   9
 6   7   8   9  10  11
 7   8   9  10  11  12  13
 8   9  10  11  12  13  14  15
 9  10  11  12  13  14  15  16  17
10  11  12  13  14  15  16  17  18  19
```

a) Análisis

Entrada: Leer el límite del triángulo.
Inicializar filas en 1 y columnas en 1.
Proceso: Se utilizan dos estructuras repetitivas; la externa controla el límite del triángulo, la interna genera cada elemento del triángulo.
Salida : Imprimir el triángulo generado.

b) Variables a utilizar

X = Columna
Y = Fila
M = Límite del ejercicio
N = Valores a imprimir por filas
X,Y = Posición a ser escrito cada valor

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        IMPRIMIR "Teclee el límite del triángulo :"
Paso 3        LEER M
Paso 4        Y ← 1
Paso 5        IMPRIMIR Y
Paso 6        PARA J DE 1 HASTA M  HACER
                  Y ← Y + 1
                  PARA N DE Y HASTA (J+Y) HACER
                     IMPRIMIR N
                  FIN_PARA
              FIN_PARA
Paso 5      FIN
```

d) Codificación PASCAL

```
PROGRAM TRIANGULO;
{GENERACION DE UN TRIANGULO}
USES CRT;
VAR
    X,Y,N,M,J  :INTEGER;
BEGIN
    WRITELN('Teclee el límite del triángulo :');
    READ(M);
    Y:= 1;
    WRITELN(Y);
    FOR J:= 1 TO M DO
       BEGIN
          X:= 1;
          Y:= Y + 1
          FOR N:= Y TO (J+Y) DO
             BEGIN
                GOTOXY(X,Y);
                WRITE(N);
                WRITE('  ');
                X:= X + 3
             END
       END
END.
```

PROBLEMAS PROPUESTOS

1. **Dado un conjunto de N valores, determinar la media aritmética según la siguiente fórmula:**

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$$

2. Se dispone de un conjunto de registros, cada uno con una terna de valores X, Y, Z, el proceso termina cuando Z = -99
   Si
   X < Y _____ $A = 3X^2 + 2Y - Z$
   X = Y _____ $A = 2X^3 - 2Y + Z$
   X > Y _____ $A = X + Y + Z$
   X > Z _____ $A = 5X + 2Y - 3Z$
   X < Z _____ $A = X - 5Y + Z^2$
   X = Z _____ $A = 7X - 7Y - 7Z$

Calcular $\sum A;\ \sum X;\ \frac{\sum Y}{\sum Z}$

3. El gerente de un almacén desea saber cuántas ventas se realizarán entre 0 y $5.000, entre $5001 y $10.000, entre $10001 y más.
   Además, desea el total de ventas realizadas así como el total recaudado por ventas en un mes determinado.

4. La función exponencial e^x se puede expresar como:

$$e^x = 1 + X + \frac{X^2}{2!} + \frac{X^3}{3!} + \frac{X^4}{4!} + \cdots + \frac{X^{10}}{10!}$$

   Calcular e^x para valores de X que se encuentren entre $0 \leq X \leq 1$ y que sufre incrementos de 0.05.

5. Leer un conjunto de valores X dado en radianes y calcular el seno X, según la siguiente expresión:

$$SenX = \frac{X}{1!} - \frac{X^3}{3!} + \frac{X^5}{5!} - \frac{X^7}{7!} + \cdots + \frac{X^{15}}{15!}$$

6. Calcular $Y = 1 + \frac{X^2}{2!} + \frac{X^4}{4!} + \frac{X^6}{6!} + \frac{X^8}{8!} + \cdots + \frac{X^{30}}{30!}$

7. Calcular $Y = 1 - \frac{X^3}{2!} + \frac{X^6}{4!} - \frac{X^9}{6!} + \frac{X^{12}}{8!} - \cdots + \frac{X^{30}}{20!}$

8. Se tiene un conjunto de N registros y por cada uno de ellos se va a leer un valor entero L. Calcular e imprimir para cada valor L, su correspondiente valor Y.

   $Y = (L - 1)! + 2$ _____ si $1 \leq L \leq 5$
   $Y = \sqrt{E^{3L} + 2}$ _____ si $6 \leq L \leq 8$
   $Y = (3L + 1)(L!)$_____ si $10 \leq L \leq 15$
   $Y = 0$ para los demás casos.

9. Calcular e imprimir el valor de Y para cada valor de X, cuando X = 1, 2, 3, ... 10 , si

$$Y = \frac{X^0}{0!} + \frac{X^2}{3!} - \frac{X^5}{7!} - \frac{X^9}{12!} + \frac{X^{14}}{18!} + \frac{X^{20}}{25!} - \frac{X^{27}}{33!} - \frac{X^{35}}{42!} + \frac{X^{44}}{52!}$$

10.Calcular e imprimir el valor de Y para cada valor de X, cuando X = 1,2,3, ... 10, si

$$Y = 1 + \frac{X^2}{2!} - \frac{X^5}{5!} - \frac{X^9}{8!} + \frac{X^{14}}{11!} + \frac{X^{20}}{14!} - \frac{X^{27}}{17!} - \frac{X^{35}}{20!} + \frac{X^{44}}{23!}$$

11.Realizar la prueba de escritorio al siguiente algoritmo y determinar qué produce.

```
INICIO
   LEER M
   F ← 1
   PARA I DE 1 HASTA M HACER
      S ← 0
      PARA J DE 1 HASTA I HACER
         S ← S + F
      FIN_PARA
      F ← S
   FIN_PARA
   IMPRIMIR M,F
FIN
```

12.El siguiente programa halla los números entre 0 y 9999 cuya parte entera de su raíz cuadrada son iguales a la suma de las cifras que conforman dicho número.

Ejemplo: $\sqrt{52} = 7$ $7 = 5 + 2$

$\sqrt{147} = 12$ $12 = 1 + 4 + 7$

Pero tiene errores de sintaxis, encontrarlos y corregirlos.

```
C       PROGRAMA NUMERO1
        WRITE(*,1)
1       FORMAT(/ , ' ',20X, 'Universidad Nacional de Colombia' // , ' ',26X,
       *'Sede Manizales')
        I = 0
9       IF (I .GT. 9999) GOTO 7
           N = I
           K = N / 1000
           N = N - (K*1000)
           L = N / 100
           N = N - (L*100)
           M = N / 10
           N = N - (M*10)
           S = K + L + M + N
           J = SQRT(I + 0.0)
           IF (S .EQ. J) GOTO 8;
```

```
10          I = I + 1
            GOTO 9
 8          WRITE(*,2)I,J,K,L,M,N,S
 2          FORMAT(' ',16X,I4,17X,F2.0,17X,I1, '+',I1, '+',I1, '+',I1, '=',I4)
        GOTO 10
 7      STOP
        END
```

13.El siguiente programa halla los números entre 0 y 99999 que son iguales a la suma de los cubos de cada una de sus cifras.
Ejemplo: $153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$
Pero tiene errores de sintaxis, encontrarlos y corregirlos.

```
        PROGRAM CUBOS
        WRITE(*,50)
 50     FORMAT(1H1,25(/ ),24X, 'Universidad Nacional de Colombia')
        K = 0
        DO  10  I = 0, 99999
           IACUM = 0;
           IAUX = I
           DO  20  J = 1, 5
C             IC = IAUX - INT(IAUX / 10)*10
              IAUX = INT(IAUX / 10)
              IACUM = IACUM + IC**3
 20        CONTINUE
           IF (IACUM .EQ. I) THEN
              K = K + 1
              WRITE(*,1)K,IACUM
  1           FORMAT(// 5X, 'Elemento número ',I2, ':',I5)
           ENDIF
100     CONTINUE
        STOP
        END
```

14.Señale 5 errores de sintaxis en el siguiente programa y determine lo que calcula mediante una prueba de escritorio.

```
PROGRAM SUMASERIE;
USES CRT,PRINTER;
CONST
   TERMINOS : 100;
VAR
   I              :INTEGER;
   POTENCIA       :REAL;
```

```
BEGIN
    SUMA := 0;
    POTENCIA:= 1;
    FOR I:= 1 TO TERMINO
        BEGIN
            POTENCIA:= POTENCIA*2;
            SUMA:= SUMA + I / POTENCIA
        END;
        WRITELN('La suma es', SUMA:3:1)
END
```

15.Señale 5 errores de sintaxis en el siguiente programa y explique en qué forma se corrigen.

```
PROGRAM MEDIA;
{PROGRAMA PARA CALCULAR LA MEDIA ARITMETICA DE 5 NUMEROS}
USES CRT,PRINTER
VAR
   NUMERO,SUMA     :REAL;
   MEDIA           :INTEGER;
BEGIN
    SUMA := 0;
    FOR J = 1 TO 5 DO
        BEGIN
            WRITELN('(Cuál es el número',J, '?');
            READLN(NUMERO);
            SUMA:= SUMA + NUMERO
        END;
        MEDIA := SUMA / 5;
        WRITELN('La media es', MEDIA:5:2)
END.
```

16.Señale 5 errores de sintaxis en el siguiente programa y explique en qué forma se corrigen.

```
PROGRAM CERTIFICADO BANCARIO;
USES CRT,PRINTER;
(*PROGRAMA QUE CALCULA CERTIFICADO BANCARIO*)
VAR
    DEPOSITO,VALOR,INTERES      :REAL;
    CANTIDAD,PERIODO,AÑO        :INTEGER
BEGIN
    (*LECTURA DE DEPOSITO, PERIODO Y CANTIDAD)
    READ(DEPOSITO,PERIODO,CANTIDAD);
```

```
    WRITELN( 'Periodo :',PERIODO:3, 'Cantidad :,CANTIDAD=6:4, 'Depósito = $',
            DEPOSITO:10);
    {SE LE ASIGNA EL VALOR INICIAL DE DEPOSITO A VALOR *}
    VALOR = DEPOSITO;
    WRITELN('Valor inicial del certificado = $',VALOR:10:2)
    WRITELN;
    (*CALCULO DEL INTERES Y DEL VALOR DE CADA AÑO*);
    WRITELM('Año interés valor');
    FOR AÑO:= 1 TO PERIODO DO
       BEGIN
          INTERES:= VALOR*CANTIDAD;
          VALOR:= VALOR + INTERES;
          WRITELN(AÑO:3,INTERES:10:2,VALOR:10:2)
END.
```

17. Una compañía cobra a sus clientes el último día de cada mes. Si el cliente paga dentro de los primeros 10 días del mes siguiente obtiene un descuento de $1000 pesos, o bien del 2% sobre la deuda, dependiendo de cual sea el mayor valor. Si el cliente paga en los siguientes 10 días, no tiene derecho a ningún descuento, es decir, deberá pagar exactamente la suma adeudada, si paga dentro de los restantes días del mes, tendrá un recargo de $1000 ó del 2% sobre la deuda, dependiendo de cual sea el menor valor.

    Leer un lote de 100 registros, cada uno de los cuales contiene el número de identificación de cada cliente, nombre del cliente y cantidad adeudada.

    Se requiere calcular e imprimir títulos, número de identificación de cada cliente, nombre del cliente, cantidad que tendría que pagar en los primeros 10 días del mes, en los segundos 10 días y en los restantes 10 días. Realizar análisis, algoritmo y codificaciones FORTRAN y PASCAL.

18. Un Banco tiene el problema de calcular los intereses que a final de año habrá de acreditar a cada uno de los clientes. los datos son organizados en registros, cada uno de los cuales contiene los siguientes campos: código del cliente, capital, día del calendario comercial (año de 360 días) en que el capital fue ingresado al Banco y tasa de interés convenida.

    La fórmula a aplicar es: $Interes = K\frac{i}{100} * \frac{360 - M}{360}$

    donde: K = Capital
    M = Día
    i = Tasa

    Se requiere calcular también el número de clientes, el total de capital depositado y el total de intereses.
    Realizar análisis, algoritmo y codificaciones Fortran y Pascal.

EJERCICIOS COMPILADOS

EJERCICIO Nro. 2.1

Calcular la prima de servicios para varios empleados de la siguiente forma:

100% del sueldo para empleados administrativos (Tipo 1)
200% del sueldo para operarios (Tipo 2).

NOTA: Los sueldos de la institución oscilan entre $200.000 y $500.000.

Codificación Pascal

```
PROGRAM PRIMA;
{Cálculo de la prima de servicios}
USES CRT,PRINTER;
VAR
    S,P     :REAL;
    T       :BYTE;
    RES     :STRING[2];
    OK      :BOOLEAN;
{   S       = Sueldo
    P       = Prima
    T       = Tipo de empleado
    OK      = Condición
    RES     = Respuesta}
BEGIN
    REPEAT
      {Validación del tipo de empleado}
      REPEAT
          WRITELN(':10, 'teclee el tipo de empleado :');
          READ(T);
          IF (T<1) OR (T>2) THEN
              WRITELN(':10, 'Error en datos, inténtelo de nuevo');
      UNTIL (T>=1) AND (T<=2);
      {Validación del sueldo del empleado}
      REPEAT
          WRIELN(':10, 'Teclee el sueldo del empleado :');
          READ(S);
          OK:= (S>=200000) AND (S<=500000);
          IF NOT OK THEN
              WRITELN(':10, 'Sueldo incorrecto, inténtelo de nuevo');
          DELAY(10000);
```

```
        CLRSCR;
      UNTIL OK;
      IF T=1 THEN
        P:= S
      ELSE
        P:= 2*S;
      WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
      WRITELN(Lst,'':10,'Para un sueldo de:',S:8:2,'':40,'La prima de servicios es de:',
              P:8:2);
      WRITELN(Lst);WRITELN(Lst);
      WRITELN('':10, 'Desea calcular otra prima ?');
      READ(RES);
    UNTIL (RES= 'NO') OR (RES='no');
END.
```

EJERCICIO Nro. 2.2

Leer N términos y hallar cuáles de ellos son primos. Codificar en lenguaje Pascal.

```
PROGRAM PRIMOS;
{Determina en una serie de datos, (comprendidos entre 1 y 32767), cuáles son números
primos. Un número es primo cuando no es divible exactamente por los números
comprendidos entre 2 y la parte entera de su raíz cuadrada}
USES CRT,PRINTER;
VAR
    I,A,N,B,C,NUM,D,E :INTEGER;
BEGIN
    FOR I:= 1 TO 6 DO
      WRITELN(Lst, '');
    WRITELN(Lst, '':24, 'Universidad Nacional de Colombia');
    FOR I:= 1 TO 4 DO
      WRITELN(Lst, '');
    WRITELN(Lst, '':33, 'Sede Manizales');
    FOR I:= 1 TO 9 DO
      WRITELN(Lst, '');
    WRITELN(Lst, '':19, 'Programa que encuentra los números primos');
    FOR I:= 1 TO 6 DO
      WRITELN(Lst, '');
    CLRSCR;
    REPEAT
      WRITE(Lst, 'Entre cuántos números desea analizar :');
      READLN(N);
      WRITELN(N);
```

```
    IF N<= 0 THEN
        WRITE(Lst,'':10,N, 'Debe pertenecer a los enteros positivos, teclee de nuevo');
  UNTIL N>0
  FOR B:= 1 TO N DO
    BEGIN
      REPEAT
        WRITELN(Lst, '':10,'Entre el número :');
        READLN(NUM);
        WRITELN(Lst,NUM);
        IF NUM<= 0 THEN
          WRITELN(Lst, '':10,NUM, 'Debe pertenecer a los enteros positivos,
                  teclee de nuevo :');
      UNTIL NUM>0;
      C:= TRUNC(SQRT(NUM));
      A:= 1;
      D:= 2;
      WHILE (A= 1) AND (D<= C) DO
        BEGIN
          E:= NUM Mod D;
          IF E= 0 THEN
            A:= 0;
          D:= D+1;
        END;
      IF A= 1 THEN
        WRITELN(Lst, '':10,NUM, 'Es un número primo');
      ELSE
        WRITELN(Lst, '':10,NUM, 'No es un número primo');
      WRITELN;
    END;
  WRITELN(Lst, '':10,'Teclee ENTER para terminar');
  READLN;
END.
```

EJERCICIO Nro. 2.3

Leer N términos y hallar cuáles de ellos son primos. Codificar en lenguaje Fortran.

```
      PROGRAM PRIMOS
C     Programa que determina si un término es primo
      WRITE(*,8)
    8 FORMAT(5(/),25X, 'Universidad nacional',3(/),25X, 'Sede Manizales')
      WRITE(*,1)
    1 FORMAT(3(/),27X, 'Programa de números primos',4(/),10X, 'Teclee el número de
```

```
     *términos a determinar si son primos :')
      READ(*,2)N
   2  FORMAT(I6)
      WRITE(*,4)
   4  FORMAT(//,10X,'Escriba los números, cada uno, después de la respuesta anterior;',
     */,10X, 'máximo de 5 cifras :')
      DO 3 J=1,N
         READ(*,5)L
   5     FORMAT(I6)
         IF (L-32678)34,34,32
  32        WRITE(*,67)
  67        FORMAT(/,10X, 'El número excede el límite')
            GOTO 3
  34     IF (L-0)22,22,23
  22        WRITE(*,24)L
  24        FORMAT(/,10X,I6,5X, 'No es primo')
            GOTO 3
  23     IF (L-3)25,25,26
  25        WRITE(*,27)L
  27        FORMAT(/,10X,I6,5X, 'Es primo')
            GOTO 3
  26     M= L**0.5
         DO 6 I= 2,M
             A= L/I
             K= A
             IF (K-A)6,9,6
   6     CONTINUE
         WRITE(*,13)L
  13     FORMAT(/,10X,I6,5X, 'Es primo')
         GOTO 3
   9     WRITE(*,10)L
  10     FORMAT(/,10X,I6,5X, 'No es primo')
   3  CONTINUE
      STOP
      END
```

EJERCICIO Nro. 2.4

Leer un conjunto de N valores y determinar cuáles son múltiplos de 3, 5 ó 7.

```
      PROGRAM MULTI
```

```
C       Definición de variables, N :Número de elementos del conjunto
C       L :Nombre del elemento
C       Programa que determian qué elementos de un conjunto de números son múltiplos
C       de 7 y/o de 5 y/o de 3.
        WRITE(*,10)
   10   FORMAT(8(/),20X, 'Programa para determinar en un conjunto',//,25X, 'de
       *números, cuáles son múltiplos',//,22X, 'de 7 y/o de 5 y/o de 3, o de ninguno',////,8X,
       *'Escriba un valor con I3',4(/),10X, 'Número',10X, 'Múltiplo',10X, 'Múltiplo',10X,
       *'Múltiplo'/,26X, 'de siete',10X, 'de cinco',10X, 'de tres')
        DO 240 N= 1,5
           READ(*,15)L
   15      FORMAT(I3)
           WRITE(*,20)L
   20      FORMAT(13X,I3)
           M=L
  170      IF (M-7)30,40,50
   30      WRITE(*,60)
   60      FORMAT(30X, 'NO')
  160      I=L
  200      IF (I-5)70,80,90
   70         WRITE(*,100)
  100         FORMAT(47X, 'NO')
  190         J=L
  230      IF (J-3)110,120,130
  110         WRITE(*,140)
  140         FORMAT(65X, 'NO')
              GOTO 240
   40      WRITE(*,150)
  150      FORMAT(30X, 'SI')
           GOTO 160
   50      M= M-7
           GOTO 170
   80      WRITE(*,180)
  180      FORMAT(47X, 'SI')
           GOTO 190
   90      I= I-5
           GOTO 200
  130      J= J-3
           GOTO 230
  120      WRITE(*,210)
  210      FORMAT(65X, 'SI')
  240   CONTINUE
        STOP
        END
```

EJERCICIO Nro. 2.5

El gerente de una compañía desea saber si los trabajadores nuevos tienden a trabajar más horas extras que los trabajadores antiguos, así mismo desea conocer la antiguedad de sus empleados según la siguiente tabla:

0	———	menos de 1 año
1 año	———	menos de 3 años
3 años	———	menos de 5 años
5 años	———	menos de 10 años
10 años y más		

Para cada categoría se quiere calcular el promedio de horas trabajadas por empleado. Preparar un reporte de acuerdo a lo solicitado, disponiendo para el efecto de la siguiente información por cada empleado: seguro social, edad, años trabajados en la compañía, tasa horaria, horas regulares trabajadas y horas extras trabajadas.

```
PROGRAM NOMINA;
USES CRT;
VAR
    NS,ED,CON2,CON3,CON4,CON5,CONN,CONV,CONT    :INTEGER;
    AT,TH,HR,HE,PT,PRO1,PRO2,PRO3,PRO4,PRO5,SHT    :REAL;
    SHT1,SHT2,SHT3,SHT4,SHT5,SUMN,SUMV,CON1          :REAL;
BEGIN
    CON1:= 0; PRO1:= 0; SHT1:= 0; SUMN:= 0;
    CON2:= 0; PRO2:= 0; SHT2:= 0; SUMV:= 0;
    CON3:= 0; PRO3:= 0; SHT3:= 0; CONN:= 0;
    CON4:= 0; PRO4:= 0; SHT4:= 0; CONV:= 0;
    CON5:= 0; PRO5:= 0; SHT5:= 0; CONT:= 0;
    CLRSCR;
    GOTOXY(20,1);
    WRITE('Universidad Nacional de Colombia');
    GOTOXY(35,2);
    WRITE('Sede Manizales');
    GOTOXY(28,7);
    WRITE('Programa nómina');
    DELAY(5000);
    CLRSCR;
    WRITELN;WRITELN;
    REPEAT
      WRITE('Número del seguro social :');
      READLN(NS);
      WRITE('Edad :');
      READLN(ED);
```

```
WRITE('Años trabajados en la compañía :');
READLN(AT);
WRITE('Tasa horaria :');
READLN(TH);
WRITE('Horas regulares trabajadas :');
READLN(HR);
WRITE('Horas extras trabajadas :');
READLN(HE);
PT:= TH*(HR+HE);
WRITELN('Paga total :', PT:6:2);
SHT:= HR+HE;
WRITELN('Total horas trabajadas :',SHT:2:2);
WRITELN;WRITELN;
DELAY(2000);
IF AT>1 THEN
   BEGIN
      CONV:= CONV+1;
      SUMV:= SUMV+HE;
   END;
IF (AT>0) AND (AT<=1) THEN
   BEGIN
      CONN:= CONN+1;
      SUMN:= SUMN+HE;
   END;
IF AT<1 THEN
   BEGIN
      CON1:= CON1+1;
      SHT1:= SHT1+SHT;
      PRO1:=SHT1/CON1;
   END;
IF (AT>=1) AND (AT<3) THEN
   BEGIN
      CON2:= CON2+1;
      SHT2:= SHT2+SHT;
      PRO2:=SHT2/CON2;
   END;
IF (AT>=3) AND (AT<5) THEN
   BEGIN
      CON3:= CON3+1;
      SHT3:= SHT3+SHT;
      PRO3:=SHT3/CON3;
   END;
IF (AT>=5) AND (AT<10) THEN
   BEGIN
```

```
            CON4:= CON4+1;
            SHT4:= SHT4+SHT;
            PRO4:=SHT4/CON4;
        END;
    IF AT>=10 THEN
        BEGIN
            CON5:= CON5+1;
            SHT5:= SHT5+SHT;
            PRO5:=SHT5/CON5;
        END;
    WRITE('Teclee 1 para parar, teclee 0 para continuar :');
    READLN(CONT);
    WRITELN;WRITELN;
UNTIL CONT=1;
CLRSCR;
WRITELN;WRITELN;
WRITELN('Total empleados antiguos :',CONV);
WRITELN('Horas extras trabajadas :',SUMV:6:2);
WRITELN;
WRITELN('Total empleados nuevos :',CONN);
WRITELN('Horas extras trabajadas :',SUMN:6:2);
WRITELN;WRITELN;
IF SUMV>SUMN THEN
    WRITELN('Empleados antiguos trabajan más horas extras que los nuevos');
IF SUMV<SUMN THEN
    WRITELN('Empleados nuevos trabajan más horas extras que los antiguos');
IF SUMV=SUMN THEN
    WRITELN('Todos los empleados trabajan las mismas horas extras');
WRITELN;WRITELN;
WRITELN('Empleados que han trabajado en la compañía menos de 1 año:',
            CON1:6:2);
WRITELN('Promedio de horas trabajadas por empleado :',PRO1:6:2);
WRITELN('Empleados que han trabajado en la compañía de 1 a menos de 3 años:',
            CON2);
WRITELN('Promedio de horas trabajadas por empleado :',PRO2:6:2);
WRITELN('Empleados que han trabajado en la compañía de 3 a menos de 5años:',
            CON3);
WRITELN('Promedio de horas trabajadas por empleado :',PRO3:6:2);
WRITELN('Empleados que han trabajado en la compañía de 5 a menos de 10 años:',
            CON4);
WRITELN('Promedio de horas trabajadas por empleado :',PRO4:6:2);
WRITELN('Empleados que han trabajado en la compañía 10 años ó más:',
            CON5);
WRITELN('Promedio de horas trabajadas por empleado :',PRO5:6:2);
```

```
    DELAY(30000);
END.
```

EJERCICIO Nro. 2.6

Se tiene un lote de N registros con la siguiente información: código del artículo, nombre, valor unitario, clase del artículo (1= importado, 2= nacional), unidades consumidas en el último mes, unidades consumidas en los últimos dos meses, unidades consumidas en los últimos tres meses, unidades consumidas en los últimos cuatro meses. Calcular

a) Promedio de consumo de los últimos cuatro meses.
b) Consumo esperado según la siguiente fórmula:
   Consumo esperado = (1+A)*Promedio
   donde A = 10 para artículos importados
   A = 15 para artículos nacionales
c) Total de unidades de consumo esperado tanto importadas como nacionales.

```
PROGRAM INVENTARIO;
USES CRT;
VAR
    I,COD,CLASE,UCMA,UCMS,UCMO,UCMN,N      :INTEGER;
    V_UNID,PROM_N,PROM_I,CEN,CEI,VCEN,VCEI :REAL;
BEGIN
    CLRSCR;
    WRITE('Digite número de registro a analizar :');
    READLN(N);
    WRITELN;WRITELN;
    FOR I:= 1 TO N DO
      BEGIN
        WRITE('Digite código del artículo :');
        READLN(COD);
        WRITELN;
        WRITE('Digite valor unitario del artículo :');
        READLN(V_UNID);
        WRITELN;
        REPEAT
          WRITELN('Entre la clase del artículo así:');
          WRITELN('  1 si el artículo es importado');
          WRITELN('  2 si el artículo es nacional');
          WRITE('La clase del artículo es :');
          READLN(CLASE);
          IF (CLASE= 1) OR (CLASE= 2) THEN
            BEGIN
```

```
            WRITELN;
            WRITE('Digite las unidades consumidas en Agosto :');
            READLN(UCMA);
        END
    ELSE
        BEGIN
            WRITELN;WRITELN;
            WRITELN('La clase del artículo es 1 ó 2 únicamente');
            WRITELN;
        END
UNTIL (CLASE=1) OR (CLASE=2);
WRITE('Digite las unidades consumidas en Septiembre :');
READLN(UCMS);
WRITE('Digite las unidades consumidas en Octubre :');
READLN(UCMO);
WRITE('Digite las unidades consumidas en Noviembre :');
READLN(UCMN);
WRITELN;WRITELN;
PROM_I:= 0;
PROM_N:= 0;
CEI:= 0;
CEN:= 0;
VCEN:= 0;
VCEI:= 0;
IF CLASE=1 THEN
    BEGIN
        PROM_I:= (UCMA+UCMS+UCMO+UCMN)/4;
        CEI:= 1.1*PROM_I;
        VCEI:= CEI*V_UNID;
        WRITELN('Código:  ',COD, '     Valor Unidad:   ',V_UNID:5:2);
        WRITELN;
        WRITELN('Artículo importado');
        WRITELN;
        WRITELN('Promedio unidades consumidas importadas:',PROM_I:5:2);
        WRITELN;
        WRITELN('Consumo esperado en art. importados:  ',CEI:5:2);
        WRITELN;
        WRITELN('Valor consumo esperado en arts. importados:  ',VCEI:5:2);
        WRITELN;WRITELN
    END
ELSE
    BEGIN
        PROM_N:= (UCMA+UCMS+UCMO+UCMN)/4;
        CEN:= 1.15*PROM_N;
```

```
                VCEN:= CEN*V_UNID;
                WRITELN('Código:  ',COD, '     Valor Unidad:   ',V_UNID:5:2);
                WRITELN;
                WRITELN('Artículo nacional');
                WRITELN;
                WRITELN('Promedio unidades consumidas nacionales:',PROM_N:5:2);
                WRITELN;
                WRITELN('Consumo esperado en art. nacionales:  ',CEN:5:2);
                WRITELN;
                WRITELN('Valor consumo esperado en arts. nacionales:  ',VCEN:5:2);
                WRITELN;WRITELN
            END;
    END;
END.
```

CAPITULO III

ARREGLOS

3.1 INTRODUCCIÓN
3.2 DEFINICIÓN
3.3 ARREGLOS DE UNA DIMENSIÓN
PROBLEMAS RESUELTOS
PROBLEMAS PROPUESTOS
3.4 ARREGLOS DE DOS DIMENSIONES
PROBLEMAS RESUELTOS
PROBLEMAS PROPUESTOS
EJERCICIOS COMPILADOS

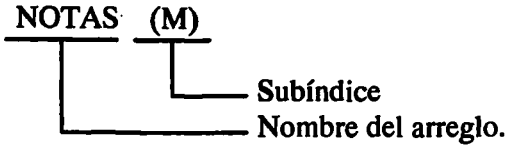
3.1 INTRODUCCIÓN

Supóngase que existe un conjunto de 30 notas pertenecientes a un grupo de estudiantes de matemáticas y se desea obtener su promedio. Para solucionar este problema con las herramientas estudiadas hasta el momento, se tendrían que realizar treinta ciclos y por cada ciclo leer una nota, para luego acumularla y por último hallar su promedio; ó para evitar las treinta lecturas, se podría realizar una sola que comprenda las 30 variables, siendo este método el menos indicado porque se tendrían tantas variables como datos se quisieran analizar. Para obviar lo anterior, se dispone de otra herramienta de programación como son los **ARREGLOS**.

3.2 DEFINICIÓN

Los arreglos permiten representar un conjunto finito de datos del mismo tipo bajo el nombre de una variable.
Los arreglos se componen del nombre del arreglo y del subíndice o suscrito, expresándose el subíndice a continuación del nombre del arreglo mediante el empleo de paréntesis.

Cuando se hace referencia a un valor específico del arreglo se le denomina **ELEMENTO.** Cada elemento del arreglo ocupa un espacio fijo y contiguo en la memoria del computador, igual al que ocuparía el elemento si fuera tratado como una variable independiente.



El valor dentro del paréntesis o suscrito puede ser:

Una variable. Ejemplo NOTAS(M), X(L)
Una constante. Ejemplo NOTAS(30), Y(8)
Una expresión. Ejemplo NOTAS(I+2), Z(2*N)

Los suscritos pueden ser variables, constantes o expresiones aritméticas, pero siempre de tipo entero.

3.3 ARREGLOS DE UNA DIMENSIÓN

Cuando el arreglo está conformado de un subíndice se le denomina **unidimensional o vectorial,** si está compuesto de dos subíndices se le conoce como **bidimensional o matricial** y compuesto por más de dos subíndices se le conoce como **multidimensional.**

Es necesario diferenciar entre el valor o dato y la posición que éste ocupa en el arreglo. Por ejemplo.

Se tienen los siguientes valores 8, 7, 1, -40, 3 y se van a almacenar en un arreglo X.

X				
X(1)	X(2)	X(3)	X(4)	X(5)
8	7	1	-40	3

El valor 8 está en la posición 1 del arreglo X; X(1) = 8
El valor -40 está en la posición 4 del arreglo X; X(4)= -40
La posición 5ª del arreglo X tiene almacenado el valor 3; X(5) = 3
La posición 2ª del arreglo X tiene almacenado el valor 7; X(2) = 7

EJERCICIO

Asumiendo que se tiene en memoria un arreglo A con los siguientes valores, calcular las diferentes formas de suscrito presentadas a continuación:

A													
A(1)	A(2)	A(3)	A(4)	A(5)	A(6)	A(7)	A(8)	A(9)	A(10)	...	L	M	J
5.	21.	7.1	4.	3.	20.	3.5	9.	13.	-80.		2	6	4
MEMORIA													

1. Una constante entera.
   Z ← A(5) * 3
   Z ← 3 * 3
   Z ← 9

2. Una variable entera.
   Z ← A(M)/2
   Z ← A(6)/2
   Z ← 20/2
   Z ← 10

3. Una expresión de la forma (V + C), donde V es una variable y C es una constante.
   Z ← A(L + 2)↑2
   Z ← A(2 + 2)↑2
   Z ← A(4)↑2
   Z ← 4↑2
   Z ← 16

4. Una expresión de la forma (V - C)
   Z ← A(J - 2) + 5
   Z ← A(4 - 2) + 5
   Z ← A(2) + 5
   Z ← 21 + 5
   Z ← 26

5. **Una expresión de la forma (C * V)**
   Z ← A(2 * J)/2
   Z ← A(2 * 4)/2
   Z ← A(8)/2
   Z ← 9/2
   Z ← 4.5

6. **Una expresión de la forma (C * V + C')**
   Z ← A(3 * L + 4)/8
   Z ← A(3 * 2 + 4)/8
   Z ← A(10)/8
   Z ← -80/8
   Z ← -10

7. **Una expresión de la forma (C * V - C')**
   Z ← A(5 * J - 15)↑2
   Z ← A(5 * 4 - 15)↑2
   Z ← A(5)↑2
   Z ← 3↑2
   Z ← 9

EJERCICIO

Leer un arreglo de N elementos y hallar la suma de todos sus elementos. Imprimir el vector leído y la suma.

Solución

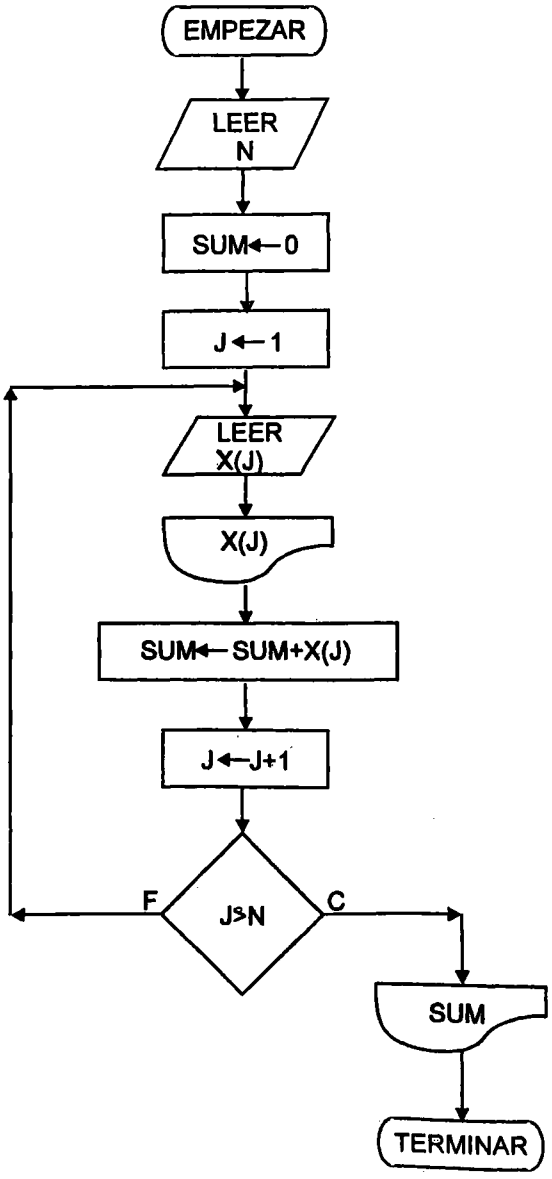
a) Análisis

Entrada: Leer el límite del arreglo N
Leer el arreglo
Iniciar sumatoria de términos en cero y el subíndice o variable que maneja la posición del vector en 1.
Proceso: Mientras el subíndice sea inferior al límite del vector se debe acumular su contenido.
Salida: Imprimir el vector leído, así como la suma.

b) Variables a utilizar

N = Límite de elementos del vector
SUM = Sumatoria de elementos
X = Nombre del arreglo
J = Subíndice o posición del arreglo

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        SUM ← 0
Paso 4        J ← 1
Paso 5        REPETIR
                 LEER X(J)
                 IMPRIMIR X(J)
                 SUM ← SUM + X(J)
                 J ← J + 1
              HASTA_QUE J > N
Paso 6        IMPRIMIR SUM
Paso 7      FIN
```

e) Prueba de escritorio

Sea el vector 1, 7, 3, 8

N	SUM	J	X(J)
4	0	1	X(1) = 1
	0+1 = 1	2	X(2) = 7
	1+7 = 8	3	X(3) = 3
	8+3 = 11	4	X(4) = 8
	11+8 = 19	5	

SEGUNDA VERSIÓN DEL ALGORITMO

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        SUM ← 0
Paso 4        J ← 1
Paso 5        MIENTRAS J <= N HACER
                 LEER X(J)
                 IMPRIMIR X(J)
                 SUM ← SUM + X(J)
                 J ← J + 1
              FIN_MIENTRAS
Paso 6        IMPRIMIR SUM
Paso 7      FIN
```

TERCERA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        SUM ← 0
Paso 3        LEER N
Paso 4        PARA J DE 1 HASTA N HACER
                LEER X(J)
                SUM ← SUM + X(J)
                IMPRIMIR X(J)
              FIN_PARA
Paso 5        IMPRIMIR SUM
Paso 6      FIN
```

EJERCICIO

Calcular la suma de los cuadrados de los 20 elementos de un arreglo unidimensional X.

Solución

a) Análisis

Entrada: Se inicializa el subíndice del vector en 1 y la sumatoria de los cuadrados en cero. Leer el arreglo.

Proceso: Se controla la posición del arreglo con el subíndice y se eleva al cuadrado el contenido de cada posición hasta completar sus 20 elementos.

Salida : Imprimir la sumatoria

b) Variables a utilizar

I = Subíndice del arreglo.
SUM = Suma de cuadrados.
X = Nombre del vector.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        I ← 1
Paso 3        SUM ← 0
Paso 4        REPETIR
                LEER X(I)
                SUM ← SUM + X(I)↑2
```

```
                I ← I + 1
              HASTA_QUE I > 20
Paso 5        IMPRIMIR SUM
Paso 6      FIN
```

Observación

La primera posición de un arreglo vectorial se puede iniciar en cero o en uno, dependiendo del lenguaje de programación que se trabaje.

d) Prueba de escritorio

Sea el arreglo X:

2	5	1	4	3	...	9
X(1)	X(2)	X(3)	X(4)	X(5)	...	X(20)

I	X(I)	SUM
1	X(1) = 2	$0+2^2 = 4$
2	X(2) = 5	$4+5^2 = 29$
3	X(3) = 1	$29+1^2 = 30$
4	X(4) = 4	$30+4^2 = 46$
5	X(5) = 3	$46+3^2 = 55$
.	.	.
.	.	.
.	.	.
20	X(20) = 9	

Otra solución al ejercicio empleando la estructura PARA es:

Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUM ← 0
Paso 3        PARA I DE 1 HASTA 20 HACER
                  LEER X(I)
                  SUM ← SUM + X(I)↑2
              FIN_PARA
Paso 4        IMPRIMIR SUM
Paso 5      FIN
```

Observación

Cuando se trabajan arreglos como parte del rango de una estructura PARA, el subíndice debe ser idéntico al índice de la estructura.

EJERCICIO

Se tienen 2 vectores A y B de 10 elementos cada uno y se desea crear otro vector que contenga la multiplicación de los vectores leídos, imprimir los tres vectores.

Solución

a) Análisis

Entrada: Inicializar el subíndice en 1
Leer los dos vectores originales.
Proceso: Se toma la primera posición del vector A y se multiplica por la primera posición del vector B, el resultado se almacena en la primera posición del vector C. Este proceso se repite tantas veces como elementos contengan los vectores.
Salida : Imprimir los 3 vectores

b) Variables a utilizar

L = Subíndice de los arreglos.
A, B = Nombre de los vectores originales.
C = Nombre del vector resultante.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        L ← 1
Paso 3        REPETIR
                  LEER A(L), B(L)
                  C(L) ← A(L) * B(L)
                  IMPRIMIR A(L), B(L), C(L)
                  L ← L + 1
              HASTA_QUE L > 10
Paso 4      FIN
```

EJERCICIO

Calcular la media aritmética ($\overline{X}$) de un conjunto de N valores mediante el empleo de arreglos. Imprimir el vector leido y su media aritmética.

Solución

a) Análisis

Entrada: Inicializar la sumatoria de elementos en cero
Leer el vector X y su límite N
Proceso: Aplicar la siguiente fórmula:

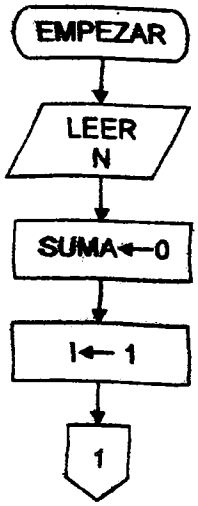
$$\overline{X} = \frac{\sum_{I=1}^{N} X_I}{N}$$

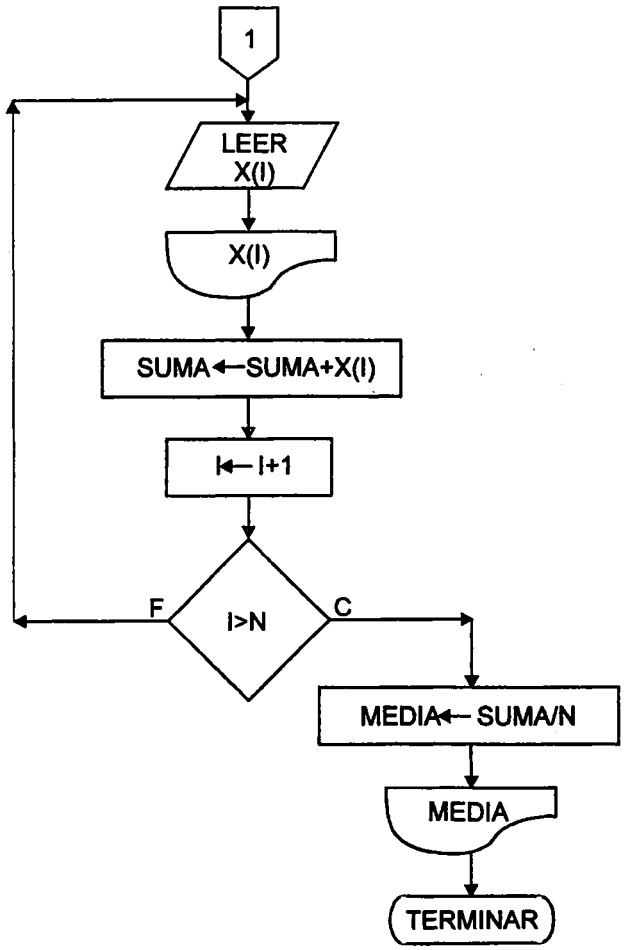
Leer cada elemento del vector y acumularlo,para finalmente dividir el acumulado en el número total de elementos y obtener la media aritmética.
Salida : Imprimir el vector leído y la media aritmética.

b) Variables a utilizar

N = Número de valores
SUMA = Sumatoria de valores
MEDIA = Media aritmética
X = Nombre del vector
I = Subíndice

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        I ← 1
Paso 4        LEER N
Paso 5        REPETIR
                  LEER X(I)
                  IMPRIMIR X(I)
                  SUMA ← SUMA + X(I)
                  I ← I + 1
              HASTA_QUE I > N
              MEDIA ← SUMA / N
Paso 6        IMPRIMIR MEDIA
Paso 7      FIN
```

Otra versión del algoritmo anterior es:

Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        LEER N
Paso 4        PARA I DE 1 HASTA N HACER
                 LEER X(I)
                 IMPRIMIR X(I)
                 SUM ← SUM + X(I)
              FIN_PARA
              MEDIA ← SUM / N
Paso 5        IMPRIMIR MEDIA
Paso 6      FIN
```

De no quererse leer o imprimir simultáneamente dentro del cuerpo de la estructura PARA se requiere hacer lo siguiente.

Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        LEER N
Paso 4        PARA I DE 1 HASTA N HACER
                 LEER X(I)
              FIN_PARA
Paso 5        PARA J DE 1 HASTA N HACER
                 SUMA ← SUMA + X(J)
              FIN_PARA
Paso 6        MEDIA ← SUMA / N
Paso 7        PARA K DE 1 HASTA N HACER
                 IMPRIMIR X(K)
              FIN_PARA
Paso 8        IMPRIMIR MEDIA
Paso 9      FIN
```

En este ejercicio los índices I, J, K de las estructuras **PARA**, señalan la posición del vector a la cual se quiere referir, por lo tanto, no presentan incompatibilidad en la utilización porque todas ellas tienen sus límites definidos, por ejemplo:

```
PARA I DE 1 HASTA N HACER
   LEER X(I)
FIN_PARA
```

Cuando I vale 1, lee el contenido de la posición 1 del vector X, es decir lee X(1), así:

```
PARA K DE 1 HASTA N HACER
   IMPRIMIR X(K)
FIN_PARA
```

Cuando K vale 1 imprime el contenido de la posición 1 del vector X, es decir, imprime X(1) siendo el contenido del arreglo el mismo, lo que cambia es la forma como se llama. Resumiendo, lo que se lee, escribe o procesa, realmente no son los subíndices I, J, K, sino los valores que esas variables representan, siendo su denominación transparente para el computador.

Observación

Cuando se trabaja con variables suscritas, se debe reservar una ó más posiciones de memoria dependiendo de las necesidades del problema, lo que ocasiona que luego de ser leído el arreglo permanezca almacenado en la memoria del computador, facilitando las operaciones y escrituras de éstos y otros vectores resultantes cuando se deseen mediante la manipulación de cualquier variable que actúe como subíndice y no necesariamente con el mismo subíndice como fue leído o escrito el arreglo en ocasiones anteriores. Esas reservaciones de memoria son transparentes a las soluciones algorítmicas, por lo tanto no requieren de alguna especificación adicional.

EJERCICIO

Calcular la media aritmética ponderada según la siguiente fórmula:

$$\overline{X}_p = \frac{\sum_{i=1}^{N} X_i W_i}{\sum_{i=1}^{N} W_i}$$

Solución

a) Análisis

Entrada: Inicializar la sumatoria del numerador, es decir, de los Xi. Wi en cero, la sumatoria del denominador, o sea, de los Wi en cero y el subíndice en 1.

Leer el límite de los 2 vectores.
Proceso: Calcular la sumatoria de los X(I) * W(I) y la sumatoria de los W(I).
Salida : Imprimir la media aritmética ponderada.

b) Variables a utilizar

SXW = Sumatoria de los Xi * Wi
SW = Sumatoria de los Wi
X, W = Vectores originales
N = Límite de elementos de los vectores
I = Subíndice, común a ambos vectores
MAP = Media aritmética ponderada

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SXW ← 0
Paso 3        SW ← 0
Paso 4        I ← 1
Paso 5        LEER N
Paso 6        REPETIR
                  LEER X(I), W(I)
                  SXW ← SXW + X(I) * W(I)
                  SW ← SW + W(I)
                  I ← I + 1
              HASTA_QUE I > N
Paso 7        MAP ← SXW / SW
Paso 8        IMPRIMIR MAP
Paso 9      FIN
```

Otra versión del algoritmo es:

Algoritmo

```
Paso 1      INICIO
Paso 2        SXW ← 0
Paso 3        SW ← 0
Paso 4        LEER N
Paso 5        PARA I DE 1 HASTA N HACER
                  LEER X(I), W(I)
                  SXW ← SXW + X(I) * W(I)
```

```
                SW ← SW + W(I)
            FIN_PARA
Paso 6      MAP ← SXW / SW
Paso 7      IMPRIMIR MAP
Paso 8    FIN
```

EJERCICIO

Leer un vector de K posiciones y ordenarlo ascendentemente. Imprimir el vector leído y el vector ordenado.

Solución

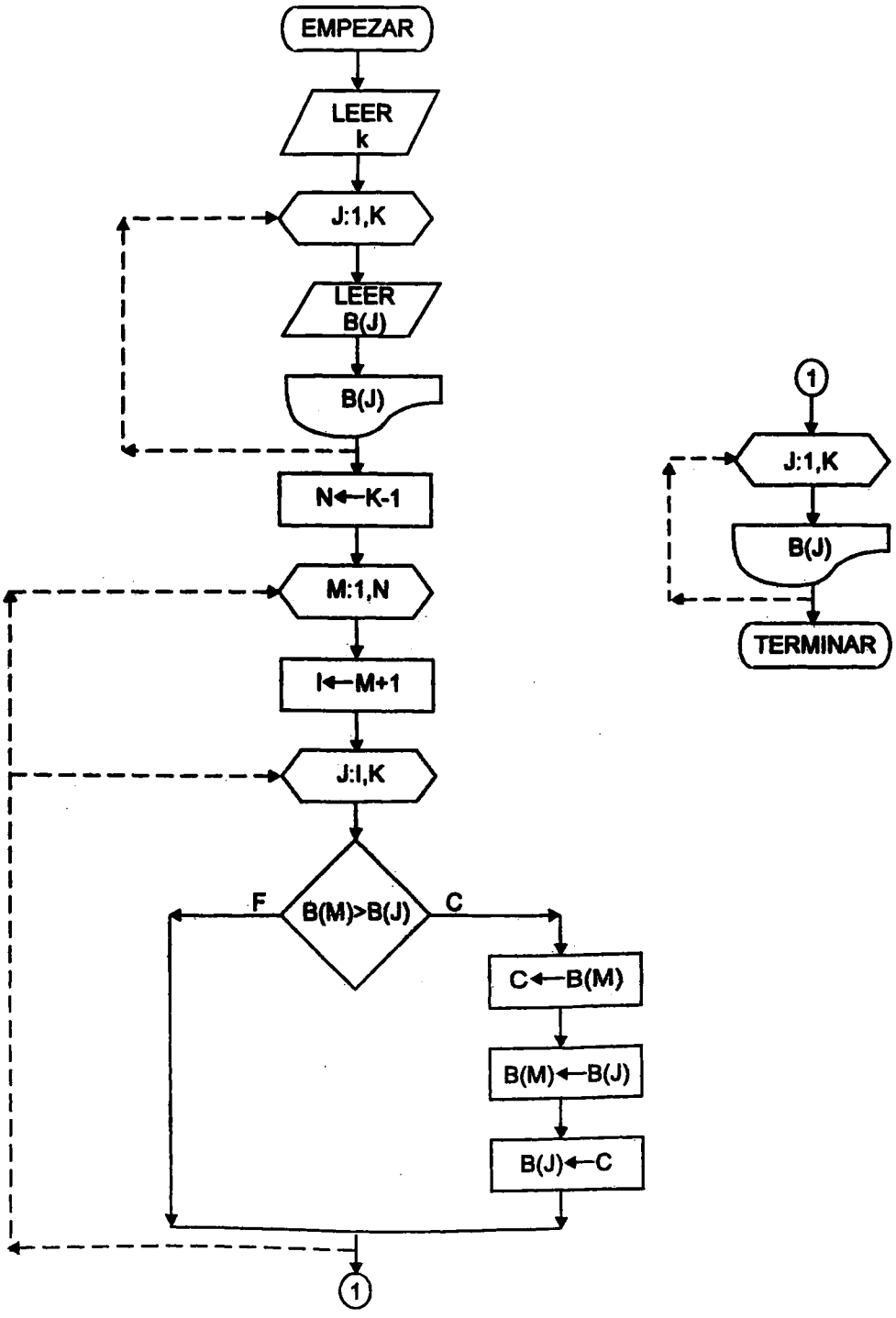
a) Análisis

Entrada: Leer el vector a ser ordenado y su límite.
Proceso: Se ordena el primer elemento del vector con respecto al resto de elementos, luego el segundo elemento y así sucesivamente.
Se utilizan 2 estructuras **PARA**, la estructura externa variando de 1 hasta el límite -1 y la estructura interna variando desde 2 hasta el límite del vector; esto con el propósito de no comparar el primer elemento consigo mismo, y también para no comparar el último término con el siguiente, que no existe.
Es necesario utilizar una variable temporal, por ejemplo si B(M) tiene un valor de 3 y B(J) tiene un valor de 2, al ordenarlo, se almacena el contenido de B(J) en B(M), perdiéndose el valor original que éste tenía; para evitarlo, se utiliza una variable temporal que almacene el valor de B(M) antes de trasladar el valor de B(J) a B(M), luego el contenido de la variable temporal se almacena en B(J) obteniéndose su ordenamiento.
Salida : Imprimir el vector original y el vector ordenado.

b) Variables a Utilizar

B = Nombre del vector
K = Número de elementos del vector B
C = Variable temporal
N = Penúltimo término del vector
I = Siguiente término del vector.
J, M = Variables índices de la estructura PARA

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER K
Paso 3        PARA J DE 1 HASTA K HACER
                LEER B(J)
                IMPRIMIR B(J)
              FIN_PARA
Paso 4        N ← K - 1
Paso 5        PARA M DE 1 HASTA N HACER
                I ← M + 1
                PARA J DE I HASTA K HACER
                  SI B(M) > B(J)
                    ENTONCES C ← B(M)
                              B(M) ← B(J)
                              B(J) ← C
                  FINSI
                FIN_PARA
              FIN_PARA
Paso 6        PARA J DE 1 HASTA K HACER
                IMPRIMIR B(J)
              FIN_PARA
Paso 7      FIN
```

e) Prueba de escritorio

Sea el vector 3, 1, 7, 0.

K	N	M	I	J	B(M)	B(J)	C	ORDENAMIENTO
4	3	1	2	2	B(1)=3	B(2)=1	3	
					B(1)=1	B(2)=3		1 | 3 | 7 | 0
				3	B(1)=1	B(3)=7		
				4	B(1)=1	B(4)=0	1	
					B(1)=0	B(4)=1		0 | 3 | 7 | 1
		2	3	3	B(2)=3	B(3)=7		
				4	B(2)=3	B(4)=1	3	
					B(2)=1	B(4)=3		0 | 1 | 7 | 3
		3	4	4	B(3)=7	B(4)=3	7	
					B(3)=3	B(4)=7		0 | 1 | 3 | 7

Otra forma de resolver el anterior ejercicio:

Algoritmo

```
Paso 1      INICIO
Paso 2        LEER K
Paso 3        PARA J DE 1 HASTA K HACER
                 LEER B(J)
                 IMPRIMIR B(J)
              FIN_PARA
Paso 4        J ← 1
Paso 5        REPETIR
                 SI B(J) > B(J+1)
                    ENTONCES C ← B(J)
                                 B(J) ← B(J+1)
                                 B(J+1) ← C
                                 J ← 1
                    SINO J ← J + 1
                 FINSI
              HASTA_QUE J = K
Paso 6        PARA J DE 1 HASTA K HACER
                 IMPRIMIR B(J)
              FIN_PARA
Paso 7      FIN
```

SINTAXIS FORTRAN

Cuando se está interesado en almacenar algunos de los valores que toma una variable durante la ejecución de un programa, se debe reservar no una, sino un conjunto de posiciones consecutivas para esta variable, mediante la instrucción **DIMENSION.**

La instrucción DIMENSION tiene la forma:

DIMENSION V1(K), V2(L,M), V3(I,J,M)

en donde:

V1, V2, V3	: representan cualquier nombre válido de variables suscritas.
K, L, N, I, J, M	: representan constantes enteras, que indican el número máximo de elementos de las filas o columnas de cada una de las variables.

Ejemplo:

```
DIMENSION X(9), Y(7,8), Z(20,10,12), M(100)
```

Cada una de las variables suscritas se escribe a continuación de la palabra DIMENSION, separadas por comas.

Cada variable suscrita tiene señalado dentro del paréntesis los máximos valores que cada uno de los suscritos puede tomar, así, DIMENSION Y(7,8) le indica al computador que la variable Y del programa es suscrita y puede tener hasta 7 filas y 8 columnas, para un total de 56 elementos. Por lo tanto el computador reservará 56 posiciones de memoria para la variable suscrita Y.

Observación

- Cada que en un programa se utilicen variables suscritas, es necesario codificar la instrucción DIMENSION para poder reservar las localizaciones de memoria necesarias.
- Las proposiciones DIMENSION deben aparecer en el programa antes de cualquier proposición ejecutable.

SINTAXIS PASCAL

Un arreglo se puede declarar mediante referencia a un tipo de arreglo previamente definido, o a través de la definición del arreglo en la declaratoria de variables mediante la utilización de la instrucción **ARRAY**, así:

```
VAR
    NOMBRE_ARREGLO :ARRAY[1..N] OF REAL;
```

donde :

NOMBRE_ARREGLO = Es el nombre del arreglo.
N = Límite de elementos del arreglo.

Ejemplo:

```
VAR
    A :ARRAY[1..100] OF REAL;
```

Indica que A es un arreglo unidimensional que puede contener hasta 100 elementos de tipo real.

```
VAR
    NOMBRE_ARREGLO  :ARRAY[1..N,1..M] OF INTEGER;
```

donde :

NOMBRE_ARREGLO = Es el nombre asignado al arreglo.
N = Es el límite de filas del arreglo.
M = Es el límite de columnas del arreglo.

Ejemplo:

```
VAR
    B :ARRAY[1..5,1..7] OF INTEGER;
```

B es un arreglo matricial que puede contener hasta 5 filas y 7 columnas, para un total de 35 elementos de tipo entero.

Observación

- Todos los elementos del arreglo deben ser del mismo tipo y el tipo al que corresponde cada índice debe ser ordinal.
- Los elementos del arreglo deben ser finitos y su número no varía durante la ejecución del programa.

EJERCICIO:

Leer un vector de n elementos y ordenarlo ascendente y descendentemente. Imprimir los tres vectores. Realizar la codificación en lenguajes FORTRAN y PASCAL.

Codificación FORTRAN

```
      PROGRAM ORDEN
      DIMENSION VL(10), VA(10), VD(10)
C     Este programa ordena un conjunto de elementos ascendente y descendentemente,
C     además muestra los tres vectores.
C     VL    = Nombre del vector leído.
C     VA    = Nombre del vector ordenado ascendentemente.
C     VD    = Nombre del vector ordenado descendentemente.
C     A     = Variable auxiliar.
C     NP    = Número de posición.
      WRITE(*,1)
    1 FORMAT(5X, 'CUANTOS ELEMENTOS TIENE EL VECTOR ?')
      READ(*,5)N
```

```
    5 FORMAT(I2)
      WRITE(*,8)
    8 FORMAT(2X, 'ENTRE LOS VALORES CON FORMATO REAL F10.3 ')
      DO 11 J=1,N
         READ(*,18)VL(J)
         VA(J)=VL(J)
   18    FORMAT(F10.3)
   11 CONTINUE
      DO 22 J=1,N
C        Se hace NP=J para almacenar el número de posición de VA(J)
         NP=J
         A=VA(J)
         DO 33 I=J-1,1,-1
            IF (A .LT. VA(I)) THEN
               NP=I
C              Cuando se encuentra que A<VA(I) se hace NP=I y se corren
C              mayores que A una posición en el vector descendente y en el ascendente
C              se disminuye una posición.
               VD(N-I)=VD(N+1-I)
               VA(I+1)=VA(I)
            ENDIF
   33    CONTINUE
C       Se inserta A en el vector ascendente y descendente de acuerdo con el valor de NP
         VA(NP)=A
         VD(N+1-NP)=A
   22 CONTINUE
      WRITE(*,23)
   23 FORMAT(5X, 'EL VECTOR LEIDO',5X, 'VEC. ORD. ASC.',5X, 'VEC. ORD.
     *DESC.' /// )
      DO 44 J=1,N
         WRITE(*,24) VL(J),VA(J),VD(J)
   24    FORMAT(6X,F10.3,7X,F10.3,7X,F10.3)
   44 CONTINUE
      STOP
      END
```

Codificación PASCAL

```
PROGRAM ORDEN;
USES
    CRT,PRINTER;
CONST
    M=50;
```

```
VAR
    VL, VA, VD    :ARRAY[1..M] OF REAL;
    N, I, J, NP   :INTEGER;
    A             :REAL;
BEGIN
  {Este programa ordena un conjunto de elementos ascendente y descendentemente,
  además muestra los tres vectores}
  {VL = Nombre del vector leído.
   VA = Nombre del vector ordenado ascendentemente.
   VD = Nombre del vector ordenado descendentemente.
   NP = Número de posición.}
  WRITELN(LST, '':5, 'CUANTOS ELEMENTOS TIENE EL VECTOR :');
  READ(N);
  WRITELN(LST, '':5, 'ENTRE LOS VALORES CON FORMATO REAL');
  FOR J:=1 TO N DO
      BEGIN
        READLN(VL[J]);
        VA[J]:=VL[J]
      END;
  FOR J:=1 TO N DO
      {Se hace NP=J para almacenar el número de posición en VA(J)}
      BEGIN
        NP:=J;
        A:=VA[J];
        FOR I:=(J-1) DOWNTO 1 DO
            BEGIN
              IF A<VA[I] THEN
                  BEGIN
                    NP:=I;
                    VD[N-I]:=VD[N+1-I];
                    VA[I+1]:=VA[I]
                  END
            END
            {Se inserta A en el vector ascendente y descendente de acuerdo con el valor
            de NP.}
        VA[NP]:=A;
        VD[N+1-NP]:=A
      END;
  WRITELN(LST, '':5, 'VECTOR LEIDO', '':5, 'VEC. ORD. ASC.', '':5, 'VEC. ORD.
  DESC.');
  FOR J:=1 TO N DO
      WRITELN(LST, '':6,VL[J]:10:3, '':7,VA[J]:10:3, '':7, VD[J]:10:3);
END.
```

PROBLEMAS RESUELTOS

EJERCICIO

La compañía Manizales requiere actualizar su inventario con base al movimiento de ventas. El inventario está compuesto de 50 artículos y dispone de la siguiente información por artículo: código, nombre, cantidad y valor unitario.
La compañía realizó 30 ventas de artículos y la información que se tiene por cada venta es: código del artículo vendido y cantidad vendida.
Imprimir: inventario inicial, movimiento de ventas, inventario final actualizado y valor total del inventario final.

Solución

a) Análisis

Entrada: Leer código, nombre, cantidad y valor unitario de cada artículo del inventario inicial, así como código del artículo vendido y cantidad vendida.
Proceso: Cuando el código del artículo vendido sea igual al código del artículo en existencias se resta la cantidad vendida de la cantidad en existencias. La venta de un artículo se calcula multiplicando la cantidad vendida por el valor unitario.
El total del inventario final se calcula sumando las ventas parciales de artículos.
Salida : Imprimir el inventario inicial, el movimiento de ventas, el inventario final actualizado y el valor total del inventario final.

b) Variables a utilizar

I = Subíndice
COD = Código del artículo en inventario
NOM = Nombre del artículo
CAN = Cantidad del artículo
VUNIT = Valor unitario del artículo
CODV = Código del artículo vendido
CANV = Cantidad vendida
VAR = Valor del artículo en existencias
VIF = Valor del inventario final

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA I DE 1 HASTA 50 HACER
                LEER COD(I), NOM(I), CAN(I), VUNIT(I)
```

```
                IMPRIMIR COD(I), NOM(I), CAN(I), VUNIT(I)
            FIN_PARA
Paso 3      PARA K DE 1 HASTA 30 HACER
                LEER CODV, CANV
                IMPRIMIR CODV, CANV
                    PARA I DE 1 HASTA 50 HACER
                        SI CODV = COD(I)
                            ENTONCES CAN(I) ← CAN(I) - CANV
                        FINSI
                    FIN_PARA
            FIN_PARA
Paso 4      VIF ← 0
Paso 5      PARA I DE 1 HASTA 50 HACER
                VAR(I) ← CAN(I) * VUNIT(I)
                IMPRIMIR COD(I),NOM(I),CAN(I),VUNIT(I),VAR(I)
                VIF ← VIF + VAR(I)
            FIN_PARA
Paso 6      IMPRIMIR VIF
Paso 7  FIN
```

EJERCICIO

Se tienen 3 arreglos unidimensionales con la siguiente información por estudiante: código del estudiante, valor por materia y número de materias. Generar un cuarto arreglo que contenga el valor a pagar por matrícula de cada uno de los N estudiantes sumándole $1.500= por concepto de derechos médicos. Calcular además el total recaudado por pago de matrículas y la matrícula promedio por estudiante. Imprimir toda la información leída, así como la calculada.

Solución

a) Análisis

Entrada: Leer el número de elementos del vector, así como el código del estudiante, valor por materia y número de materias por estudiante.
Inicializar en cero el total pagado por matrícula.

Proceso: Calcular:
Total matrícula del estudiante = valor matrícula x número de materias + 1500
Total pagado por matrículas = total pagado por matrículas + total matrícula del estudiante.
Promedio = (total pagado por matrículas) / (número de estudiantes).

Salida : Imprimir código del estudiante, valor por materia, número de materias del estudiante, total matrícula del estudiante, total pagado por matrículas y matrícula promedio.

b) Variables a utilizar

N = Número de estudiantes
COD = Código de estudiante
VM = Valor materia
NM = Número de materias
PROM = Promedio
TPM = Total pagado por matrículas
TM = Total matrícula del estudiante.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        TPM ← 0
Paso 3        LEER N
Paso 4        PARA I DE 1 HASTA N HACER
                LEER COD(I), VM(I), NM(I)
                TM(I) ← VM(I) * NM(I) + 1500
                TPM ← TPM + TM(I)
                IMPRIMIR COD(I), VM(I), NM(I), TM(I)
              FIN_PARA
Paso 5        PROM ← TPM / N
Paso 6        IMPRIMIR TPM, PROM
Paso 7      FIN
```

EJERCICIO

Leer en N registros los valores de A y B, calcular e imprimir Y, siendo:

$$Y = \sum_{i=1}^{N} X_i$$

donde: $X_i = A + B^2$ _____ *si* $A^2 - B^2 < 0$

$X_i = A - 3B$ _____ *si* $A^2 - B^2 = 0$

$X_i = A + B - 7$ _____ *si* $A^2 - B^2 > 0$

Solución

a) Análisis

Entrada: Inicializar la sumatoria.
Leer el límite de los datos a ser procesados y por cada registro leer los valores A y B.

Proceso: Leída una pareja de datos A y B, se calcula $A^2 - B^2$ y el resultado se compara contra cero para determinar su correspondiente valor X(I) y luego acumularlo.
Salida : Imprimir la sumatoria.

b) Variables a utilizar

SUM = Sumatoria
N = Límite de datos a ser procesados
A, B = Valores a ser leídos
X(I) = Vector resultante

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SUM ← 0
Paso 3        LEER N
Paso 4        PARA I DE 1 HASTA N HACER
                  LEER A,B
                  SI (A↑2 - B↑ 2) < 0
                     ENTONCES X(I) ← A+B↑ 2
                     SINO SI (A↑ 2 - B↑ 2) = 0
                                 ENTONCES X(I) ← A-3*B
                                 SINO X(I) ← A+B-7
                           FINSI
                  FINSI
                  SUM ← SUM + X(I)
              FIN_PARA
Paso 5        IMPRIMIR SUM
Paso 6      FIN
```

EJERCICIO

Calcular la desviación estándar para un conjunto de K elementos, según las siguientes fórmulas:

$$DESV = \sqrt{\frac{\sum_{i=1}^{K}(X_i - \overline{X})^2}{K}} \qquad \overline{X} = \frac{\sum_{i=1}^{K} X_i}{K}$$

Solución

a) Análisis

Entrada: Inicializar la sumatoria de las desviaciones, o sea,la sumatoria del numerador y la sumatoria de los Xi en cero.
Leer el vector X y su límite

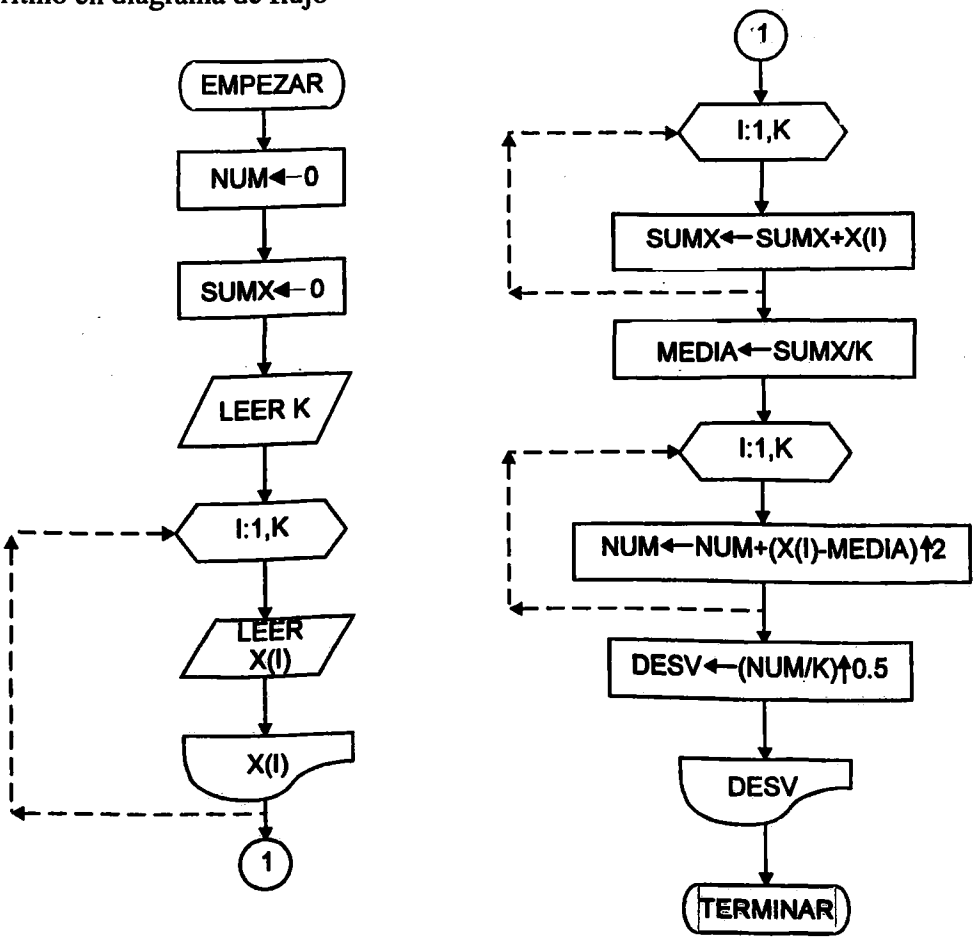
Proceso: Calcular primero la media aritmética $\overline{X}$, para luego calcular la sumatoria de las desviaciones o el numerador y por último se aplica la fórmula.

Salida : Imprimir el vector y la desviación estándar.

b) Variables a utilizar

K = Número de elementos del vector
SUMX = Sumatoria de los elementos del vector
NUM = Sumatoria de las desviaciones o numerador
MEDIA = Media aritmética
DESV = Desviación estándar.

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        NUM ← 0
Paso 3        SUMX ← 0
Paso 4        LEER K
Paso 5        PARA I DE 1 HASTA K HACER
Paso 6            LEER X(I)
                  IMPRIMIR X(I)
              FIN_PARA
Paso 7        PARA I DE 1 HASTA K HACER
                  SUMX ← SUMX + X(I)
              FIN_PARA
Paso 8        MEDIA ← SUMX / K
Paso 9        PARA I DE 1 HASTA K HACER
                  NUM ← NUM + (X(I) - MEDIA) ↑ 2
              FIN_PARA
Paso 10       DESV ← (NUM / K) ↑ 0.5
Paso 11       IMPRIMIR DESV
Paso 12     FIN
```

EJERCICIO

Leer un arreglo de K posiciones. Determinar cual es el elemento mayor y la(s) posición(es) que éste ocupa dentro del arreglo. Imprimir el vector leído, el elemento mayor y su(s) posición(es).

Solución

a) Análisis

Entrada: Leer el número de elementos del vector, así como el vector.
Proceso: Se asigna la primera posición del vector a una variable y luego se compara ésta contra la segunda posición del vector para determinar cual de las dos es mayor. Esta comparación se debe hacer a través de todo el vector.
Tan pronto se determina cuál es el elemento mayor, se hace otro recorrido a lo largo del vector y se compara el elemento mayor contra cada posición del vector, en donde coincidan se ordena imprimir su posición.
Salida : Imprimir el vector leído, el elemento mayor y su posición.

b) Variables a utilizar

K = Total de elementos del vector
Z = Nombre del vector
MAYOR = Elemento mayor (Inicialmente se asume igual a Z(1))
J = Posición del vector

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER K
Paso 3        PARA J DE 1 HASTA K HACER
                  LEER Z(J)
                  IMPRIMIR Z(J)
              FIN_PARA
Paso 4        MAYOR ← Z(1)
Paso 5        PARA J DE 2 HASTA K HACER
                  SI MAYOR < Z(J)
                     ENTONCES MAYOR ← Z(J)
                  FINSI
              FIN_PARA
Paso 6        IMPRIMIR MAYOR
Paso 7        PARA J DE 1 HASTA K HACER
                  SI MAYOR = Z(J)
                     ENTONCES IMPRIMIR J
                  FINSI
              FIN_PARA
Paso 8      FIN
```

EJERCICIO

Leer un vector de N elementos. Determinar la suma de los elementos positivos, la suma de los elementos negativos, la suma de las diferencias entre números adyacentes, además encontrar cuántos elementos iguales a cero hay en el vector. Imprimir el vector leído y todo lo calculado.

Solución

a) Análisis

Entrada: Inicializar en cero los contadores de números ceros, la sumatoria de números positivos, la sumatoria de números negativos y la sumatoria de las diferencias

entre números adyacentes.
Leer el vector así como el límite de sus elementos.

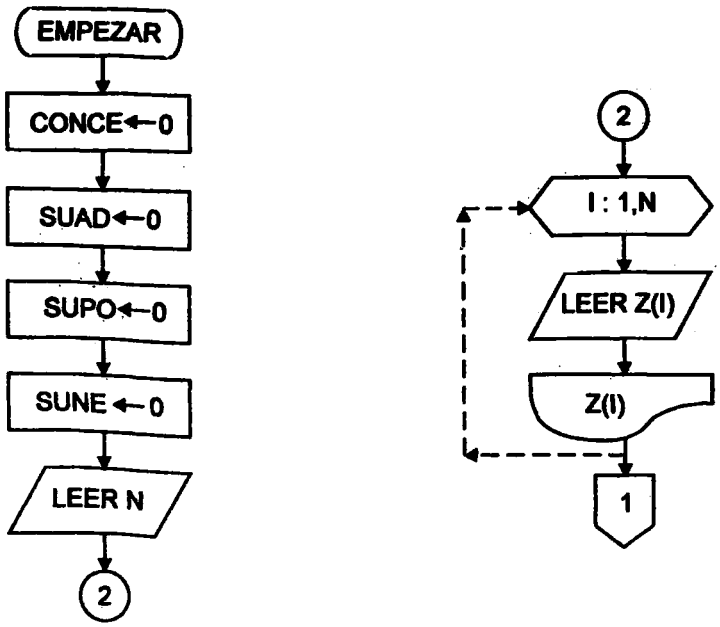
Proceso: Se utiliza una estructura **PARA** con el propósito de calcular la diferencia entre números adyacentes y la sumatoria de éstos. En otra estructura **PARA** se determina la sumatoria de elementos positivos, de elementos negativos y el contador de números ceros, mediante la comparación de cada posición del vector contra cero.

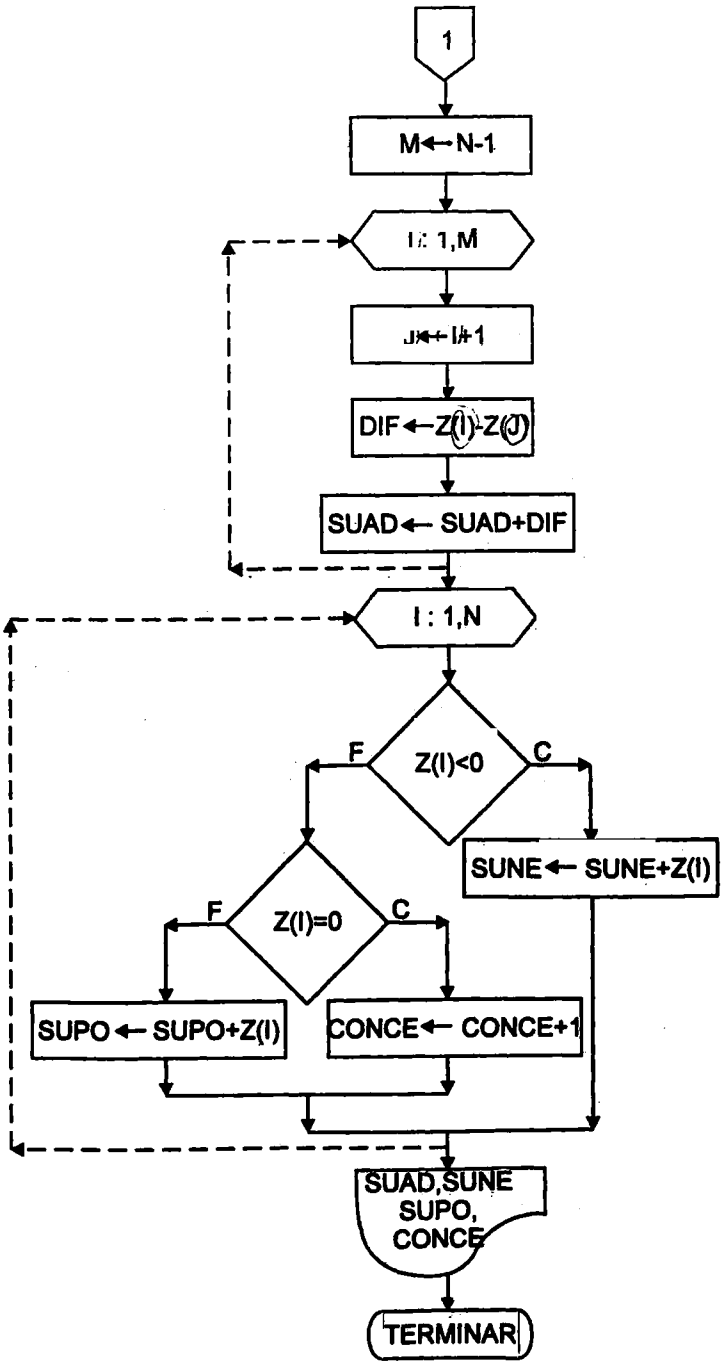
Salida : Imprimir el vector original así como la suma de los elementos positivos,elementos negativos, la suma de las diferencias entre números adyacentes y el número de elementos cero que tiene el vector.

b) Variables a utilizar

N = Número de elementos del vector
Z = Nombre del vector
SUAD = Sumatoria de elementos adyacentes
SUPO = Sumatoria de elementos positivos
SUNE = Sumatoria de elementos negativos
CONCE = Contador de números ceros
DIF = Diferencia de números adyacentes

c) Algoritmo en diagrama de Flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2          CONCE ← 0
Paso 3          SUAD ← 0
```

```
Paso 4      SUPO ← 0
Paso 5      SUNE ← 0
Paso 6      LEER N
Paso 7      PARA I DE 1 HASTA N HACER
                LEER Z(I)
                IMPRIMIR Z(I)
            FIN_PARA
Paso 8      M ← N - 1
Paso 9      PARA I DE 1 HASTA M HACER
                J ← I + 1
                DIF ← Z(I) - Z(J)
                SUAD ← SUAD + DIF
            FIN_PARA
Paso 10     PARA I DE 1 HASTA N HACER
                SI Z(I) < 0
                   ENTONCES SUNE ← SUNE + Z(I)
                   SINO SI Z(I) = 0
                                ENTONCES CONCE ← CONCE + 1
                                SINO SUPO ← SUPO + Z(I)
                            FINSI
                FINSI
            FIN_PARA
Paso 11     IMPRIMIR SUAD. SUNE, SUPO, CONCE
Paso 12   FIN
```

EJERCICIO

Se tienen 2 vectores X, Y, de N elementos cada uno y se desea calcular el promedio de acuerdo a la siguiente fórmula:

$$PROM = \frac{\sum_{i=1}^{N} X_i Y_i + \sum_{i=1}^{N} (X_i Y_i)^2}{\frac{1}{N}\sum_{i=1}^{N} X_i^2 - \left(\sum_{i=1}^{N} Y_i^2\right)^2}$$

Solución

a) Análisis

Entrada: Leer los dos vectores y su límite, e inicializar las sumatorias.
Proceso: Calcular: la sumatoria de los $X_i Y_i$

la sumatoria de los $(X_iY_i)^2$
la sumatoria de los $(X_i)^2$
la sumatoria de los $(Y_i)^2$

Salida : Imprimir el promedio

b) Variables a utilizar

N = Número de elementos de los vectores
X, Y = vectores originales
SXY = Sumatoria de $X_i * Y_i$
SX2 = Sumatoria de $(X_i)^2$
SY2 = Sumatoria de $(Y_i)^2$
SXY2 = Sumatoria de los $(X_i Y_i)^2$
PROM = Promedio

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        PARA I DE 1 HASTA N HACER
                LEER X(I), Y(I)
              FIN_PARA
Paso 4        SXY ← 0
Paso 5        SX2 ← 0
Paso 6        SY2 ← 0
Paso 7        SXY2 ← 0
Paso 8        PARA I DE 1 HASTA N HACER
                SXY ← SXY + X(I) * Y(I)
                SXY2 ← SXY2 + (X(I) * Y(I))↑ 2
                SX2 ← SX2 + X(I) ↑ 2
                SY2 ← SY2 + Y(I) ↑ 2
              FIN_PARA
Paso 9        PROM ← (SXY + SXY2)/(1/N * SX2 - SY2↑2)
Paso 10       IMPRIMIR PROM
Paso 11     FIN
```

EJERCICIO

Leer un vector de N elementos y conformar un segundo vector que contenga todos los elementos positivos del vector original, un tercer vector que contenga todos elementos negativos del vector original y un cuarto vector que contenga todos los elementos cero del vector original. Imprimir los 4 vectores.

Solución

a) Análisis

Entrada: Se inicializa la posición de los vectores de elementos negativos, de elementos positivos, y de elementos cero en cero.
Leer el límite N y el vector de datos.

Proceso: Cuando se encuentra un elemento bien sea positivo, negativo o cero, se debe llevar un controlador de su posición en donde va a ser almacenado para que quede consecutivo con el elemento anteriormente encontrado.De no hacerse esto, se tendría un vector de datos positivos con elementos en blanco intercalados y así mismo sucedería con el vector de elementos negativos y con el vector de elementos ceros.

Salida : Imprimir los 3 vectores.

b) Variables a utilizar

PN = Posición de elementos negativos
PP = Posición de elementos positivos
PC = Posición de elementos ceros
N = Límite del vector
X = Nombre del vector original
VN = Vector resultante de elementos negativos
VP = Vector resultante de elementos positivos
VC = Vector resultante de elementos ceros

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PN ← 0
Paso 3        PP ← 0
Paso 4        PC ← 0
Paso 5        LEER N
Paso 6        PARA J DE 1 HASTA N HACER
                 LEER X(J)
                 IMPRIMIR X(J)
              FIN_PARA
Paso 7        PARA J DE 1 HASTA N HACER
                 SI X(J) > 0
                    ENTONCES PP ← PP + 1
                             VP(PP) ← X(J)
                    SINO SI X(J) < 0
```

```
                                ENTONCES PN ← PN + 1
                                         VN(PN) ← X(J)
                                SINO     PC ← PC+1
                                         VC(PC) ← X(J)
                            FINSI
                        FINSI
                    FIN_PARA
Paso 8              PARA J DE 1 HASTA PP HACER
                        IMPRIMIR VP(J)
                    FIN_PARA
Paso 9              PARA J DE 1 HASTA PN HACER
                        IMPRIMIR VN(J)
                    FIN_PARA
Paso 10             PARA J DE 1 HASTA PC HACER
                        IMPRIMIR VC(J)
                    FIN_PARA
Paso 11         FIN
```

PROBLEMAS PROPUESTOS

1- Analizar el siguiente algoritmo mediante una prueba de escritorio.

```
INICIO
    IMPRIMIR "llenado de un vector"
    LEER N
    PARA I DE 1 HASTA N HACER
       LEER X
       A(I) ← X
       IMPRIMIR A(I)
    FIN_PARA
FIN
```

2- Probar el siguiente algoritmo con los datos: 1, 3, 5, 7, 9.

```
INICIO
    PARA K DE 1 HASTA 5 HACER
       LEER X(K)
    FIN_PARA
    K ← 5
    PARA J DE 1 HASTA 5 HACER
       Y(K) ← X(J)
       K ← K - 1
```

```
    FIN_PARA
    PARA J DE 1 HASTA 5 HACER
       X(J) ← Y(J)
       IMPRIMIR X(J)
    FIN_PARA
FIN
```

3- Analizar el algoritmo presentado a continuación mediante una prueba de escritorio tomando como base los siguientes datos: 1, 3, 0, 15, 12, 2, 13, 7, 14, 6, 3, 1, 18, 5

```
INICIO
    SUMA ← 0
    J ← 1
    REPETIR
       LEER Y (J)
       SUMA ← SUMA + Y (J)
       J ← J + 2
    HASTA_QUE J > 13
    IMPRIMIR SUMA
FIN
```

4-
```
INICIO
    I ← 0
    REPETIR
       LEER A (I)
       I ← I + 1
    HASTA_QUE I > 9
FIN
```
Cuántos datos se leen en el anterior algoritmo ?

5-
```
INICIO
    PARA I DE 1 HASTA 5 HACER
       PARA J DE 1 HASTA 3 HACER
          LEER X (J)
       FIN_PARA
    FIN_PARA
FIN
```

Cuántas lecturas se realizan ?

6- Leer un vector de 30 elementos y calcular la suma de los elementos de orden impar. Imprimir el vector leído y la suma.

7- La ecuación de la recta es: $Y = a + bX$

donde:

$$b = \frac{\sum XY - n\overline{XY}}{\sum X^2 - n\overline{X}^2}$$

$$a = \overline{Y} - b\overline{X}$$

Según las anteriores fórmulas del método de mínimos cuadrados, determinar la ecuación de la recta para los siguientes valores.

X	Y
1	0
2	1
3	2
4	3
5	4
6	5
7	6

8- Leer un vector A de N elementos y calcular la varianza, según la siguiente fórmula.

$$Varianza = \delta X^2 = \frac{1}{N-1}\left[\sum_{I=1}^{N}(X_i - \overline{X})^2\right]$$

9- Se tiene un par de vectores R y S de N elementos cada uno, calcular C como:

$$C = \sum_{K=1}^{N} B_K$$

$$B_K = R_K * S_K \_\_\_\_\_ si\ R_K = S_K$$
$$B_K = R_K^{S_K} \_\_\_\_\_ si\ R_K < S_K$$
$$B_K = R_K - S_K \_\_\_\_\_ si\ R_K > S_K$$

10-Leer 2 arreglos de K posiciones y encontrar los elementos que sean iguales en los dos arreglos.

11-Calcular la raíz cuadrada de la suma de los cuadrados de los primeros cincuenta elementos pares de una lista numérica. Dicho de otra forma. Calcular:

$$Z = \sqrt{\left(A(2)^2 + A(4)^2 + A(6)^2 + \cdots + A(100)^2\right)}$$

12-Leer un vector de 20 elementos y determinar cual es el elemento mayor del vector e imprimirlo, así como el vector leído.

13-Dado un vector de N elementos y cada elemento formado por un número de 4 dígitos, calcular e imprimir la suma de la primera cifra de cada elemento, la suma de las dos cifras siguientes de cada elemento y la suma de la última cifra de cada elemento.
Ejemplo: 5082, 1320, 5241, 7196, 9764, 0547.
Suma de las primeras cifras: 5+1+5+7+9+0=27.
Suma de las cifras intermedias: 08+32+24+19+76+54=213.
Suma de las últimas cifras: 2+0+1+6+4+7=20.

14-Leer un vector de N elementos y determinar cuáles números están repetidos y cuántas veces, además encontrar cuáles números faltan entre el menor y el mayor.
Ejemplo: 1, 4, 3, 3, 6, 7, 9, 7, 8, 7
El número 3 está repetido 2 veces.
El número 7 está repetido 3 veces.
Los números que faltan entre 1 y 9 son el 2 y el 5.

15-Leer un vector de N elementos y calcular su media armónica (H), según la siguiente fórmula:

$$H = \frac{N}{\sum_{i=1}^{N} \frac{1}{X_i}}$$

16-Leer los vectores X e Y de N elementos cada uno. Calcular Z según la siguiente fórmula:

$$Z = \sqrt{\sum_{i=1}^{N} (4X_i + 3Y_i)^2}$$

17-Leer un vector B de N elementos, calcular e imprimir Z según la siguiente fórmula:

$$Z = \frac{1}{N} \sum_{i=1}^{N} C_i$$

donde:

$$C_i = \begin{cases} 1.0 & si \quad B_i < 100 \\ 2.0 & si \quad 100 \leq B_i < 200 \\ 3.0 & si \quad B_i \geq 200 \end{cases}$$

18-Leer un arreglo Z de N elementos, encontrar Q según la siguiente fórmula:

$$Q = \frac{\sum_{i=1}^{N} Z_i}{\left[\sum_{i=1}^{N}\left(\sum_{i=1}^{N} Z_i\right) * Z_i\right]^2}$$

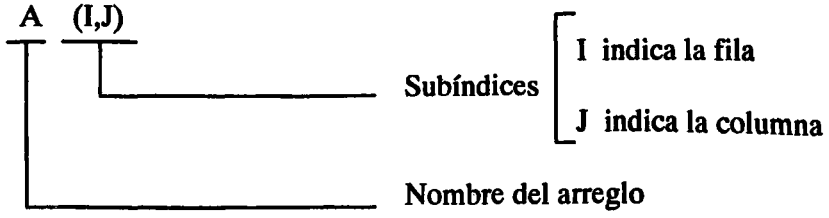
19-Leer los arreglos X , Y de N elementos cada uno, calcular e imprimir el coeficiente de correlación de Pearson (R) según la siguiente fórmula:

$$R = \frac{\sum_{i=1}^{N}(X_i Y_i) - \overline{XY}}{\sqrt{\left[\frac{1}{N}\sum_{i=1}^{N}(X_i^2) - \overline{X}^2\right] * \left[\frac{1}{N}\sum_{i=1}^{N}(Y_i^2) - \overline{Y}^2\right]}}$$

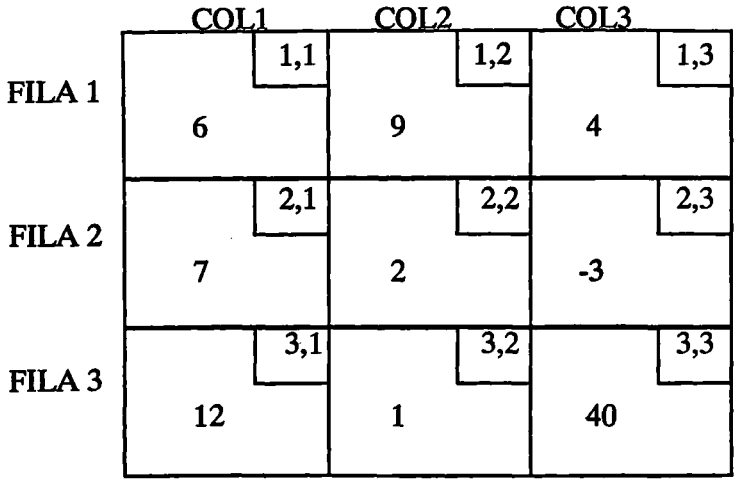
donde: $\overline{X} = \frac{\sum_{i=1}^{N} X_i}{N}$ $\overline{Y} = \frac{\sum_{i=1}^{N} Y_i}{N}$

3.4 ARREGLOS DE DOS DIMENSIONES

También llamados arreglos bidimensionales. Está conformado por el nombre del arreglo y dos subíndices separados por coma, el primero de ellos especifica la fila y el segundo la columna. Debido a que éste tipo de formaciones es muy común, también se le conoce con el nombre de **MATRICES.**



Al igual que en los arreglos de una dimensión, es necesario diferenciar entre el valor o dato y la posición que éste ocupa dentro del arreglo bidimensional. Por ejemplo. Se tiene la matriz A de 3 filas y 3 columnas con los siguientes datos:



El valor 6 está en la fila 1 columna 1 del arreglo A, es decir A(1,1) = 6.
El valor -3 está en la fila 2 columna 3 del arreglo A, es decir A(2,3) = -3
En la posición A(3,2), se tiene almacenado el valor 1
En la posición A(1,3), se tiene almacenado el valor 4.

EJERCICIO

Leer una matriz A de N filas y M columnas. Calcular la suma de los elementos de la última fila.

Solución

a) Análisis

Entrada: Leer el número de filas N y el número de columnas M. Inicializar la sumatoria de la última fila en cero.

Proceso: Haciendo referencia a la matriz anteriormente graficada, se pide sumar el contenido de las posiciones A(3,1) + A(3,2) + A(3,3) = 12+1+40 = 53. Para lograrlo es necesario utilizar un anidamiento de estructuras **PARA**, una de ellas para que maneje con su índice las filas y la otra para que controle las columnas. Dentro del cuerpo del anidamiento se efectúa la comparación del índice que maneja las filas contra el límite de filas, es decir, de I contra N, cuando esta comparación sea igual, indica que el control está ubicado en la última fila y se procede a sumar sus correspondientes valores.

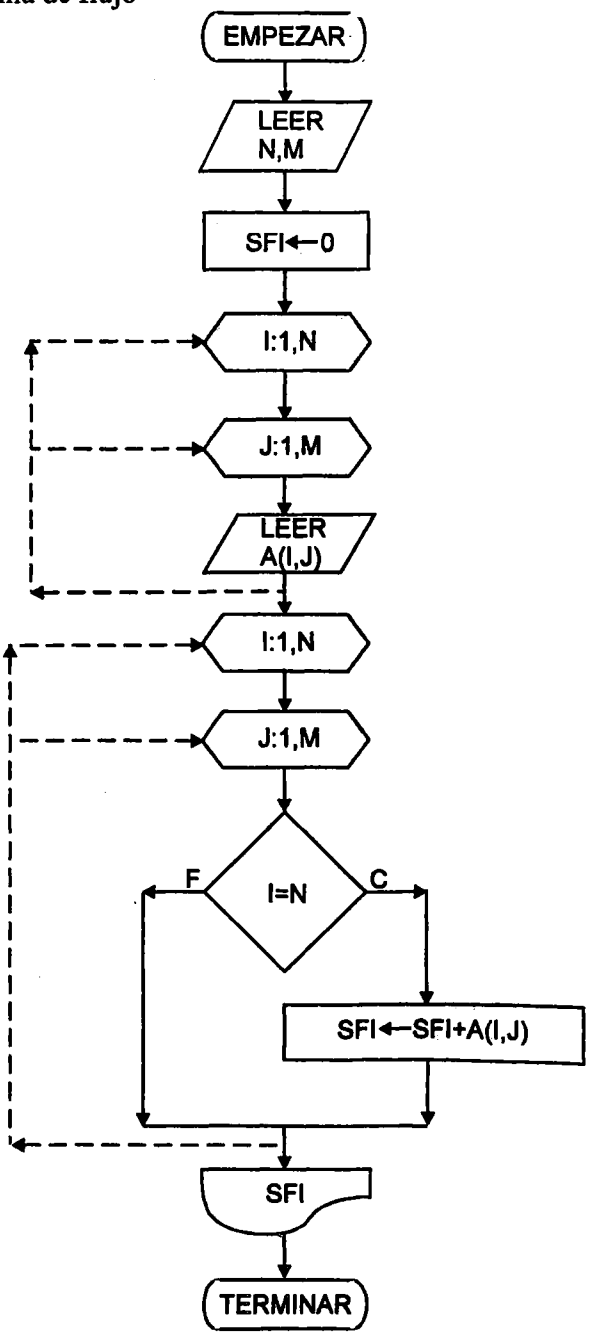
Salida : Imprimir la sumatoria de la última fila.

b) Variables a utilizar

SFI = Sumatoria de la fila

A = Nombre de la matriz
N = Número de filas de la matriz A
M = Número de columnas de la matriz A
I = Índice de filas
J = Índice de columnas

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N, M
Paso 3        SFI ← 0
Paso 4        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA M HACER
                  LEER A(I, J)
                FIN_PARA
              FIN_PARA
Paso 5        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA M HACER
                  SI I = N
                    ENTONCES SFI ← SFI + A(I, J)
                  FINSI
                FIN_PARA
              FIN_PARA
Paso 6        IMPRIMIR SFI
Paso 7      FIN
```

e) Prueba de escritorio

Se debe recordar que cuando hay anidamiento, hasta tanto el ciclo más interno no se ejecuta totalmente, no se sale al ciclo más externo.

N	M	I	J	A(I, J)	SFI	EXPLICACIÓN
3	3	1	1	A(1,1)= 6	0	En este punto se leyó la matriz por filas, es decir, completamente la primera fila, luego la segunda y así sucesivamente.
			2	A(1,2)= 9		
			3	A(1,3)= 4		
		2	1	A(2,1)= 7		
			2	A(2,2)= 2		
			3	A(2,3)= 3		
		3	1	A(3,1)=12		
			2	A(3,2)= 1		
			3	A(3,3)=40		
		1	1			Se compara I contra N, es decir, I contra 3, como no son iguales no se tiene en cuenta para la suma de la última fila.
			2			
			3			
		2	1			

N	M	I	J	A(I, J)	SFI	EXPLICACIÓN
		3	1		0+A(3,1)=0+12=12	En este momento está
			2		12+A(3,2)=12+1=13	ubicado en la última fila
			3		13+A(3,3)=13+40=53	y procede a sumar sus elementos.

EJERCICIO PROPUESTO

Repetir la prueba de escritorio hasta que se entienda.

SEGUNDA VERSIÓN DEL ALGORITMO EN SEUDOCÓDIGO:

```
Paso 1      INICIO
Paso 2        LEER N, M
Paso 3        SFI ← 0
Paso 4        PARA I DE 1 HASTA N HACER
                 PARA J DE 1 HASTA M HACER
                    LEER A(I, J)
                 FIN_PARA
              FIN_PARA
Paso 5        I ← N
Paso 6        PARA J DE 1 HASTA M HACER
                 SFI ← SFI + A(I, J)
              FIN_PARA
Paso 7        IMPRIMIR SFI
Paso 8      FIN
```

Observación

a) Cuando se desea procesar la matriz por filas, se deben hacer variar sus columnas, o sea que el índice del ciclo interno debe ser el que maneja las columnas y el índice del ciclo externo debe ser el que controla las filas.

Ejemplo:

```
INICIO
  LEER N,M {N límite de filas, M límite de columnas}
  PARA I DE 1 HASTA N HACER
     PARA J DE 1 HASTA M HACER
        LEER A(I,J)
```

```
        FIN_PARA
    FIN_PARA
FIN
```

Prueba de escritorio

N	M	I	J	A(I, J)	
2	3	1	1	A(1,1)	LECTURA DE LA PRIMERA FILA DE LA MATRIZ
			2	A(1,2)	
			3	A(1,3)	
			4		
		2	1	A(2,1)	LECTURA DE LA SEGUNDA FILA DE LA MATRIZ
			2	A(2,2)	
			3	A(2,3)	
			4		
		3			

Como se puede apreciar en la prueba de escritorio, se está leyendo la matriz por filas.

b) Cuando se desea procesar la matriz por columnas, se deben hacer variar sus filas, o sea que el índice del ciclo interno debe ser el que maneja las filas y el índice del ciclo externo debe ser el que controla las columnas.

Ejemplo:

```
INICIO
   LEER N,M {N límite de filas, M límite de columnas}
   PARA J DE 1 HASTA M HACER
      PARA I DE 1 HASTA N HACER
         LEER A(I,J)
      FIN_PARA
   FIN_PARA
FIN
```

Prueba de escritorio

N	M	I	J	A(I, J)	
2	3	1	1	A(1,1)	LECTURA DE LA PRIMERA COLUMNA DE LA MATRIZ
			2	A(2,1)	
			3		

2	1	A(1,2)	⌜LECTURA DE LA SEGUNDA
	2	A(2,2)	
	3		⌞COLUMNA DE LA MATRIZ
3	1	A(1,3)	⌜LECTURA DE LA TERCERA
	2	A(2,3)	
	3		⌞COLUMNA DE LA MATRIZ
4			

A través de la prueba de escritorio se puede observar la lectura de la matriz por columnas.

EJERCICIO

Los elementos de la matriz Z se encuentran precedidos por otro registro que indica el número de filas y el número de columnas que tiene la matriz. Leer la matriz por columnas e imprimirla por filas, además calcular la suma de los elementos de la última columna.

Solución

a) Análisis

Entrada: Inicializar la sumatoria de la columna en cero.
Leer el límite de filas y el límite de columnas.
Proceso: Para leer la matriz por columnas se deben hacer variar sus filas, es decir, el ciclo interno serán las filas.
Para calcular los elementos de la última columna se deben hacer variar sus filas.
Para imprimir la matriz por filas se deben hacer variar sus columnas, o sea que el ciclo interno serán las columnas.
Salida : Imprimir la matriz así como la suma de los elementos de la última columna.

b) Variables a utilizar

Z = Nombre de la matriz
I = Índice de filas
J = Índice de columnas
K = Límite de filas de la matriz
L = Límite de columnas de la matriz
SCO = Sumatoria de la columna.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SCO ← 0
Paso 3        LEER K, L
Paso 4        PARA J DE 1 HASTA L HACER
                 PARA I DE 1 HASTA K HACER
                    LEER Z(I, J)
                 FIN_PARA
              FIN_PARA
Paso 5        PARA J DE 1 HASTA L HACER
                 PARA I DE 1 HASTA K HACER
                    SI J = L
                       ENTONCES SCO ← SCO + Z(I, J)
                    FINSI
                 FIN_PARA
              FIN_PARA
Paso 6        PARA I DE 1 HASTA K HACER
                 PARA J DE 1 HASTA L HACER
                    IMPRIMIR Z(I, J)
                 FIN_PARA
              FIN_PARA
Paso 7        IMPRIMIR SCO
Paso 8      FIN
```

Otra solución algorítmica al ejercicio anterior es:

```
Paso 1      INICIO
Paso 2        SCO ← 0
Paso 3        LEER K, L
Paso 4        PARA J DE 1 HASTA L HACER
                 PARA I DE 1 HASTA K HACER
                    LEER Z(I, J)
                    SI J = L
                       ENTONCES SCO ← SCO + Z(I, J)
                    FINSI
                 FIN_PARA
              FIN_PARA
```

```
Paso 5          PARA I DE 1 HASTA K HACER
                    PARA J DE 1 HASTA L HACER
                        IMPRIMIR Z(I, J)
                    FIN_PARA
                FIN_PARA
Paso 6          IMPRIMIR SCO
Paso 7      FIN
```

EJERCICIO

Leer una matriz de N filas, M columnas y un valor Z. Averiguar en qué posiciones de la matriz se repite ese número Z y cuántas veces.

Solución

a) Análisis

Entrada: Iniciar el contador de valores Z en cero. Leer el límite de filas,límite de columnas y la matriz.

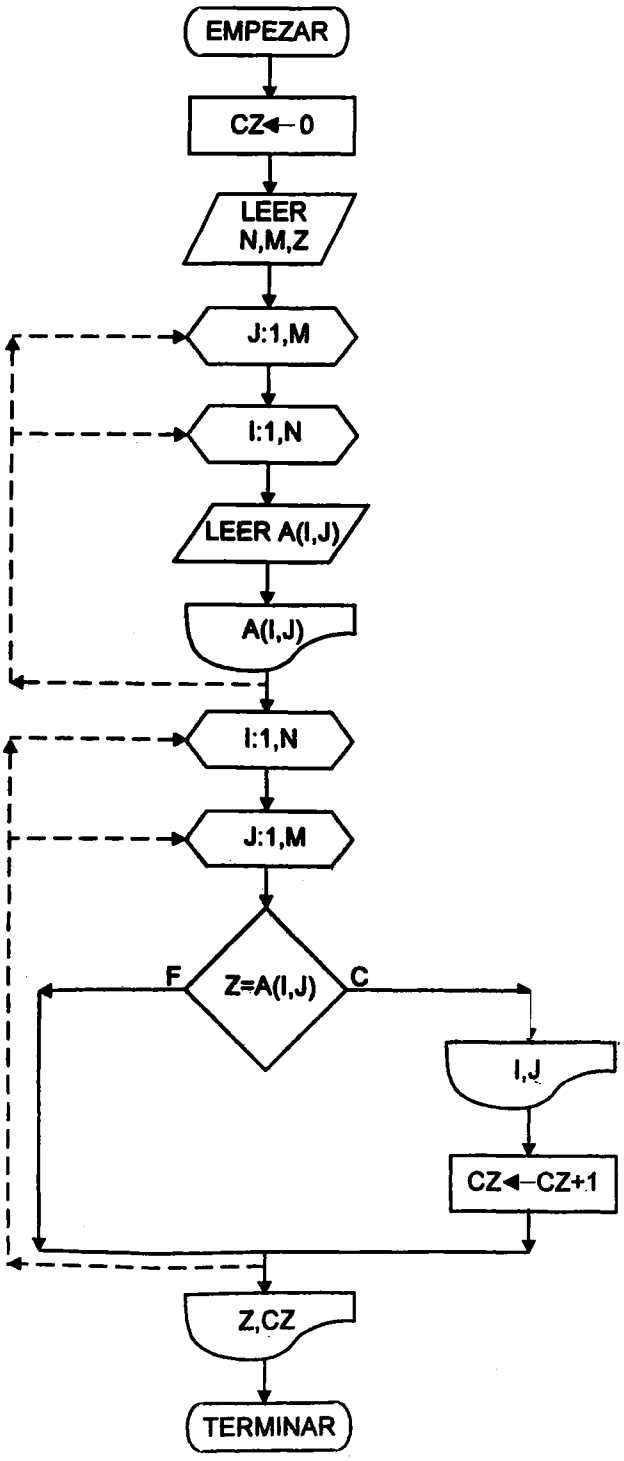
Proceso: Como se trata de averiguar en qué posiciones de la matriz se encuentra un determinado valor, entonces hay que recorrer totalmente la matriz y no importa si se hace por filas o por columnas. Se debe comparar Z contra A(I,J),si son iguales, implica que en la posición A(I,J) se encuentra repetido el valor Z y se procede a incrementar el contador de repeticiones en uno.

Salida : Imprimir la matriz, el valor Z y las posiciones en que se repite.

b) Variables a utilizar

Z = Número a ser buscado
A = Nombre de la matriz
I = Índice de las filas
J = Índice de las columnas
N = Límite de filas
M = Límite de columnas
CZ= Contador de veces que se repite Z

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        CZ ← 0
Paso 3        LEER N, M, Z
Paso 4        PARA J DE 1 HASTA M HACER
                 PARA I DE 1 HASTA N HACER
                    LEER A(I, J)
                    IMPRIMIR A(I, J)
                 FIN_PARA
              FIN_PARA
Paso 5        PARA I DE 1 HASTA N HACER
                 PARA J DE 1 HASTA M HACER
                    SI Z = A(I, J)
                       ENTONCES IMPRIMIR I, J
                                CZ ← CZ + 1
                    FINSI
                 FIN_PARA
              FIN_PARA
Paso 6        IMPRIMIR Z, CZ
Paso 7      FIN
```

PROBLEMAS RESUELTOS

EJERCICIO

Se tiene un grupo de 30 estudiantes, cada uno tomando 6 asignaturas. Calcular el promedio del grupo por cada asignatura y el promedio total. En cada registro se leen las calificaciones pertenecientes a un estudiante.

Solución

a) Análisis

Entrada: Leer la matriz de calificaciones. Inicializar la sumatoria de las calificaciones de cada materia y la sumatoria total de calificaciones en cero.

Proceso: Se debe leer la matriz de calificaciones por filas, por lo tanto, es necesario hacer variar sus columnas.

Se generan dos vectores,uno para almacenar la sumatoria de cada asignatura y el otro para almacenar el promedio por asignatura.

El promedio de cada asignatura se calcula como la sumatoria de la asignatura dividida en 30.

El promedio total se puede calcular como la sumatoria de todas las notas dividida en 180 (30 estudiantes x 6 asignaturas), ó también sumar el vector de promedios

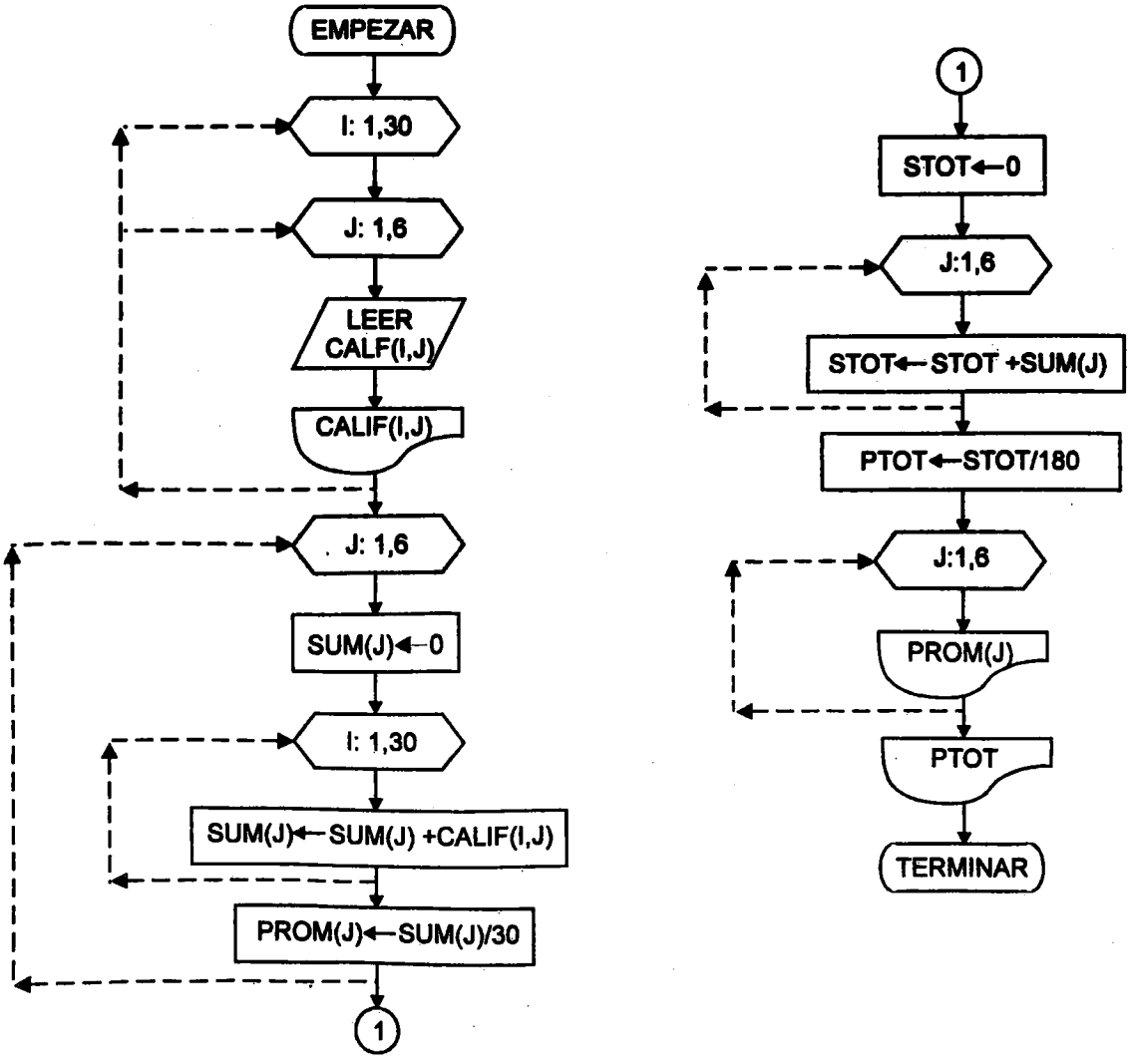
de notas y dividirlo en 6.

Salida : Imprimir la matriz de calificaciones, el vector de promedios y el promedio total.

b) Variables a utilizar

CALIF = Nombre de la matriz de calificaciones
I,J = Subíndices de la matriz de calificaciones
SUM = Sumatoria de notas de cada asignatura
PROM = Promedio de notas
STOT = Sumatoria total de notas del grupo
PTOT = Promedio total

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        PARA I DE 1 HASTA 30 HACER
                PARA J DE 1 HASTA 6 HACER
                  LEER CALIF (I, J)
                  IMPRIMIR CALIF (I, J)
                FIN_PARA
              FIN_PARA
Paso 3        PARA J DE 1 HASTA 6 HACER
                SUM (J) ← 0
                  PARA I DE 1 HASTA 30 HACER
                    SUM (J) ← SUM (J) + CALIF (I, J)
                  FIN_PARA
                  PROM (J) ← SUM (J) / 30
              FIN_PARA
Paso 4        STOT ← 0
Paso 5        PARA J DE 1 HASTA 6 HACER
                STOT ← STOT + SUM (J)
              FIN_PARA
Paso 6        PTOT ← STOT / 180
Paso 7        PARA J DE 1 HASTA 6 HACER
                IMPRIMIR PROM (J)
              FIN_PARA
Paso 8        IMPRIMIR PTOT
Paso 9      FIN
```

EJERCICIO

Leer una matriz de N filas y M columnas. Calcular la suma de los elementos de la triangular superior, la suma de los elementos de la triangular inferior y la suma de los elementos de la diagonal principal. Imprimir la matriz leída, así como todo lo calculado.

Solución

a) Análisis

Entrada: Inicializar sumatoria de la triangular inferior, sumatoria de la triangular superior, y sumatoria de la diagonal principal, en cero.
Leer el límite de filas y columnas, así como la matriz original.

Proceso: Se debe controlar si el número de filas es igual al número de columnas, para proceder a calcular las diferentes diagonales mediante comparación del índice de las filas contra el índice de las columnas.
En una matriz cuadrática de 3x3, las posiciones de la triangular inferior son: (1,2), (1,3), (2,3), de la triangular superior son: (2,1), (3,1), (3,2), de la diagonal

principal son: (1,1), (2,2), (3,3).

Salida : Imprimir la matriz, la triangular superior, la triangular inferior y la diagonal principal.

b) Variables a utilizar

B = Nombre de la matriz
I = Índice de las filas
J = Índice de las columnas
N = Límite de filas
M = Límite de columnas
STI = Sumatoria triangular inferior
STS = Sumatoria triangular superior
SDP= Sumatoria diagonal principal

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        STI ← 0
Paso 3        STS ← 0
Paso 4        SDP ← 0
Paso 5        LEER N, M
Paso 6        PARA I DE 1 HASTA N HACER
                 PARA J DE 1 HASTA M HACER
                    LEER B(I, J)
                    IMPRIMIR B(I, J)
                 FIN_PARA
              FIN_PARA
Paso 7        SI  N = M  Y  M>0
                 ENTONCES
                    PARA I DE 1 HASTA N HACER
                       PARA J DE 1 HASTA M HACER
                          SI  I < J
                             ENTONCES STI ← STI + B(I, J)
                             SINO SI I = J
                                        ENTONCES SDP ← SDP+B(I,J)
                                        SINO STS ← STS + B(I, J)
                                     FINSI
                          FINSI
                       FIN_PARA
                    FIN_PARA
                    IMPRIMIR SDP. STI. STS
              FINSI
Paso 8      FIN
```

SEGUNDA VERSIÓN DEL ALGORITMO:

```
Paso 1      INICIO
Paso 2        STI ← 0
Paso 3        STS ← 0
Paso 4        SDP ← 0
Paso 5        LEER N, M
Paso 6        PARA I DE 1 HASTA N HACER
                 PARA J DE 1 HASTA M HACER
                    LEER B(I, J)
                    IMPRIMIR B(I, J)
                 FIN_PARA
              FIN_PARA
Paso 7        SI  N = M  Y  M>0
                 ENTONCES
                    PARA I DE 1 HASTA N HACER
                       PARA J DE (I+1) HASTA M HACER
                          STI ← STI + B(I, J)
                       FIN_PARA
                       SDP ← SDP+B(I,I)
                       PARA J DE  1 HASTA  (I-1)  HACER
                          STS ← STS + B(I, J)
                       FIN_PARA
                    FIN_PARA
                    IMPRIMIR SDP, STI, STS
              FINSI
Paso 9      FIN
```

EJERCICIO

Leer 2 matrices A y B. Generar una tercera matriz que contenga la suma de éstas. Imprimir las 3 matrices.

Solución

a) Análisis

Entrada: Leer el límite de filas y columnas de la matriz A y de la matriz B. Luego leer las matrices A y B.

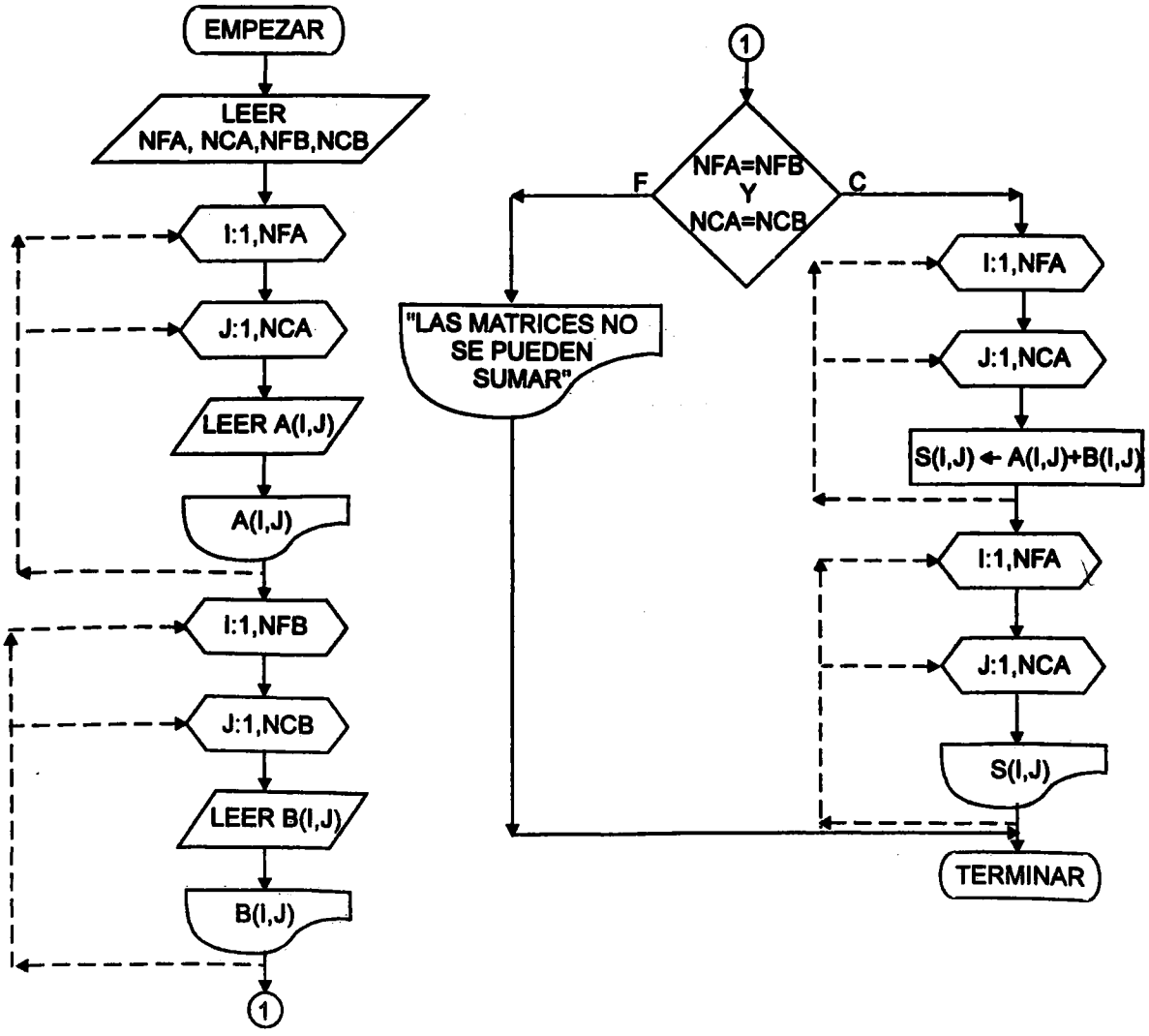
Proceso: Se debe controlar que el número de filas de la matriz A sea igual al número de filas de la matriz B y el número de columnas de la matriz A sea igual al número de columnas de la matriz B; en caso de que ésta condición no se cumpla, se imprime un mensaje indicando la inconsistencia presentada.

Salida : Imprimir las matrices originales y la matriz suma.

b) Variables a utilizar

NFA = Número de filas de la matriz A
NCA = Número de columnas de la matriz A
NFB = Número de filas de la matriz B
NCB = Número de columnas de la matriz B
A,B = Nombres matrices originales
S = Nombre de la matriz suma
I = Índice de filas
J = Índice de columnas

c) Algoritmo en diagrama de flujo



d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER NFA, NCA, NFB, NCB
Paso 3        PARA I DE 1 HASTA NFA HACER
                 PARA J DE 1 HASTA NCA HACER
                    LEER A(I, J)
                    IMPRIMIR A(I, J)
                 FIN_PARA
              FIN_PARA
Paso 4        PARA I DE 1 HASTA NFB HACER
                 PARA J DE 1 HASTA NCB HACER
                    LEER B(I, J)
                    IMPRIMIR B(I, J)
                 FIN_PARA
              FIN_PARA
Paso 5        SI (NFA = NFB) Y (NCA = NCB)
                 ENTONCES PARA I DE 1 HASTA NFA HACER
                                PARA J DE 1 HASTA NCA HACER
                                   S(I,J) ← A(I,J) + B(I,J)
                                FIN_PARA
                             FIN_PARA
                             PARA I DE 1 HASTA NFA HACER
                                PARA J DE 1 HASTA NCA HACER
                                   IMPRIMIR S(I,J)
                                FIN_PARA
                             FIN_PARA
                 SINO IMPRIMIR "las matrices no se  pueden sumar"
              FINSI
Paso 6      FIN
```

EJERCICIO

Leer 2 matrices A y B. Generar una tercera matriz que contenga el producto de éstas. Imprimir las 3 matrices.

Solución

a) Análisis

Entrada: Leer el límite de filas y columnas de la matriz A y de la matriz B. Leer las matrices A y B.

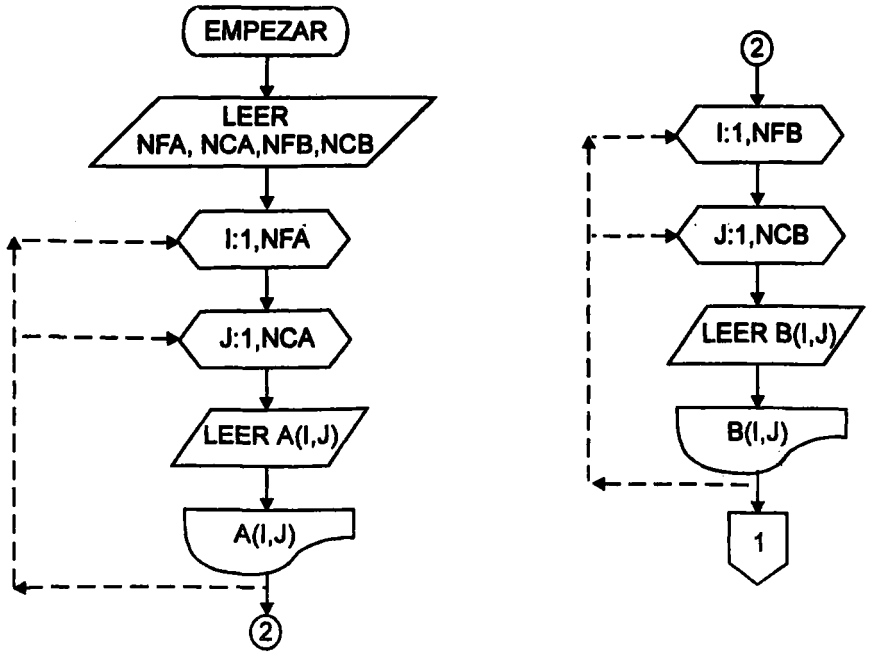
Proceso: Para poder efectuar el producto, se debe controlar que el número de columnas de la matriz A sea igual al número de filas de la matriz B o el número de columnas de la matriz B sea igual al número de filas de la matriz A,en caso de no cumplirse ésta condición se imprime un mensaje indicando la inconsistencia presentada.

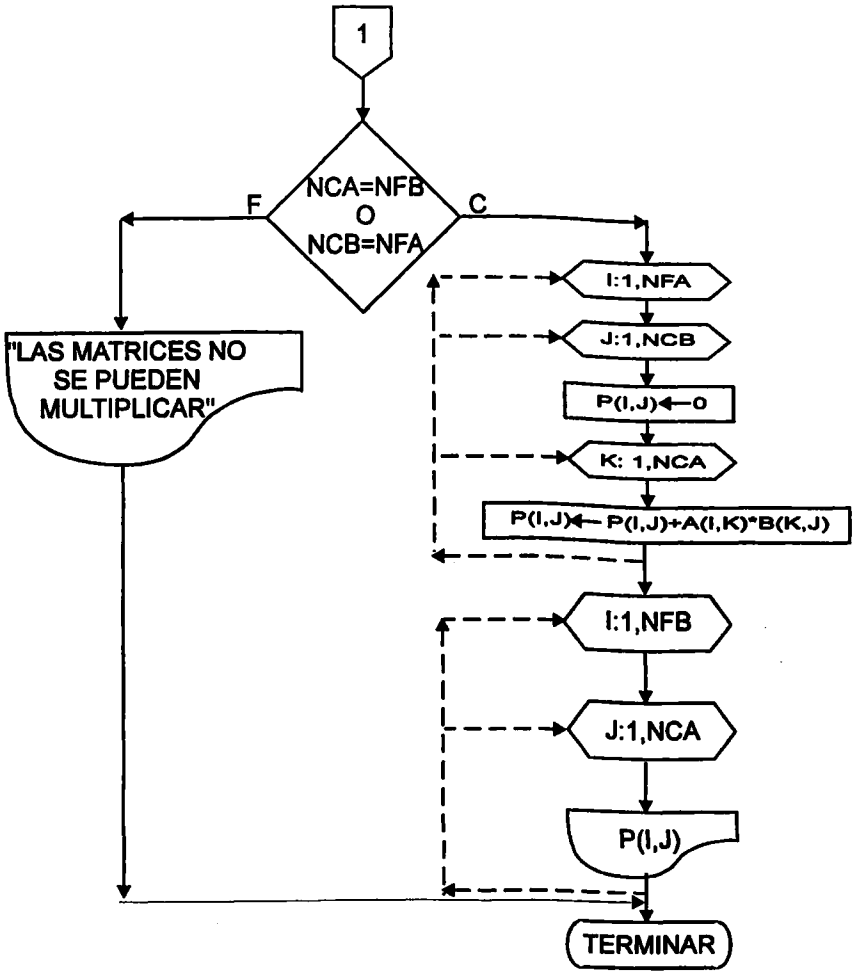
Salida : Imprimir las matrices originales y la matriz producto.

b) Variables a utilizar

NFA = Número de filas de la matriz A
NCA = Número de columnas de la matriz A
NFB = Número de filas de la matriz B
NCB = Número de columnas de la matriz B
A, B = Nombre de las matrices originales
P = Nombre de la matriz producto
I = Índice de las filas
J = Índice de las columnas

c) Algoritmo en diagrama de flujo





d) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER NFA, NCA, NFB, NCB
Paso 3        PARA I DE 1 HASTA NFA HACER
                PARA J DE 1 HASTA NCA HACER
                  LEER A(I, J)
                  IMPRIMIR A(I, J)
                FIN_PARA
              FIN_PARA
Paso 4        PARA I DE 1 HASTA NFB HACER
                PARA J DE 1 HASTA NCB HACER
                  LEER B(I, J)
                  IMPRIMIR B(I, J)
                FIN_PARA
```

```
                FIN_PARA
Paso 5          SI (NCA = NFB) O (NCB=NFA)
                    ENTONCES PARA I DE 1 HASTA NFA HACER
                                PARA J DE 1 HASTA NCB HACER
                                    P(I, J) ← 0
                                    PARA K DE 1 HASTA NCA HACER
                                        P(I,J) ← P(I,J)+A(I,K)*B(K,J)
                                    FIN_PARA
                                FIN_PARA
                            FIN_PARA
                            PARA I DE 1 HASTA NFB HACER
                                PARA J DE 1 HASTA NCA HACER
                                    IMPRIMIR P(I,J)
                                FIN_PARA
                            FIN_PARA
                    SINO IMPRIMIR "las matrices no se pueden multiplicar"
                FINSI
Paso 6          FIN
```

e) Codificación FORTRAN

```
      PROGRAM MATRICES
      DIMENSION A(10,10),B(10,10),S(10,10),P(10,10)
C     Calcular la suma y multiplicación de dos matrices A y B, las matrices resultantes
C     serán S y P
      WRITE(*,1)
    1 FORMAT(11(/),32X, 'PROGRAMA SUMA Y PRODUCTO DE MATRICES',
     *8(/),40X, 'ELABORADO POR ALONSO TAMAYO ALZATE',5(/),25X, 'UNI
     *VERSIDAD NACIONAL DE COLOMBIA' /// 20X, 'FACULTAD DE CIENCIAS
     *Y ADMINISTRACIÓN')
   23 WRITE(*,2)
    2 FORMAT('1',5(/),15X, 'NÚMERO DE FILAS DE A:'\)
      READ(*,3)NFA
    3 FORMAT(I2)
      WRITE(*,4)
    4 FORMAT(/,15X, 'NÚMERO DE COLUMNAS DE A:'\)
      READ(*,5)NCA
    5 FORMAT(I2)
      WRITE(*,6)
    6 FORMAT(/,15X, 'NÚMERO DE FILAS DE B:'\)
      READ(*,7)NFB
    7 FORMAT(I2)
```

```
      WRITE(*,8)
 8    FORMAT(/,15X, 'NÚMERO DE COLUMNAS DE B:'\)
      READ(*,9)NCB
 9    FORMAT(I2)
      WRITE(*,10)
10    FORMAT(///,15X, 'Iintroduzca los datos en el orden señalado'///)
      DO 11 I=1,NFA
      DO 11 J=1,NCA
        WRITE(*,12)I,J
12      FORMAT(10X, 'ENTRE A(',I2, ',',I2 , '):'\)
        READ(*,13)A(I,J)
13      FORMAT(F5.0)
11    CONTINUE
      DO 14 I=1,NFB
      DO 14 J=1,NCB
        WRITE(*,15)I,J
15      FORMAT(10X, 'ENTRE B(',I2, ',',I2, '):'\)
        READ(*,16)B(I,J)
16      FORMAT(F5.0)
14    CONTINUE
      IF (NFA .EQ. NFB .AND. NCA .EQ. NCB) GOTO 19
         WRITE(*,17)
17       FORMAT(10X, 'LA SUMA NO SE PUEDE EFECTUAR')
         GOTO 18
19    DO 20 I=1,NFA
      DO 20 J=1,NCA
        S(I,J)=A(I,J)+B(I,J)
20    CONTINUE
18    IF (NCA .EQ. NFB) .OR. (NCB .EQ. NFA)GOTO 21
        WRITE(*,22)
22      FORMAT(10X, 'EL PRODUCTO NO SE PUEDE EFECTUAR')
        GOTO 23
21    DO 24 I=1,NFA
      DO 24 J=1,NCA
24      P(I,J)=0
      DO 25 I=1,NFA
      DO 25 J=1,NCB
      DO 25 K=1,NFB
        P(I,J)=P(I,J)+A(I,K)*B(K,J)
25    CONTINUE
      WRITE(*,26)
26    FORMAT(/// 6X, 'MATRIZ A',38X, 'MATRIZ B')
      DO 27 I=1,NFA
```

```
      WRITE(*,28)(A,(I,J),J=1,NCA),(B(I,J),J=1,NCB)
28    FORMAT(4X,3F10.1,12X,3F10.1,/)
27  CONTINUE
    WRITE(*,29)
29  FORMAT(/// 6X, 'MATRIZ SUMA',28X, 'MATRIZ PRODUCTO')
    DO 30 I=1,NFA
      WRITE(*,31)(S(I,J),J=1,NCA),(P(I,J),J=1,NCA)
31    FORMAT(4X,3F12.1,4X,3F12.1,/)
30  CONTINUE
    STOP
    END
```

Observación:

En FORTRAN se puede leer o imprimir la matriz de tres formas:

1. *Forma explícita.*
   Explicada y analizada algorítmicamente en las páginas 267 y 268.
   Ejemplo:

```
      DO 2 Y=1,4
      DO 2 J=1,50
        WRITE(*,30)A(I,J)
    2 CONTINUE
```

2. *Forma semi-implícita.*
   Ejemplo:

```
      DO 2 J=1,4
        WRITE(*,30)(A(I,J),I=1,50)
    2 CONTINUE
```

   DO INTERNO

   En este caso se está haciendo variar el subíndice I implícitamente, por lo tanto actúa como DO interno.

3. *Forma implícita.*
   Ejemplo:

```
      WRITE(*,30)((A(I,J),I=1,50),J=1,4)
```

   DO EXTERNO
   DO INTERNO

   Esta es una modalidad totalmente implícita, donde el subíndice I actúa como DO interno y el subíndice J actúa como DO externo.

f) Codificación PASCAL

```
PROGRAM MATRICES;
USES
     CRT,PRINTER;

VAR
    A,B,S,P                    :ARRAY[1..10,1..10] OF REAL;
    NFA,NCA,NFB,NCB,I,J,K      :INTEGER;

BEGIN
  FOR I:=1 TO 5 DO
      WRITELN(LST, '');
  WRITELN(LST, '':10, 'PROGRAMA SUMA Y PRODUCTO DE MATRICES');
  FOR I:=1 TO 3 DO
      WRITELN(LST, '');
  WRITELN(LST, '':10, 'Elaborado por ALONSO TAMAYO ALZATE');
  WRITELN(LST, '');WRITELN(LST, '');
  WRITELN(LST, '':10, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
  WRITELN(LST, '');WRITELN(LST, '');
  WRITELN(LST, '':13, 'SEDE MANIZALES');
  WRITELN(LST);WRITELN(LST);
  {Calcula la suma y multiplicación de dos matrices A y B, las matrices resultantes serán
   S y P}
  WRITELN(LST, '':5, 'NÚMERO DE FILAS DE LA MATRIZ A :');
  READ(NFA);
  WRITELN(LST);WRITELN(LST);
  WRITELN(LST, '':5, 'NÚMERO DE COLUMNAS DE LA MATRIZ A :');
  READ(NCA);
  WRITELN(LST);WRITELN(LST);
  WRITELN(LST, '':5, 'NÚMERO DE FILAS DE LA MATRIZ B :');
  READ(NFB);
  WRITELN(LST);WRITELN(LST);
  WRITELN(LST, '':5, 'NÚMERO DE COLUMNAS DE LA MATRIZ B :');
  READ(NCB);
  WRITELN(LST);WRITELN(LST);
  FOR I:=1 TO NFA DO
      BEGIN
        FOR J:=1 TO NCA DO
            READ(A[I,J])
      END;
  FOR I:=1 TO NFB DO
      BEGIN
        FOR J:=1 TO NCB DO
```

```
            READ(B[I,J])
        END;
    WRITELN(LST);WRITELN(LST);WRITELN(LST);
    WRITELN(LST, '':6, 'MATRIZ A');
    FOR I:=1 TO NFA DO
        FOR J:=1 TO NCA DO
            WRITE(LST, '':4,A[I,J]:4:0):
    WRITELN(LST);WRITELN(LST);
    WRITELN(LST, '':6, 'MATRIZ B');
    FOR I:=1 TO NFB DO
        FOR J:=1 TO NCB DO
            WRITE(LST, '':4,B[I,J]:4:0);
    IF (NFA=NFB) AND (NCA=NCB) THEN
        BEGIN
            FOR I:=1 TO NFA DO
                FOR J:=1 TO NCA DO
                    S[I,J]:=A[I,J]+B[I,J];
        END;
            WRITELN(LST);WRITELN(LST);
            WRITELN(LST, '':6, 'MATRIZ SUMA');
            FOR I:=1 TO NFA DO
              FOR J:=1 TO NCA DO
                  WRITE(LST, '':4,S[I,J]:4:0)
        ELSE
            WRITELN(LST, '':10, 'LA SUMA NO SE PUEDE EFECTUAR');
    IF (NCA=NFB) OR (NCB=NFA) THEN
        BEGIN
            FOR I:=1 TO NFA DO
                FOR J:=1 TO NCA DO
                    P[I,J]:= 0;
            FOR I:=1 TO NFA DO
                FOR J:=1 TO NCB DO
                    FOR K:=1 TO NFB DO
                        P[I,J]:=P[I,J]+A[I,K]*B[K,J];
            WRITELN(LST);WRITELN(LST);
            WRITELN(LST, '':6, 'MATRIZ PRODUCTO');
            FOR I:=1 TO NFA DO
                FOR J:=1 TO NCA DO
                    WRITE(LST, '':4,P[I,J]:4:0):
        END;
    ELSE
        WRITELN(LST, '':10, 'EL PRODUCTO NO SE PUEDE EFECTUAR');
END.
```

EJERCICIO

Se tiene una matriz cuadrática de N filas y N columnas. Encontrar el elemento mayor de cada fila y de cada columna, e imprimirlos así como la matriz leída.

Solución

a) Análisis

Entrada: Leer el límite de filas y columnas de la matriz y luego leer la matriz original.
Proceso: Se asume que la primera posición de la matriz es la mayor de la primera fila y se compara contra cada uno de los demás elementos de esa misma fila, en caso de encontrar un elemento mayor al asumido inicialmente entonces es reemplazado. Al finalizar el análisis de la fila, el elemento mayor es almacenado en un vector de elementos mayores por fila. Igual procedimiento se hace con cada una de las demás filas y columnas de la matriz.
Salida : Imprimir la matriz original y los vectores de elementos mayores de las filas y de elementos mayores de las columnas.

b) Variables a utilizar

N = Número de filas y columnas de la matriz
B = Nombre de la matriz original
FIMA = Nombre del vector de filas mayores
COMA = Nombre del vector de columnas mayores
I = Índice de filas
J = Índice de columnas

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA N HACER
                  LEER B(I, J)
                  IMPRIMIR B(I, J)
                FIN_PARA
              FIN_PARA
Paso 4        PARA I DE 1 HASTA N HACER
                FIMA(I) ← B(I, 1)
                PARA J DE 1 HASTA N HACER
                  SI FIMA(I) < B(I, J)
                    ENTONCES FIMA(I) ← B(I, J)
```

```
                    FINSI
                FIN_PARA
            FIN_PARA
Paso 5      PARA J DE 1 HASTA N HACER
                COMA(J) ← B(1, J)
                PARA I DE 1 HASTA N HACER
                    SI COMA(J) < B(I, J)
                        ENTONCES COMA(J) ← B(I, J)
                    FINSI
                FIN_PARA
            FIN_PARA
Paso 6      PARA I DE 1 HASTA N HACER
                IMPRIMIR FIMA(I), COMA(I)
            FIN_PARA
Paso 7    FIN
```

EJERCICIO

Se tiene una matriz de N filas y M columnas y se desea calcular el promedio de cada fila así como el promedio de cada columna. Imprimir la matriz leída y los promedios calculados.

Solución

a) Análisis

Entrada: Leer el límite de filas, el límite de columnas y la matriz.
Proceso: Se calculan los promedios y almacenan en dos vectores, uno para los promedios de las filas y el otro para los promedios de las columnas.
Se inicializa la sumatoria de las filas y sumatoria de las columnas en cero por cada ciclo ejecutado.
Salida : Imprimir la matriz, el vector de promedios de filas y el vector de promedios de columnas.

b) Variables a utilizar

B = Nombre de la matriz
I = Índice de las filas
J = Índice de las columnas
N = Límite de las filas
M = Límite de las columnas
PROFI = Vector promedio de las filas
PROCO= Vector promedio de las columnas.

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        LEER N, M
Paso 3        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA M HACER
                  LEER A(I, J)
                  IMPRIMIR A(I, J)
                FIN_PARA
              FIN_PARA
Paso 4        PARA I DE 1 HASTA N HACER
                PROFI(I) ← 0
                PARA J DE 1 HASTA M HACER
                  PROFI(I) ← PROFI(I)+A(I, J)/M
                FIN_PARA
                IMPRIMIR PROFI(I)
              FIN_PARA
Paso 5        PARA J DE 1 HASTA M HACER
                PROCO(J) ← 0
                PARA I DE 1 HASTA N HACER
                  PROCO(J) ← PROCO(J)+A(I, J)/N
                FIN_PARA
                IMPRIMIR PROCO(J)
              FIN_PARA
Paso 6      FIN
```

EJERCICIO

Leer una matriz de N filas y M columnas. Calcular la suma de los elementos de la 1ª fila, la suma de los elementos de la 1ª columna, la suma de los elementos de la diagonal principal y la suma de los elementos de la diagonal secundaria. Imprimir la matriz leída y todo lo calculado.

Solución

a) Análisis

Entrada: Leer el límite de filas, límite de columnas y la matriz.
Inicializar en cero las sumatorias de, la primera fila, la primera columna, la diagonal principal y la diagonal secundaria.

Proceso: Controlar si el número de filas es igual al número de columnas para poder hallar la diagonal principal y diagonal secundaria, si no lo son, se debe calcular la

sumatoria de los elementos de la primera fila y la suma de los elementos de la primera columna.
Cuando la matriz es de 3 filas por 3 columnas, la diagonal secundaria se calcula sumando los elementos de las posiciones (1,3), (2,2), (3,1). La diagonal principal se calcula sumando los elementos de las posiciones (1,1), (2,2), (3,3).

Salida : Imprimir la matriz leída y todo lo calculado.

b) Variables a utilizar

SF = Sumatoria de la primera fila
SC = Sumatoria de la primera columna
SDP = Sumatoria de la diagonal principal
SDS = Sumatoria de la diagonal secundaria
N = Límite de filas de la matriz
M = Límite de columnas de la matriz
MAT = Nombre de la matriz
I = Índice de las filas
J = Índice de las columnas

c) Algoritmo en seudocódigo

```
Paso 1      INICIO
Paso 2        SF ← 0
Paso 3        SC ← 0
Paso 4        SDP ← 0
Paso 5        SDS ← 0
Paso 6        LEER N, M
Paso 7        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA M HACER
                  LEER MAT(I, J)
                  IMPRIMIR MAT(I, J)
                FIN_PARA
              FIN_PARA
Paso 8        PARA I DE 1 HASTA N HACER
                PARA J DE 1 HASTA M HACER
                  SI J = 1
                    ENTONCES SC ← SC + MAT(I, J)
                  FINSI
                  SI I = 1
                    ENTONCES SF ← SF + MAT(I, J)
                  FINSI
```

```
                FIN_PARA
            FIN_PARA
Paso 9      IMPRIMIR SC, SF
Paso 10     SI N = M
                ENTONCES PARA I DE 1 HASTA N HACER
                                PARA J DE 1 HASTA M HACER
                                    SI  I = J
                                        ENTONCES SDP ← SDP+MAT(I,J)
                                    FINSI
                                    SI J = M
                                        ENTONCES SDS ← SDS+MAT(I,J)
                                                    M ← M - 1
                                    FINSI
                                FIN_PARA
                            FIN_PARA
                            IMPRIMIR SDP, SDS
                SINO IMPRIMIR "la matriz no es cuadrática".
            FINSI
Paso 11     FIN
```

Otra forma de calcular la diagonal secundaria consiste en comparar la sumatoria de los valores de las variables índice de las estructuras PARA contra el límite de filas o columnas de la matriz aumentada en una unidad, en caso de ser iguales se almacena el elemento de la matriz que se encuentre en esa posición, así:

```
Paso 1      INICIO
Paso 2          LEER N,M
Paso 3          SI N=M
                    ENTONCES SDS ← 0
                                PARA I DE 1 HASTA N HACER
                                    PARA J DE 1 HASTA M HACER
                                        SI (I+J)=(M+1)
                                            ENTONCES SDS ← SDS+A(I,J)
                                        FINSI
                                    FIN_PARA
                                FIN_PARA
                                IMPRIMIR "La diagonal secundaria es :", SDS
                    SINO IMPRIMIR "La matriz no es cuadrática"
                FINSI
Paso 4      FIN
```

PROBLEMAS PROPUESTOS

1- **INICIO**
  PARA I DE 1 HASTA 3 HACER
    PARA J DE 1 HASTA 6 HACER
      K(I,J) ← I
    FIN_PARA
  FIN_PARA
FIN

Qué valores tendrá la matriz K al culminar los ciclos ?

2- Encontrar el elemento mayor y el menor de una matriz. Imprimirlos.

3- Leer una matriz cuadrática e imprimir la matriz leída y la matriz identidad.
La matriz identidad es una matriz cuadrática en donde los elementos de la diagonal principal son iguales a uno y los demás elementos son iguales a cero.

4- Leer una matriz de N x M. Hallar e imprimir la transpuesta de la matriz. La matriz transpuesta convierte las filas de la matriz original en columnas y las columnas en filas.

5- Leer una matriz R de N filas y M columnas. Calcular las sumas de cada fila y almacenarlas en un primer vector, calcular las sumas de cada columna y almacenarlas en un segundo vector. Imprimir la matriz leída y los vectores hallados. Calcular además la suma total de cada uno de los vectores e imprimirlas.

EJERCICIOS COMPILADOS

EJERCICIO Nro. 3.1

Para un lote de M vendedores se desea calcular la comisión a pagar por concepto de ventas, de la siguiente forma: 10% de comisión para ventas menores o iguales a $100.000 y 20 % para ventas superiores a $100.000; al final se desea un reporte por vendedor que contenga nombre del vendedor, sueldo, total ventas realizadas, comisión y total a pagar. Realizar el anterior ejercicio mediante la aplicación de arreglos.

```
Program Ventas;
{Programa que calcula la comisión para M vendedores}
Uses crt;
Type
   Vector     =Array[1..50] of real;
   Nom        =Array[1..50] of string[30];
```

```
Var
   I,M,K                          :Integer;
   Sueldo,Vn,Comision,Total_pagar:Vector;
   Nombre                         :Nom;
BEGIN
   Clrscr;
   Gotoxy(19,3);
   Write('UNIVERSIDAD NACIONAL DE COLOMBIA');
   Gotoxy(26,4);
   Write('SEDE MANIZALES');
   Delay(2000);
   Clrscr;
   Write('NÚMERO DE VENDEDORES :');
   Readln(M);
   Writeln;Writeln;
   For I:=1 to M do
       Begin
          Clrscr;
          Gotoxy(6,4);
          Write('NOMBRE :');
          Readln(Nombre[I]);
          Gotoxy(6,6);
          Write('SUELDO :');
          Readln(Sueldo[I]);
          Gotoxy(6,8);
          Write('VENTAS :');
          Readln(Vn[I]);
       End;
   For I:=1 to M do
       If Vn[I]<=100000.0 Then
          Comision[I]:=Vn[I]*0.10
       Else
          Comision[I]:=Vn[I]*0.20;
   For I:=1 to M do
       Total_pagar[I]:=Sueldo[I]+Comision[I];
   K:=79;
   For I:=2 to K do
       Begin
          Gotoxy(I,1)
          Write('-');
       End;
   Gotoxy(31,2);
   Write('DEPARTAMENTO DE CALDAS');
```

```
    K:=79;
    For I:=2 to K do
        Begin
            Gotoxy(I,4)
            Write('-');
        End;
    K:=24;
    For I:=1 to K do
        Begin
            Gotoxy(2,I)
            Write('¦');
        End;
    K:=24;
    For I:=1 to K  do
        Begin
            Gotoxy(79,I)
            Write('¦');
        End;
    Gotoxy(5,6);
    Write('NOMBRE');
    Gotoxy(23,6);
    Write('SUELDO');
    Gotoxy(36,6);
    Write('VENTAS');
    Gotoxy(46,6);
    Write('COMISIÓN');
    Gotoxy(57,6);
    Write('TOTAL A PAGAR');
    K:=6;
    For I:=1 to M do
        Begin
            K:=K+2
            Gotoxy(6,K)
            Write(Nombre[I]);
        End;
    K:=6;
    For I:=1 to M do
        Begin
            K:=K+2
            Gotoxy(17,K)
            Write(Sueldo[I]:12:2,Vn[I]:12:2);
            Write(Comision[I]:12:2,Total_pagar[I]:12:2);
        End;
END.
```

EJERCICIO Nro. 3.2

Leer dos vectores A y B, formar un nuevo vector cuyos elementos comprendan la unión de los vectores leídos. El vector unión deberá contener todos los elementos del vector A y del vector B, excepto aquellos elementos que son comunes. Imprimir los vectores originales y el vector unión.

```
Program Union;
Uses Crt,Printer;
Var
   I,J,N,M,D  :Integer;
   A,B,C,E    :Array[1..30] of Integer;

BEGIN
   Writeln(lst);Writeln(lst);
   Writeln(lst,'                    UNIVERSIDAD NACIONAL DE COLOMBIA');
   Writeln(lst,'                          SEDE MANIZALES');
   Writeln(lst);
   For I:=1 to 7 do
       Writeln(lst);
   Writeln(lst,'     PROGRAMA QUE CALCULA LA UNIÓN DE DOS VECTORES');
   For I:=1 to 9 do
       Writeln(lst);
   Writeln(lst,'                    ALONSO TAMAYO ALZATE');
   Clrscr;
   For I:=1 to 10 do
       Writeln(lst);
   Writeln('ELEMENTOS DEL VECTOR A :');
   Readln(N);
   Writeln(lst,'ELEMENTOS DEL VECTOR B :');
   Readln(M);
   Writeln;
   For I:=1 to N do
       Begin
          Write(lst,'EL VECTOR A',I, ':');
          Readln(A[I]);
       End;
   Writeln;
   For J:=1 to M do
       Begin
          Write(lst,'EL VECTOR B',J, ':');
          Readln(B[J]);
       End;
   Writeln;
```

```
   For I:=1 to N do
       C[I]:= A[I];
   For J:=1 to M do
       C[J+N]:= B[J];
   {La unión de los dos vectores}
   For I:=1 to N+M-1 do
       For J:=I+1 to N+M do
          If  C[I] = C[J] Then
              E[I]:=I;
   For I:=1 to M+N do
       If  I <> E[I]  Then
        E[I]:=9999;
   Writeln(lst);
   Writeln(lst,'ELEMENTOS DEL VECTOR A  :',N);
   For I:=1 to N do
       Writeln(lst,'EL VECTOR A ',I, ':', A[I]);
   Writeln(lst);
   Writeln(lst,'ELEMENTOS DEL VECTOR B  :',M);
   For J:=1 to M do
       Writeln(lst,'EL VECTOR B ',J, ':', B[J]);
   Writeln(lst);
   Writeln(lst,'ESTE ES EL VECTOR UNIÓN  :');
   For D:=1 to M+N do
       If  D <> E[D]  Then
         Writein(lst,          C:', C[D]);
   Readln;
END.
```

EJERCICIO Nro. 3.3

Leer dos vectores A y B, formar un nuevo vector cuyos elementos comprendan la intersección de los vectores leídos. El vector intersección deberá contener todos los elementos que son comunes tanto al vector A como al vector B. Imprimir los vectores originales y el vector intersección.

```
Program Interseccion;

Uses Crt,Printer;

Const
   V1=100;
   V2=100;
```

```
   V3=200;
Var
   A                 :Array [1..V1] of real;
   B                 :Array [1..V2] of real;
   C                 :Array [1..V3] of real;
   I,J,M,N,NC,Q,R    :Integer;

BEGIN
   Clrscr;
   For I:=1 to 4 do
       Writeln(lst);
   Writeln(lst, '':22, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
   Writeln(lst, '':31, 'SEDE MANIZALES');
   For I:=1 to 4 do
       Writeln(lst);
   Writeln(lst, '':27,'PROGRAMA INTERSECCIÓN');
   Writeln(lst, '':31, 'ELABORADO POR');
   Writeln(lst, '':31, 'ALONSO TAMAYO ALZATE');
   For I:=1 to 4 do
       Writeln(lst);
   Delay(10000);
   Clrscr;
   Repeat
       Write(lst, 'TECLEE EL LÍMITE DEL VECTOR A  ');
       Readln(N);
       If  N<= 0  Then
                 Writeln(lst, 'ERROR EN DATOS, INTENTE DE NUEVO');
       Write(lst, 'TECLEE EL LÍMITE DEL VECTOR B ');
       Readln(M);
       If  M<=0 Then
                 Writeln(lst, 'ERROR EN DATOS, INTENTE DE NUEVO)
   Until (N>0) and (M>0);
   Clrscr;
   Writeln(lst, 'EL VECTOR A ES :');
   For J:=1 to N do
       Begin
          Write(lst, 'EL ELEMENTO A(',J, ') ES :');
          Read(A[J])
       End;
   Writeln(lst);
   Writeln(lst, 'EL VECTOR B ES :');
   For I:=1 to M do
       Begin
          Write(lst, 'EL ELEMENTO B(',I, ') ES :');
```

```
            Read(B[I])
        End;
    Writeln;
    Writeln(lst, 'LOS ELEMENTOS COMUNES EN A Y B SON :');
    NC:=0;
    For I:=1 to M do
        For J:=1 to N do
            Begin
                If A[J]= B[I] Then
                    Begin
                        NC:=NC+1;
                        C[NC]:= A[J]
                    End
            End;
    Clrscr;
    Writeln(lst, 'EL VECTOR A ES :');
    For J:=1 to N do
        Write(A[J]:8:2);
    Writeln(lst):Writeln(lst);
    Writeln(lst, 'EL VECTOR B ES :');
    For I:=1 to M do
        Write(B[I]:8:2);
    Writeln(lst);Writeln(lst);
    Writeln(lst, 'LA INTERSECCIÓN ENTRE A Y B ES :');
    For J:=1 to NC do
        Write(C[J]:8:2);
    Readln;
    Delay(10000);
End.
```

EJERCICIO Nro. 3.4

Leer un vector de N elementos y hallar su mediana y promedio. Mediana es el elemento que se encuentra en la mitad, luego de que el vector ha sido ordenado.

```
Program Mediana;
Uses Crt,Printer;
Var
   J,N,L              :Integer;
   F,PROM,MED,E       :Real;
   X                  :Array [1..100] of real;
BEGIN
```

```
Clrscr;
For J:=1 to 7 do
    Writeln(lst);
Writeln(lst, '':30, 'UNIVERSIDAD NACIONAL');
Writeln(lst, '':33, 'SEDE MANIZALES');
For J:=1 to 16 do
    Writeln(lst);
Writeln(lst, '':24, 'PROGRAMA QUE CALCULA LA MEDIANA Y PROMEDIO DE
       UN VECTOR');
For J:=1 to 13 do
    Writeln(lst);
Writeln(lst, '':23, 'ELABORADO POR: ALONSO TAMAYO ALZATE');
Clrscr;
For J:=1 to 5 do
    Writeln(lst);
Repeat
    Writeln(lst, '':5, 'DIGITE EL LÍMITE DE DATOS :');
    Readln(N);
    Writeln(lst, '':5,N);
    If N <= 0 Then
       Writeln(lst, '':5, 'ERROR EN DATOS');
    Writeln(lst);
Until N>0;
F:=0;
For J:=1 to N do
    Begin
       Writeln(lst, '':5, 'TECLEE LOS VALORES DEL VECTOR');
       Readln(X [J]);
       F:=F+ X[J];
       Writeln(lst, '':5, X[J]:4:3);
    End;
For J:=1 to N-1 do
    Begin
       For L:=J+1 to N do
           If X[J] > X[L] Then
              Begin
                 E:= X[J];
                 X[J]:= X[L];
                 X[L]:=E;
              End;
    End;
L:=N div 2;
If (N mod 2) = 0 Then
```

```
        Med:= (X[L]+ X[L+1])/2
    Else Med:= X[L+1];
    Prom:=F/N;
    Writeln(lst, '':5, 'EL PROMEDIO DEL VECTOR ES :', Prom:4:3);
    Writeln(lst, '':5, 'LA MEDIANA DEL VECTOR ES :',Med:4:3);
    Readln;
End.
```

EJERCICIO Nro. 3.5

Leer un arreglo de N elementos que se encuentra ordenado ascendentemente. Leer adicionalmente un valor e insertarlo dentro del vector ordenado, de tal forma que mantenga su ordenamiento. Si el valor ya forma parte del vector, no se puede insertar. Imprimir el vector original, el elemento a ser insertado y el vector resultante.

```
      PROGRAM INSER
      DIMENSION A(100),B(100)
      WRITE(*,100)
100   FORMAT(1H0,15(/),T45, 'Universidad Nacional de Colombia'//T52, 'Sede
     *Manizales'///T54, 'Elaborado por ALONSO TAMAYO ALZATE',//)
      WRITE(*,102)
102   FORMAT(1H1,10(/),40X, 'Vector Original')
      WRITE(*,105)
105   FORMAT(1X, 'Entre el número máximo de elementos (N) que contiene el vector,
     *con formato I3')
 10   READ(*,20)N
 20   FORMAT(I3)
      IF (N-100)25,25,35
 25   READ(*,30)(A(I),I=1,N)
 30   FORMAT(30F4.1)
      WRITE(*,103)
103   FORMAT(2X, 'Entre el elemento a insertar con formato F4.1')
      READ(*,40)X
 40   FORMAT(F4.1)
      WRITE(*,200)X
200   FORMAT(5(/),20X, 'El elemento a insertar es  ',3X,F4.1)
      DO 1 I=1,N
  1     B(I)=A(I)
      DO 5 I=1,N
        IF (X-A(I))50,60,70
 50     IF (X-A(I+1))80,80,90
 80     GOTO 5
 90     K=I+1
```

```
      GOTO 7
 5  CONTINUE
    K=I+1
 7  M=K-1
    DO 3 I=1,M
 3     B(I)=A(I)
    B(K)=X
    MI=K+1
    MF=N+1
    DO 4 I=MI,MF
 4     B(I)=A(I-1)
    K=N+1
    GOTO 65
 60 K=N
 65 WRITE(*,110)(A(I),I=1,N)
110 FORMAT(///(20X,10F6.1))
    WRITE(*,115)(B(I),I=1,K)
115 FORMAT(/////40X, 'Vector Resultante'///(20X,10F6.1))
    STOP
 70 K=1
    B(K)=X
    J=N+1
    DO 2 I=2,J
 2     B(I)=A(I-1)
    GOTO 65
 35 WRITE(*,120)
120 FORMAT(10X, 'El valor de N es mayor que 100, entre otro valor,con formato I3')
    GOTO 10
    END
```

EJERCICIO Nro. 3.6

BÚSQUEDA BINARIA

Consiste en buscar un elemento en un arreglo, el cual debe estar ordenado.
Se toma el valor a ser buscado y se compara contra el elemento que se encuentra en la posición central del arreglo, una vez que éste ha sido ordenado, si es igual, se ha encontrado el elemento buscado y termina la búsqueda, sino se determina si el elemento buscado se encuentra en la primera o segunda mitad del arreglo y se realiza igual proceso entrando a comparar el elemento buscado contra el elemento central de la mitad donde se determinó su ubicación.

Ejemplo:

Sean los elementos:

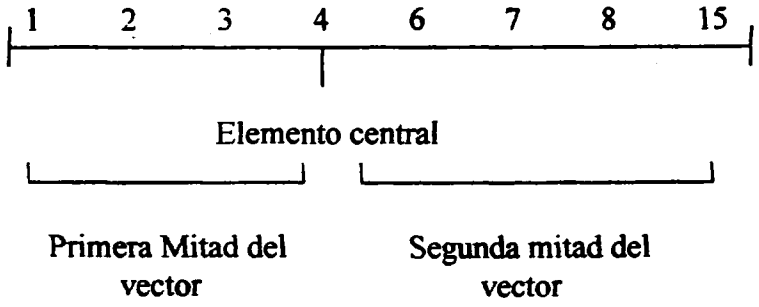
1 7 2 8 4 3 6 15

y 7 el valor buscado.

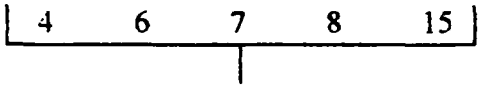
Se ordena el vector y tenemos:

1 2 3 4 6 7 8 15

Se compara 7 contra el elemento central del vector. Como no es posible ubicar el elemento central, se toma el inmediatamente anterior que es 4.



Como 7 es mayor que el elemento central, se procede a buscar el elemento en la segunda mitad del vector, eliminando la primera mitad.



Elemento central

Nuevamente se determina el elemento central, que en este caso es 7 y se compara contra el valor buscado, como son iguales, el elemento ha sido encontrado.

```
Program Binario;
Uses Crt,Printer;
Var
   J,I,Primer,Central,Ultimo,Limite,F,Numero :Integer;
   Buscado    :Boolean;
   Aux        :Array[1..1] of real;
   Vector     :Array[1..1000] of real;

BEGIN
   Clrscr;
   For J:=1 to 12 do
       Writeln(lst);
```

```
Writeln(lst, '':31, 'BÚSQUEDA BINARIA');
For J:=1 to 10 do
    Writeln(lst);
Writeln(lst,'':23, 'ELABORADO POR ALONSO TAMAYO ALZATE');
For J:=1 to 10 do
    Writeln(lst);
Writeln(lst, '':24, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
Clrscr;
Write(lst, '':10,'TECLEE LÍMITE DEL VECTOR :');
Readln(Limite);
For J:=1 to Limite do
    Begin
       Write(lst, '':10,'TECLEE NÚMERO DE LA POSICIÓN :',J:2, ':');
       Readln(Vector[J]);
    End;
F:=Limite;
{Ordenamiento del vector}
For J:=1 to (F-1) do
    For I:=(J+1) to F do
       If Vector[J] >= Vector[I] Then
          Begin
             Aux[1]:=Vector[J];
             Vector[J]:=Vector[I];
             Vector[I]:=Aux[1];
          End;
For I:=1 to F do
    Write(Vector[I]:5:0);
Primer:=1;
Ultimo:=F;
Buscado:=False;
Writeln(lst);
Write(lst,'TECLEE EL NÚMERO A SER BUSCADO :');
Readln(Numero);
While (Primer <= Ultimo) and (Buscado = False) do
    Begin
       {Determinación del elemento central}
       Central:=(Primer + Ultimo) div 2;
       {Búsqueda del elemento en la mitad del vector}
       If Numero = Vector[Central] Then
          Buscado:=True
       Else
          If Numero > Vector[Central] Then
             Primer:=Central + 1
```

```
            Else
               Ultimo:=Central - 1;
        End;
   If Buscado=True Then
        Writeln(lst, '':10,Numero, 'NÚMERO EXISTENTE EN EL VECTOR')
   Else
        Writeln(lst, '':10,Numero, 'NÚMERO INEXISTENTE');
END.
```

EJERCICIO Nro. 3.7

Leer una matriz de N filas y M columnas, encontrar el elemento mayor y el elemento menor de la matriz.

```
Program Matriz;
{Encuentra el elemento mayor y menor de una matriz}
Uses crt;
Var
   J,I,T,N,M,TEMP  :Integer;
   Ordenado        :Boolean;
   X               :Array[1..10,1..10] of integer;

BEGIN
   Clrscr;
   Gotoxy(24,3);
   Write('UNIVERSIDAD NACIONAL DE COLOMBIA');
   Gotoxy(29,5);
   Write('SEDE MANIZALES');
   Gotoxy(20,10);
   Write('FACULTAD DE CIENCIAS Y ADMINISTRACIÓN');
   Gotoxy(30,19);
   Write('ELABORADO POR ALONSO TAMAYO ALZATE');
   Readln;
   Clrscr;
   Write('TECLEE EL NÚMERO DE FILAS :');
   Readln(N);
   Write('TECLEE EL NÚMERO DE COLUMNAS :');
   Readln(M);
   For J:=1 to N do
      For I:=1 to M do
         Begin
            Write('X[',J, ',',I, ']:');
            Readln(X[J,I])
```

```
            End;
    For J:=1 to N do
        Repeat
            Ordenado:=True;
            For T:=1 to M-1 do
                If X[J,T] > X[J,T+1] Then
                    Begin
                        Temp:=X[J,T];
                        X[J,T]:=X[J,T+1];
                        X[J,T+1]:=Temp;
                        Ordenado:=False
                    End;
        Until (Ordenado=True);
    For J:=1 to M do
        Repeat
            Ordenado:=True;
            For  I:=1 to N-1 do
                If X[I,J] > X[I+1,J] Then
                    Begin
                        Temp:=X[I,J];
                        X[I,J]:=X[I+1,J];
                        X[I+1,J]:=Temp;
                        Ordenado:=False
                    End;
        Until (Ordenado=True);
    Write('EL ELEMENTO MENOR ES :',X[1,1]);Writeln;
    Write('EL ELEMENTO MAYOR ES :',X[N,M];
END.
```

EJERCICIO Nro. 3.8

Leer una matriz de N filas y M columnas. Hallar e imprimir la transpuesta de la matriz.

```
Program Trans;
{Programa que lee una matriz M*N, calcula su transpuesta e imprime ambas matrices}
Uses crt;
Var
   M,N,F,R    :Integer;
   A          :Array[1..50,1..50] of real;

BEGIN
   Clrscr;
```

```
Gotoxy(21,2);
Write('PROGRAMA MATRIZ TRANSPUESTA');
Gotoxy(33,10);
Write('ELABORADO POR ALONSO TAMAYO ALZATE');
Gotoxy(20,20);
Write('UNIVERSIDAD NACIONAL DE COLOMBIA');
Gotoxy(27,21);
Write('SEDE MANIZALES');
Clrscr;
Repeat
    Write('ENTRE EL NÚMERO DE FILAS DE LA MATRIZ :');
    Readln(N);
    If (N < 0) Then
       Writeln('ERROR EN LA ENTRADA DEL DATO, REINTENTE');
Until (N>0);
Writeln;
Repeat
    Write('AHORA ENTRE EL NÚMERO DE LAS COLUMNAS :');
    Readln(M);
    If (M<=0) Then
       Writeln(ERROR EN LA ENTRADA DEL DATO, REINTENTE');
Until (M>0);
Writeln;
Writeln('ENTRE LAS COMPONENTES DE LA MATRIZ');
Writeln;
For R:=1 to N do
    Begin
       For F:=1 to M do
           Begin
              Gotoxy(5*R,10+3*F);
              Read(A[R,F]);
           End;
    End;
Clrscr;
Writeln('LA MATRIZ INICIAL ES :');
Writeln;
For F:=1 to M do
    Begin
      For R:=1 to N do
           Begin
              Write(' ',A[R,F]:6:1);
           End;
       Writeln;
    End;
```

```
    Writeln;Writeln;Writeln;
    Writeln('LA MATRIZ TRANSPUESTA ES :');
    Writeln;
    For F:=1 to N do
        Begin
            For R:=1 to M do
                Begin
                    Write('',A[F,R]:6:1);
                End;
            Writeln;
        End;
END.
```

CAPITULO IV

DISEÑO DESCENDENTE

4.1 INTRODUCCION
4.2 PROGRAMACIÓN MODULAR
4.3 CLASIFICACIÓN DE LOS MÓDULOS
PROBLEMAS RESUELTOS
PROBLEMAS PROPUESTOS
EJERCICIOS COMPILADOS

4.1 INTRODUCCIÓN

Una de las ideas esenciales en la programación de computadores se refiere a la división de problemas grandes y complejos, en subproblemas más simples, comprensibles y fáciles de implementar, obviando de ésta forma inconvenientes como los siguientes:

- Repetición de un conjunto de instrucciones o pasos a través del algoritmo.
- La inclusión de pequeños cambios a un algoritmo, se refleja en profundas reformas al mismo.
- La puesta en marcha del algoritmo se vuelve compleja.

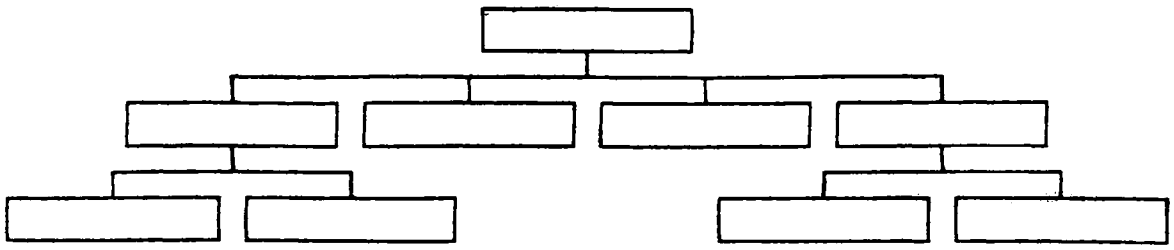
Los anteriores problemas se evitan con la utilización de la Programación Modular.

4.2 PROGRAMACIÓN MODULAR

Conocida en ocasiones como Divide y Vencerás, consiste en fraccionar un problema complejo en varios subproblemas, para que la solución se vuelva más simple. A su vez, un subproblema se puede dividir en otros subproblemas y así sucesivamente. A ésta metodología también se le llama DISEÑO DESCENDENTE o TOP_DOWN, porque parte del problema general al diseño de soluciones específicas para cada una de las divisiones, hasta obtener una solución efectiva del problema principal. En esta forma será posible resolver y probar cada subproblema por separado, permitiendo inclusive una distribución de tareas entre programadores, acortando notablemente el tiempo demandado para la solución.

Cuando se plantea adecuadamente el análisis descendente, obliga a examinar todos los aspectos del problema antes de continuar con el nivel inferior siguiente, siendo el resultado final un conjunto de algoritmos que se pueden implementar usando las estructuras estudiadas en los capítulos anteriores.

Las relaciones de cada módulo del Diseño Descendente pueden ser representadas gráficamente en un diagrama de estructura similar a un organigrama, mediante bloques o rectángulos que señalan una actividad asociada con un nivel particular.



Las normas de esta metodología son:

- Un módulo se ejecuta sólo cuando el control le llega del bloque precedente. Cuando el bloque termina su ejecución, el control regresa al módulo que le antecede jerárquicamente.
- El control de la información se enruta hacia abajo, los resultados se enrutan hacia arriba.
- Cada módulo debe resolver un problema específico.

4.3 CLASIFICACIÓN DE LOS MÓDULOS

Los módulos se pueden clasificar en Funciones y Procedimientos.

4.3.1 Funciones

Cuando se llama a una función, el control pasa a las instrucciones que la definen y una vez que la función ha sido evaluada, el control regresa con el correspondiente resultado al punto donde fue llamado en el algoritmo principal.

Los usuarios de los lenguajes de programación tienen a su disposición cierta cantidad de funciones definidas por el compilador, también llamadas funciones **Internas** o **estándar**, y pueden ser funciones trigonométricas y matemáticas, entre otras, por ejemplo SIN (Seno), COS (Coseno), TAN (Tangente), SQRT (Raiz cuadrada), SQR (cuadrado), LN (logaritmo natural) etc. Se recomienda al estudiante consultar el manual de referencia del compilador que esté utilizando, para que conozca en detalle la relación de funciones estándar que le proporciona el compilador. Las funciones **Internas** se pueden usar directamente en las expresiones, mediante el llamado de su nombre con los argumentos encerrados entre paréntesis.

Una función interna es evocada de la siguiente forma:

NOMBRE_FUNCION (Lista de argumentos actuales)

EJERCICIO

Elaborar una tabla de Senos y Cosenos para valores de X que oscilan entre 0 y 360 grados y que sufre incrementos de 5.

Solución

a) Análisis

Entrada : No requiere de lectura de datos ya que éstos se generan automáticamente.
Proceso : No se hace necesario definir las funciones de Seno y Coseno puesto que la mayoría de los compiladores las tienen definidas, por lo tanto se hará uso de ellas directamente.
Salida : Imprimir el valor de X, el Seno y Coseno respectivamente.

b) Variables a utilizar

X = Valor a ser tratado

SIN = Función Seno
COS = Función Coseno
SENO = Seno de X
COSENO = Coseno de X

c) Algoritmo

```
Paso 1      INICIO
Paso 2        PARA X DE 0 HASTA 360,5 HACER
                  SENO ← SIN (X)
                  COSENO ← COS (X)
                  IMPRIMIR X, SENO, COSENO
              FIN_PARA
Paso 3      FIN
```

El argumento en este caso es la variable X, la cual es objeto del cálculo del Seno y Coseno en forma directa.

Cuando las funciones **Internas** no realizan una operación determinada o cálculo en particular, se hace necesario que sean definidas por el usuario y son llamadas funciones **Externas**. Tanto las funciones **Internas** como las **Externas**, retornan un sólo valor bajo el nombre como fue definida, al mismo lugar del algoritmo principal donde se hizo la llamada, una vez han sido ejecutadas. Este valor puede ser cualquier tipo elemental de dato.

Su forma general es:

```
FUNCIÓN < NOMBRE_FUNCION > (ARGUMENTOS)
  INICIO
      ACCIÓN 1
      ACCIÓN 2
          .
          .
          .
      ACCIÓN N
  FIN
```

y es llamada de la siguiente forma:

```
NOMBRE_FUNCION (ARGUMENTOS)
```

Donde: El nombre de la función se debe caracterizar por lo que realiza.
Los argumentos son nombres de variables, de funciones o de procedimientos y sólo son utilizados dentro del cuerpo de la función.

El conjunto de argumentos se llaman **Ficticios** o **Temporales** cuando son declarados en la función y se llaman **Reales** o **Actuales** cuando son declarados en el algoritmo principal.

EJERCICIO

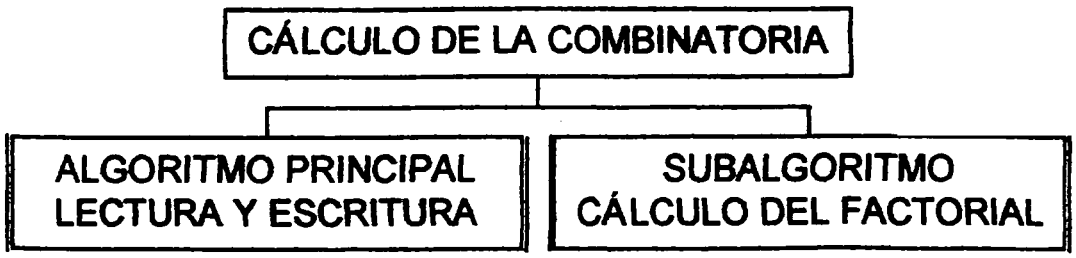
Calcular la combinatoria de N en M según la siguiente fórmula:

$$\begin{bmatrix} N \\ M \end{bmatrix} = \frac{N!}{M!(N-M)!}$$

Solución

a) Análisis

Estructura del algoritmo



Entrada : Leer N y M
Proceso : Por restricción del problema se controla que N>M y M>=0.
Se hace un subalgoritmo general que calcula el factorial de un valor y se elabora el algoritmo principal del que se hacen tantas llamadas al subalgoritmo como se requieran, por ejemplo para calcular el factorial del numerador (N), factorial del denominador (M) y el factorial de la diferencia (N - M).
El argumento del subalgoritmo es un argumento temporal,el cual es reemplazado por el argumento actual del algoritmo principal para posteriormente calcular la combinatoria.
Salida : Imprimir N, M y la combinatoria

b) Variables a utilizar

N, M, K = Valores a ser factorizados (Argumentos Actuales)

FACN = Factorial de N
FACM = Factorial de M
FACK = Factorial de la diferencia (N-M)
FACT = Factorial General
COMBI = Combinatoria
I = Variable controladora de ciclos
J = Argumento temporal

c) Algoritmo principal

```
Paso 1      INICIO
Paso 2        Leer N, M
Paso 3        SI  (N > M) Y (M >= 0)
                  ENTONCES
                     FACN ← FACTORIAL(N)
                     FACM ← FACTORIAL(M)
                     K ← N - M
                     FACK ← FACTORIAL(K)
                     COMBI ← FACN / (FACM * FACK)
                     IMPRIMIR N, M, COMBI
              FIN_SI
Paso 4      FIN
```

Subalgoritmo

```
            FUNCIÓN FACTORIAL(J)
Paso 1      INICIO
Paso 2        FACT ← 1
Paso 3        I ← 1
Paso 4        REPETIR
                 FACT ← FACT * I
                 I ← I + 1
              HASTA_QUE I > J
Paso 5        FACTORIAL ← FACT
Paso 6      FIN
```

En la primera llamada se evoca desde el algoritmo principal al subalgoritmo y el argumento actual (N) es reemplazado por el argumento temporal (J), se evalúa el factorial y el resultado, almacenado bajo el nombre de la función FACTORIAL, es retornado al algoritmo principal al mismo lugar donde se hizo la llamada, para continuar con la ejecución de éste.

EJERCICIO

Mediante la utilización de una función externa calcular la media armónica H, de la siguiente manera:

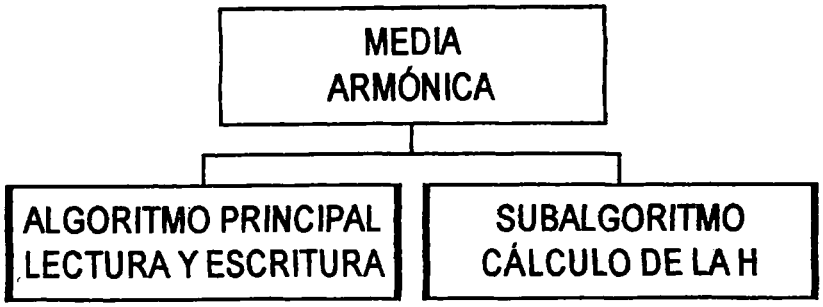
$$H = \frac{N}{\sum_{I=1}^{N} \frac{1}{X_I}} \quad siendo \quad \sum_{I=1}^{N} \frac{1}{X_I} = \frac{1}{X_1} + \frac{1}{X_2} + \ldots + \frac{1}{X_N}$$

Para un vector Z de M elementos.

Solución

a) Análisis

Estructura del algoritmo



Entrada : Leer el vector a-ser analizado y su límite.
Inicializar la sumatoria de inversos y el contador de elementos sumados en cero.

Proceso : Chequear que el elemento a ser sumado sea diferente de cero, en este caso, se incrementa el contador de términos sumados en la unidad. Si el elemento es igual a cero no lo tiene en cuenta porque su inverso daría como resultado un indeterminado y toma el siguiente elemento para hacerle el mismo análisis. Por último se divide la sumatoria de inversos entre el contador de términos sumados. Todo este proceso se hace en un subalgoritmo.
El algoritmo principal se encarga de leer el vector Z y hacer el llamado al subalgoritmo, mediante la entrega del nombre del vector como argumento, para que proceda a realizar el cálculo respectivo.

Salida : Imprimir el vector leído y la media armónica.

b) Variables a utilizar

CE = Contador de elementos

Z = Nombre del vector
M = Límite de elementos del vector
Z,M = Argumentos Actuales
SUMA = Acumulador de inversos
X, N = Argumentos temporales
H = Media armónica

c) Algoritmo Principal

```
Paso 1      INICIO
Paso 2        Leer M
Paso 3        PARA I DE 1 HASTA M HACER
                 Leer Z(I)
                 IMPRIMIR Z(I)
              FIN_PARA
Paso 4        H ← ARMÓNICA(Z, M)
Paso 5        IMPRIMIR "La media armónica del vector Z es", H
Paso 6      FIN
```

Subalgoritmo

```
            FUNCIÓN ARMÓNICA(X, N)
Paso 1      INICIO
Paso 2        SUMA ← 0
Paso 3        CE ← 0
Paso 4        PARA I DE 1 HASTA N HACER
                 SI X(I) <> 0
                    ENTONCES SUMA ← SUMA + 1/X(I)
                                CE ← CE + 1
                 FINSI
              FIN_PARA
Paso 5        ARMÓNICA ← CE / SUMA
Paso 6      FIN
```

Observación

Los argumentos del algoritmo principal y del subalgoritmo no necesariamente deben coincidir en el nombre, puesto que los argumentos **Actuales** son reemplazados por los argumentos **Temporales** y una vez han sido evaluados, son reemplazados de nuevo en los argumentos **Actuales** para posterior utilización en cálculos e impresión. Es indispensable

que los argumentos **Temporales** correspondan en número, orden y tipo con los argumentos **Actuales**.

4.3.2 Procedimientos

No en todos los algoritmos se hace necesario la utilización de Funciones ó Procedimientos, de hecho se pueden presentar algoritmos compuestos por cientos de pasos o actividades que jamás utilicen un Procedimiento. Sin embargo, esta metodología presenta entre otros, los siguientes problemas: la solución se vuelve más extensa de lo necesario; requiere mayor cantidad de memoria del computador; a mayor número de instrucciones es mayor la probabilidad de cometer errores en la digitación; es más dificultoso su entendimiento; al elaborar una prueba de escritorio es más complejo su seguimiento y se hace más difícil la detección y corrección de errores. Por lo anterior se puede deducir que el **Procedimiento** es más general y recomendable.

La estructura del **Procedimiento** básicamente es la misma en la mayoría de los lenguajes de programación, aunque en unos se llama Subrutina ó Subprograma, en otros se llama Módulo o Procedimiento.

Los **Procedimientos** se declaran en forma similar a las Funciones, con pequeñas variaciones en la forma como se le referencia. A diferencia de las Funciones: a) los **Procedimientos** devuelven 0, 1 ó más valores al algoritmo que lo llama, la función sólo devuelve un valor. b) El nombre del **Procedimiento** no está asociado a los resultados que produce. c) Una función se utiliza haciendo mención a su nombre en una expresión, en cambio el procedimiento es empleado mediante un llamado al mismo.

Su forma general es:

```
PROCEDIMIENTO < NOMBRE_PROCEDIMIENTO > (ARGUMENTOS)
INICIO
   ACCIÓN 1
   ACCIÓN 2


   ACCION N
FIN
```

El **procedimiento** es evocado de la siguiente forma:

LLAMAR NOMBRE_PROCEDIMIENTO (lista de argumentos actuales).

Observación

Algunos compiladores exigen que los procedimientos o subprogramas sean definidos antes que el programa principal, para otros es indiferente. En este estudio se define primero el algoritmo principal y a partir de él se hace el llamado a los diferentes procedimientos en la medida en que se requieran.

EJERCICIO

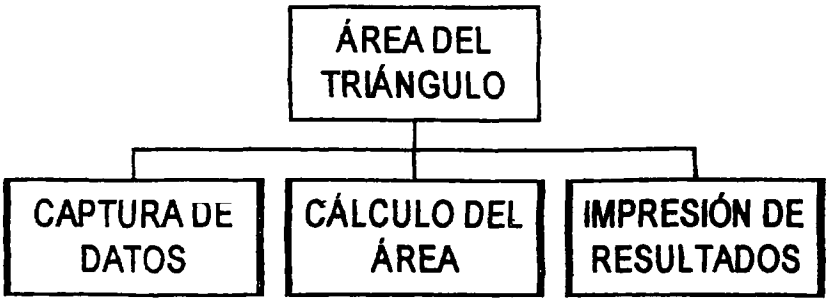
Se tiene un conjunto de 100 registros con la siguiente información por registro: Altura y Base del triángulo.
Calcular el área del triángulo según la siguiente fórmula:

$$AREA = \frac{BASE * ALTURA}{2}$$

Solución

a) Análisis

Estructura del algoritmo



Entrada : Leer la base y la altura. Estos datos se solicitan por pantalla.
Proceso : Se elaboran 3 procedimientos, uno para la captura de datos, otro para el cálculo del área del triángulo y el último para producir los resultados. A partir del algoritmo principal se hace el llamado a cada procedimiento.
Salida : Imprimir base, altura y área del triángulo.

b) Variables a utilizar

BASE = Base
ALTURA = Altura
AREA = Área

c) Algoritmo principal

```
Paso 1      INICIO
Paso 2        PARA I DE 1 HASTA 100 HACER
                 LLAMAR DATOS(BASE, ALTURA)
                 LLAMAR CALCULO_AREA(BASE,ALTURA,AREA)
                 LLAMAR IMPRIMIR_RESULTADOS(BASE, ALTURA, AREA)
              FIN_PARA
Paso 3      FIN
```

```
PROCEDIMIENTO DATOS (BASE, ALTURA)
Paso 1      INICIO
Paso 2        ESCRIBIR "Teclee la altura del triángulo"
Paso 3        Leer ALTURA
Paso 4    ESCRIBIR "Teclee la base del triángulo"
Paso 5        Leer BASE
Paso 6      FIN
```

```
PROCEDIMIENTO CALCULO_AREA (BASE, ALTURA, AREA)
Paso 1      INICIO
Paso 2        AREA ← (BASE * ALTURA) / 2
Paso 3      FIN
```

```
PROCEDIMIENTO IMPRIMIR_RESULTADOS (BASE, ALTURA, AREA)
Paso 1      INICIO
Paso 2        IMPRIMIR "La base del triángulo es", BASE
Paso 3        IMPRIMIR "La altura del triángulo es", ALTURA
Paso 4        IMPRIMIR "El área del triángulo es", AREA
Paso 5      FIN
```

En el paso 2 del procedimiento DATOS se escribe un mensaje por pantalla en donde se solicita teclear la altura del *triángulo*, diferente del paso 2 del procedimiento IMPRIMIR_RESULTADOS, porque allí se refiere a una salida por impresora.

Las instrucciones empleadas en el algoritmo principal consisten únicamente de los nombres de los procedimientos y sus argumentos. Después de ejecutado un procedimiento, el programa principal continúa con la siguiente instrucción y así sucesivamente hasta su terminación.

Observación

Los argumentos están conformados por las variables que requiere el procedimiento para poder efectuar los diferentes procesos y por aquellas variables que se generan dentro de él y que serán de posterior utilización, bien puede ser en otros cálculos, o para su correspondiente impresión por parte del programa principal.

SINTAXIS FORTRAN

En FORTRAN existen 3 clases de subprogramas:

- Función proposición o instrucción FUNCTION.
- Subprograma FUNCTION.
- Subprograma SUBROUTINE.

Una instrucción FUNCTION especifica las operaciones que se deben realizar cada vez que el nombre de una instrucción FUNCTION aparece referenciada en otra proposición en el mismo programa. El nombre usado para llamarla debe coincidir con el nombre empleado en la definición.

Forma general:

NOMBRE (A1,A2,A3...AN) = EXPRESION

donde: NOMBRE = Es el nombre de la función.
A1,A2,...AN = Son los argumentos de la función.
EXPRESION = Es una expresión aritmética.

Se llama:

VARIABLE = NOMBRE (A1,A2,A3...AN)

Consideraciones con la instrucción FUNCTION

- Los argumentos de la función pueden aparecer sólo una vez en el listado de argumentos.
- Los argumentos pueden ser sólo variables, la expresión puede contener constantes.
- Todas las instrucciones FUNCTION definidas en un programa deben preceder a la primera instrucción ejecutable del programa.
- Las expresiones no pueden contener elementos de un arreglo (elementos subindicados).
- No es posible definir una instrucción FUNCTION sin argumentos.

Ejemplo

Calcular el pago mensual según la siguiente fórmula:

$$PAGO\ MENSUAL = C * \frac{t(1+t)^n}{(1+t)^n - 1}$$

siendo C = capital.
t = tasa de interés.
n = tiempo.

Calcular el pago mensual para los siguientes tres planes:

1. C = 100000 t = 3% n = 24 meses.
2. C = 150000 t = 2.5% n = 12 meses.
3. C = 200000 t = 3.5% n = 30 meses.

Codificación:

```
      PROGRAM PLAN
      CUOTA(C,T,N) = (C*T*(1+T)**N)/((1+T)**N-1)
      PAGO1 = CUOTA (100000,0.03, 24)
      PAGO2 = CUOTA (150000,0.025,12)
      PAGO3 = CUOTA (200000,0.035,30)
      WRITE(*,1)PAGO1,PAGO2,PAGO3
1     FORMAT(1H1,40X,'PROGRAMA ELABORADO POR ALONSO
     *TAMAYO ALZATE' /// 50X, 'CÁLCULO DE ANUALIDADES'
     *///15X, 'PLAN 1 = ', F8.2,3X,'PLAN 2 = ', F8.2,3X, 'PLAN 3 = ', F8.2)
      STOP
      END
```

SUBPROGRAMA FUNCTION

Forma general:

FUNCTION NOMBRE(A1,A2,...AN)

donde FUNCTION : Palabra reservada del FORTRAN.
NOMBRE : Es el nombre asignado a la función.
A1,A2,...AN : Son los argumentos DUMMY y pueden ser variables o nombres de arreglos o nombre DUMMY de una subrutina u otro subprograma. FUNCTION.

El subprograma FUNCTION, consiste de la palabra reservada FUNCTION seguida por otras proposiciones incluyendo RETURN y END.

ESQUEMA DEL SUBPROGRAMA FUNCTION

```
FUNCTION CALCULO(A,B,J)
      .
      .
      .
   I=J*2
      .
      .
      .
   CALCULO=A**I/B
   RETURN
END
```

Se llama

VARIABLE=NOMBRE(A1,A2,... AN)

donde NOMBRE : Es el nombre asignado a la función.
A1,A2,... AN : Son los argumentos actuales.

Ejemplo de programa llamante

```
RESP=CALCULO(X,Y,K)
```

En este ejemplo los valores X,Y,K son reemplazados en el subprograma FUNCTION por los argumentos A,B,J respectivamente; se ejecuta el subprograma FUNCTION y los resultados producidos son trasladados al programa principal al lugar donde se hizo la llamada, a través de una operación inversa de sustitución de argumentos.

Consideraciones con el subprograma FUNCTION

- La instrucción FUNCTION debe ser la primera en el subprograma.
- Cada subprograma FUNCTION se programa y compila como un programa FORTRAN separado.
- En un subprograma se pueden usar todas las proposiciones FORTRAN, excepto otras proposiciones FUNCTION o SUBROUTINES.
- Los argumentos de un subprograma FUNCTION no se deben modificar durante la ejecución del subprograma.
- La proposición RETURN permite que el control vuelva al mismo lugar del programa principal donde se efectuó la llamada.
- En un mismo subprograma se pueden utilizar varias proposiciones RETURN, pero mínimo debe aparecer una.

- La proposición RETURN es la última ejecutada en el subprograma, pero no tiene que ser físicamente la última proposición del programa.
- La proposición RETURN no puede aparecer en el programa principal.

SUBPROGRAMA SUBROUTINE

Forma general:

SUBROUTINE NOMBRE(A1,A2,...AN)

donde NOMBRE : Es el nombre asignado a la subrutina.
A1,A2,... AN : Son los argumentos DUMMY y pueden ser variables o nombres de arreglo o nombre DUMMY de otra subrutina o subprograma FUNCTION.

El subprograma SUBROUTINE es similar al subprograma FUNCTION en muchos aspectos. Las reglas para ambos subprogramas son similares, ambos requieren de una proposición END y una proposición RETURN y ambos contienen el mismo orden de los argumentos DUMMY.

Al subprograma subroutine se le llama mediante la proposición CALL, así:

Forma general:
CALL NOMBRE(A1,A2,...AN)

donde NOMBRE : Es el nombre del subprograma subrutina.
A1,A2,... AN : Son los argumentos que han sido proporcionados al subprograma subrutina. Los argumentos pueden ser variables, elementos de un arreglo, nombre de un arreglo, constantes, expresiones aritméticas, expresiones lógicas o nombre de un subprograma.

Consideraciones con el subprograma SUBROUTINE

- La instrucción SUBROUTINE debe ser la primera en el subprograma.
- El nombre de la subrutina no puede aparecer en cualquier otra instrucción en el subprograma SUBROUTINE.
- Cada argumento DUMMY puede aparecer sólo una vez en la lista de argumentos.
- Si no hay argumentos DUMMY, los paréntesis deben ser omitidos.
- Un subprograma SUBROUTINE debe contener una última instrucción RETURN y END.
- Los argumentos DUMMY deben corresponder en número, orden y tipo con los argumentos actuales o argumentos de programa principal.

Ejemplo

Para un salario bruto entre	Retención en la fuente
0 - 200000	0
200001 - 400000	4% salario bruto.
400001 y más.	7% salario bruto.

Imprimir código, nombre, retención en la fuente y salario neto para una nómina de 50 empleados.

```
      PROGRAM NOMINA
      CHARACTER*2 NOM
C     NOM = NOMBRE DEL TRABAJADOR
C     SN = SALARIO NETO
      WRITE(*,10)
 10   FORMAT(1H1///42X,'UNIVERSIDAD NACIONAL DE COLOMBIA SEDE
     *MANIZALES'//38X,'LIQUIDACIÓN DE SALARIOS SEGÚN
     *RETENCIÓN EN LA FUENTE'///40X,'PROGRAMA ELABORADO
     *POR ALONSO TAMAYO ALZATE'///10X, 'CÓDIGO',10X,
     *'NOMBRE', 25X,'SALARIO BRUTO', 10X,'RETENCIÓN', 5X,
     *'SALARIO NETO'///)
      DO 20 L=1.50
         READ(*,30)ICOD,NOM,SB
 30      FORMAT(I4, 15A2, F8.2)
         CALL NOMBRE(SB,RF,SN)
         WRITE(*,40)ICOD,NOM,SB,RF,SN
 40      FORMAT(//11X,I4,10X,15A2,5X,F8.2,2(10X,F8.2))
 20   CONTINUE
      STOP
      END

      SUBROUTINE NOMBRE(SB,RF,SN)
C        SB = SALARIO BRUTO
C        RF = RETENCIÓN EN LA FUENTE
C        SN = SALARIO NETO
         IF (SB-200000)2,2,3
   2        RF=0
            GOTO 12
   3     IF (SB-400000)4,4,5
   5        RF=SB*0.07
            GOTO 12
   4     RF=SB*0.04
  12     SN=SB-RF
```

```
      RETURN
    END
```

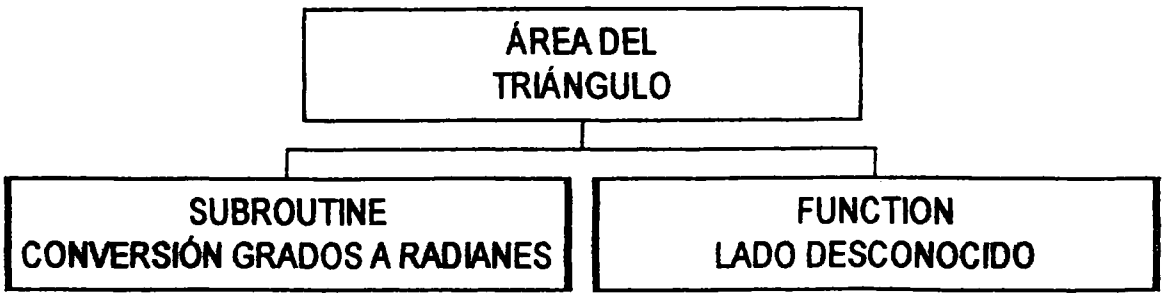
Observación

En FORTRAN la instrucción FUNCTION se debe declarar dentro del cuerpo del programa principal, mientras que los subprogramas FUNCTION o SUBROUTINE se declaran independientemente del programa principal y es indiferente si se codifica primero el subprograma o programa principal, ya que cada uno debe ser compilado separadamente antes de ser llamado por el programa principal.

EJERCICIO

Calcular el área de un triángulo conociendo dos de sus lados y el ángulo comprendido entre ellos, mediante la aplicación de un subprograma FUNCTION y un subprograma SUBROUTINE que convierta grados a radianes.

a) Estructura del algoritmo



b) Compilación FORTRAN

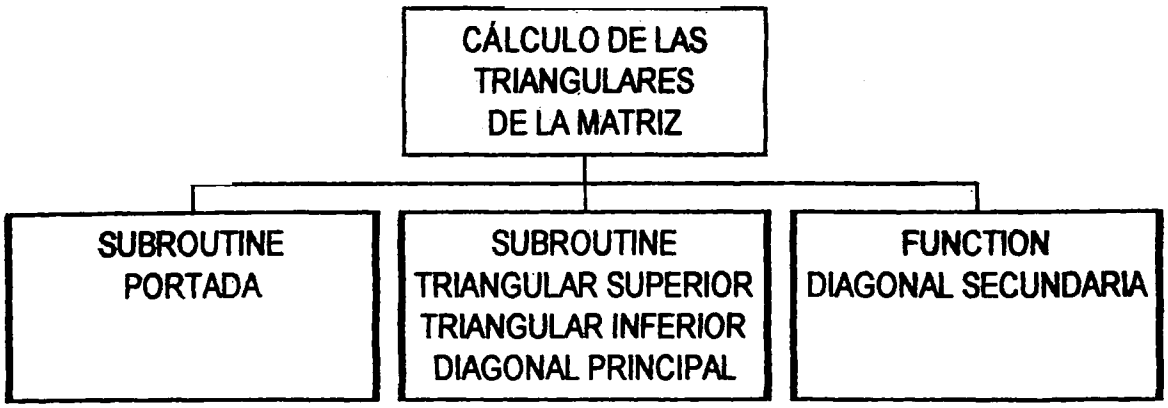
```
      PROGRAM AREA
C     ESTE PROGRAMA CALCULA EL AREA DE UN TRIÁNGULO CONOCIDO
C     DOS LADOS Y EL ÁNGULO COMPRENDIDO ENTRE ELLOS
C     DEFINICIÓN DE VARIABLES: A y B LADOS DEL TRIÁNGULO DADO.
C     IGRA: GRADOS
C     IMIN: MINUTOS
C     ISEG: SEGUNDOS
C     C: LADO DESCONOCIDO
      WRITE(*,1)
1     FORMAT(2X,'TECLEE LOS DATOS')
      WRITE(*,*) 'LADO A= '
```

```
      READ(*,3)A
3     FORMAT(F4.1)
      WRITE(*,*) 'LADO B= '
      READ(*,5)B
5     FORMAT(F4.1)
      WRITE(*,*) 'GRADOS= '
      READ(*,6)IGRA
6     FORMAT(I3)
      WRITE(*,*) 'MINUTOS= '
      READ(*,7)IMIN
7     FORMAT(I3)
      WRITE(*,*) 'SEGUNDOS= '
      READ(*,8)ISEG
8     FORMAT(I3)
      CALL CONV(IGRA,IMIN,ISEG,RAD)
      P=(A+B+C(A,B,RAD))/2.0
      S=P*(P-A)*(P-B)*(P-C(A,B,RAD))
      AR=S**0.5
      WRITE(*,50)AR
50    FORMAT(2X, 'AREA= ',F6.2,/)
      STOP
      END

      SUBROUTINE CONV(IGRA,IMIN,ISEG,RAD)
        ANG=IGRA+(IMIN/60)+(ISEG/3600)
        RAD=(ANG*3.141592)/180
        RETURN
      END

      FUNCTION C(A,B,RAD)
        A2=A**2
        B2=B**2
        D=A*B*(COS(RAD))
        C=(A2+B2-D)**0.5
        RETURN
      END
```

EJERCICIO

Leer una matriz de N filas y M columnas. Calcular la suma de los elementos de la triangular superior, la suma de los elementos de la triangular inferior, la suma de los elementos de la diagonal principal y la suma de los elementos de la diagonal secundaria, mediante el empleo de un subprograma FUNCTION y subprogramas SUBROUTINE.

a) Estructura del algoritmo



b) Compilación FORTRAN

```
      PROGRAM TRIANG
      DIMENSION A(10,10)
      CALL PORTADA
C     LEER UNA MATRIZ DE N FILAS Y M COLUMNAS, CALCULAR LA SUMA
C     DE LA TRIANGULAR SUPERIOR, LA SUMA DE LA TRIANGULAR
C     INFERIOR, LA SUMA DE LA DIAGONAL PRINCIPAL Y LA SUMA DE LA
C     DIAGONAL SECUNDARIA
      WRITE(*,1)
1     FORMAT('1',5X, 'ESCRIBA EL NÚMERO DE FILAS: '\)
      READ(*,2)N
2     FORMAT(I2)
      WRITE(*,3)
3     FORMAT(/5X, 'ESCRIBA EL NUMERO DE CULUMNAS: '\)
      READ(*,4)M
   4  FORMAT(I2)
      DO 7 I=1,N
         DO 7 J=1,M
            WRITE(*,5)I,J
   5        FORMAT(10X, 'ESCRIBA EL ELEMENTO A(' ,I2, ',', I2, ') : '\)
            READ(*,6)A(I,J)
   6        FORMAT(F4.0)
   7  CONTINUE
      WRITE(*,20)
  20  FORMAT(///10X, 'LA MATRIZ LEIDA ES' /)
      DO 16 I=1,N
         WRITE(*,15)(A(I,J),J=1,M)
  15     FORMAT(15X,/3F10.1)
```

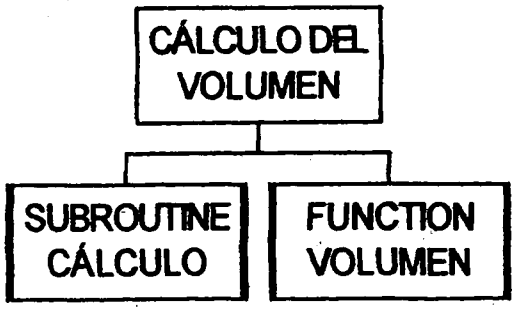
```
16    CONTINUE
      IF (N .EQ. M) THEN
17       CALL MATRIZ(N,M,A,SUMI,SUMS,SUMD)
         WRITE(*,8)SUMI,SUMS,SUMD
 8       FORMAT(//5X, 'LA SUMA DE LA TRIANGULAR INFERIOR ES : ',F8.1,//
     *   5X, 'LA SUMA DE LA TRIANGULAR SUPERIOR ES: ',F8.1,//5X, 'LA SU
     *   MA DE LA DIAGONAL PRINCIPAL ES: ',F8.1,/)
         ADIAG=DIAG(N,A)
         WRITE(*,9)ADIAG
 9       FORMAT(5X, 'LA SUMA DE LA DIAGONAL SECUNDARIA ES: ',F8.1)
      ELSE
         GOTO 18
      ENDIF
18    STOP
      END

      SUBROUTINE MATRIZ(N,M,A,SUMI,SUMS,SUMD)
         DIMENSION A(10,10)
         SUMS=0
         SUMI=0
         SUMD=0
         DO 13 I=1,N
            DO 13 J=1,M
               IF (I-J)10,11,12
10                SUMI=SUMI+A(I,J)
                  GOTO 13
11                SUMD=SUMD+A(I,J)
                  GOTO 13
12                SUMS=SUMS+A(I,J)
13       CONTINUE
         RETURN
      END

      FUNCTION DIAG(N,A)
         DIMENSION A(10,10)
         SUMC=0
         J=1
         DO 14 I=N,1,-1
            SUMC=SUMC+A(I,J)
            J=J+1
14       CONTINUE
         DIAG=SUMC
         RETURN
      END
```

```
      SUBROUTINE PORTADA
         WRITE(*,21)
21       FORMAT(10(/),//40X, 'ELABORADO POR ALONSO TAMAYO ALZATE',
     *   11(/),25X, 'UNIVERSIDAD NACIONAL DE COLOMBIA' //31X, 'SEDE
     *   MANIZALES'//23X, 'FACULTAD DE CIENCIAS Y ADMINISTRACIÓN')
         RETURN
      END
```

EJERCICIO

Calcular el volumen a llenar para la construcción de terraplenes mediante la aplicación de un subprograma FUNCTION y un subprograma SUBROUTINE.

a) Estructura del algoritmo



b) Compilación FORTRAN

```
      PROGRAM TERRAPLEN
      DIMENSION V(10),DI(10)
C     ESTE PROGRAMA CALCULA EL VOLUMEN A LLENAR PARA LA
C     CONSTRUCCIÓN DE TERRAPLENES, INTERPRETÁNDOSE ASI LAS
C     SIGUIENTES ABREVIACIONES:
C     CN = COTA NEGRA DE LA ESTACIÓN.
C     CRPI = COTA ROJA DEL PUNTO DE INFLEXIÓN.
C     L Y LL = PROPORCIÓN DEL TERRAPLEN.
C     CT = COTA DE TRABAJO.
C     CR = COTA ROJA.
      WRITE(*,80)
 80   FORMAT(6(/),21X, 'UNIVERSIDAD NACIONAL',///,25X, 'SEDE
     *MANIZALES',15(/),35X, 'ELABORADO POR: ALONSO TAMAYO ALZATE')
      WRITE(*,2)
  2   FORMAT('ESCRIBA EL LÍMITE DE DATOS QUE DESEA ENTRAR:')
      READ(*,1)N
```

```
 1    FORMAT(I2)
      DO 20 J=1,N
         WRITE(*,3)
 3       FORMAT('ESTACIÓN, CN, PENDIENTE, CRPI, L, LL, ANCHO, DIST')
         READ(*,4)ESTACION,CN,PENDIENTE,CRPI,L,LL,ANCHO,DI(J)
 4       FORMAT(F3.0,/,F6.3,/,F5.2,/,F6.3,/,I2,/,I2,/,F5.2,/,F6.2)
         DIST=DI(J)
         CALL CALCULO(PENDIENTE,DIST,CRPI,CN,CT,AREA,L,LL,ANCHO,CR)
         WRITE(*,5)CN,CR,CT,AREA
 5       FORMAT('CN=',4X,F6.3,/,3X, 'CR=',4X,F6.3,/,3X, 'CT=',4X,F6.3,/,3X,
     *   'AREA=',2X,F6.3)
         V(J)=AREA
20    CONTINUE
      VOLUME=VOLUMX(DI,V,N)
      WRITE(*,6)VOLUME
 6    FORMAT('EL VOLUMEN A ESCAVAR ES=',2X,F10.2)
      STOP
      END

      SUBROUTINE CALCULO(PENDIENTE,DIST,CRPI,CN,CT,AREA,L,LL,
     *ANCHO,CR)
         CR=(PENDIENTE*DIST)+CRPI
         CT=CR-CN
         AREA=(L*CT*ANCHO)+((L*LL)*(CT)**2)
         RETURN
      END

      FUNCTION VOLUMX(DI,V,N)
         DIMENSION V(10),DI(10)
         SUM=0
         DO 60 K=1,N-1
            VO = ((V(K)+V(K+1))/2)*(DI(K)-DI(K+1))
            WRITE(*,7)VO
 7          FORMAT('EL VOLUMEN PARCIAL ES=',F10.2)
            SUM=SUM+VO
60       CONTINUE
         VOLUMX=SUM
         RETURN
      END
```

SINTAXIS PASCAL

En Pascal existen dos clases de **subprogramas** :

- Funciones.
- Procedimientos.

Las funciones son subprogramas que devuelven un valor.
Las funciones pueden ser estándar o definidas por el programador. Las funciones estándar se encuentran definidas en el respectivo compilador; al igual que sucede con el FORTRAN, se recomienda consultar los manuales de referencia.

A continuación se relacionan algunas funciones estandar de uso frecuente

SIN (X)	Seno de X.
COS (X)	Coseno de X.
EXP (X)	Exponencial de X.
LN (X)	Logaritmo Natural de X.
SQRT (X)	Raíz Cuadrada de X.
SQR (X)	Cuadrado de X.
ABS (X)	Valor Absoluto de X.
ARCTAN (X)	Arco Tangente de X.
PI	Devuelve el valor de Pi.

Las funciones son utilizadas directamente en una expresión, tantas veces sean requeridas.

Ejemplo

X:=SQRT (ABS (SIN(A)));

Extrae la raíz cuadrada del valor absoluto del SENO de A y el resultado lo almacena en X.

Ejemplo

C:=SQRT (SQR (A) + SQR (B))

Saca la raíz cuadrada a la sumatoria de los cuadrados de A más B y el resultado lo almacena en C.

FUNCIONES DEFINIDAS POR EL PROGRAMADOR

La función devuelve un único valor al lugar donde se le llamó.

Forma general:

FUNCTION NOMBRE (A1, A2, A3, ..., AN) : TIPO;

donde FUNCTION : Es una palabra reservada del Pascal.
A1, A2, ..., AN : Son los argumentos DUMMY o Temporales.
TIPO : Tipo de dato del resultado que devuelve la función.

ESQUEMA DEL SUBPROGRAMA FUNCTION

```
FUNCTION NOMBRE (A1, A2, A3, ... AN) : TIPO;
{DECLARACIONES LOCALES}
BEGIN
      INSTRUCCIONES QUE CONFORMAN LA FUNCIÓN
      ____________________
      ____________________
      ____________________

      NOMBRE:=RESULTANTE DE LA FUNCIÓN
END;
```

Se llama:

```
IDENTIFICADOR:=NOMBRE(A1,A2,... AN);
```

donde NOMBRE : Es el nombre asignado a la función.
A1,A2,... AN : Son los argumentos actuales y pueden ser constantes, variables o expresiones.

Consideraciones con el subprograma FUNCTION

- La instrucción FUNCTION debe ser la primera en el subprograma.
- Cada función se llama utilizando su nombre en una expresión con los argumentos actuales encerrados entre paréntesis.
- Los argumentos DUMMY deben corresponder en número, orden y tipo con los argumentos actuales o argumentos del programa principal.
- Cada argumento DUMMY puede aparecer sólo una vez en la lista de argumentos.
- Los argumentos de un subprograma FUNCTION no se deben modificar durante la ejecución del subprograma.

SUBPROGRAMA PROCEDURE

Forma general:

PROCEDURE NOMBRE(A1,A2,...AN);

donde PROCEDURE : Es una palabra reservada del PASCAL.

NOMBRE : Es el nombre asignado al procedimiento.
A1,A2,...AN : Son los argumentos DUMMY o temporales.

ESQUEMA DEL SUBPROGRAMA PROCEDURE

```
PROCEDURE NOMBRE(A1,A2,...AN);
{DECLARACIONES LOCALES}
BEGIN
      INSTRUCCIONES QUE CONFORMAN EL PROCEDIMIENTO;
      ____________________
      ____________________
      ____________________
END;
```

Se llama:

```
NOMBRE(A1,A2,...AN);
```

donde NOMBRE : Es el nombre asigando al procedimiento.
A1,A2,...AN : Son los argumentos actuales y pueden ser constantes, variables o expresiones.

Consideraciones con el subprograma PROCEDURE

- La instrucción PROCEDURE debe ser la primera en el subprograma.
- Los argumentos DUMMY deben corresponder en número, orden y tipo con los argumentos actuales o argumentos del programa principal.
- Si no hay argumentos DUMMY, los paréntesis deben ser omitidos.
- Cada procedimiento se llama utilizando su nombre y los argumentos actuales encerrados entre paréntesis, si no existen argumentos, se llama sólo por su nombre.
- Los argumentos de un subprograma PROCEDURE no se deben modificar durante la ejecución del subprograma.
- Los argumentos DUMMY sólo están definidos durante la ejecución del subprograma, antes y después de la ejecución están indefinidos.
- No se pueden usar nombres de subprogramas como argumentos.
- Un subprograma FUNCTION o PROCEDURE debe estar definido antes de ser utilizado.

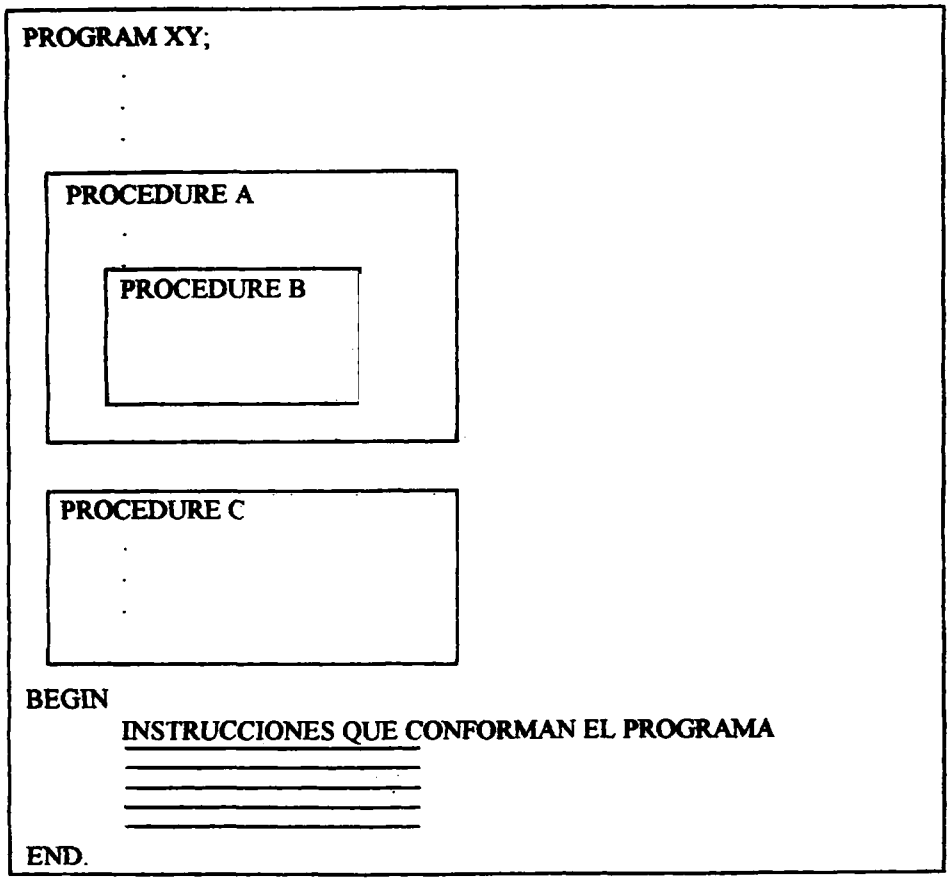
Las variables utilizadas en los subprogramas y programas principales pueden ser LOCALES y GLOBALES.
Cuando son declaradas en el subprograma se llaman variables LOCALES y sólo tienen vigencia durante la ejecución del subprograma, haciéndolos más independientes y económicas en el consumo de memoria.

Cuando las variables son declaradas en el programa principal se llaman variables GLOBALES y tienen la ventaja de compartir información con cualquiera de los subprogramas llamados por el programa principal, es decir, las variables GLOBALES pueden ser utilizadas en el programa principal y en todos los subprogramas.

En PASCAL tanto las funciones como los procedimientos se deben declarar dentro del cuerpo del programa, así:

```
PROGRAM XY;
.
.
.
  FUNCTION
  ┌──────────┐
  │          │
  └──────────┘

  PROCEDURE A
  ┌──────────┐
  │          │
  └──────────┘

  PROCEDURE B
  ┌──────────┐
  │          │
  └──────────┘

BEGIN
      INSTRUCCIONES QUE CONFORMAN EL PROGRAMA
      ──────────────────────
      ──────────────────────
      ──────────────────────
      ──────────────────────
      ──────────────────────
END.
```

OTRO ESQUEMA



NOTA: Si un subprograma llama a otro, el subprograma llamado debe declararse primero.

Resumiendo se tiene:

- La reglas para ambos subprogramas son similares.
- El procedimiento se declara igual que la función, pero su nombre no está asociado a ninguno de los resultados obtenidos como sucede con la función.
- Las funciones retornan un valor, los procedimientos devuelven 0,1 o más valores a través de la lista de argumentos.

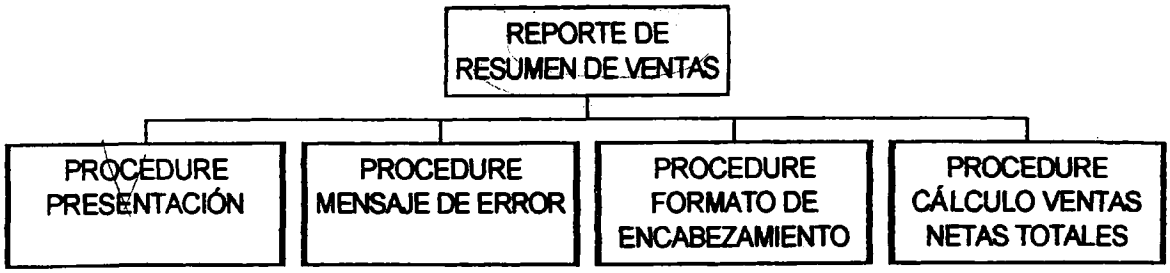
Observación

Cuando en un subprograma, bien sea SUBROUTINE O PROCEDURE, no se especifican argumentos, se debe a que el cuerpo del subprograma contiene las instrucciones de lectura y/o escritura requeridas por el mismo, es decir no hay trasmisión de datos entre el programa principal y el subprograma, o que se trata de un subprograma cuyo fin es producir sólo salidas o mensajes a ser editados.

EJERCICIO

Programa que produce un reporte resumen para N ventas mediante el empleo de procedimientos.

a) Estructura del algoritmo



b) Compilación PASCAL

```
PROGRAM VENTAS;
USES
      CRT;
{PROGRAMA QUE PRODUCE REPORTE RESUMEN DE VENTAS}
VAR
      N,J,FILA,CANTIDAD : INTEGER;
      PRODUCTO  : STRING[20];
      RES : CHAR;
      PRECIO, VENTAS_NETAS,TOTAL : REAL;

PROCEDURE PANTALLA;
{PRESENTACION INICIAL DEL PROGRAMA}
BEGIN
      CLRSCR;
      GOTOXY(22,10); WRITE('UNIVERSIDAD NACIONAL DE COLOMBIA');
      GOTOXY(25,13); WRITE('REPORTE RESUMEN DE VENTAS');
      GOTOXY(25,15); WRITE('Autor');
      GOTOXY(40,20); WRITE('Alonso Tamayo Alzate');
      GOTOXY(25,24); WRITE('Manizales, Octubre de 1995');
      DELAY(7000);
END;

PROCEDURE ERROR;
{PRODUCE MENSAJE DE ERROR}
BEGIN
      GOTOXY(1,24):WRITE('Error en datos.Presione ENTER para continuar');
```

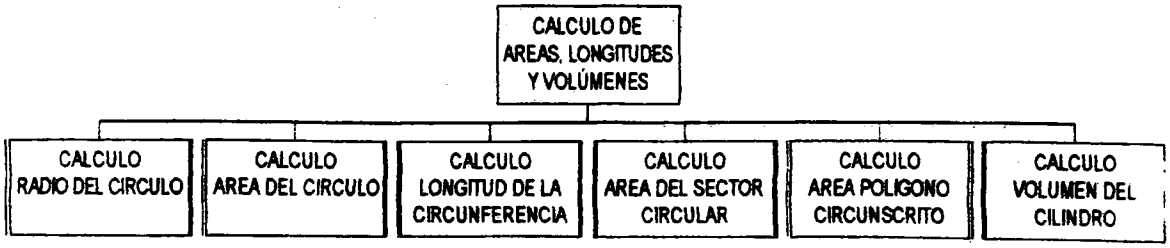
```
        GOTOXY(60,24);READ(RES);
        GOTOXY(1,24);CLREOL;
END;

PROCEDURE FORMATO;
{PRODUCE EL FORMATO DE SALIDA}
BEGIN
        GOTOXY(30,3);WRITE('COMPAÑIA MOSQUITO');
        GOTOXY(26,5);WRITE('REPORTE RESUMEN DE VENTAS');
        GOTOXY(2,11);WRITE('NOMBRE PRODUCTO');
        GOTOXY(22,11);WRITE('CANTIDAD');
        GOTOXY(36,11);WRITE('PRECIO UNITARIO');
        GOTOXY(59,11);WRITE('VENTAS NETAS');
END;

PROCEDURE SALIDA_FINAL;
{PRODUCE REPORTE DE LAS VENTAS NETAS TOTALES}
BEGIN
        GOTOXY(7,FILA-1);CLREOL;
        GOTOXY(36,FILA+3);WRITE('VENTAS NETAS TOTALES.....',TOTAL:7:2);
END;

BEGIN {Comienzo del programa principal}
    CLRSCR;
    GOTOXY(10,10);WRITE('TECLEE EL NÚMERO DE VENTAS A PROCESAR');
    READ (N);
    CLRSCR;
    PANTALLA; {EJECUTA EL PROCEDIMIENTO LLAMADO PANTALLA}
    CLRSCR;
    FORMATO;{EJECUTA EL PROCEDIMIENTO LLAMADO FORMATO}
    TOTAL:=0;
    FILA:=13; {SE INICIALIZA FILA EN 13, CON EL PROPOSITO DE OBTENER
    UNA SALIDA MEJOR DISTRIBUIDA}
    FOR J:=1 TO N DO
        BEGIN
            REPEAT
                GOTOXY(7,FILA);
                READ(PRODUCTO);
                IF PRODUCTO=' ' THEN ERROR {VALIDACIÓN DEL NOMBRE DEL
                                            PRODUCTO}
            UNTIL PRODUCTO <>' ';
            REPEAT
                GOTOXY(24,FILA);
                READ(CANTIDAD);
```

```
            IF CANTIDAD <= 0  THEN ERROR;{VALIDACIÓN DE LA CANTIDAD
                                          DE ARTÍCULOS}
        UNTIL CANTIDAD >= 0.0;
        REPEAT
            GOTOXY(41,FILA);
            READ(PRECIO);
            IF PRECIO <= 0.0 THEN ERROR  {VALIDACIÓN DEL PRECIO DEL
                                          PRODUCTO}
        UNTIL PRECIO > 0.0;
        {CÁLCULO DE LAS VENTAS NETAS POR ARTÍCULO}
        VENTAS_NETAS:=CANTIDAD*PRECIO;
        TOTAL:=TOTAL+VENTAS_NETAS;
        GOTOXY(7,FILA);WRITE(PRODUCTO);
        GOTOXY(24,FILA);WRITE(CANTIDAD);
        GOTOXY(41,FILA);WRITE(PRECIO:7:2);
        GOTOXY(61,FILA);WRITE(VENTAS_NETAS:7:2);
        FILA:=FILA+1; {INCREMENTA LA FILA Y PROCEDE A LEER OTRO
                       REGISTRO}
    END;
    SALIDA_FINAL;
END.
```

EJERCICIO

Calcular áreas, longitudes y volúmenes de varias figuras geométricas mediante el empleo de procedimientos.

a) Estructura del algoritmo



b) Compilación PASCAL

```
PROGRAM AREAS;
{CALCULA AREAS, LONGITUDES Y VOLÚMENES}
USES
    CRT;
```

```
VAR
      I, PERIMETRO, H, OPCION, R: INTEGER;
      VC, APC, ASC, LC, ANGULO, AC: REAL;
      RES : CHAR;


PROCEDURE RADIO;
BEGIN
      WRITE (' ' : 8, 'ENTRE EL RADIO DEL CIRCULO : ' );
      READLN ( R ) ;
      IF ( R<= 0 ) THEN
         WHILE ( R<= 0 ) DO
         BEGIN
         WRITE (' ': 8,' ERROR EN DATOS, POR FAVOR TECLEE DE NUEVO : ');
         READLN (R);
         END;
         WRITELN;
END;

PROCEDURE AREA_CIRCULO;
BEGIN
      AC : = (3.1416) * SQR ( R );
      WRITELN (' ': 8, 'EL AREA DEL CIRCULO ES : ', AC:5 :4);
      WRITELN; WRITELN;
      WRITELN (' ': 8, 'OPRIMA CUALQUIER TECLA PARA CONTINUAR');
      READKEY;
END;

PROCEDURE LONGITUD_CIRCUNFERENCIA;
BEGIN
      LC: =2*(3.1416)*R;
      WRITELN (' ':8, 'LA LONGITUD DE LA CIRCUNFERENCIA ES: ', LC: 5: 4);
      WRITELN, WRITELN;
      WRITELN (' ': 8,'OPRIMA CUALQUIER TECLA PARA CONTINUAR');
      READKEY;
END;

PROCEDURE AREA_SECTOR_CIRCULAR;
BEGIN
      WRITELN(' ':8,'ENTRE  ANGULO DEL SECTOR CIRCULAR EN GRADOS:
      READLN (ANGULO);
      IF ANGULO<0 THEN
         WHILE ANGULO<0  DO
         BEGIN
```

```
            WRITE (' ':8,'ERROR EN DATOS, POR FAVOR TECLEE DE NUEVO:');
            READLN (ANGULO);
          END;
      ASC: =(3.1416)*SQR( R )*ANGULO/360;
      WRITELN (' ': 8,'EL AREA DEL SECTOR CIRCULAR ES: ', ASC: 5: 4,'
              UNIDADES CUADRADAS');
      WRITELN; WRITELN;
      WRITELN (' ': 8,'OPRIMA CUALQUIER TECLA PARA CONTINUAR');
      READKEY;
END;

PROCEDURE AREA_POLIGONO_CIRCUNSCRITO;
BEGIN
      WRITE (' ': 8,'ENTRE EL PERIMETRO DEL POLIGONO CIRCUNSCRITO: ');
      READLN (PERIMETRO);
      IF PERIMETRO <= 0 THEN
         WHILE PERIMETRO <= 0  DO
         BEGIN
           WRITE (' ': 8,'ERROR EN DATOS, POR FAVOR TECLEE DE NUEVO: ');
           READLN (PERIMETRO);
         END;
      APC: =R*PERIMETRO /2;
      WRITELN (' ':8,'EL AREA DEL POLIGONO CIRCUNSCRITO ES: ',APC: 5: 4):
      WRITELN; WRITELN;
      WRITELN;
      WRITELN (' ': 8, ' OPRIMA CUALQUIER TECLA PARA CONTINUAR ' );
      READKEY;
END;

PROCEDURE VOLUMEN_CILINDRO;
BEGIN
      WRITE (' ':8,'ENTRE LA ALTURA DEL CILINDRO: ');
      READLN (H);
      IF H <= 0 THEN
         WHILE H <= 0  DO
         BEGIN
           WRITE (' ':8,'ERROR EN ALTURA, POR FAVOR TECLEE DE NUEVO:');
           READLN ( H );
         END;
      VC: =(3.1416)*SQR( R )*H;
      WRITELN (' ':8,'EL VOLUMEN DEL CILINDRO ES: ', VC: 5: 4,'UNIDADES
      CUBICAS');
      WRITELN; WRITELN;
      WRITELN (' ': 8, ' OPRIMA CUALQUIER TECLA PARA CONTINUAR ' );
```

```
        READKEY;
END;

BEGIN
        WRITELN (' ': 8, 'PROGRAMA QUE CALCULA AREAS, LONGITUDES Y
        VOLUMENES MEDIANTE EL ENPLEO DE PROCEDIMIENTOS ');
        DELAY (10000); CLRSCR;
        REPEAT
            RADIO; CLRSCR;
            AREA_CIRCULO; CLRSCR;
            LONGITUD_CIRCUNFERENCIA; CLRSCR;
            AREA_SECTOR_CIRCULAR; CLRSCR;
            AREA_POLIGONO_CIRCUNSCRITO; CLRSCR;
            VOLUMEN_CILINDRO; CLRSCR;
            WRITELN (' ': 8, 'DESEA REALIZAR MAS CALCULOS (S/N)? ' );
            READ (RESP);
        UNTIL RES=N;
END.
```

EJERCICIO

Se tiene un lote de N registros con la siguiente información: código, nombre, salario bruto y total de días trabajados. Se desea calcular el interés sobre la cesantía, como el total pagado por intereses de la siguiente forma :

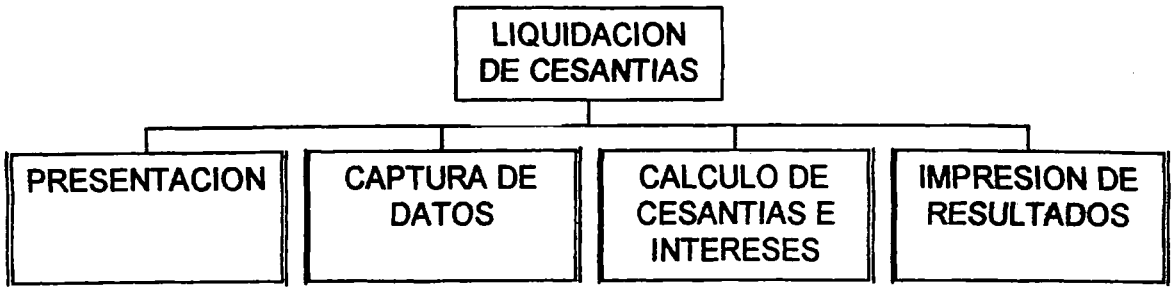
$$Interes\ a\ la\ cesantia = \frac{Cesantia * 12\%\ anual * total\ dias\ trabajados}{360\ dias}$$

$$Cesantia = \frac{Salario\ bruto * total\ dias\ trabajados}{360\ dias}$$

Nota: si el total de días trabajados es inferior a 30, no se tiene derecho a cesantías.

Solución

a) Estructura del algoritmo



Entrada : Leer el límite de datos a procesar.
Inicializar el total de intereses en cero.
Leer código, nombre, salario bruto y total de días trabajados.

Proceso : Se elaboran 4 procedimientos para:

- Presentación del trabajo, el cual no contiene argumentos por que no requiere de variables de entrada como de salida y tiene solo fines de documentación.
- Captura de datos. Se solicitan los datos por pantalla.
- Cálculo de cesantías e intereses a las cesantías.
- Impresión de resultados. La salida va acompañada de los títulos correspondientes.

Salida : Imprimir código, nombre, salario bruto, total de días trabajados, cesantías e intereses a las cesantías y el total de intereses.

b) Variables a utilizar

N = Límite de datos a procesar
COD = Código
NOM = Nombre
SB = Salario bruto
TDT = Total de días trabajados
CES = Cesantías
INT = Intereses a las cesantías
TINT = Total de intereses.

c) Algoritmo Principal

```
Paso 1      INICIO
Paso 2        LLAMAR PRESENTACION
Paso 3        Leer N
Paso 4        TINT ← 0
Paso 5        PARA I DE 1 HASTA N HACER
```

```
                    LLAMAR CAPTURA_DATOS(COD,NOM,SB,TDT)
                    LLAMAR CESANTIAS(SB,TDT,CES,INT)
                    TINT ← TINT + INT
                    LLAMAR RESULTADOS(COD, NOM, SB, TDT,  CES, INT)
                FIN_PARA
Paso 6          IMPRIMIR "El total de intereses es", TINT
Paso 7      FIN


PROCEDIMIENTO PRESENTACION
Paso 1      INICIO
Paso 2          IMPRIMIR "UNIVERSIDAD NACIONAL DE COLOMBIA"
Paso 3          IMPRIMIR "SEDE MANIZALES"
Paso 4          IMPRIMIR "CALCULO DE CESANTIAS"
Paso 5          IMPRIMIR "MANIZALES, ENERO DE 1990"
Paso 6      FIN


PROCEDIMIENTO CAPTURA_DATOS (COD, NOM, SB, TDT)
Paso 1      INICIO
Paso 2          ESCRIBIR "Teclee el código del trabajador"
Paso 3          Leer COD
Paso 4          ESCRIBIR" Teclee el nombre del trabajador"
Paso 5          Leer NOM
Paso 6          ESCRIBIR "Teclee salario bruto"
Paso 7          Leer SB
Paso 8          ESCRIBIR "Teclee total de días trabajados"
Paso 9          Leer TDT
Paso 10     FIN


PROCEDIMIENTO CESANTIAS (SB, TDT, CES, INT)
Paso 1      INICIO
Paso 2          SI TDT >= 30
                    ENTONCES CES ← SB * TDT / 360
                    INT ← CES * 0.12 * TDT / 360
                    SINO CES ← 0
                    INT ← 0
                FINSI
Paso 3      FIN

PROCEDIMIENTO RESULTADOS (COD, NOM, SB, TDT, CES, INT)
Paso 1      INICIO
```

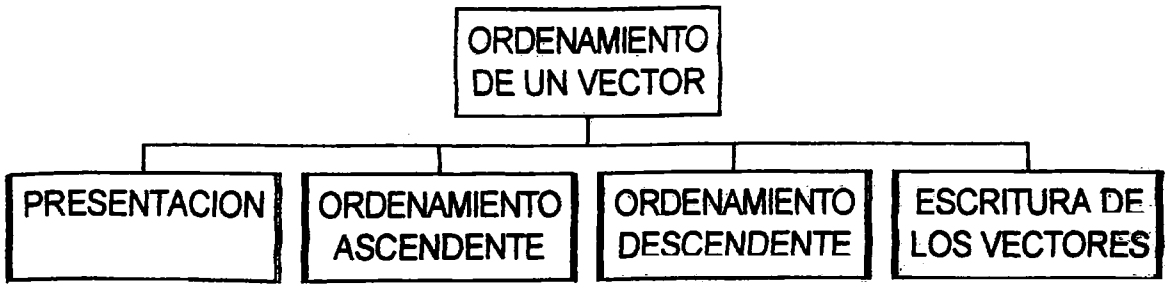
```
Paso 2        IMPRIMIR "Código del trabajador", COD
Paso 3        IMPRIMIR "Nombre del trabajador", NOM
Paso 4        IMPRIMIR "Salario bruto", SB
Paso 5        IMPRIMIR "Total días trabajados", TDT
Paso 6        IMPRIMIR "Cesantías", CES
Paso 7        IMPRIMIR "Intereses a las cesantías", INT
Paso 8      FIN
```

EJERCICIO

Leer un vector K de N elementos y ordenarlo en forma ascendente y descendente. Imprimir el vector original y los vectores ordenados.

Solución

a) Estructura del Algoritmo



Entrada : Leer el límite N y el vector K a ser ordenado.
Proceso : Se utilizan cuatro subalgoritmos para realizar lo siguiente :

- Presentación del ejercicio
- Ordenamiento ascendente
- Ordenamiento descendente
- Impresión de los vectores

Salida : Imprimir el vector original y los vectores ordenados.

b) Variables a Utilizar

K = Nombre del vector
N = Número de elementos del vector
T = Variable temporal
J = Posición del vector, a su vez, índice de la estructura PARA.

c) Algoritmo Principal

```
Paso 1      INICIO
Paso 2        LLAMAR PORTADA
Paso 3        LEER N
Paso 4        PARA J DE 1 HASTA N HACER
                  LEER K(J)
              FIN PARA
Paso 5        IMPRIMIR "EL VECTOR ORIGINAL ES".
Paso 6        LLAMAR SALIDA(N,K)
Paso 7        LLAMAR ASCENDENTE(N,K)
Paso 8        IMPRIMIR "EL VECTOR ORDENADO ASCENDENTEMENTE ES"
Paso 9        LLAMAR SALIDA(N,K)
Paso 10       LLAMAR DESCENDENTE(N,K)
Paso 11       IMPRIMIR "EL VECTOR ORDENADO DESCENDENTEMENTE
                        ES"
Paso 12       LLAMAR SALIDA(N,K)
Paso 13     FIN
```

```
PROCEDIMIENTO PORTADA
Paso 1      INICIO
Paso 2        IMPRIMIR "ORDENAMIENTO DE UN VECTOR"
Paso 3        IMPRIMIR "MEDIANTE LA UTILIZACIÓN DE"
Paso 4        IMPRIMIR "PROCEDIMIENTOS"
Paso 5      FIN
```

```
PROCEDIMIENTO SALIDA(N,K)
Paso 1      INICIO
Paso 2        PARA J DE 1 HASTA N HACER
                  IMPRIMIR K(J)
              FIN_PARA
Paso 3      FIN
```

```
PROCEDIMIENTO ASCENDENTE(N,K)
Paso 1      INICIO
Paso 2        J ← 1
Paso 3        REPETIR
                  SI K(J) > K(J+1)
                      ENTONCES T ← K(J)
                      K(J) ← K(J+1)
```

```
                    K(J+1) ← T
                    J ← 1
                    SINO J ← J+1
                FINSI
            HASTA_QUE J=N
Paso 4      FIN
```

```
PROCEDIMIENTO DESCENDENTE(N,K)
Paso 1      INICIO
Paso 2        J ← 1
Paso 3        REPETIR
                SI K(J) < K(J+1)
                  ENTONCES T ← K(J)
                  K(J) ← K(J+1)
                  K(J+1) ← T
                  J ← 1
                  SINO J ← J+1
                FINSI
              HASTA_QUE J=N
Paso 4      FIN
```

EJERCICIO

Una compañía de sistemas efectúo un estudio relacionando los puntos de una prueba de aptitud con la productividad en computación del nuevo personal.

CÓD. EMPLEADO	APTITUD (X)	PRODUCTIVIDAD (Y)
1	7	35
2	13	53
3	17	75
4	19	81
5	21	85
6	23	93
7	27	97

La compañía desea obtener la ecuación de la recta que pueda usarse para predecir la productividad de futuros trabajadores, así como la media aritmética y desviación típica de las variables X, Y, de acuerdo con las siguientes fórmulas:

$$\overline{X} = \frac{\sum X_I}{N} \qquad Dx = \sqrt{\frac{\sum X^2}{N} - \overline{X}^2} \qquad Dy = \sqrt{\frac{\sum Y^2}{N} - \overline{Y}^2}$$

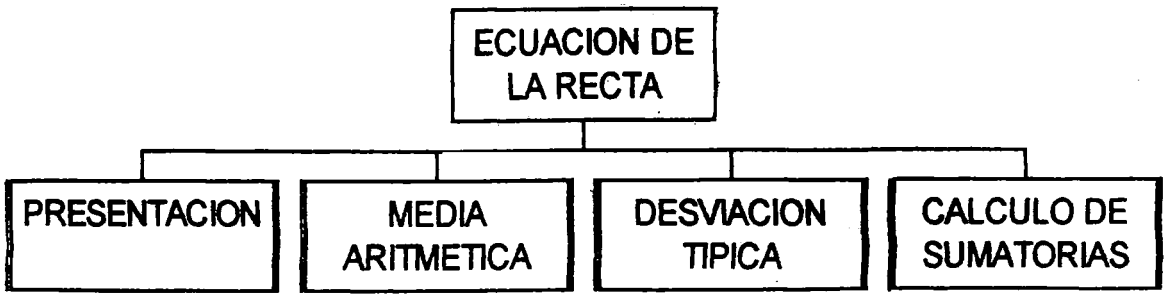
$$A = \frac{\sum XY + N * \overline{X} * \overline{Y}}{\sum X^2 - N * \overline{X}^2} \qquad b = \overline{Y} - A * \overline{X}$$

Ecuacion de la recta: $Y = AX + b$

Imprimir toda la información leída como la calculada, mediante el empleo de procedimientos.

Solución

a) Estructura del algoritmo



Entrada : Leer el límite de datos
Leer la identificación de cada empleado
Leer la matriz de datos compuesta de N filas por 2 columnas, correspondientes a la aptitud y a la productividad.

Proceso : Se emplean cuatro subalgoritmos para realizar lo siguiente:

- Presentación del ejercicio
- Cálculo de la media aritmética
- Cálculo de la desviación típica
- Cálculo de las sumatorias X, Y, y Sumatoria de X^2.

Salida : Imprimir la media aritmética de X, Y. La desviación típica de X, Y, y la ecuación de la recta Y.

b) Variables a utilizar

N = Límite de empleados
I = Variable índice de la estructura **PARA**
CODE = código del Empleado

DATOS = Matriz de datos
MED = Media Aritmética
DES = Desviación Típica
SUMXC = Sumatoria de los valores de X elevados al cuadrado
SUMXY = Sumatoria de los valores X*Y
SUM = Sumatoria
SUMD = Sumatoria de la Desviación
SDC = Sumatoria de la Desviación al cuadrado

c) Algoritmo principal

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        PARA I DE 1 HASTA N HACER
                 LEER CODE(I)
                 IMPRIMIR CODE(I)
              FIN_PARA
Paso 4        PARA I DE 1 HASTA N HACER
                 PARA J DE 1 HASTA 2 HACER
                    LEER DATOS(I,J)
                    IMPRIMIR DATOS(I,J)
                 FIN_PARA
              FIN_PARA
Paso 5        LLAMAR MEDIA(N, DATOS, MED)
Paso 6        LLAMAR DESVIACIÓN(N, DATOS, MED, DES)
Paso 7        LLAMAR SUMATORIAS(N, DATOS, SUMXY, SUMXC)
Paso 8        IMPRIMIR "La media aritmética de X es", MED(1)
Paso 9        IMPRIMIR "La desviación típica de X es", DES(1)
Paso 10       IMPRIMIR "La media aritmética de Y es", MED(2)
Paso 11       IMPRIMIR "La desviación típica de Y es", DES(2)
Paso 12       A ← (SUMXY + N * MED(1) * MED(2))/(SUMXC - N *
              MED(1)↑2)
Paso 13       B ← MED(2) - A * MED(1)
Paso 14       IMPRIMIR "La ecuación de la recta para predecir la productividad es
              "Y= ",A," X +",B
Paso 15     FIN
```

```
PROCEDIMIENTO PRESENTACION
Paso 1      INICIO
Paso 2        IMPRIMIR "Cálculo estadístico. Ecuación de la"
```

```
Paso 3          IMPRIMIR "Recta para predecir la productividad"
Paso 4          IMPRIMIR "De los Trabajadores de la"
Paso 5          IMPRIMIR "Compañía XXX"
Paso 6      FIN
```

```
PROCEDIMIENTO MEDIA(N, DATOS, MED)
Paso 1      INICIO
Paso 2          PARA J DE 1 HASTA 2 HACER
                    SUM ← 0
                    PARA I DE 1 HASTA N HACER
                        SUM ← SUM + DATOS(I,J)
                    FIN_PARA
                    MED(J) ← SUM / N
                FIN_PARA
Paso 3      FIN
```

```
PROCEDIMIENTO DESVIACIÓN(N, DATOS, MED, DES)
Paso 1      INICIO
Paso 2          PARA J DE 1 HASTA 2 HACER
                    SUMD ← 0
                    PARA I DE 1 HASTA N HACER
                        SUMD ← SUMD + DATOS(I,J)↑2
                    FIN_PARA
                    SDC(J) ← SUMD / N
                    DES(J) ← (SDC(J) - MED(J)^|2)↑(1/2)
                FIN_PARA
Paso 3      FIN
```

```
PROCEDIMIENTO SUMATORIAS(N, DATOS, SUMXY, SUMXC)
Paso 1      INICIO
Paso 2          SUMXY ← 0
Paso 3          PARA I DE 1 HASTA N HACER
                    SUMXY ← SUMXY + DATOS(I,1) * DATOS(I,2)
                FIN_PARA
Paso 4          SUMXC ← 0
Paso 5          PARA I DE 1 HASTA N HACER
                    SUMXC ← SUMXC + DATOS(I,1)↑2
                FIN_PARA
Paso 6      FIN
```

PROBLEMAS RESUELTOS

EJERCICIO

Calcular la depreciación de activos por el método de la línea recta. La información por cada registro consta de: año de compra del activo, costo del activo y vida útil. La salida debe contener la misma información leída, además de la depreciación anual, depreciación acumulada y saldo.
Ejemplo de depreciación en línea recta: Depreciar un vehículo (a 5 años) cuyo precio es de $1'000.000.

$$\text{Depreciación 1er año} = \frac{1000000}{5} = 200000$$

$$\text{Depreciación 2° año} = \frac{1000000}{5} = 200000$$

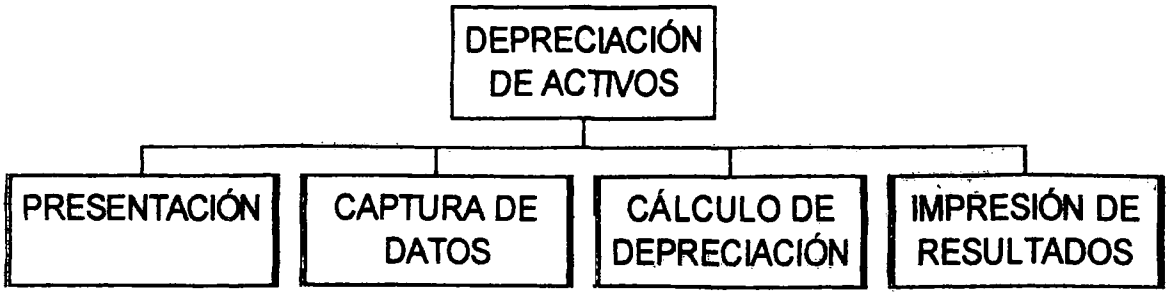
. .
. .
. .
. .
. .

$$\text{Depreciación 5° año} = \frac{1000000}{5} = 200000$$

La depreciación por el método de la línea recta es constante durante la vida útil del activo.

Solución

a) Estructura del algoritmo



Entrada : Leer el limite de datos a procesar.
Inicializar la depreciación acumulada en cero.
Leer año de compra del activo, costo del activo y vida útil.
Proceso : Se elaboran los siguientes procedimientos
- Presentación del informe

- Captura de datos
- Cálculo de la depreciación
- Impresión de resultados.

Salida : Imprimir año de compra del activo, costo, vida útil, depreciación, depreciación acumulada y saldo.

b) Variables a utilizar

N = Límite de datos a procesar
AÑO = Año de compra del activo
COS = Costo del activo
VU = Vida útil
DEP = Depreciación
DEPAC = Depreciación acumulada
SDO = Saldo

c) Algoritmo Principal

```
Paso 1      INICIO
Paso 2        LLAMAR PRESENTACION
Paso 3        ESCRIBIR "Teclee el número de datos a  procesar".
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                 DEPAC ← 0
                 LLAMAR LEER_DATOS(AÑO,COS,VU)
                 IMPRIMIR "Costo del Activo", COS
                 IMPRIMIR " Vida Útil del Activo", VU
                 PARA L DE 1 HASTA VU HACER
                    LLAMAR DEPRECIACIÓN(COS, VU, DEP,  DEPAC, SDO)
                    LLAMAR IMPRIMIR_DATOS(AÑO,DEP, DEPAC, SDO)
                    AÑO ← AÑO + 1
                 FIN_PARA
              FIN_PARA
Paso 6      FIN
```

```
PROCEDIMIENTO PRESENTACION
Paso 1      INICIO
Paso 2        IMPRIMIR "Cálculo de Depreciación de Activos"
Paso 3        IMPRIMIR "Por el método de la Línea Recta"
Paso 4        IMPRIMIR "Mediante la aplicación de la metodología"
Paso 5        IMPRIMIR "   DISEÑO DESCENDENTE   "
Paso 6      FIN
```

```
PROCEDIMIENTO LEER_DATOS (AÑO, COS, VU)
Paso 1      INICIO
Paso 2         ESCRIBIR "Teclee el año de compra del activo"
Paso 3         LEER AÑO
Paso 4         ESCRIBIR "Teclee el costo del activo"
Paso 5         LEER COS
Paso 6         ESCRIBIR "Teclee la vida útil del activo"
Paso 7         LEER VU
Paso 8      FIN
```

```
PROCEDIMIENTO DEPRECIACIÓN (COS, VU, DEP, DEPAC, SDO)
Paso 1      INICIO
Paso 2         DEP ← COS / VU
Paso 3         DEPAC ← DEPAC + DEP
Paso 4         SDO ← COS - DEPAC
Paso 5      FIN
```

```
PROCEDIMIENTO IMPRIMIR_DATOS (AÑO, DEP, DEPAC, SDO)
Paso 1      INICIO
Paso 2         IMPRIMIR "Año de depreciación del Activo", AÑO
Paso 3         IMPRIMIR "Depreciación", DEP
Paso 4         IMPRIMIR "Depreciación acumulada", DEPAC
Paso 5         IMPRIMIR "Saldo", SDO
Paso 6      FIN
```

EJERCICIO

Calcular la Depreciación de Activos por el método de la suma de los dígitos de los años.
Ejemplo. Depreciar un vehículo (a 5 años) cuyo precio es de $1'000.000=
La suma de los dígitos de los años = 1+2+3+4+5 =15
Se suman los dígitos de los años de acuerdo a la vida útil.

Depreciación 1er año = $\frac{5}{15}$*1000000

Depreciación 2° año = $\frac{4}{15}$*1000000

. .
. .
. .

Depreciación 5° año = $\frac{1}{15}$*1000000

Solución

a) Análisis

Tanto la entrada como la salida de datos son idénticas al ejercicio anterior y por estar ya definidas en los procedimientos no se especificarán nuevamente.

Proceso: Se acoplan dos procedimientos, uno de presentación para indicar cuál es el método de depreciación a utilizar, el otro para el cálculo de la depreciación correspondiente a este método.
Se aplican las siguientes fórmulas adicionales:
Sumatoria de dígitos = (1+vida útil)*vida útil/ 2.
Disminución de la vida útil = vida útil + 1 - el valor del índice de la estructura interna **PARA**.

b) Variables a utilizar

N = Límite de datos a procesar
AÑO = Año de compra del Activo
COS = Costo del Activo
VU = Vida útil
DEP = Depreciación
DEPAC = Depreciación acumulada
SDO = Saldo
SUMD = Sumatoria de dígitos
DVU = Disminución de la vida útil (NUMERADOR)
J, L = Índices de las estructuras PARA

c) Algoritmo Principal

```
Paso 1      INICIO
Paso 2        LLAMAR OTRA_PRESENTACION
Paso 3        ESCRIBIR "Teclee el número de datos a procesar"
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                  DEPAC ← 0
                  LLAMAR LEER_DATOS(AÑO,COS,VU)
                  IMPRIMIR "Costo del Activo", COS
                  IMPRIMIR "Vida Útil del Activo", VU
                  SUMD ← (1 + VU) * VU / 2
                  PARA L DE 1 HASTA VU HACER
                     LLAMAR SUMA_DIGITOS(L,COS,VU,DEP,DEPAC,SDO)
                     LLAMAR IMPRIMIR_DATOS(AÑO,DEP, DEPAC, SDO)
                     AÑO ← AÑO + 1
```

```
                FIN_PARA
            FIN_PARA
Paso 6      FIN
```

```
PROCEDIMIENTO OTRA_PRESENTACION
Paso 1      INICIO
Paso 2        IMPRIMIR "Cálculo de la Depreciación de Activos"
Paso 3        IMPRIMIR "Por el método de la Suma de los"
Paso 4        IMPRIMIR "Dígitos de los Años mediante la"
Paso 5        IMPRIMIR "Aplicación del Diseño Descendente"
Paso 6      FIN
```

```
PROCEDIMIENTO SUMA_DIGITOS (L, COS, VU, DEP, DEPAC, SDO)
Paso 1      INICIO
Paso 2        DVU ← VU + 1 - L
Paso 3        DEP ← DVU / SUMD * COS
Paso 4        DEPAC ← DEPAC + DEP
Paso 5        SDO ← COS - DEP
Paso 6      FIN
```

Otra solución al ejercicio de Depreciación de Activos por el método de la Suma de los Dígitos de los Años es la siguiente:

a) Análisis

Proceso: El cálculo de la suma de los dígitos de los años se realiza mediante el empleo de una estructura **PARA**.
Se crea otro procedimiento para el cálculo de la suma de los dígitos de los años; los demás procedimientos ya están definidos en soluciones anteriores, por lo tanto se hará uso de ellos.

Algoritmo Principal

```
Paso 1      INICIO
Paso 2        LLAMAR OTRA_PRESENTACION
Paso 3        ESCRIBIR "Teclee el número de datos a procesar"
Paso 4        LEER N
Paso 5        PARA J DE 1 HASTA N HACER
                DEPAC ← 0
                SUMD ← 0
                LLAMAR LEER_DATOS(AÑO, COS, VU)
```

```
                    IMPRIMIR "Costo del Activo", COS
                    IMPRIMIR "Vida Útil del Activo", VU
                    PARA L DE 1 HASTA VU HACER
                        SUMD ← SUMD + L
                    FIN_PARA
                    REPETIR
                       LLAMAR DEPRECIACION_DIGITOS(COS, VU, SUMD,
                       DEP, DEPAC, SDO)
                       LLAMAR IMPRIMIR_DATOS(AÑO, DEP, DEPAC, SDO)
                       VU ← VU - 1
                       AÑO ← AÑO + 1
                    HASTA_QUE VU = 0
                 FIN_PARA
Paso 6      FIN

PROCEDIMIENTO DEPRECIA_DIGITO (COS, VU, SUMD, DEP,DEPAC, SDO)
Paso 1      INICIO
Paso 2         DEP ← VU / SUMD * COS
Paso 3         DEPAC ← DEPAC + DEP
Paso 4         SDO ← COS - DEPAC
Paso 5      FIN
```

EJERCICIO

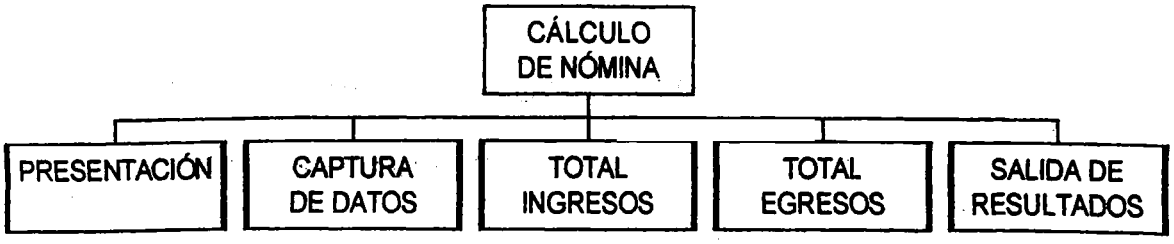
Se tiene un lote de registros cada uno con la siguiente información: código, nombre, salario bruto y número de hijos. El proceso termina cuando el código ≤ 0.
Calcular el salario neto, el valor devengado por bonificación anual, auxilio de transporte y subsidio familiar. Calcular además, el total de ingresos, total de egresos y total pagado por toda la nómina.

La bonificación anual es del 50% para salarios brutos inferiores a $60.000, en caso contrario es del 30%.
El auxilio de transporte es de $3.000 para salarios brutos inferiores a $80.000.
El subsidio familiar es de $2000 por hijo, para salarios brutos inferiores a $100.000.
Los seguros sociales y la Retención en la fuente se calculan según la siguiente tabla:

SALARIO BRUTO	SEGURO SOCIAL	RETENCIÓN EN LA FUENTE
Hasta 40.000	1% del S.B.	0% del S.B.
40.001 - 60.000	3% del S.B.	2% del S.B.
60.001 - 80.000	5% del S.B.	4% del S.B.
80.001 - 100.000	7% del S.B.	6% del S.B.
100.001 - y más	10% del S.B.	8% del S.B.

Solución

a) Estructura del algoritmo



Entrada : Leer código, nombre, salario bruto y húmero de hijos.
Inicializar el acumulador de salarios netos.
Se deben procesar tantos registros mientras el código sea menor o igual a cero.
Proceso : Se emplean cinco subalgoritmos para realizar lo siguiente:
- Presentación del ejercicio
- Capturar datos
- Totalizar ingresos
- Totalizar egresos
- Presentación de los cálculos hallados.

Salida : Imprimir todo lo leído y calculado

b) Variables a utilizar

COD = Código
NOM = Nombre
SB = Salario bruto
NH = Número de hijos
BON = Bonificación anual
ST = Subsidio de transporte
SF = Subsidio familiar
SS = Seguro social
RET = Retención en la fuente
TOTIN = Total de Ingresos
TOTEG = Total de egresos
SN = Salario neto
TOTNOM = Total nómina pagada

c) Algoritmo principal

Paso 1 **INICIO**
Paso 2 **LLAMAR** PORTADA
Paso 3 TOTNOM ←

```
Paso 4          LLAMAR CAPTURA_DATOS(COD,NOM,SB,NH)
Paso 5          MIENTRAS COD > 0 HACER
                    LLAMAR INGRESOS(SB,NH,BON,ST,SF,TOTIN)
                    LLAMAR EGRESOS(SB,SS,RET,TOTEG)
                    SN ← TOTIN - TOTEG
                    TOTNOM ← TOTNOM + SN
                    LLAMAR SALIDA(COD, NOM, SB, NH, BON, ST, SF, SS, RET,
                    TOTIN, TOTEG, SN)
                    LLAMAR CAPTURA  DATOS(COD, NOM, SB, NH)
                FIN_MIENTRAS
Paso 6          IMPRIMIR "El total pagado por nómina es", TOTNOM
Paso 7      FIN
```

```
PROCEDIMIENTO PORTADA
Paso 1      INICIO
Paso 2          IMPRIMIR "Compañía XYZ"
Paso 3          IMPRIMIR "Cálculo de la Nómina Mensual"
Paso 4          IMPRIMIR "Correspondiente a Enero de 1990"
Paso 5      FIN
```

```
PROCEDIMIENTO CAPTURA_DATOS (COD, NOM, SB, NH)
Paso 1      INICIO
Paso 2          ESCRIBIR "Teclee el código del trabajador"
Paso 3          LEER COD
Paso 4          ESCRIBIR "Teclee el nombre del trabajador"
Paso 5          LEER NOM
Paso 6          ESCRIBIR "Teclee el salario bruto"
Paso 7          LEER SB
Paso 8          ESCRIBIR "Teclee el número de hijos"
Paso 9          LEER NH
Paso 10     FIN
```

```
PROCEDIMIENTO INGRESOS(SB, NH, BON, ST, SF, TOTIN)
Paso 1      INICIO
Paso 2          SI SB < 60.000
                    ENTONCES BON ← SB * 0.5
                    SINO BON ← SB * 0.3
                FINSI
Paso 3          SI SB < 80.000
                    ENTONCES ST ← 3.000
                    SINO ST ← 0
```

```
                FINSI
Paso 4          SI SB < 100.000
                    ENTONCES SF ← NH * 2000
                    SINO SF ← 0
                FINSI
Paso 5          TOTIN ← SB + BON + ST + SF
Paso 6      FIN
```

```
PROCEDIMIENTO EGRESOS (SB, SS, RET, TOTEG)
Paso 1      INICIO
Paso 2          SI SB <= 40.000
                    ENTONCES SS ← SB * 0.01
                    RET ← 0
                    SINO SI SB <= 60.000
                       ENTONCES SS ← SB * 0.03
                       RET ← SB * 0.02
                       SINO SI SB <= 80.000
                           ENTONCES SS ← SB * 0.05
                           RET ← SB * 0.04
                           SINO SI SB <= 100.000
                               ENTONCES SS ← SB * 0.07
                               RET ← SB * 0.06
                               SINO SS ← SB * 0.1
                               RET ← SB * 0.08
                           FINSI
                       FINSI
                    FINSI
                FINSI
Paso 3          TOTEG ← SS + RET
Paso 4      FIN
```

```
PROCEDIMIENTO SALIDA (COD, NOM, SB, NH, BON, ST, SF, SS, RET, TOTIN,
                      TOTEG, SN)
Paso 1      INICIO
Paso 2          IMPRIMIR "Código del trabajador", COD
Paso 3          IMPRIMIR "Nombre del trabajador", NOM
Paso 4          IMPRIMIR "Salario bruto", SB
Paso 5          IMPRIMIR "Número de hijos", NH
Paso 6          IMPRIMIR "Bonificación", BON
Paso 7          IMPRIMIR "Subsidio de transporte", ST
Paso 8          IMPRIMIR "Subsidio familiar", SF
```

```
Paso 9        IMPRIMIR "Seguro social", SS
Paso 10       IMPRIMIR "Retención en la fuente", RET
Paso 11       IMPRIMIR "Total de Ingresos", TOTIN
Paso 12       IMPRIMIR "Total de Egresos", TOTEG
Paso 13       IMPRIMIR "Salario Neto", SN
Paso 14     FIN
```

Observación:

Uno de los estilos de programación que permite mayor interacción entre la máquina y el usuario, se logra mediante la utilización de MENÚS como herramienta de programación, de tal manera que al usuario se le facilita bastante la operación del sistema, limitándose a escoger la opción que desea ejecutar; simultáneamente brinda seguridad al sistema de información, ya que se puede condicionar su operación a ciertas opciones obligatorias o MENÚS obligatorios.
Su elaboración es sencilla y consiste en definir una serie de actividades, las cuales son agrupadas usualmente por medio de subprogramas, de tal forma que al ser seleccionada una opción del menú, lo que realmente se ejecuta es una subrutina o un procedimiento.

EJERCICIO

Para un lote de N datos no superior a 1.000, calcular el rango, media aritmética, desviación media, desviación estándar y varianza, según las siguientes fórmulas:

$$RANGO = DATO\ MAYOR - DATO\ MENOR$$

$$MEDIA\ ARITMÉTICA = \bar{X} = \frac{\sum X_I}{N}$$

$$DESVIACIÓN\ MEDIA = \frac{\sum |X_I - \bar{X}|}{N}$$

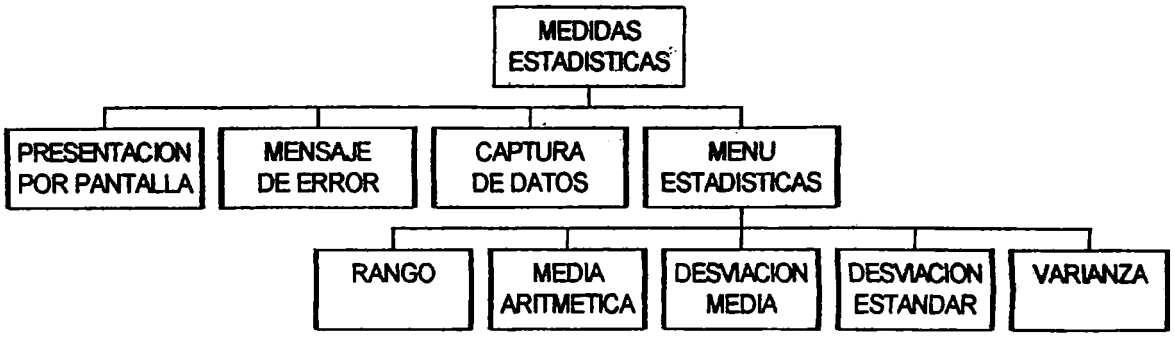
$$DESVIACIÓN\ ESTÁNDAR = \sqrt{\frac{\sum (X_I - \bar{X})^2}{N}}$$

$$VARIANZA = DESVIACIÓN\ ESTÁNDAR^2 = \frac{\sum (X_I - \bar{X})^2}{N}$$

Calcular e imprimir lo anterior mediante la utilización de un menú.

Solución

a) Estructura del Algoritmo



Entraaa : Leer el límite y los datos a procesar, los cuales son almacenados en un vector. Inicializar el acumulador de datos en cero.

Proceso : Se utilizan los siguientes subalgoritmos

- Presentación por pantalla
- Mensaje de error
- Menú estadísticas
- Captura de datos
- Cálculo del rango
- Cálculo de la media aritmética
- Cálculo de la desviación media
- Cálculo de la desviación estándar
- Cálculo de la varianza

Se emplean validaciones de datos, tanto en el procedimiento menú para controlar que la operación seleccionada se encuentre dentro del rango de alternativas permitidas, como en el procedimiento de captura de datos para validar que el total de datos no exceda de 1.000 tal como reza el enunciado del ejercicio. En ambas validaciones cuando se detecta una inconsistencia, se activa el procedimiento Error el cual envía un mensaje al usuario.

Todo el ejercicio se desarrolla con base en procedimientos, buscando simplicidad y mayor entendimiento en cuanto al diseño del mismo se refiere.

Salida : Cualquiera de las alternativas del MENÚ seleccionada genera su propia salida impresa, algunas por pantalla (escribir), otras por impresora (imprimir).

b) Variables a utilizar

N = Límite de datos a procesar
X = Dato a ser procesado
Respuesta = Valor asumido cuando se produce el mensaje de error.
Opción = Valor seleccionado del menú
Mayor = Elemento mayor de los datos analizados
Menor = Elemento menor de los datos analizados
R = Rango

SUM = Sumatoria de elementos
XMED = Media aritmética
DM = Desviación estándar
S = Varianza

c) Algoritmo principal

```
{ CÁLCULOS ESTADÍSTICOS MEDIANTE LA UTILIZACIÓN DE MENÚ }
Paso 1      INICIO
Paso 2        LLAMAR PANTALLA
Paso 3        REPETIR
                 LLAMAR MENU(ERROR, OPCIÓN)
                 EN CASO DE OPCIÓN HACER
                    1: LLAMAR RANGO(N,X,MAYOR,MENOR,R,RESPUESTA)
                    2: LLAMAR MEDIA(CAPTURA, MED, RESPUESTA)
                    3: LLAMAR DESMED(CAPTURA, MED, DM, RESPUESTA)
                    4: LLAMAR ESTÁNDAR(SUMDES, DESV, RESPUESTA)
                    5: LLAMAR VARIANZA(SUMDES, S, RESPUESTA)
                 FIN_CASO
              HASTA_QUE OPCIÓN = 6
Paso 4      FIN


PROCEDIMIENTO PANTALLA
{ PRESENTACION INICIAL DEL PROGRAMA }
Paso 1      INICIO
Paso 2        ESCRIBIR "UNIVERSIDAD NACIONAL"
Paso 3        ESCRIBIR "PROGRAMA PARA CALCULAR"
Paso 4        ESCRIBIR "ALGUNAS MEDIDAS ESTADÍSTICAS"

Paso 5        ESCRIBIR "A TRAVÉS DEL MANEJO DE"
Paso 6        ESCRIBIR "      MENÚ"
Paso 7        ESCRIBIR "AUTOR:  ALONSO TAMAYO ALZATE"
Paso 8        ESCRIBIR "MANIZALES, OCTUBRE DE 1995"
Paso 9      FIN


PROCEDIMIENTO ERROR(RESPUESTA)
{ PRODUCE MENSAJE DE ERROR POR PANTALLA }
Paso 1      INICIO
Paso 2        ESCRIBIR "ERROR EN DATOS. PRESIONE LA TECLA
              ENTER PARA CONTINUAR"
Paso 3        LEER RESPUESTA
```

```
Paso 4          ESCRIBIR "
Paso 5      FIN


PROCEDIMIENTO MENU(ERROR, OPCIÓN)
{ CREA EL MENÚ DE ESTADÍSTICA }
Paso 1      INICIO
Paso 2          ESCRIBIR "MENÚ DE ESTADÍSTICA"
Paso 3          ESCRIBIR "              "
Paso 4          ESCRIBIR " 1 - RANGO"
Paso 5          ESCRIBIR " 2 - MEDIA ARITMÉTICA"
Paso 6          ESCRIBIR " 3 - DESVIACIÓN MEDIA"
Paso 7          ESCRIBIR " 4 - DESVIACIÓN ESTÁNDAR"
Paso 8          ESCRIBIR " 5 - VARIANZA"
Paso 9          ESCRIBIR " 6 - FIN DEL PROCESO"
Paso 10         ESCRIBIR "TECLEE LA OPCIÓN SELECCIONADA"
Paso 11         REPETIR
                    LEER OPCIÓN
                    SI (OPCIÓN < 1) o (OPCIÓN > 6)
                        ENTONCES LLAMAR ERROR(RESPUESTA)
                    FINSI
                HASTA_QUE (OPCIÓN >= 1) y (OPCIÓN <= 6)
Paso 12     FIN


PROCEDIMIENTO CAPTURA(N,X)
{ PRODUCE CAPTURA DE DATOS }
Paso 1      INICIO
Paso 2          ESCRIBIR "SON DATOS NUEVOS PARA PROCESAR? (S/N)"
Paso 3          LEER RESPUESTA
Paso 4          SI (RESPUESTA="S") O (RESPUESTA="s")
Paso 5              ENTONCES ESCRIBIR "TECLEE EL TOTAL DE
                    DATOS A PROCESAR"
                    REPETIR
                        LEER N
                        SI (N<=000) O (N>999)
                            ENTONCES LLAMAR ERROR(RESPUESTA)
                        FINSI
                    HASTA_QUE (N>000) Y (N<=999)
                    PARA J DE 1 HASTA N HACER
                        X(J) ← 0
                    FIN_PARA
                    ESCRIBIR "DIGITE CADA DATO Y PRESIONE LA
```

```
                    TECLA ENTER"
                    PARA J DE 1 HASTA N HACER
                        LEER X(J)
                    FIN_PARA
                    ESCRIBIR "LOS DATOS ORIGINALES SON: "
                    PARA J DE 1 HASTA N HACER
                        IMPRIMIR X(J)
                    FIN_PARA
                FINSI
Paso 6      FIN


PROCEDIMIENTO RANGO(N, X, MAYOR, MENOR, R, RESPUESTA)
{ CALCULA EL RANGO DE UN CONJUNTO DE DATOS }
Paso 1      INICIO
Paso 2          LLAMAR CAPTURA(N, X)
Paso 3          MAYOR ← X(1)
Paso 4          MENOR ← X(1)
Paso 5          PARA J DE 2 HASTA N HACER
                    SI X(J) > MAYOR
                        ENTONCES MAYOR ← X(J)
                        SINO SI X(J) < MENOR
                            ENTONCES MENOR ← X(J)
                        FINSI
                    FINSI
                FIN_PARA
Paso 6          R ← MAYOR - MENOR
Paso 7          IMPRIMIR "LOS DATOS ANALIZADOS SON: "
Paso 8          PARA J DE 1 HASTA N HACER
                    IMPRIMIR X(J)
                FIN_PARA
Paso 9          IMPRIMIR "EL ELEMENTO MAYOR ES ", MAYOR
Paso 10         IMPRIMIR "EL ELEMENTO MENOR ES ", MENOR
Paso 11         IMPRIMIR "EL RANGO ES ", R
Paso 12         ESCRIBIR "PRESIONE LA TECLA ENTER PARA CONTINUAR"
Paso 13         LEER RESPUESTA
Paso 14     FIN


PROCEDIMIENTO MED(N, X, XMED)
{ CALCULO DE LA MEDIA ARITMÉTICA }.
Paso 1      'NICIO
Paso 2          SUM ← 0
```

```
Paso 3          PARA J DE 1 HASTA N HACER
                   SUM ← SUM + X(J)
                FIN_PARA
Paso 4          XMED ← SUM / N
Paso 5       FIN

PROCEDIMIENTO MEDIA(CAPTURA, MED, RESPUESTA)
{ IMPRESIÓN DE LA MEDIA ARITMÉTICA }
Paso 1       INICIO
Paso 2          LLAMAR CAPTURA(N, X)
Paso 3          LLAMAR MED(N, X, XMED)
Paso 4          IMPRIMIR "LOS DATOS ANALIZADOS SON: "
Paso 5          PARA J DE 1 HASTA N HACER
                   IMPRIMIR X(J)
                FIN_PARA
Paso 6          IMPRIMIR "LA MEDIA ARITMÉTICA ES ", XMED
Paso 7          ESCRIBIR "PRESIONE LA TECLA ENTER PARA CONTINUAR"
Paso 8          LEER RESPUESTA
Paso 9       FIN

PROCEDIMIENTO DESMED(CAPTURA, MED, DM, RESPUESTA)
{ CALCULO DE LA DESVIACIÓN MEDIA }
Paso 1       INICIO
Paso 2          LLAMAR CAPTURA(N, X)
Paso 3          LLAMAR MED(N, X, XMED)
Paso 4          SUM ← 0
Paso 5          PARA J DE 1 HASTA N HACER
                   SUM ← SUM + | X(J)-XMED |
                FIN_PARA
Paso 6          DM ← SUM/N
Paso 7          IMPRIMIR "LOS DATOS ANALIZADOS SON: "
Paso 8          PARA J DE 1 HASTA N HACER
                   IMPRIMIR X(J)
                FIN_PARA
Paso 9          IMPRIMIR "LA DESVIACIÓN MEDIA ES: ", DM
Paso 10         ESCRIBIR "PRESIONE LA TECLA ENTER PARA CONTINUAR"
Paso 11         LEER RESPUESTA
Paso 12      FIN

PROCEDIMIENTO SUMDES(CAPTURA, MED, SUM)
{ CALCULO DE LA SUMATORIA DE LAS DESVIACIONES}
Paso 1       INICIO
Paso 2          LLAMAR CAPTURA(N, X)
```

```
Paso 3          LLAMAR MED(N, X, XMED)
Paso 4          SUM ← 0
Paso 5          PARA J DE 1 HASTA N HACER
                   SUM ← SUM + (X(J) - XMED)↑|2
                FIN_PARA
Paso 6       FIN
```

```
PROCEDIMIENTO ESTÁNDAR(SUMDES, DESV, RESPUESTA)
{ CALCULO DE LA DESVIACIÓN ESTÁNDAR }
Paso 1       INICIO
Paso 2          LLAMAR SUMDES(CAPTURA, MED, SUM)
Paso 3          DESV ← (SUM/N)↑0.5
Paso 4          IMPRIMIR "LOS DATOS ANALIZADOS SON: "
Paso 5          PARA J DE 1 HASTA N  HACER
                   IMPRIMIR X(J)
                FIN_PARA
Paso 6          IMPRIMIR "LA DESVIACIÓN ESTÁNDAR ES", DESV
Paso 7          ESCRIBIR "PRESIONE LA TECLA ENTER PARA CONTINUAR"
Paso 8          LEER RESPUESTA
Paso 9       FIN
```

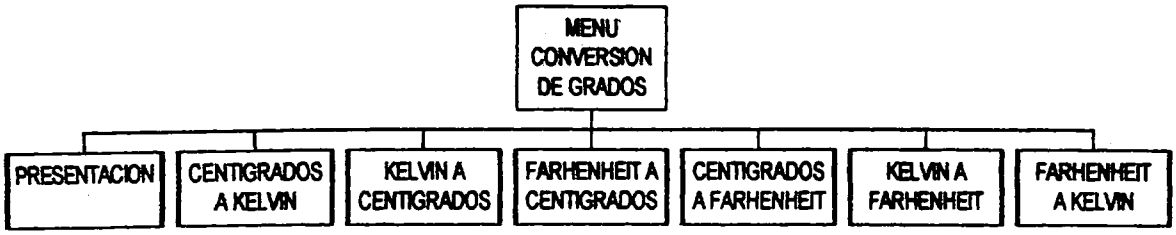
```
PROCEDIMIENTO VARIANZA(SUMDES, S, RESPUESTA)
{ CALCULO DE LA VARIANZA }
Paso 1       INICIO
Paso 2          LLAMAR SUMDES(CAPTURA, MED, SUM)
Paso 3          S ← SUM/N
Paso 4          IMPRIMIR "LOS DATOS ANALIZADOS SON: "
Paso 5          PARA J DE 1 HASTA N HACER
                   IMPRIMIR X(J)
                FIN_PARA
Paso 6          IMPRIMIR "LA VARIANZA ES: ", S
Paso 7          ESCRIBIR "PRESIONE LA TECLA ENTER PARA CONTINUAR"
Paso 8          LEER RESPUESTA
Paso 9       FIN
```

EJERCICIO

Elaborar un programa que permita convertir grados mediante la utilización de un menú.

Solución

a) Estructura del algoritmo



b) Compilación PASCAL

```
PROGRAM TEMPERATURA;
USES CRT,PRINTER;
VAR
      I,OPCION : INTEGER;
      VAL : REAL;

PROCEDURE MENUPPAL;
BEGIN
      WRITELN(LST);
      WRITELN(LST);
      WRITELN(LST, ' ':40, 'TEMPERATURA');
      WRITELN(LST);
      WRITELN(LST);
      WRITELN(LST, ' ':15, '1. PRESENTACIÓN');
      WRITELN(LST, ' ':15, '2. CENTÍGRADOS A KELVIN');
      WRITELN(LST, ' ':15, '3. KELVIN A CENTÍGRADOS');
      WRITELN(LST, ' ':15, '4. FARHENHEIT A CENTÍGRADOS');
      WRITELN(LST, ' ':15, '5. CENTÍGRADOS A FARHENHEIT');
      WRITELN(LST, ' ':15, '6. KELVIN A FARHENHEIT');
      WRITELN(LST, ' ':15, '7. FARHENHEIT A KELVIN');
      WRITELN(LST, ' ':15, '8. TERMINAR');
      WRITELN(LST);
      WRITELN(LST);
      REPEAT
          WRITE(LST, ' ':10, 'ELIJA UNA OPCIÓN:     ');
          READ(OPCION);
          WRITELN(LST,OPCION:5);
          IF (OPCION < 1) OR (OPCION > 8) THEN
          BEGIN
             WRITELN(LST);
             WRITELN(LST, ' ':15, 'LA OPCIÓN MARCADA ES INCORRECTA');
          END;
```

```
        UNTIL (OPCION >=1) AND (OPCION <= 8);
END;

PROCEDURE PRESENTACION;
BEGIN
        FOR I:=1 TO 7 DO
            WRITELN(LST);
        WRITELN(LST, ' ':30, 'ALONSO TAMAYO ALZATE');
        WRITELN(LST);
        FOR I:=1 TO 9 DO
            WRITELN(LST);
        WRITELN(LST);
        WRITELN(LST);
        WRITELN(LST, ' ':25, '  ****************************** ');
        WRITELN(LST, ' ':25, '  *                              * ');
        WRITELN(LST, ' ':25, '  *      CONVERSIÓN DE           * ');
        WRITELN(LST, ' ':25, '  *                              * ');
        WRITELN(LST, ' ':25, '  *      TEMPERATURAS            * ');
        WRITELN(LST, ' ':25, '  *                              * ');
        WRITELN(LST, ' ':25, '  ****************************** ');
        WRITELN(LST);
        WRITELN(LST);
        FOR I:=1 TO 9 DO
           WRITELN(LST);
        WRITELN(LST, ' ':30, 'REALIZADO POR:');
        WRITELN(LST);
        WRITELN(LST, ' ':30, '              ALONSO TAMAYO ALZATE');
        FOR I:=1 TO 9 DO
           WRITELN(LST);
        WRITELN(LST, ' ':25, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
        WRITELN(LST);
        WRITELN(LST, ' ':30, '            SEDE MANIZALES');
        WRITELN(LST);
        WRITELN(LST, ' ';27, 'FACULTAD DE CIENCIAS Y ADMINISTRACIÓN');
        WRITELN(LST);
        WRITELN(LST, ' ':30, '                 1995');
        FOR I:=1 TO 9 DO
            WRITELN(LST);
END;

FUNCTION CEN_A_KEL (CEN:REAL):REAL;
BEGIN
        CEN_A_KEL:=CEN+273;
END;
```

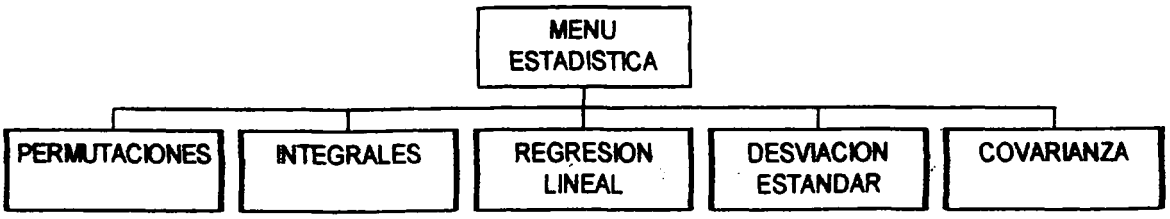
```
FUNCTION KEL_A_CEN (KEL:REAL):REAL;
BEGIN
      KEL_A_CEN:=KEL-273;
END;

FUNCTION FAR_A_CEN (FAR:REAL):REAL;
BEGIN
      FAR_A_CEN:=(5/9)*(FAR-32);
END;

FUNCTION CEN_A_FAR (CEN:REAL):REAL;
BEGIN
      CEN_A_FAR:=(9/5)*(CEN+32);
END;

PROCEDURE KEL_A_FAR (KEL:REAL);
VAR
      AUX,AUX1 : REAL;
BEGIN
      AUX:= KEL_A_CEN (KEL);
      AUX1:=CEN_A_FAR (AUX);
      WRITELN(LST, ' ':10, 'VALOR:  ',AUX1:5:5);
END;

PROCEDURE FAR_A_KEL (FAR:REAL);
VAR
      AUX,AUX1 : REAL;
BEGIN
      AUX:= FAR_A_CEN (FAR);
      AUX1:=CEN_A_KEL (AUX);
      WRITELN(LST, ' ':10, 'VALOR:  ',AUX1:5:5);
END;

BEGIN
      REPEAT
         MENUPPAL;
         IF (OPCION > 1) AND (OPCION < 8) THEN
         BEGIN
            WRITELN(LST, ' ':10, 'TECLEE VALOR A CONVERTIR:  ');
            READLN(VAL);
            WRITELN(LST, ' ':10,VAL:5:5);
         END;
         CASE OPCION OF
            1: PRESENTACION
```

```
            2: WRITELN(LST,' ':10,VAL:4:4,' GRADOS CENTÍGRADOS SON ',
                       CEN_A_KEL(VAL):5:5,' GRADOS KELVIN');
            3: WRITELN(LST,' ':10,VAL:4:4,' GRADOS KELVIN SON',
                       KEL_A_CEN(VAL):5:5,' GRADOS CENTÍGRADOS');
            4: WRITELN(LST,' ':10,VAL:4:4,' GRADOS FARHENHEIT SON ',
                       FAR_A_CEN(VAL):5:5,' GRADOS CENTÍGRADOS');
            5: WRITELN(LST,' ':10,VAL:4:4,' GRADOS CENTÍGRADOS SON ',
                       CEN_A_FAR(VAL):5:5,' GRADOS FARHENHEIT');
            6: WRITELN(LST,' ':10,VAL:4:4,' GRADOS KELVIN SON ',
                       KEL_A_FAR(VAL):5:5,' GRADOS FARENHEIT');
            7: WRITELN(LST,' ':10,VAL:4:4,' GRADOS FARENHEIT SON ',
                       FAR_A_KEL(VAL):5:5,' GRADOS KELVIN');
         END;
      UNTIL OPCION = 8;
END.
```

EJERCICIO

Cálculos estadísticos y matemáticos mediante la aplicación de un menú.

a) Estructura del algoritmo



b) Compilación FORTRAN

```
      PROGRAM ESTAD
      CALL PORTADA
      NO = 6
80    N = MENU(NO)
      IF (NO .EQ. 1) THEN
         CALL OPCION1
         GOTO 80
      ENDIF
      IF (NO .EQ. 2) THEN
         CALL OPCION2
         GOTO 80
      ENDIF
```

```
      IF (NO .EQ. 3) THEN
         CALL OPCION3
         GOTO 80
      ENDIF
      IF (NO .EQ. 4) THEN
         CALL OPCION4
         GOTO 80
      ENDIF
      IF (NO .EQ. 5) THEN
         CALL OPCION5
         GOTO 80
      ENDIF
      IF (NO.NE.1.AND.NO.NE.2.AND.NO.NE.3.AND.NO.NE.4.AND.NO.NE.5)
      STOP
      END

      SUBROUTINE PORTADA
         WRITE(1,1)
1        FORMAT(10(/),40X,'MENÚ ESTADÍSTICA',14(/),30X,'ELABORADO POR
     *   ALONSO TAMAYO ALZATE',10(/),24X,'UNIVERSIDAD NACIONAL DE
     *   COLOMBIA')
         RETURN
      END


      FUNCTION MENU(NO)
         WRITE(*,10)
10       FORMAT(25(/),25X,'1. PERMUTACIONES')
         WRITE(*,20)
20       FORMAT(25X,'2. INTEGRALES')
         WRITE(*,30)
30       FORMAT(25X,'3. REGRESIÓN LINEAL')
         WRITE(*,40)
40       FORMAT(25X,'4. DESVIACIÓN ESTÁNDAR')
         WRITE(*,50)
50       FORMAT(25X,'5. COVARIANZA')
         WRITE(*,60)
60       FORMAT(25X,'6. TERMINAR')
         WRITE(*,70)
70       FORMAT(/25X,'TECLEE SU OPCIÓN: ')
         READ(*,*)NO
         MENU = NO
         RETURN
      END
```

```
      SUBROUTINE OPCION1
         WRITE(*,100)
100      FORMAT(25(/),21X,'*** CÁLCULO DE PERMUTACIONES "nPr" ***')
         WRITE(*,101)
101      FORMAT(8X,'ENTRE EL VALOR DE n: '\)
         READ(*,*)NP
         WRITE(*,102)
102      FORMAT(8X,'ENTRE EL VALOR DE r: '\)
         READ(*,*)RP
         C = 1
         DO 103 I = 1,NP
            C = C*I
103      CONTINUE
         L = NP-RP
         FACT = 1
         DO 104 J = 1,L
            FACT = FACT*L
104      CONTINUE
         PER = C/FACT
         WRITE(*,105)PER
105      FORMAT(8X,'LA PERMUTACIÓN ES: ',F8.2)
         PAUSE
         RETURN
      END

      SUBROUTINE OPCION2
         F(NX) = NX**2
         WRITE(*,206)
206      FORMAT(25(/),16X,'* INTEGRAL POR EL MÉTODO DE LOS TRAPECIOS
     *   *')
         WRITE(*,200)
200      FORMAT(8X,'ENTRE EL LÍMITE INFERIOR: '\)
         READ(*,*)A
         WRITE(*,201)
201      FORMAT(8X,'ENTRE EL LÍMITE SUPERIOR: '\)
         READ(*,*)B
         WRITE(*,202)
202      FORMAT(8X,'ENTRE EL VALOR DE N: '\)
         READ(*,*)ND
         DX = (B-A)/ND
         J = 0
         DO 203 I = 1,ND
            Y = F(I)
            J = J+Y
```

```
203     CONTINUE
        IR = DX*J
        Y0 = F(0)
        YN = F(ND)
        C = (Y0 + YN)/2
        D = ND-1
        L = 0
        DO 204 K = 1,D
           YK = F(K)
           L = L+YK
204     CONTINUE
        T = DX*(C+L)
        WRITE(*,205)T
205     FORMAT(8X,'EL VALOR DE LA INTEGRAL ES:  ',F9.2)
        PAUSE
        RETURN
      END

      SUBROUTINE OPCION3
        F(NX) = NX**2
        G(NX) = NX**3
        WRITE(*,300)
300     FORMAT(25(/),22X,'*** CÁLCULO DE LA REGRESIÓN LINEAL ***')
        C = 0
        D = 0
        P = 0
        S = 0
        W = 0
        WRITE(*,301)
301     FORMAT(8X,'ENTRE EL VALOR DE  nr:  '\)
        READ(*,*)NR
        DO 302 I = 1,NR
           YR = F(I)
           XR = G(I)
           C = C+YR
           D = D+XR
           Z = XR**2
           P = P+Z
           E = YR**2
           S = S+E
           H = XR*YR
302     CONTINUE
        W = W+H
        Q = NR*W - D*C
```

```
      FI = NR*Z - D**2
      B = Q/FI
      A = (C - B*D)/NR
      O = (NR*Z - D**2) * (W*S - C**2)
      R = (NR*W - C*D)/O**0.5
      WRITE(*,303)R
303   FORMAT(8X,'LA REGRESIÓN LINEAL ES: ',F8.2)
      PAUSE
      RETURN
    END

    SUBROUTINE OPCION4
      M(NX) = NX**3
      WRITE(*,400)
400   FORMAT(25(/),20X,'*** CÁLCULO DE LA DESVIACIÓN ESTÁNDAR
     *  ***')
      WRITE(*,401)
401   FORMAT(8X,'ENTRE EL VALOR DE N: '\)
      READ(*,*)NDE
      C = 0
      DO 402 I = 1,NDE
        M1 = M(I)
        C = C+M1
402   CONTINUE
      XM = C/NDE
      DO 403 J=1,NDE
        M2= M(J)
        X1= (M2 - XM)**2
403   CONTINUE
      D1 = M(J)
      DES = (D1**0.5)
      WRITE(*,404)DES
404   FORMAT(8X,'LA DESVIACIÓN ESTÁNDAR ES  ',F9.2)
      PAUSE
      RETURN
    END


    SUBROUTINE OPCION5
      Y(NX) = NX**2
      X(NX) = NX**3
      WRITE(*,500)
500   FORMAT(25(/),25X,'*** CALCULO DE COVARIANZA ***')
      WRITE(*,501)
```

```
501     FORMAT(8X,'ENTRE EL VALOR DE N: '\)
        READ(*,*)NV
        S = 0
        D = 0
        C = 0
        DO 502 I = 1.NV
           S = S+X(I)
           D = D+Y(I)
           E = X(I)+Y(I)
           C = C+E
502     CONTINUE
        XMV = S/NV
        YMV = D/NV
        COV = (C - NV*XMV*YMV)/(NV-1)
        WRITE(*,503)COV
503     FORMAT(8X,'COVARIANZA: ',F9.2)
        PAUSE
        RETURN
      END
```

PROBLEMAS PROPUESTOS

1. Determinar que valor se imprime de acuerdo al siguiente algoritmo:

Algoritmo Principal

```
Paso 1      INICIO
Paso 2        A ← 2
Paso 3        B ← 5
Paso 4        LLAMAR X(A, B, Z)
Paso 5        IMPRIMIR Z
Paso 6      FIN
```

```
PROCEDIMIENTO X (A, B, Z)
Paso 1      INICIO
Paso 2        Y ← A - 2 * B ↑ 2
Paso 3        D ← A ↑ 2 - B ↑ 2
Paso 4        E ← -Y / D ↑ (1/5)
Paso 5        F ← 1/(Y * D ↑ (1/3))
```

Paso 6 Z ← E - F
Paso 7 **FIN**

2- Elaborar un procedimiento para reducir un ángulo expresado en radianes a grados, minutos y segundos.

3- Elaborar un procedimiento para convertir el peso de un articulo dado en arrobas a Kilogramos, libras y gramos.

4- Para un grupo de N estudiantes se dispone de 3 notas de evaluaciones por registro cada una con un valor del 20% y una nota del proyecto final con un valor del 40%. Calcular la nota definitiva mediante la aplicación de la metodología del Diseño Descendente.

5- Calcular mediante la aplicación de un procedimiento el valor del servicio de energía, si si el valor del Kilovatio es de $20 =, además se cobra un cargo fijo según la siguiente tabla.

ESTRATO	CARGO FIJO
1	50
2	100
3	300
4	500
5	1.000
6	2.000

Por cada registro se lee el número del contador, consumo y estrato social. El proceso termina cuando el estrato sea menor o igual a cero y mayor que seis.

6- Realizar una prueba de escritorio al siguiente algoritmo y determinar que calcula.

Algoritmo principal

```
Paso 1      INICIO
Paso 2        LEER N
Paso 3        M ← N + 1
Paso 4        PARA I DE 1 HASTA M HACER
                  A(I) ← FACT(N)/(FACT(I-1)*FACT(N-I+1))
```

```
                FIN_PARA
Paso 5          PARA I DE 1 HASTA M HACER
                    IMPRIMIR A(I)
                FIN_PARA
Paso 6          FIN
```

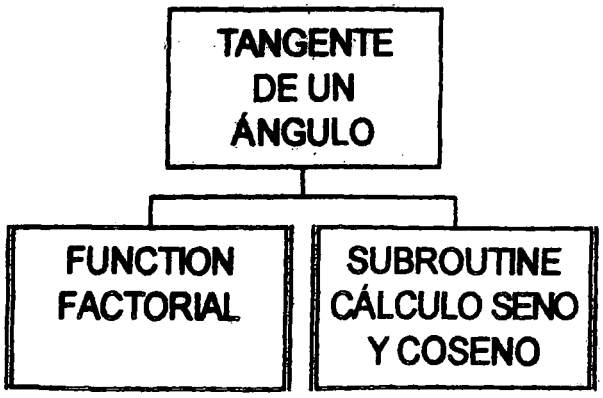
```
FUNCIÓN FAC(N)
Paso 1          INICIO
Paso 2              FACT ← 1
Paso 3              SI N > 1
                        ENTONCES PARA I DE 2 HASTA N HACER
                            FACT ← FACT * I
                        FIN_PARA
                    FINSI
Paso 4              FAC ← FACT
Paso 5          FIN
```

EJERCICIOS COMPILADOS

EJERCICIO Nro. 4.1

Determinar la tangente de un ángulo hallados el seno y el coseno, mediante la utilización de subprogramas.

a) Estructura del Algoritmo



b) Compilación Fortran

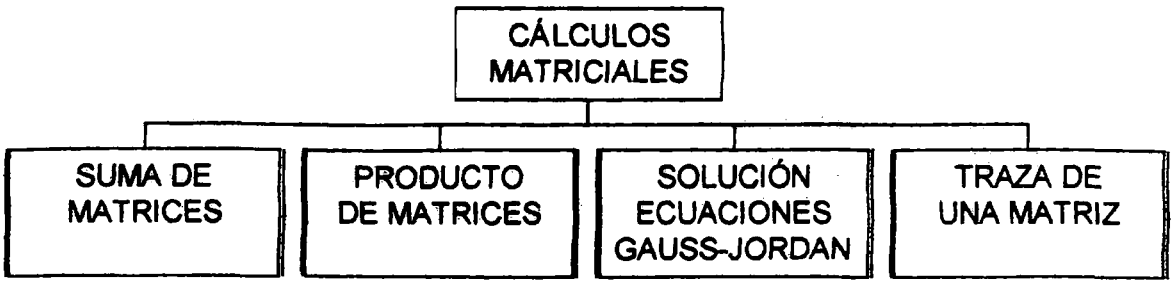
```
      PROGRAM TRIGONO
C     Programa para determinar la tangente de un ángulo hallados el seno y el coseno.
C     Para encontrar el seno y el coseno se utilizaron sus respectivas series de Taylor.
C     Definición de variables
C     SEN :Seno del ángulo en radianes.
C     COS :Coseno del ángulo en radianes.
C     TAN :Tangente del ángulo en radianes.
      WRITE(*,5)
    5 FORMAT(10(/),5X, 'Teclee el número de ángulos a evaluar :',\)
      READ(*,8)NO
    8 FORMAT(I3)
      DO 100 IL=1,NO
         SEN= 0
         COS= 0
         WRITE(*,10)
   10    FORMAT(2(/),5X, 'Teclee el valor del ángulo en radianes :',\)
         READ(*,20)X
   20    FORMAT(F10.8)
         WRITE(*,30)
   30    FORMAT(2(/),5X, 'Teclee el límite superior de la serie :',\)
         READ(*,40)N
   40    FORMAT(I4)
         DO 45 JI= 0,N
            D= FAC(JI)
            CALL SECOS(X,JI,TS,TC)
            SEN= TS/D+SEN
            COS= TC/D+COS
   45    CONTINUE
         TAN= SEN/COS
         WRITE(*,50)SEN,COS,TAN
   50    FORMAT(3(/),5X, 'El Seno del ángulo es :',F12.10,3(/),5X, 'El Coseno del
     *   ángulo es :',F12.10,3(/),5X, 'La Tangente del ángulo es :',F20.10,4(/))
  100 CONTINUE
      STOP
      END

      FUNCTION FAC(N)
         IF (N .EQ. 0) THEN
            FAC= 1
         ELSE
            FAC= 1
            M= 2*N
            DO 10 I=1,M
               FAC= FAC*I
```

```
10          CONTINUE
          ENDIF
          RETURN
        END

        SUBROUTINE SECOS(X,N,TSE,TCE)
          TSA= 0
          TCA= 0
          T= (-1)**N
          S= X**(2*N)
          TCA= T*S+TCA
          O= X**(2*N-1)*2*N
          P= (-1)**(N-1)
          TSA= P*O*TSA
          TSE= TSA
          TCE= TCA
          RETURN
        END
```

EJERCICIO Nro. 4.2

Cálculos matriciales mediante la aplicación de subprogramas.

a) Estructura del algoritmo



b) Compilación Fortran

```
      PROGRAM MATRIZ
      DIMENSION A(5,5), B(5,5), C(5,5), D(5,5)
*************************************************************************
C                          PROGRAMA PRINCIPAL
C     El programa calcula la suma y multiplicación de dos matrices dadas, con sus
```

```
C       respectivas condiciones,además calcula la solución de un sistema de ecuaciones por
C       el método de Gauss-Jordan y la traza de una matriz dada.
*************************************************************************
        WRITE(*,1)
    1   FORMAT(8(/),30X, 'ALONSO TAMAYO ALZATE',10(/),24X, 'UNIVERSIDAD
       *NACIONAL DE COLOMBIA',/// 30X, 'SEDE MANIZALES')
        WRITE(*,2)
    2   FORMAT(19(/),5X, 'El programa calcula la suma y multiplicación de dos
       *matrices',//,5X, 'dadas, con sus respectivas condiciones',//,5X, 'calcula además la
       *solución de un sistema de ecuaciones por',//,5X, 'el método de Gauss-Jordan y la
       *traza de una matriz dada.',/)
 1000   CALL MENU(KOP)
        GOTO (100,200,300,400,500),KOP
*************************************************************************
C
C                   CÁLCULO DE LA SUMA DE MATRICES
C
*************************************************************************
  100   WRITE(*,3)
    3   FORMAT(10(/),30X, 'SUMA DE MATRICES')
        CALL DATOS(L,K,A)
        CALL DATOS(M,N,B)
        IF (L .NE. M) .OR. (K .NE. N) THEN
           WRITE(*,4)
    4      FORMAT(10(/),25X, 'Las matrices no se pueden sumar',//28X, 'No son
       *   cuadradas',/)
           GOTO 100
        ENDIF
        CALL SUMA(L,K,A,B,C)
        CALL PRES(1,L,K,M,N,A,B,C)
        WRITE(*,5)
    5   FORMAT(10(/),25X, '-1 PARA CONTINUAR SUMA :'\)
        READ(*,*)KOUT
        IF (KOUT .EQ. -1) GOTO 100
        GOTO 1000
*************************************************************************
C
C                   CÁLCULO DEL PRODUCTO DE DOS MATRICES
C
*************************************************************************
  200   WRITE(*,6)
    6   FORMAT(10(/),20X, 'PRODUCTO DE MATRICES')
        CALL DATOS(L,K,A)
        CALL DATOS(M,N,B)
```

```
      IF (K .NE. M) THEN
        WRITE(*,7)
    7   FORMAT(10(/),20X, 'Las matrices no se pueden multiplicar', //,10X, 'El número
     *  de columnas de la matriz A no es igual'//,16X,'al número de filas de la matriz B')
        GOTO 200
      ENDIF
      CALL PRODUC(L,K,M,N,A,B,C)
      CALL PRES(2,L,K,M,N,A,B,C)
      WRITE(*,8)
    8 FORMAT(10(/),20X,'-1 PARA CONTINUAR PRODUCTO :',\)
      READ(*,*)KOUT
      IF (KOUT .EQ. -1)GOTO 200
      GOTO 1000
*****************************************************************************
C
C                    CÁLCULO DE GAUSS-JORDAN
C
*****************************************************************************
  300 WRITE(*,9)
    9 FORMAT(10(/),30X, 'GAUSS-JORDAN')
      CALL DATOS(N,MA,A)
      IF (N .GE. MA) THEN
        WRITE(*,11)
   11   FORMAT(10(/),30X, 'DATOS MAL SUMINISTRADOS',/)
        GOTO 300
      ENDIF
      WRITE(*,12)
   12 FORMAT(10(/),30X, 'DATOS DADOS',/)
      DO 101 I=1,N
        WRITE(*,13)(A(I,J),J=1,MA)
   13   FORMAT(15X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1)
  101 CONTINUE
      CALL GAUSS(N,A,D)
      WRITE(*,14)
   14 FORMAT(///28X, 'VECTOR SOLUCIÓN',/)
      WRITE(*,15)(D(I),I=1,N)
   15 FORMAT(15X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1)
      WRITE(*,16)
   16 FORMAT(10(/),21X, '-1 PARA CONTINUAR GAUSS-JORDAN :',\)
      READ(*,*)KOUT
      IF (KOUT .EQ. -1) GOTO 300
      GOTO 1000
*****************************************************************************
C
```

```
C                    CÁLCULO DE LA TRAZA DE UNA MATRIZ
C
*************************************************************************
 400  WRITE(*,17)
  17  FORMAT(10(/),30X, 'TRAZA')
      CALL DATOS(L,M,A)
      IF (L .NE. M) THEN
         WRITE(*,18)
  18     FORMAT(10(/),26X, 'LA MATRIZ NO ES CUADRADA',/)
         GOTO 400
      ENDIF
      S= TRAZA(L,M,A)
      WRITE(*,19) S
  19  FORMAT(10(/),10X, 'LA TRAZA ES :',F10.5)
      WRITE(*,21)
  21  FORMAT(5(/),26X, '-1 PARA CONTINUAR TRAZA :',\)
      READ(*,*)KOUT
      IF (KOUT .EQ. -1) GOTO 400
         GOTO 1000
*************************************************************************
C
C                                FINAL
C
*************************************************************************
 500  STOP
      END

*************************************************************************
C
C                    COMIENZO DE LAS SUBRUTINAS
C
*************************************************************************

      SUBROUTINE MENU(KOP)
  10     WRITE(*,1)
   1     FORMAT(10(/),35X, '-MENU-',/30X, '1-SUMA',/30X, '2- PRODUCTO',/30X,
     *   '3- GAUS-JORDAN',/30X, '4- TRAZA',/30X, '5- FIN',5(/),30X, 'CUÁL ES SU
     *   OPCIÓN :'\)
         READ(*,*)KOP
         IF ((KOP .LT. 1) .OR. (KOP  GT. 5)) THEN
            WRITE(*,2)
   2        FORMAT(14(/),30X, 'OPCIÓN INVÁLIDA'//30X, 'ESCOJA DE NUEVO')
            GOTO 10
         ENDIF
```

```
      RETURN
      END

      SUBROUTINE DATOS(NF,NC,X)
         DIMENSION X(5,5)
         WRITE(*,1)
    1    FORMAT(3(/),29X, 'CARGANDO DATOS',///)
         WRITE(*,2)
    2    FORMAT(//,5X, 'NÚMERO DE FILAS :'\)
         READ(*,*)NF
         WRITE(*,3)
    3    FORMAT(//,5X, 'NÚMERO DE COLUMNAS :',\)
         READ(*,*)NC
         DO 10 I=1,NF
         DO 10 J=1,NC
            X(I,J)= 0.0
            WRITE(*,4)I,J
    4       FORMAT(/,5X, 'ELEMENTO (',I3, ',',I3, ') :',\)
            READ(*,*)X(I,J)
   10    CONTINUE
         WRITE(*,5)
    5    FORMAT(5(/),30X, 'DATOS TERMINADOS',/)
         RETURN
      END

      SUBROUTINE SUMA(NFA,NCA,X,Y,Z)
         DIMENSION X(5,5), Y(5,5), Z(5,5)
         DO 10 I=1,NFA
         DO 10 J=1,NCA
            Z(I,J)= 0.0
            Z(I,J)= X(I,J)+Y(I,J)
   10    CONTINUE
         RETURN
      END

      SUBROUTINE PRES(KL,NFA,NCA,NFB,NCB,X,Y,Z)
         DIMENSION X(5,5), Y(5,5), Z(5,5)
         WRITE(*,1)
    1    FORMAT(//,10X, 'MATRIZ A :',28X, 'MATRIZ B :',/)
         DO 10 I=1,NFB
            WRITE(*,2)(X(I,J), J=1,NCA)
    2       FORMAT(2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1)
            WRITE(*,9)(Y(I,J), J=1,NCB)
    9       FORMAT(42X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1,2X,F4.1)
```

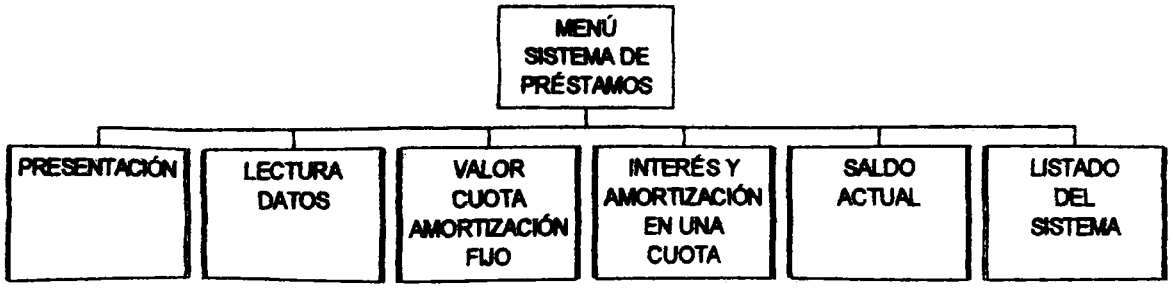
```
10      CONTINUE
        IF (KL .EQ. 1) THEN
           WRITE(*,3)
 3         FORMAT(///,36X, 'A + B',/)
           DO 20 I=1,NFA
              WRITE(*,4)(Z(I,J), J=1,NCA)
 4            FORMAT(28X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1)
20         CONTINUE
        ELSE
           WRITE(*,5)
 5         FORMAT(///,33X, 'A + B',/)
           DO 30 I=1,NFA
              WRITE(*,6)(Z(I,J), J=1,NCB)
 6            FORMAT(28X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1,3X,F4.1)
30         CONTINUE
        ENDIF
        RETURN
      END

      SUBROUTINE GAUSS(N,A,X)
        DIMENSION A(5,5), X(5)
        M= N+1
        L= N-1
        DO 10 K=1,L
           KP= K+1
           DO 20 I= KP,N
              QT= A(I,K)/A(K,K)
              DO 30 J= KP,M
                 A(I,J)= A(I,J)-QT*A(K,J)
30            CONTINUE
20         CONTINUE
           DO 40 I= KP,N
              A(I,K)= 0
40         CONTINUE
10      CONTINUE
        X(N)= A(N,M)/A(N,N)
        DO 50 NN=1,L
           SUM= 0
           I= N-NN
           IP= I+1
           DO 60 J= IP,N
              SUM= SUM+A(I,J)*X(J)
60         CONTINUE
           X(I)= (A(I,M)-SUM)/A(I,I)
```

```
50      CONTINUE
        RETURN
      END

      FUNCTION TRAZA(NFA,NCB,X)
        DIMENSION X(5,5)
        TRAZA= 0.
        DO 10 I=1,NFA
        DO 10 J=1,NCB
          IF (I .EQ. J) THEN
            TRAZA= TRAZA+X(I,J)
          ENDIF
10      CONTINUE
        RETURN
      END

      SUBROUTINE PRODUC(NFA,NCA,NFB,NCB,A,B,C)
        DIMENSION A(5,5), B(5,5), C(5,5)
        DO 10 I=1,NFA
        DO 10 J=1,NFA
          C(I,J)= 0.
          DO 20 K=1,NFB
            C(I,J)= C(I,J)+A(I,K)*B(K,J)
20        CONTINUE
10      CONTINUE
        RETURN
      END
```

EJERCICIO Nro. 4.3

Sistema de préstamos bancarios aplicando un menú.

a) Estructura del algoritmo



b) Compilación Pascal

```
PROGRAM INTERESES;
USES CRT,PRINTER;
VAR
    Opcion,Plazo,Ncuota                                    :INTEGER;
    Capital,Tasainteres,Cuotafija,Intcuo,Amocuo,Saldohoy   :REAL;

PROCEDURE PRESENTACION;
VAR
    D,T,O :INTEGER;
BEGIN
   CLRSCR,
   WRITELN(Lst);
   WRITELN(Lst, '':25, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
   WRITELN(Lst, '':34, 'SEDE MANIZALES');
   WRITELN(Lst, '':29, 'MANIZALES, OCTUBRE DE 1995');
   READLN;
END;

PROCEDURE MENU(VAR Opcion :INTERGER);
BEGIN
   CLRSCR;
   WRITELN('       MENÚ SISTEMA DE PRÉSTAMOS CUOTA FIJA');
   WRITELN(' 1. Presentación');
   WRITELN(' 2. Lectura de capital, interés, y  plazo');
   WRITELN(' 3. Valor de la cuota en sistema de amortización fijo');
   WRITELN(' 4. Interés y amortización en una cuota dada');
   WRITELN(' 5. Saldo actual');
   WRITELN(' 6. Listar comportamiento del sistema');
   WRITELN(' 7. Salida');
   REPEAT
      WRITELN('Teclee la opción elegida');
      READLN(Opcion);
      IF (Opcion<1) OR (Opcion>7) THEN
         WRITELN('ERROR EN EL VALOR ENTRADO, TECLEE DE NUEVO');
   UNTIL (Opcion>=1) AND (Opcion<=7);
END;

PROCEDURE LECTURA(VAR K,TI :REAL; VAR PL :INTEGER);
BEGIN
   WRITELN('Teclee el valor del préstamo :');
   REPEAT
      READLN(K);
      IF K<0 THEN
         WRITELN('ERROR EN EL VALOR, TECLEE DE NUEVO');
```

```
    UNTIL K>=0;
    WRITELN('Teclee el valor de la tasa de interés (%)   :');
    REPEAT
       READLN(TI);
       IF TI<1 THEN
          WRITELN('ERROR EN EL VALOR, TECLEE DE NUEVO');
    UNTIL TI>=1;
    WRITELN('Teclee el plazo a pagar de la deuda (meses)  :');
    REPEAT
       READLN(PL);
       IF PL<= 0 THEN
          WRITELN('ERROR EN EL VALOR, TECLEE DE NUEVO');
    UNTIL PL>0;
END;

FUNCTION POTENCIA(NUMERO :REAL; EXPONENTE :INTEGER):REAL;
VAR
     VECES,I              :INTEGER;
     NUMEROTOTAL          :REAL;
BEGIN
    NUMEROTOTAL:= 1;
    VECES:= ABS(EXPONENTE);
    FOR I:= 1 TO VECES DO
       NUMEROTOTAL:= NUMEROTOTAL*NUMERO;
    IF EXPONENTE<0 THEN
       NUMEROTOTAL:= 1/NUMEROTOTAL;
    POTENCIA:= NUMEROTOTAL;
END;


FUNCTION VALORCUOTAFIJA(K,TI :REAL; P :INTEGER):REAL;
VAR
     TEMP :REAL;
BEGIN
    TI:= TI/100;
    TEMP:= (1-POTENCIA((1+TI),-P);
    IF TEMP<>0 THEN
       VALORCUOTAFIJA:= (K*TI)/TEMP
    ELSE
       BEGIN
          WRITELN('División por cero');
          READLN;
       END;
END;
```

```
PROCEDURE LECTURANUMERODECUOTA(VAR NCUOT,P :INTEGER);
BEGIN
   WRITELN('Teclee el número de la cuota que desea averiguar :');
   REPEAT
      READLN(NCUOT);
      IF (NCUOT<0) OR (NCUOT>P) THEN
         WRITELN('ERROR EN EL VALOR ENTRADO, TECLEE DE NUEVO');
   UNTIL (NCUOT>=0) AND (NCUOT<P);
END;

PROCEDURE INTERAMORTIZFIJA(CUOTA,TI :REAL; P,NCTA :INTEGER;
                                    VAR INTCUOTA,AMCUOTA :REAL);
BEGIN
   TI:= TI/100;
   IF NCTA>0 THEN
      BEGIN
         AMCUOTA:= CUOTA/POTENCIA((1+TI),(P-NCTA+1));
         INTCUOTA:= CUOTA-AMCUOTA;
      END
   ELSE
      BEGIN
         AMCUOTA:= 0;
         INTCUOTA:= 0;
      END;
END;

FUNCTION SALDOACTUAL(SALDOACT,CUOTA,TI :REAL; PL,NCTA
                              :INTEGER):REAL;
VAR
    I                                  :INTEGER;
    AMORTACTUAL,INTERACTUAL   :REAL;
BEGIN
   FOR I:= 0 TO NCTA DO
      BEGIN
         INTERAMORTIZFIJA(CUOTA,TI,PL,I,INTERACTUAL,AMORTACTUAL);
         SALDOACT:= SALDOACT-AMORTACTUAL;
      END;
   SALDOACTUAL:= SALDOACT;
END;

PROCEDURE LISTARCOMPORTAMIENTO(SALDOACT,CUOTA,TI :REAL;
                                          PL :INTEGER);
VAR
    INTERACTUAL,AMORTACTUAL   :REAL;
```

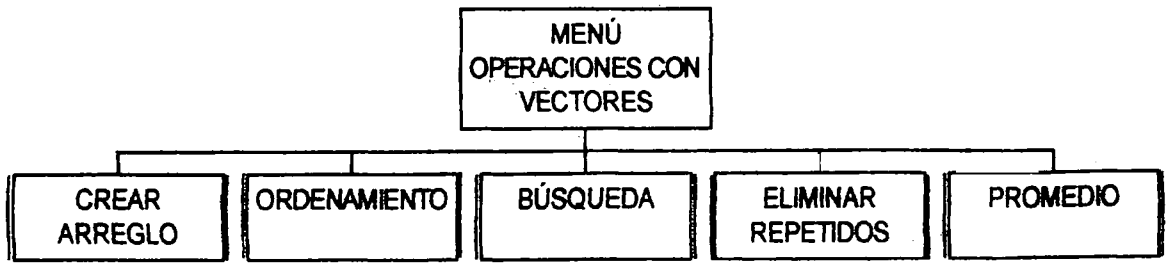
```
    I                                   :INTEGER;
BEGIN
   FOR I:= O TO PL DO
      BEGIN
         INTERAMORTIZFIJA(CUOTA,TI,PL,I,INTERACTUAL,AMORTACTUAL);
         SALDOACT:= SALDOACT-AMORTACTUAL;
         WRITELN(Lst, 'Cuota ',I,' Interés ',INTERACTUAL:5:2,' Amortización',
                  AMORTACTUAL:5:2, 'Saldo Actual ',SALDOACT:5:2);
         READLN;
      END;
END;

BEGIN {PROGRAMA PRINCIPAL}
   REPEAT
      MENU(OPCION);
      CASE (OPCION) OF
         1:PRESENTACION;
         2:LECTURA(CAPITAL,TASAINTERES,PLAZO);
         3:BEGIN
            WRITELN('Valor cuota fija ',VALORCUOTAFIJA(CAPITAL,
                     TASAINTERES,PLAZO):5:2);
            READLN;
           END;
         4:BEGIN
            LECTURANUMERODECUOTA(NCUOTA,PLAZO);
            CUOTAFIJA:= VALORCUOTAFIJA(CAPITAL,TASAINTERES,PLAZO);
            INTERAMORTIZFIJA(CUOTAFIJA,TASAINTERES,PLAZO,NCUOTA,
                              INTCUO,AMOCUO);
            WRITELN(' CUOTA ',NCUOTA, 'INTERÉS ',INTCUO:5:2);
            WRITELN(' CUOTA ',NCUOTA, 'AMORTIZACIÓN ',AMOCUO:5:2);
            READLN;
           END;
         5:BEGIN
            LECTURANUMERODECUOTA(NCUOTA,PLAZO);
            CUOTAFIJA:=VALORCUOTAFIJA(CAPITAL,TASAINTERES,PLAZO);
            WRITELN('En la cuota ',NCUOTA, 'El saldo es ',SALDOACTUAL
                     (CAPITAL,CUOTAFIJA,TASAINTERES,PLAZO,NCUOTA):5:2;
            READLN;
           END;
         6:BEGIN
            CUOTAFIJA:= VALORCUOTAFIJA(CAPITAL,TASAINTERES,PLAZO);
            LISTARCOMPORTAMIENTO(CAPITAL,CUOTAFIJA,TASAINTERES,
                                 PLAZO);
           END;
```

```
        END
    UNTIL (OPCION=7);
END.
```

EJERCICIO Nro. 4.4

Operaciones con vectores mediante la utilización de un menú.

a) Estructura del algoritmo



b) Compilación Pascal

```
PROGRAM MENU;
USES CRT,PRINTER;
CONST
      N=50;
TYPE
    ARREGLO=ARRAY[1..N] OF INTEGER;
VAR
    PRIMERO,ULTIMO,T,CENTRAL,I,X,DATO,Y,J,Z,NUM  :INTEGER;
    L                                            :ARREGLO;
    ENCONTRADO                                   :BOOLEAN;

PROCEDURE PORTADA_PRINCIPAL;
BEGIN
   {********************** PORTADA**********************}
   CLRSCR;
   WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
   WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
   WRITELN(Lst, '':25, 'PROGRAMA QUE REALIZA');
   WRITELN(Lst);
   WRITELN(Lst, '':25, 'OPERACIONES CON VECTORES');
   WRITELN(Lst);
   WRITELN(Lst, '':25, 'MEDIANTE UN MENÚ');
```

```
    WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':20, 'Elaborado por ALONSO TAMAYO ALZATE');
    WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':21, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
    WRITELN(Lst);
    WRITELN(Lst, '':30, 'SEDE MANIZALES');
    DELAY(3000);
    CLRSCR;
END;

PROCEDURE LLENAR_ARREGLO(VAR L:ARREGLO);
BEGIN
    CLRSCR;
    I:= 1;
    GOTOXY(20,2);WRITELN('***LLENAR ARREGLO***');
    GOTOXY(25,5);WRITELN('Digite la cantidad de números :');
    GOTOXY(25,6);READLN(X);
    Y:= 10;
    FOR I:= 1 TO X DO
        BEGIN
            GOTOXY(25,9);WRITELN('Digite el número');
            GOTOXY(25,Y);READLN(DATO);
            L[I]:=DATO;
            Y:=Y+1;
        END;
    GOTOXY(45,9);WRITE('LA LISTA ES');
    Y:= 10;
    FOR I:=1 TO X DO
        BEGIN
            GOTOXY(45,Y);WRITELN(L[I]);
            Y:=Y+1;
        END;
    DELAY(3000);
END;

PROCEDURE ORDENAR(VAR L:ARREGLO);
BEGIN
    CLRSCR;
    GOTOXY(20,2);WRITE('***ORDENAR ARREGLO***');
    FOR J:=1 TO (X-1) DO
        BEGIN
            FOR I:= (J+1) TO X DO
                BEGIN
                    IF (L[J]>L[I]) THEN
```

```
                    BEGIN
                       Z:=L[I];
                       L[I]:=L[J];
                       L[J]:=Z;
                    END;
              END;
        END;
    Y:=10;
    GOTOXY(25,9);WRITE('ARREGLO ORDENADO');
    FOR I:=1 TO X DO
        BEGIN
           GOTOXY(25,Y);WRITELN(L[I]);
           Y:=Y+1;
        END;
    DELAY(3000);
END;

PROCEDURE BUSQUEDA(VAR I:ARREGLO);
BEGIN
    GOTOXY(20,2);WRITE('***BUSCAR UN NÚMERO***');
    GOTOXY(25,9);WRITE('Digite el número a buscar');
    GOTOXY(25,10);READ(NUM);
    PRIMERO:=1;
    ULTIMO:=X;:
    ENCONTRADO:= FALSE;
    WHILE (PRIMERO<= ULTIMO) AND (NOT ENCONTRADO) DO
        BEGIN
           CENTRAL:= (PRIMERO+ULTIMO) DIV 2;
           IF NUM=L[CENTRAL] THEN
               BEGIN
                  ENCONTRADO:=TRUE;
               END
           ELSE
               IF NUM> L[CENTRAL] THEN
                  BEGIN
                      PRIMERO:= CENTRAL+1;
                  END
               ELSE
                  ULTIMO:= CENTRAL-1;
        END;
    IF ENCONTRADO=TRUE THEN
        BEGIN
           GOTOXY(25,15);WRITE('El número existe en la posición :',CENTRAL);
```

```
            DELAY(3000);
        END
    ELSE
        BEGIN
            GOTOXY(25,16);WRITE('El número no existe');
            DELAY(3000);
        END;
END;

PROCEDURE ELIMINAR_REPETIDOS(VAR L:ARREGLO; X:INTEGER);
VAR
    R,Y      :INETEGER;
BEGIN
    GOTOXY(20,2);WRITE('***ELIMINAR REPETIDOS***');
    Y:= 10;
    GOTOXY(25,9);WRITE('ARREGLO SIN NÚMEROS REPETIDOS');
    FOR R:=1 TO (X+1) DO
        IF L[R]<>L[R+1] THEN
            BEGIN
                GOTOXY(25,Y);WRITELN(L[R]);
                Y:=Y+1;
            END;
    DELAY(3000);
END;

PROCEDURE PROMEDIO(VAR L:ARREGLO; X:INTEGER);
VAR
    PROM   :REAL;
    SUM,Z  :INTEGER;
BEGIN
    GOTOXY(20,2);WRITE('***PROMEDIO DE LA LISTA***');
    SUM:= 0;
    FOR Z:=1 TO X DO
        SUM:=SUM+L[Z];
    PROM:=SUM/X;
    GOTOXY(25,9);WRITE('El promedio de la lista es :',PROM);
    DELAY(3000);
END;

PROCEDURE CUADRO;
VAR
    X,Y      :INTEGER;
BEGIN
    FOR X:= 5 TO 74 DO
```

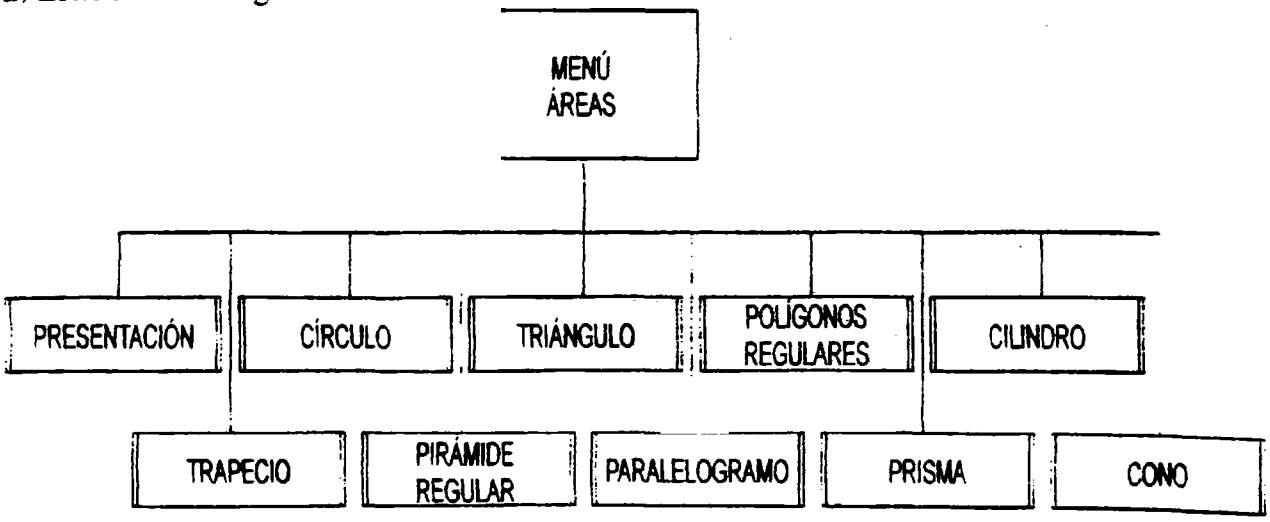
```
        BEGIN
          GOTOXY(X,2);WRITE('-');
          GOTOXY(X,20);WRITE('-');
        END;
    FOR Y:=2 TO 20 DO
        BEGIN
          GOTOXY(5,Y);WRITE('¦');
          GOTOXY(74,Y);WRITE('¦');
        END;
    GOTOXY(5,2);WRITE('┌');
    GOTOXY(74,2);WRITE('┐');
    GOTOXY(5,20);WRITE('└');
    GOTOXY(74,20);WRITE('┘');
END;

PROCEDURE MENU_PRINCIPAL;
VAR
    OP        :CHAR;
BEGIN
    REPEAT
        CLRSCR;
        CUADRO;
        GOTOXY(29,7);WRITE('MENU DE OPCIONES');
        GOTOXY(25,10);WRITE('1. LLENAR ARREGLO');
        GOTOXY(25,11);WRITE('2. ORDENAR ARREGLO');
        GOTOXY(25,12);WRITE('3. BUSCAR NÚMERO');
        GOTOXY(25,13);WRITE('4. ELIMINAR REPETIDOS');
        GOTOXY(25,14);WRITE('5. PROMEDIO');
        GOTOXY(25,15);WRITE('6. TERMINAR');
        GOTOXY(30,18);WRITE('Digite la opción :');
        GOTOXY(48,18);READ(OP);
        CASE OP OF
          '1': BEGIN
                 CLRSCR;
                 LLENAR_ARREGLO(L);
               END;
          '2': BEGIN
                 CLRSCR;
                 ORDENAR(L);
               END;
          '3': BEGIN
                 CLRSCR;
                 BUSQUEDA(L);
               END;
```

```
            '4': BEGIN
                 CLRSCR;
                 ELIMINAR_REPETIDOS(L,X);
               END;
            '5': BEGIN
                 CLRSCR;
                 PROMEDIO(L,X);
               END;
        END
    UNTIL OP='6';
END;

BEGIN {PROGRAMA PRINCIPAL}
    MENU_PRINCIPAL;
END.
```

EJERCICIO Nro. 4.5

Cálculo de áreas de algunas figuras geométricas mediante el empleo de un menú.

a) Estructura del algoritmo.



b) Compilación Pascal

```
PROGRAM AREAS;
USES CRT,PRINTER;

VAR
    I,OPCION        :INTEGER;
```

```
PROCEDURE PRESENTACION;
BEGIN
   CLRSCR;
   FOR I:=1 TO 8 DO
      WRITELN(Lst);
   WRITELN(Lst, '':17, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
   WRITELN(Lst, '':23, 'SEDE MANIZALES');
   FOR I:=1 TO 7 DO
      WRITELN(Lst);
   WRITELN(Lst, '':14, 'PROGRAMA PARA CALCULAR AREAS');
   WRITELN(Lst, '':14, 'DE LAS FIGURAS GEOMÉTRICAS');
   READLN;
   WRITELN(Lst, 'Este programa ha sido elaborado para el cálculo de áreas de las');
   WRITELN(Lst, 'figuras geométricas más importantes tales como el círculo, el');
   WRITELN(Lst, 'triángulo, el paralelogramo, el trapecio y los polígonos regulares');
   READLN;
END;

PROCEDURE CIRCULO;
VAR
    AC,L,R  :REAL;
    C       :INTEGER;
BEGIN
   WRITELN(Lst, '':20, '* AREA DEL CIRCULO *');
   REPEAT
      CLRSCR;
      WRITELN(Lst, 'Teclee el radio del círculo :');
      READ(R);
      IF (R<= 0) THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READ(R);
            UNTIL (R>0);
         END;
      AC:= PI*SQR(R);
      WRITELN(Lst, 'El área del círculo es:  ',AC:8:2);
      L:= 2*PI*R;
      WRITELN(Lst, 'La longitud del círculo es:  ',L:8:2);
      READLN;
      REPEAT
         WRITELN(Lst);
         WRITELN(Lst, 'DESEA CONTINUAR ? (S=1/N=0)');
         READ(C);
```

```
      UNTIL (C=1) OR (C=0);
   UNTIL (C=0);
   READLN;
END;

PROCEDURE TRAPECIO;
VAR
    A,H,SUM,B1,B2          :REAL;
    RES                    :INTEGER;
BEGIN
   CLRSCR;
   WRITELN(Lst);WRITELN(Lst);WRITELN(Lst);
   WRITELN(Lst,'':20, ' * AREA DEL TRAPECIO *');
   REPEAT
      WRITELN(Lst, 'Teclee el valor de la altura :');
      READLN(H);
      IF (H<= 0) THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READLN(H);
            UNTIL H>0;
         END;
      WRITELN(Lst, 'Teclee el valor de la base mayor :');
      READLN(B1);
      IF (B1<= 0) THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READLN(B1);
            UNTIL B1>0;
         END;
      WRITELN(Lst, 'Teclee el valor de la base menor :');
      READLN(B2);
      IF (B2>=B1) OR (B2<= 0) THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READLN(B2);
            UNTIL (B2<B1) AND (B2>0);
         END;
      SUM:= 0;
      A:= (H/2)*B1+B2;
      SUM:= SUM+A;
```

```
        WRITELN(Lst);
        WRITELN(Lst, 'El área del trapecio es:     ',A:8:2);
        WRITELN(Lst);
        REPEAT
           WRITELN(Lst, 'DESEA CONTINUAR ? (S=1/N=0)');
           READLN(RES);
        UNTIL (RES=1) OR (RES=0);
    UNTIL (RES=0);
    WRITELN(Lst);
    WRITELN(Lst, 'El total de áreas es de   ',SUM:8:2);
    READLN
END;

PROCEDURE TRIANGULO;
VAR
     L1,L2,L3,AR,R,P          :REAL;
     D1,D2,NT,RES             :INTEGER;
BEGIN
    CLRSCR;
    NT:= 0;
    R:= 0;
    D1:= 0;
    D2:= 0;
    WRITELN(Lst, '* ÁREA DE TRIÁNGULOS *');
    REPEAT
        WRITELN(Lst, 'Ingrese el valor del lado Nro. 1 :');
        READLN(L1);
        IF (L1<= 0) THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READ(L1);
              UNTIL L1>0;
           END;
        WRITELN(Lst, 'Ingrese el valor del lado Nro. 2 :');
        READLN(L2);
        IF (L2<= 0) THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(L2);
              UNTIL L2>0;
           END;
        WRITELN(Lst, 'Ingrese el valor del lado Nro. 3 :');
```

```
        READLN(L3);
        IF (L3<= 0) THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READ(L3);
              UNTIL L3>0;
           END;
        P:= (L1+L2+L3);
        AR:= (SQRT(P*(P-L1)*(P-L2)*(P-L3)));
        WRITELN(Lst, 'El área del triángulo es:     ',AR:9:2);
        WRITELN(Lst, 'El perímetro del triángulo es :',P:2:3);
        IF (L1=L2) AND (L2=L3) THEN
           BEGIN
              NT:= NT+1;
              R:= R+AR;
              IF AR>32.34 THEN
                 D1:= D1+1;
           END;
        IF (AR>42.36) AND (AR<70.80) THEN
           D2:= D2+1;
        REPEAT
           WRITELN(Lst, 'DESEA CONTINUAR ? (S=1/N=0)');
           READ(RES);
        UNTIL (RES=1) OR (RES=0);
     UNTIL (RES=0);
     CLRSCR;
     WRITELN(Lst, 'Triángulos equiláteros :');
     WRITELN(Lst, 'Cantidad :',NT);
     WRITELN(Lst, 'Sumatoria de áreas :  ',R:9:2);
     WRITELN(Lst, 'Con área mayor a 32.34:',D1);
     WRITELN(Lst);
     WRITELN(Lst, 'Triángulos en general');
     WRITELN(Lst, '45.36<Área<70.80',D2);
     DELAY(3000);
     READLN
END;

PROCEDURE PARALELOGRAMO;
VAR
     H,B,A    :REAL;
     RES      :INTEGER;
BEGIN
   CLRSCR;
```

```
    WRITELN(Lst, '':20, '* ÁREA DEL PARALELOGRAMO *');
    REPEAT
       WRITELN(Lst, 'Teclee el valor de la altura :');
       READLN(H);
       IF (H<= 0) THEN
          BEGIN
             REPEAT
                WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                READLN(H);
             UNTIL H>0;
          END;
       WRITELN(Lst, 'Teclee el valor de la base :');
       READLN(B);
       IF (B<= 0) THEN
          BEGIN
             REPEAT
                WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                READLN(B);
             UNTIL B>0;
          END;
       WRITELN(Lst);
       A:= B*H;
       WRITELN(Lst, 'El área del Paralelogramo es:     ',A:8:2);
       REPEAT
          WRITELN(Lst, 'DESEA CONTINUAR ? (S=1/N=0)');
          READLN(RES);
       UNTIL (RES=1) OR (RES=0);
    UNTIL (RES=0);
    READLN;
END;

PROCEDURE POLIGONOS_REGULARES;
VAR
     AP,C,P,A        :REAL;
     N,RES           :INTEGER;
BEGIN
    CLRSCR;
    WRITELN(Lst, '':15, '* ÁREA DE LOS POLÍGONOS REGULARES');
    REPEAT
       WRITELN(Lst, 'Introduzca el valor de la base :');
       READLN(C);
       IF (C<= 0) THEN
          BEGIN
             REPEAT
```

```
            WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
            READLN(C);
          UNTIL C>0;
      END;
    WRITELN(Lst, 'Teclee número de lados del polígono :');
    READLN(N);
    IF (N< 3) THEN
      BEGIN
        REPEAT
          WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
          READLN(N);
        UNTIL N>=3;
      END;
    P:= C*N;
    WRITELN(Lst, 'El perímetro del polígono es:     ',P:8:2);
    WRITELN(Lst);
    WRITELN(Lst, 'Ingrese el apotema del polígono :');
    READLN(AP);
    IF (AP>C) OR (AP<= 0) THEN
      BEGIN
        REPEAT
          WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
          READLN(AP);
        UNTIL (AP<= C) OR (AP> 0);
      END;
    A:= (P*AP)/2;
    WRITELN(Lst, 'El área del polígono de: ',N, 'lados y de base: ',C:2:2,'es:',
            A:2:2);
    REPEAT
      WRITELN(Lst, 'DESEA CONTINUAR ? (S=1/N=0)');
      READLN(RES);
    UNTIL (RES=1) OR (RES=0);
  UNTIL (RES=0);
END;

PROCEDURE PRISMA;
VAR
    AB,AP,AL,AT,SR,P,P1,H,L    :REAL;
    RES,C                      :INTEGER;
BEGIN
  CLRSCR;
  WRITELN(Lst, '':20, '* CÁLCULO DEL ÁREA DEL PRISMA REGULAR');
  REPEAT
    WRITELN(Lst, 'Ingrese el valor de la altura :');
```

```
READLN(H);
IF (H<= 0) THEN
   BEGIN
      REPEAT
         WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
         READLN(H);
      UNTIL H>0;
   END;
WRITELN(Lst, 'Ingrese el valor de la longitud :');
READLN(L);
IF (L<= 0) THEN
   BEGIN
      REPEAT
         WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
         READLN(L);
      UNTIL L>0;
   END;
WRITELN(Lst, 'Ingrese el numero de caras del prisma :');
READLN(C);
IF C< 3 THEN
   BEGIN
      REPEAT
         WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
         READLN(C);
      UNTIL C>=3;
   END;
P:= (L+L)*C;
WRITELN(Lst, 'El perímetro de la base es :',P:8:2);
WRITELN(Lst);
WRITELN(Lst, 'Ingrese el valor del apotema de la base :');
READLN(AP);
IF (AP>L) OR (AP<= 0) THEN
   BEGIN
      REPEAT
         WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
         READ(AP);
      UNTIL (AP<=L) OR (AP>0);
   END;
AB:= P*AP;
SR:= L+L
P1:= (L+L)*C;
AL:= (P1*AP);
AT:= AB+AL;
WRITELN(Lst, 'El área de la base del prisma es:    ',AB:8:2);
```

```
        WRITELN(Lst);
        WRITELN(Lst, 'El área lateral del prisma es:    ',AL:8:2);
        WRITELN(Lst);
        WRITELN(Lst, 'El área total del prisma es:    ',AT:8:2);
        REPEAT
           WRITELN(Lst, 'DESEA CONTINUAR ? (SI=1/ NO = 0)');
           READLN(RES);
        UNTIL (RES=1) OR (RES=0);
    UNTIL (RES=0);
END;

PROCEDURE PIRAMIDE;
VAR
     A,P,H,AP,L      :REAL;
     RES,C           :INTEGER;
BEGIN
    CLRSCR;
    WRITELN(Lst, '':20, '* ÁREA DE LA PIRÁMIDE REGULAR');
    REPEAT
        WRITELN(Lst, 'Ingrese el valor de la longitud de la base :');
        READLN(L);
        IF (L<= 0) THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(L);
              UNTIL L>0;
           END;
        WRITELN(Lst, 'Ingrese el número de caras de la pirámide :');
        READLN(C);
        IF C<= 0 THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READ(C);
              UNTIL C>0;
           END;
        WRITELN(Lst, 'Ingrese el valor del apotema de la pirámide :');
        READLN(AP);
        IF AP <= 0 THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(AP);
```

```
            UNTIL AP>= 0;
         END;
      P:= L*C;
      WRITELN(Lst, 'El perímetro de la base de la pirámide es :',P:8:2);
      WRITELN;
      A:= (P/2)*AP
      WRITELN(Lst, 'El área de la pirámide es:     ',A:8:2);
      REPEAT
         WRITELN(Lst, 'DESEA CONTINUAR ? (SI=1 / NO = 0)');
         READLN(RES);
      UNTIL (RES=1) OR (RES=0);
   UNTIL (RES=0);
END;

PROCEDURE CILINDRO;
VAR
    AB,AL,AT,R,H  :REAL;
    RES           :INTEGER;
BEGIN
   CLRSCR;
   WRITELN(Lst, '':20, '* ÁREA DEL CILINDRO');
   REPEAT
      WRITELN(Lst, 'Ingrese el radio de la circunferencia de la base :');
      READLN(R);
      IF R<= 0 THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READLN(R);
            UNTIL R>0;
         END;
      WRITELN(Lst, 'Ingrese la altura del cilindro :');
      READLN(H);
      IF H<= 0 THEN
         BEGIN
            REPEAT
               WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
               READLN(H);
            UNTIL H>0;
         END;
      AB:= PI*SQR(R);
      AL:= 2*(PI*R*H);
      AT:= AL+(2*AB);
      WRITELN(Lst, 'El área del cilindro que tiene como área de su base:  ',AB:8:2);
```

```
        WRITELN(Lst, 'y como área lateral :  ',AL:8:2, 'es :  ',AT:8:2);
        REPEAT
           WRITELN(Lst, 'DESEA CONTINUAR ? (SI=1/ NO = 0)');
           READLN(RES);
        UNTIL (RES=1) OR (RES=0);
    UNTIL (RES=0);
END;

PROCEDURE CONO;
VAR
     AL,AB,AT,G,R,H        :REAL;
     RES                   :INTEGER;
BEGIN
    CLRSCR;
    WRITELN(Lst, '':20, '* ÁREA DEL CONO DE REVOLUCIÓN *');
    REPEAT
        WRITELN(Lst, 'Ingrese el valor del radio de la base :');
        READLN(R);
        IF R<= 0 THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(R);
              UNTIL R>0;
           END;
        WRITELN(Lst, 'Ingrese el valor de la generatriz del cono :');
        READLN(G);
        IF G <= 0 THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(G);
              UNTIL G>0;
           END;
        WRITELN(Lst, 'Ingrese el valor de la altura del cono :');
        READLN(H);
        IF (H>=G) AND (H<=0)  THEN
           BEGIN
              REPEAT
                 WRITELN(Lst, 'ERROR EN DATOS, TECLEE DE NUEVO');
                 READLN(H);
              UNTIL (H<G) AND (H>0);
           END;
        AB:= PI*SQR(R);
```

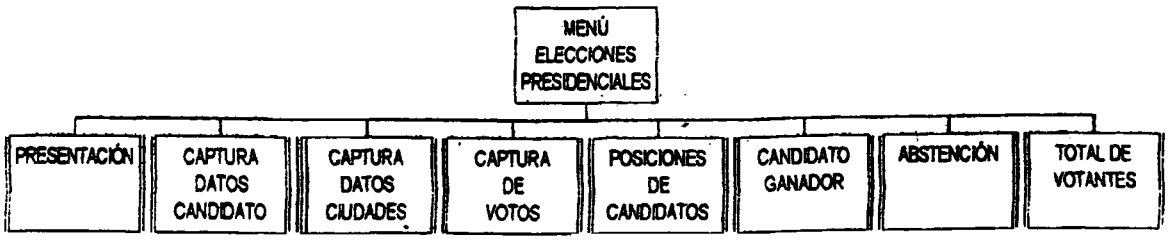
```
        AL:= PI*R*G;
        AT:= AB+AL;
        WRITELN(Lst, 'El área total del cono que tiene ');
        WRITELN(Lst, 'como área de su base:   ',AB:8:2);
        WRITELN(Lst, 'y como área lateral:   ',AL:8:2, 'es:  ',AT:8:2);
        REPEAT
           WRITELN(Lst, 'DESEA CONTINUAR ? (SI=1/ NO = 0)');
           READLN(RES);
        UNTIL (RES=1) OR (RES=0);
    UNTIL (RES=0);
END;

PROCEDURE MENU(VAR OPCION :INTEGER);
BEGIN
    CLRSCR;
    WRITELN(Lst, '1. PRESENTACIÓN');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '2. ÁREA DEL CÍRCULO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '3. ÁREA DEL TRAPECIO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '4. ÁREA DEL TRIÁNGULO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '5. ÁREA DEL PARALELOGRAMO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '6. ÁREA DE LOS POLÍGONOS REGULARES');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '7. ÁREA DEL PRISMA');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '8. ÁREA DEL CILINDRO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '9. ÁREA DEL CONO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '10. ÁREA DE LA PIRÁMIDE REGULAR');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '11. FIN DEL PROCESO');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, 'TECLEE UNA OPCIÓN :');
END;

BEGIN {PROGRAMA PRINCIPAL}
    REPEAT
        MENU(OPCION);
        READ(OPCION);
```

```
        CASE OPCION OF
            1:PRESENTACION;
            2:CIRCULO;
            3:TRAPECIO;
            4:TRIANGULO;
            5:PARALELOGRAMO;
            6:POLIGONOS_REGULARES;
            7:PRISMA;
            8:CILINDRO;
            9:CONO;
            10:PIRAMIDE;
        END;
    UNTIL OPCION=11;
END.
```

EJERCICIO Nro. 4.6

Elecciones presidenciales mediante la aplicación de un menú:

a) Estructura del algoritmo



b) Compilación Pascal

```
PROGRAM ELECCION;
USES CRT,PRINTER;
VAR
    CANDIDATOS :ARRAY[1..30] OF STRING[30];
    POSICION   :ARRAY[1..30] OF LONGINT;
    CIUDADES   :ARRAY[1..100] OF STRING[30];
    VOTOS      :ARRAY[1..30,1..30] OF LONGINT;
    I,J,OPCION,NUMCANDIDATOS,NUMCIUDADES :INTEGER;

PROCEDURE MENUPPAL;
BEGIN
```

```
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':40, 'ELECCIONES');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':15, '1. PRESENTACIÓN');
    WRITELN(Lst, '':15, '2. CANDIDATOS');
    WRITELN(Lst, '':15, '3. CIUDADES');
    WRITELN(Lst, '':15, '4. SUMINISTRO DE VOTOS');
    WRITELN(Lst, '':15, '5. POSICIONES');
    WRITELN(Lst, '':15, '6. GANADOR');
    WRITELN(Lst, '':15, '7. ABSTENCIONISTAS');
    WRITELN(Lst, '':15, '8. TOTAL DE COLOMBIANOS QUE VOTARON');
    WRITELN(Lst, '':15, '9. TERMINAR');
    WRITELN(Lst);WRITELN(Lst);
    REPEAT
       WRITE(Lst, '':10, 'Elija la opción :');
       READ(OPCION);
       WRITELN(Lst, OPCIÓN:5);
       IF (OPCION<1) OR (OPCION>9) THEN
          BEGIN
             WRITELN(Lst);
             WRITELN(Lst, '':15, 'LA OPCIÓN MARCADA ES INCORRECTA');
          END;
    UNTIL (OPCION>=1) AND (OPCION<=9);
END;

PROCEDURE PRESENTACION;
BEGIN
    WRITELN(Lst);
    WRITELN(Lst, '':30,'Elaborado por ALONSO TAMAYO ALZATE');
    FOR I:=1 TO 9 DO
       WRITELN(Lst);
    WRITELN(Lst, '':25, 'UNIVERSIDAD NACIONAL DE COLOMBIA');
    WRITELN(Lst, '':30, 'SEDE MANIZALES');
    FOR I:=1 TO 5 DO
       WRITELN(Lst);
END;

PROCEDURE ENTRADACANDIDATOS;
BEGIN
    WRITELN(Lst);WRITELN(Lst);
    REPEAT
       WRITELN(Lst, '':15, 'Introduzca número de candidatos a participar :');
       READLN(NUMCANDIDATOS);
       IF (NUMCANDIDATOS<1) OR (NUMCANDIDATOS>30) THEN
```

```
            WRITELN(Lst, '':15, 'Número de candidatos excede el límite');
    UNTIL (NUMCANDIDATOS>=1) AND (NUMCANDIDATOS<=30);
    FOR J:=1 TO NUMCANDIDATOS DO
        BEGIN
            WRITE(Lst, '':15, 'Introduzca candidato No. ',J:4, ':');
            READLN(CANDIDATOS[J]);
            WRITELN(Lst, CANDIDATOS[J]);
        END;
END;

PROCEDURE ENTRADACIUDADES;
BEGIN
    WRITELN(Lst);WRITELN(Lst);
    REPEAT
        WRITELN(Lst, '':15, 'Introduzca número de ciudades inscritas :');
        READLN(NUMCIUDADES);
        IF (NUMCIUDADES<1) OR (NUMCIUDADES>100) THEN
            WRITELN(Lst, '':15, 'Número de ciudades excede el límite');
    UNTIL (NUMCIUDADES>=1) AND (NUMCIUDADES<=100);
    FOR I:=1 TO NUMCIUDADES DO
        BEGIN
            WRITE(Lst, '':15, 'Introduzca ciudad No. ',I:4, ':');
            READLN(CIUDADES[I]);
            WRITELN(Lst,CIUDADES[I]);
        END;
END;


PROCEDURE SUMINISTRODEVOTOS;
BEGIN
    FOR I:=1 TO NUMCANDIDATOS DO
        FOR J:=1 TO NUMCIUDADES DO
            BEGIN
                REPEAT
                    WRITE(Lst, '':15, 'Nro. De votos de  ',CANDIDATOS[I], 'en  ',
                            CIUDADES[J], ' : ');
                    READ(VOTOS[I,J]);
                    WRITELN(LST,VOTOS[I,J]);
                    IF VOTOS[I,J]<1 THEN
                        WRITELN(Lst, '':15, 'VOTOS INCORRECTOS');
                UNTIL VOTOS[I,J]>=1;
            END;
END;
```

```
PROCEDURE POSICIONES;
VAR
    AUX     :LONGINT;
    AUX1    :STRING[30];
BEGIN
   FOR I:=1 TO NUMCANDIDATOS DO
      POSICION[I]:= 0;
   FOR I:=1 TO NUMCANDIDATOS DO
      FOR J:=1 TO NUMCIUDADES DO
         POSICION[I]:= POSICION[I]+VOTOS[I,J];
   FOR I:=1 TO (NUMCANDIDATOS-1) DO
      FOR J:= (I+1) TO NUMCANDIDATOS DO
         IF POSICION[I]<= POSICION[J] THEN
            BEGIN
               AUX:= POSICION[I];
               POSICION[I]:= POSICION[J];
               POSICION[J]:= AUX;
               AUX1:= CANDIDATOS[I];
               CANDIDATOS[I]:= CANDIDATOS[J];
               CANDIDATOS[J]:= AUX1;
            END;
   WRITELN(Lst);WRITELN(Lst);
   WRITELN(Lst, '':15, 'POSICIONES');
   WRITELN(Lst);WRITELN(Lst);
   FOR I:= 1 TO NUMCANDIDATOS DO
      WRITELN(Lst, '':15,CANDIDATOS[I]);
END;

FUNCTION PORCENTAJE(TOTAL,TOTALCAN :LONGINT):REAL;
BEGIN
   PORCENTAJE:= (TOTALCAN*100)/TOTAL;
END;

PROCEDURE GANADOR;
VAR
    TOTAL,AUX   :LONGINT;
    POR         :REAL;
    AUX1        :STRING[30];
BEGIN
   FOR I:=1 TO NUMCANDIDATOS DO
      POSICION[I]:= 0;
   FOR I:= 1 TO NUMCANDIDATOS DO
      FOR J:=1 TO NUMCIUDADES DO
```

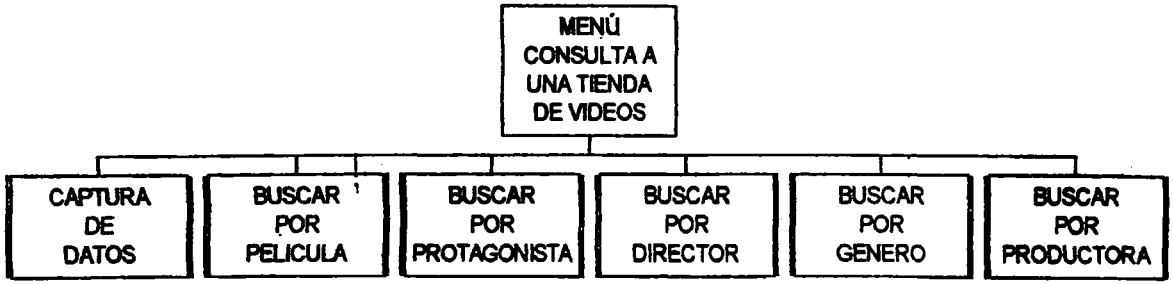
```
            POSICION[I]:= POSICION[I]+VOTOS[I,J];
    FOR I:=1 TO (NUMCANDIDATOS-1) DO
        FOR J:= (I+1) TO NUMCANDIDATOS DO
            IF POSICION[I]<= POSICION[J] THEN
                BEGIN
                    AUX:= POSICION[I];
                    POSICION[I]:= POSICION[J];
                    POSICION[J]:= AUX;
                    AUX1:= CANDIDATOS[I];
                    CANDIDATOS[I]:= CANDIDATOS[J];
                    CANDIDATOS[J]:= AUX1;
                END;
    TOTAL:= 0;
    FOR I:=1 TO NUMCANDIDATOS DO
        TOTAL:= TOTAL+POSICION[I];
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':15, 'GANADOR');
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':15, CANDIDATOS[1]);
    WRITELN(Lst);
    AUX:= POSICION[1];
    POR:= PORCENTAJE(TOTAL,AUX);
    WRITELN(Lst, '':15, 'El candidato obtuvo el  ',POR:5:3, 'por ciento de los votos');
END;

PROCEDURE ABSTENCION;
VAR
     AUX1,TOTAL,AUX    :LONGINT;
     POR               :REAL;
BEGIN
    FOR I:=1 TO NUMCANDIDATOS DO
        POSICION[I]:= 0;
    FOR I:=1 TO NUMCANDIDATOS DO
        FOR J:=1 TO NUMCIUDADES DO
            POSICION[I]:= POSICION[I]+VOTOS[I,J];
    TOTAL:=0;
    FOR I:=1 TO NUMCANDIDATOS DO
        TOTAL:= TOTAL+POSICION[I];
    AUX:= 99999999;
    POR:= 100-(PORCENTAJE(AUX,TOTAL));
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':15, 'Hubo un abstencionismo del  ',POR:5:3, 'por ciento.');
    WRITELN(Lst);
```

```
    AUX1:= AUX-TOTAL;
    WRITELN(Lst, '':15, 'Ciudadanos que no votaron  :  ', AUX1:7);
END;

PROCEDURE VOTANTES;
VAR
     TOTAL,AUX    :LONGINT;
     POR          :REAL;
BEGIN
    FOR I:=1 TO NUMCANDIDATOS DO
        POSICION[I]:= 0;
    FOR I:=1 TO NUMCANDIDATOS DO
        FOR J:=1 TO NUMCIUDADES DO
            POSICION[I]:= POSICION[I]+VOTOS[I,J];
    TOTAL:= 0;
    FOR I:=1 TO NUMCANDIDATOS DO
        TOTAL:= TOTAL+POSICION[I];
    WRITELN(Lst);WRITELN(Lst);
    WRITELN(Lst, '':15, 'Ciudadanos que votaron:  ',TOTAL:10);
END;

BEGIN {PROGRAMA PRINCIPAL}
    REPEAT
        MENUPPAL;
        CASE OPCION OF
            1: PRESENTACION;
            2: ENTRADACANDIDATOS;
            3: ENTRADACIUDADES;
            4: SUMINISTRODEVOTOS;
            5: POSICIONES;
            6: GANADOR;
            7: ABSTENCION;
            8: VOTANTES;
        END;
    UNTIL OPCION= 9;
END.
```

EJERCICIO Nro. 4.7

Consultas por diferentes conceptos a una tienda de videos mediante la aplicación de un menú.

a) Estructura del algoritmo



b) Compilación Pascal

```
PROGRAM VIDEO;
USES CRT,PRINTER;
VAR
    PEL,DIR,PRO,GEN,PROD    :ARRAY[1..300] OF STRING[40];
    OPCION                  :BYTE;
    NUM,I                   :INTEGER;

FUNCTION SALIDA:BOOLEAN;
VAR
    SAL     :CHAR;
BEGIN
   REPEAT
      WRITE('Desea procesar (S/N):  ');
      READLN(SAL);
      IF (SAL<>'S') AND (SAL<>'N') AND (SAL<>'n') AND (SAL<>'s') THEN
         WRITELN('OPCIÓN INCORRECTA');
   UNTIL (SAL='s') OR (SAL='n') OR (SAL='S') OR (SAL='N');
   IF (SAL='N') OR (SAL='n') THEN
      SALIDA:= TRUE
   ELSE
      SALIDA:= FALSE;
END;

PROCEDURE NUM_DE_PELICULAS;
BEGIN
   REPEAT
      WRITE('Introduzca número de películas a consignar');
      READLN(NUM);
      IF (NUM<1) OR (NUM>300) THEN
         WRITELN('NÚMERO ERRADO DE PELÍCULAS');
   UNTIL (NUM>=1) AND (NUM<=300);
END;
```

```
PROCEDURE ENTRADA_DATOS;
BEGIN
   FOR I:=1 TO NUM DO
      BEGIN
         WRITE('Película :');
         READLN(PEL[I]);
         WRITE('Protagonista :');
         READLN(PRO[I]);
         WRITE('Director :');
         READLN(DIR[I]);
         WRITE('Género :');
         READLN(GEN[I]);
         WRITE('Productora :');
         READLN(PROD[I]);
      END;
END;


PROCEDURE MOSTRAR_POR_TITULO;
VAR
    PELI     :STRING[40];
    AUX      :INTEGER;
BEGIN
   AUX:= 0;
   REPEAT
      WRITE('Introduzca película a buscar: ');
      READLN(PELI);
      FOR I:=1 TO NUM DO
         BEGIN
            IF PELI = PEL[I] THEN
               BEGIN
                  WRITELN('Película:       ',PEL[I]);
                  WRITELN('Protagonista: ',PRO[I]);
                  WRITELN('Director:      ',DIR[I]);
                  WRITELN('Género:        ',GEN[I]);
                  WRITELN('Productora:   ',PROD[I]);
                  AUX:= 1;
               END;
         END;
      IF AUX = 0 THEN
         WRITELN('PELÍCULA INEXISTENTE');
   UNTIL SALIDA;
END;
```

```
PROCEDURE MOSTRAR_POR_PROTAGONISTA;
VAR
    PROT    :STRING[40];
    AUX     :INTEGER;
BEGIN
   AUX:= 0;
   REPEAT
      WRITE('Introduzca nombre del protagonista: ');
      READLN(PROT);
      FOR I:=1 TO NUM DO
         BEGIN
            IF PROT = PRO[I] THEN
               BEGIN
                  WRITELN('Película:      ',PEL[I]);
                  WRITELN('Protagonista: ',PRO[I]);
                  WRITELN('Director:      ',DIR[I]);
                  WRITELN('Género:        ',GEN[I]);
                  WRITELN('Productora:    ',PROD[I]);
                  AUX:= 1;
               END;
         END;
      IF AUX = 0 THEN
         WRITELN('PROTAGONISTA INEXISTENTE');
   UNTIL SALIDA;
END;

PROCEDURE MOSTRAR_POR_DIRECTOR
VAR
    DIRE    :STRING[40];
    AUX     :INTEGER;
BEGIN
   AUX:= 0;
   REPEAT
      WRITE('Introduzca nombre del director: ');
      READLN(DIRE);
      FOR I:=1 TO NUM DO
         BEGIN
            IF DIRE = DIR[I] THEN
               BEGIN
                  WRITELN('Película:      ',PEL[I]);
                  WRITELN('Protagonista: ',PRO[I]);
                  WRITELN('Director:      ',DIR[I]);
                  WRITELN('Género:        ',GEN[I]);
```

```
                    WRITELN('Productora:   ',PROD[I]);
                    AUX:= 0;
                END;
            END;
        IF AUX = 0 THEN
            WRITELN('DIRECTOR INEXISTENTE');
    UNTIL SALIDA;
END;

PROCEDURE MOSTRAR_POR_GENERO;
VAR
     GENE    :STRING[40];
     AUX     :INTEGER;
BEGIN
    AUX:= 0;
    REPEAT
        WRITE('Introduzca el género de la película : ');
        READLN(GENE);
        FOR I:=1 TO NUM DO
            BEGIN
                IF GENE = GEN[I] THEN
                    BEGIN
                        WRITELN('Película:      ',PEL[I]);
                        WRITELN('Protagonista: ',PRO[I]);
                        WRITELN('Director:      ',DIR[I]);
                        WRITELN('Género:        ',GEN[I]);
                        WRITELN('Productora:   ',PROD[I]);
                        AUX:= 1;
                    END;
            END;
        IF AUX = 0 THEN
            WRITELN('PELÍCULAS DE ESTE GÉNERO NO HAY');
    UNTIL SALIDA;
END;

PROCEDURE MOSTRAR_POR_PRODUCTORA;
VAR
     PRODU  :STRING[40];
     AUX    :INTEGER;
BEGIN
    AUX:= 0;
    REPEAT
        WRITE('Introduzca productora de la película : ');
```

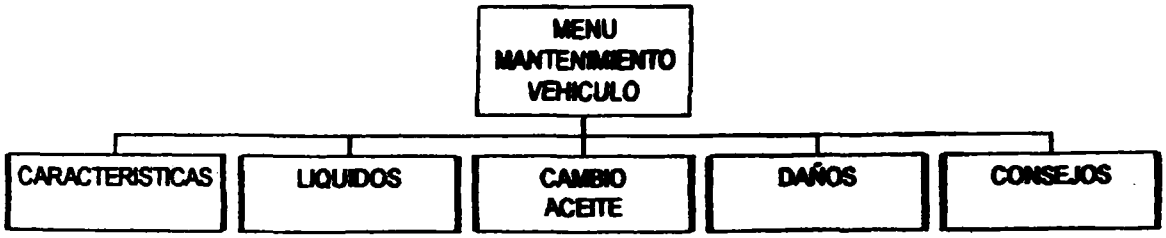
```
        READLN(PRODU);
        FOR I:=1 TO NUM DO
           BEGIN
              IF PRODU = PROD[I] THEN
                 BEGIN
                    WRITELN('Película:      ',PEL[I]);
                    WRITELN('Protagonista: ',PRO[I]);
                    WRITELN('Director:      ',DIR[I]);
                    WRITELN('Género:        ',GEN[I]);
                    WRITELN('Productora:    ',PROD[I]);
                    AUX:= 1;
                 END;
           END;
        IF AUX = 0 THEN
           WRITELN('PRODUCTORA INEXISTENTE');
     UNTIL SALIDA;
END;

PROCEDURE MENU;
BEGIN
   REPEAT
      CLRSCR;
      WRITELN('          TIENDA DE VIDEOS');
      WRITELN;WRITELN;
      WRITELN('    1. ENTRADA DE DATOS');
      WRITELN('    2. BUSCAR PELÍCULA');
      WRITELN('    3. BUSCAR POR PROTAGONISTA');
      WRITELN('    4. BUSCAR POR DIRECTOR');
      WRITELN('    5. BUSCAR POR GÉNERO');
      WRITELN('    6. BUSCAR POR PRODUCTORA');
      WRITELN('    7. SALIR');
      REPEAT
         WRITELN('Elija una opción');
         READLN(OPCION);
         IF (OPCION<1) OR (OPCION>7) THEN
            WRITELN('OPCION INCORRECTA');
      UNTIL (OPCION>=1) AND (OPCION<=7);
      CASE OPCION OF
         1 : BEGIN
                NUM_DE_PELICULAS;
                ENTRADA_DATOS;
             END;
         2 : MOSTRAR_POR_TITULO;
```

```
            3 : MOSTRAR_POR_PROTAGONISTA;
            4 : MOSTRAR_POR_DIRECTOR;
            5 : MOSTRAR_POR_GENERO;
            6 : MOSTRAR_POR_PRODUCTORA;
        END;
    UNTIL (OPCION=7);
END;

BEGIN {PROGRAMA PRINCIPAL}
    MENU;
END.
```

EJERCICIO Nro. 4.8

Mantenimiento vehicular realizado mediante la aplicación de subprogramas.

a) Estructura del algoritmo



b) Compilación Fortran

```
      PROGRAM VEHICULO
      WRITE(*,1)
    1 FORMAT(11(/),40X, 'PROGRAMA VAHÍCULO',8(/), 'Elaborado por ALONSO
     *TAMAYO ALZATE' /,23X, 'UNIVERSIDAD NACIONAL DE COLOMBIA' //)
   14 WRITE(*,2)
    2 FORMAT('1',7(/),27X,'Señale la opción (1-5)',///,10x, '(1) CARACTERÍSTICAS',
     *//,10X, '(2) LÍQUIDOS', //,10X, '(3)' ANOMALÍAS', //,10X, '(4) CAMBIOS', //,
     *10X, '(5) CONSEJOS PRÁCTICOS')
      WRITE(*,3)
    3 FORMAT(5(/),20X, 'Seleccione su opción : '\)
      READ(*,4)KOP
    4 FORMAT(I2)
      GOTO(5,6,7,8,9),KOP
   70 WRITE(*,12)
   12 FORMAT(//,20X, 'Desea usted algo más (1-SI, 2-NO): '\)
      READ(*,13)KIP
```

```
 13   FORMAT(I2)
      GOTO(14,15),KIP
  5   CALL CARACTER
      GOTO 70
  6   CALL LIQUIDOS
      GOTO 70
  7   CALL DANOS
      GOTO 70
  8   CAMB= CAMBIO(CAM)
      GOTO 70
  9   CALL CONSEJOS
      GOTO 70
 15   WRITE(*,35)
 35   FORMAT(//,10X, 'Está usted seguro que desea abandonar el programa :
     *(1-SI, 2-NO):  '\)
      READ(*,36)KUP
 36   FORMAT(I2)
      GOTO(80,14),KUP
 80   STOP
      END

     SUBROUTINE CARACTER
      WRITE(*,11)
 11   FORMAT(4(/),32X,'CARACTERÍSTICAS',5(/),7X, 'PESO (Kgr) : 685',//,7X,
     * 'DIMENSIONES (MTS):  ',/,8X, 'LARGO:          3.66',/,8X, 'ANCHO:
     * 1.48',/,8X, 'ALTO: (VACÍO)  1.55',//,7X, 'CILINDRADA:      1022cc',//,
     * 7X, 'CAPACIDAD DE GASOLINA:  34 LITROS (9 GALONES)',//,7X, 'TIPO
     * DE GASOLINA:  EXTRA O CORRIENTE',//,7X, 'BATERÍA:  12  VOLTS',
     * ///,31X, 'PARTES DEL MOTOR',//,7X, 'BLOQUE:  ES EL CORAZÓN
     * DEL MOTOR CONSTRUIDO EN HIERRO-ALUMINIO CON UNAS PERFO
     * RACIONES LLAMADAS CILINDROS',//,7X, 'CARBURADOR: UTILIZADO
     * PARA CONTROLAR LA MEZCLA DE GASOLINA Y AIRE NECESARIA
     * PARA LA COMBUSTIÓN',//,7X, 'BOMBA DE GASOLINA:  SUMINISTRA
     * LA GASOLINA AL CARBURADOR DESDE EL TANQUE',//,7X, 'VÁLVULAS:
     * LA MEZCLA DE AIRE Y GASOLINA ENTRA A LOS CILINDROS A TRAVÉS
     * DE LAS VÁLVULAS DE ADMISIÓN. LOS GASES GENERADOS DESPUÉS
     * DE LA COMBUSTIÓN SALEN A TRAVÉS DE LAS VÁLVULAS DE ESCAPE',
     * //,7X, 'DISTRIBUIDOR:  LA MEZCLA SE QUEMA EN LOS CILINDROS ME
     * DIANTE UNA CHISPA PRODUCIDA POR EL DISTRIBUIDOR Y LA BOBI
     * NA',//,7X, 'FILTRO DEL AIRE:  PURIFICA EL AIRE PARA EVITAR SUCIE
     * DADES EN LOS CILINDROS',//,7X, 'PISTONES:  LA GASOLINA ENCENDI
     * DA GENERA ENERGÍA QUE OBLIGA A LOS PISTONES A MOVERSE HA
     * CIA ARRIBA Y HACIA ABAJO EN LOS CILINDROS')
      DO 90 I=1,1000000,2
```

```
 90   CONTINUE
      WRITE(*,50)
 50   FORMAT(//,7X, 'CIGUEÑAL: LOS PISTONES A SU VEZ HACEN ROTAR EL
     * CIGUEÑAL QUE SE ENCUENTRA EN LA PARTE INFERIOR DEL MOTOR',
     * //,7X, 'VOLANTE:  EL CIGUEÑAL HACE GIRAR EL VOLANTE. EL VOLAN
     * TE SUAVIZA LA POTENCIA TRANSMITIDA POR EL MOTOR A LAS RUE
     * DAS DEL CARRO',//,7X, 'ÁRBOL DE LEVAS:  GIRA ARRASTRADO A LA
     * MITAD DE VELOCIDAD DEL CIGUEÑAL, ABRE Y CIERRA LAS VÁLVU
     * LAS DE ADMISIÓN Y ESCAPE',//,7X, 'CADENA DE DISTRIBUCIÓN: EL ÁR
     * BOL DE LEVAS GIRA ARRASTRADO POR EL CIGUEÑAL MEDIANTE UNA
     * CADENA DE DISTRIBUCIÓN',//,7X, 'BOMBA DE AGUA:  IMPULSA EL A
     * GUA PARA ELIMINAR EL CALOR DEL MOTOR POR MEDIO DEL RADIA
     * DOR',//,7X, 'CORREA-VENTILADOR:  LA BOMBA DE AGUA ES IMPULSA
     * DA POE EL CIGUEÑAL MEDIANTE UNA CORREA LLAMADA CORREA
     * DEL VENTILADOR',//;7X, 'TERMOSTATO:  PARA EVITAR QUE EL AGUA
     * CIRCULE ANTES DE QUE EL MOTOR HAYA ALCANZADO LA TEMPERA
     * TURA ADECUADA SE INSTALA EL TERMOSTATO. ÉSTE ACTÚA COMO
     * UNA VÁLVULA QUE PERMITE O EVITA EL PASO DEL AGUA',//,7X, 'BOM
     * BA Y FILTRO DE ACEITE:  EL ACEITE EVITA EL DESGASTE DE PARTES
     * EN MOVIMIENTO DEL MOTOR. EXISTEN CONDUCTOS EN EL BLOQUE
     * DEL MOTOR A TRAVÉS DE LOS CUALES CIRCULA EL ACEITE BAJO PRE
     * SIÓN IMPULSADO POR LA BOMBA DEL ACEITE. EL ACEITE SE MANTIE
     * EN LIMPIO MEDIANTE UN FILTRO');
      DO 100 I=1,1000000,2
100   CONTINUE
      WRITE(*,60)
 60   FORMAT(//,7X, 'CARTER:  EN LA PARTE INFERIOR DEL RECIPIENTE DON
     * DE GIRA EL CIGUEÑAL SE ENCUENTRA EL CARTER, EN EL CUAL DRENA
     * EL ACEITE. EL ACEITE QUE SE CALIENTA POR EL MOTOR EN MOVIMIEN
     * TO, LLEGA AL CARTER PARA ENFRIARSE',//,7X, 'ALTERNADOR:  COMO
     * ELEMENTO PRINCIPAL DEL SISTEMA ELÉCTRICO DE LOS CARROS MO
     * DERNOS ESTÁ EL ALTERNADOR QUE CARGA LA BATERÍA.ÉSTE ES TAM
     * BIÉN IMPULSADO POR LA CORREA DEL VENTILADOR',////,28X, 'FUNCIO
     * NAMIENTO DEL MOTOR',//,7X, 'ADMISIÓN:  CUANDO EL PISTÓN DES
     * CIENDE, LA MEZCLA DE AIRE Y GASOLINA ENTRA AL CILINDRO A TRA
     * VÉS DE LA VÁLVULA DE ADMISIÓN',//,7X, 'COMPRESIÓN:  EL CIGUE
     * ÑAL AL GIRAR MUEVE EL PISTÓN NUEVAMENTE HACIA ARRIBA Y
     * COMPRIME LA MEZCLA DE AIRE Y GASOLINA',//,7X,'EXPLOSIÓN: CUAN
     * DO EL PISTÓN SE ACERCA A LA PARTE SUPERIOR, SALTA UNA CHISPA
     * EN LA BUJÍA QUE ENCIENDE LA MEZCLA DE AIRE Y GASOLINA. EL EN
     * CENDIDO DE LA MEZCLA ORIGINA UNA EXPANSIÓN DE LOS GASES
     * QUE EMPUJA EL PISTÓN HACIA ABAJO',//,7X, 'ESCAPE:  EL PISTÓN INI
     * CIA NUEVAMENTE SU CARRERA HACIA ARRIBA, EXPULSANDO LOS
     * GASES QUEMADOS QUE HAN ENTREGADO SU ENERGÍA A TRAVÉS DE
```

```
     * LA VALVULA DE ESCAPE')
       RETURN
      END

      SUBROUTINE LIQUIDOS
        WRITE(*,16)
 16     FORMAT(4(/),36X, 'LÍQUIDOS',//,7X, 'ACEITE: DEBE HACERSE SU CAM
     * BIO CADA 3000 KMS SI ES ACEITE NORMAL O CADA 6000 KMS SI EL
     * ACEITE ES (SE,SF,SG)',//,7X, 'LÍQUIDO DE REFRIGERACIÓN: ES MUY
     * IMPORTANTE YA QUE ES EL QUE MANTIENE EL MOTOR A UNA TEMPE
     * RATURA ESTABLE SEA CUAL SEA EL ESTADO DEL CLIMA Y COMO SU
     * NOMBRE LO DICE REFRIGERA EL MOTOR YA QUE POR LA FRICCIÓN DE
     * LOS PISTONES CON LAS CAMISAS PRODUCE ALTAS TEMPERATU
     * RAS',//,8X, 'NOTA: HAY QUE TENER EXCESIVO CUIDADO CON ESTE LÍ
     * QUIDO QUE NO ES MÁS QUE AGUA, YA QUE SU AUSENCIA PUEDE CAU
     * SAR DAÑOS EN LA CULATA Y BOMBA DE AGUA. LOS DAÑOS EN LA CU
     * LATA SE CAUSAN PRINCIPALMENTE POR DAÑOS EN EL TERMOSTATO
     * EL CUAL AVISA CUANDO HAY AUSENCIA DE AGUA A TRAVÉS DE UNA
     * SEÑAL EN EL TABLERO DE CONTROLES',//,7X, 'COMBUSTIBLE EXTRA O
     * CORRIENTE ESTO VA SEGÚN LA NECESIDAD DEL VEHÍCULO, SE PUEDE
     * UTILIZAR SEGÚN INDICACIÓN DEL CONCESIONARIO',//,7X, 'BATERÍA:
     * COMO TODOS LOS ANTERIORES TIENE UN NIVEL MÍNIMO Y UN NIVEL
     * MÁXIMO, LA BATERÍA ES UN ARTÍCULO CASI DESECHABLE POR LO
     * CUAL ES SUGERIBLE CAMBIARLA CADA 2 AÑOS'/)
        DO 110 I=1,1000000,2
 110    CONTINUE
        WRITE(*,40)
  40    FORMAT(/,7X, 'LÍQUIDO DE FRENOS: ES UN LÍQUIDO INDISPENSABLE
     * EN CUANTO A LA SEGURIDAD DE LOS PASAJEROS Y PROTECCIÓN DEL
     * AUTOMÓVIL, YA QUE SI SE PRESENTA UNA BAJA SÚBITA DEL LÍQUIDO
     * PUEDE SER CAUSADA POR UN DAÑO EN EL SISTEMA Y ASÍ FALLAR
     * LOS FRENOS, ES ACONSEJABLE FRENAR CON CAJA SUAVEMENTE AN
     * TES DE FRENAR DE UNA BOMBEADA DEL PEDAL, YA QUE ES POSIBLE
     * QUE ENTRE AIRE A LOS CONDUCTOS')
        RETURN
       END

        FUNCTION CAMBIO(CMS)
          WRITE(*,18)
   18     FORMAT(////,32X, 'CAMBIOS',//,7X, '(1) CAMBIO CADA 3000 KMS',//,7X,
     *  '(2) CAMBIO CADA 6000 KMS')
          WRITE(*,19)
   19     FORMAT(//,15X, 'ESCRIBA (1 Ó 2) SEGÚN SU ELECCIÓN:'\)
```

```
      READ(*,20)KEP
 20   FORMAT(I2)
      WRITE(*,21)
 21   FORMAT(//,10X, 'ESCRIBA EL NÚMERO DE KMS RECORRIDOS POR SU
     * VEHÍCULO:'\)
      READ(*,*)CAM
      GOTO(23,26),KEP
 23   CAM= CAM-3000
      IF (CAM .GE. 0 .AND. CAM .LE. 3000) GOTO 24
      GOTO 23
 26   CAM= CAM-6000
      IF (CAM .GE. 0 .AND. CAM .LE. 6000) GOTO 24
      GOTO 26
 24   CMS= CAM
      WRITE(*,200)CMS
200   FORMAT(///,10X, 'SU PRÓXIMO CAMBIO ES A LOS: ',F10.2, 'KMS')
      RETURN
     END

    SUBROUTINE DANOS
      WRITE(*,30)
 30   FORMAT(////,30X, 'ANOMALÍAS GENERALES',//,7X, 'SI AL ACCIONAR EL
     * MOTOR DE ARRANQUE:',//,7X, 'NO HAY REACCIÓN ALGUNA, LAS LÁM
     * PARAS TESTIGO NO SE ENCIENDEN Y EL MOTOR DE ARRANQUE NO GI
     * RA,PUEDE SER QUE EL CABLE ELÉCTRICO DE LA BATERÍA ESTÁ DESCO
     * NECTADO DE LOS TERMINALES, RÁSPELOS Y LÍMPIELOS SI ESTÁN OXI
     * DADOS Y APRIÉTELOS. TAMBIÉN EL QUE LA BATERÍA ESTÉ FUERA DE
     * USO LA SOLUCIÓN ES CAMBIARLA',//,7X, 'LAS LÁMPARAS TESTIGO SE
     * DEBILITAN Y EL MOTOR DE ARRANQUE GIRA LENTAMENTE, LAS CAU
     * SAS PUEDEN SER TERMINALES MAL APRETADOS Y BORNES DE BATE
     * RÍA OXIDADOS, ASÍ QUE COMPRUEBE EL CONTACTO DE LOS TERMINA
     * LES. LA BATERÍA PUEDE ESTAR SUCIA O DESCARGADA POR LO TANTO
     * HAGA QUE LE EMPUJEN CON EL CONTACTO PUESTO. HAY QUE PONER
     * EL CARRO EN SEGUNDA VELOCIDAD Y EMBRAGAR CUANDO EL VEHÍ
     * CULO HA TOMADO SUFICIENTE IMPULSO, O CONECTAR LA BATERÍA
     * DESCARGADA CON OTRA BATERÍA')
      DO 120 I=1,1000000,2
120   CONTINUE
      WRITE(*,31)
 31   FORMAT(/,7X, 'EL MOTOR DA EXPLOSIONES PERO NO ARRANCA O A
     * RRANCA CON DIFICULTAD, LA CAUSA PUEDE SER UNA ALIMENTACIÓN
     * INCORRECTA: EXCESO DE GASOLINA (MOTOR AHOGADO) ASÍ QUE SA
     * QUE LAS BUJÍAS, LÍMPIELAS Y SÉQUELAS;COMPRUEBE LA SEPARACIÓN
     * DE ELECTRODOS (0.55-0.65mm) Y VUÉLVALAS A PONER.TAMBIÉN PUEDE
```

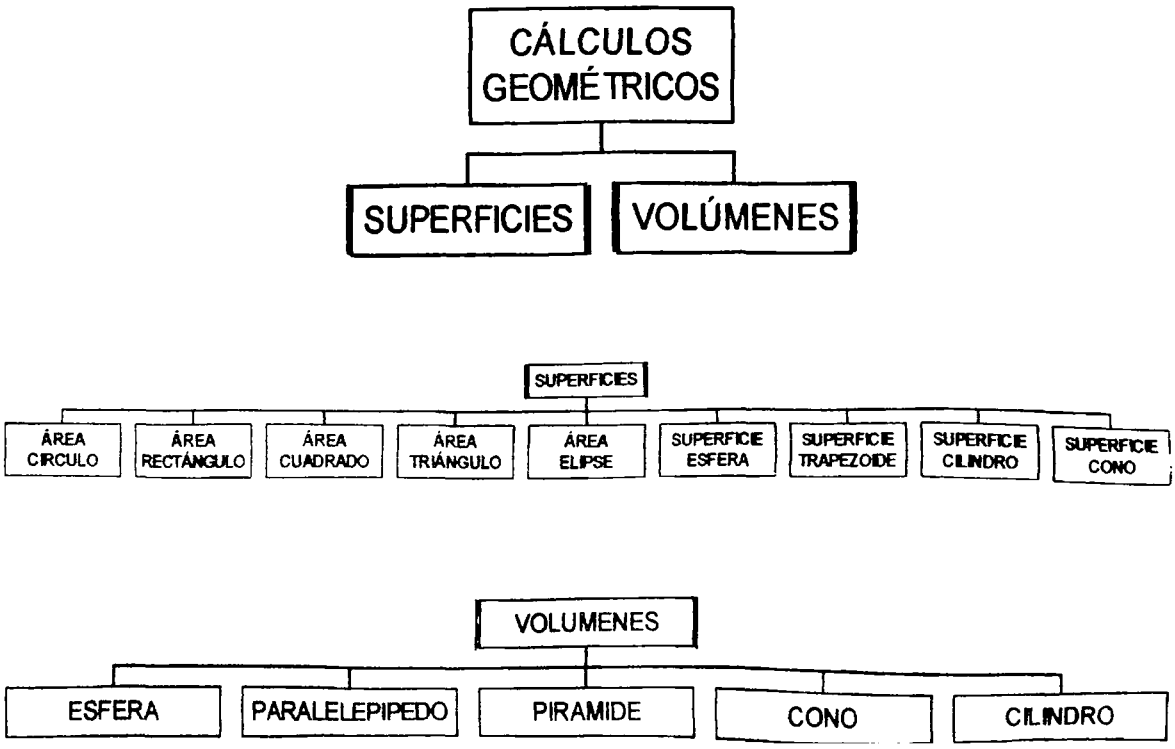
```
     *  SER UNA ALIMENTACIÓN DEFECTUOSA POR EL FUNCIONAMIENTO
     *  INCORRECTO DEL STARTER. COMPRUEBE LA POSICIÓN DEL ESTRAN
     *  GULADOR DE ARRANQUE EN FRÍO, SACANDO EL FILTRO DE AIRE.
     *  EL ESTRANGULADOR DEBE HALLARSE EN POSICIÓN CERRADA DEL
     *  STARTER SACADO A FONDO',//,7X, 'EL MOTOR ARRANCA DIFÍCILMEN
     *  TE POR TIEMPO HÚMEDO O DESPUÉS DE LAVAR EL VEHÍCULO, O SE
     *  DETIENE: PUEDE HABER UN ENCENDIDO DEFECTUOSO POR HUMEDAD
     *  EN EL SISTEMA DE ENCENDIDO PARA SOLUCIONAR EL PROBLEMA SE
     *  DEBEN SECAR LOS CABLES DE BUJÍAS Y DE LA BOBINA, RETIRE LA TA
     *  PA DEL DISTRIBUIDOR Y SEQUE EL EXTERIOR Y EL INTERIOR CON UN
     *  TRAPO SECO')
          DO 130 I=1,1000000,2
130       CONTINUE
          WRITE(*,32)
 32       FORMAT(/,7X, 'EL MOTOR ARRANCA DIFÍCILMENTE EN CALIENTE, PUE
     *  DE SER POR UN ENCENDIDO DEFECTUOSO, UNA CARBURACIÓN DEFEC
     *  TUOSA (BURBUJAS DE GAS EN EL CIRCUITO), O FALTA DE COMPRE
     *  SIÓN, POR LO TANTO DEJE ENFRIAR EL MOTOR. PISE A FONDO EL ACE
     *  LERADOR Y ACCIONE EL MOTOR DE ARRANQUE DE 10 A 20seg',///,7X,
     *  'EN LOS APARATOS ELÉCTRICOS: ',//,7X, 'CUANDO EL LIMPIABRISAS
     *  NO FUNCIONA PUEDE OCURRIR QUE HAY UN FUSIBLE ROTO, POR LO
     *  TANTO CAMBIÉLO, TAMBIÉN UN AGARROTAMIENTO DE LOS EJES ASÍ
     *  QUE TRATE DE DESAGARROTARLOS CON UN PRODUCTO ADECUADO
     *  O UNA AVERÍA DEL MOTOR, ACUDA A SU CONCESIONARIO',//,7X, 'SI
     *  LAS LUCES INTERMITENTES NO FUNCIONAN PUEDE HABER UN FUSI
     *  BLE ROTO POR LO TANTO CÁMBIELO O QUE LA CENTRAL DE INTERMI
     *  TENCIA ESTÁ DEFECTUOSA CÁMBIELA',//,7X, 'LOS FAROS NO FUNCIO
     *  NAN, PUEDE SER QUE UNA BOMBILLA ESTÉ FUNDIDA; CÁMBIELA. HILO
     *  DESCONECTADO, COMPRUÉBELO Y CONÉCTELO. TAMBIÉN PUEDE OCU
     *  RRIR QUE LA PUESTA A MASA ESTÉ DEFECTUOSA,POR ESTO REVISE LA
     *  CONEXIÓN')
          RETURN
        END

        SUBROUTINE CONSEJOS
          WRITE(*,33)
 33       FORMAT(/////,27X, 'CONSEJOS ÚTILES EN GENERAL',//,7X, 'MEJOR QUE
     *  CALENTAR EL MOTOR CON EL VAHÍCULO PARADO, CONVIENE HACER
     *  LO CONDUCIENDO SUAVEMENTE HASTA ALCANZAR LA TEMPERATU
     *  RA NORMAL. INTRODUZCA EL CHOKE, TAN PRONTO COMO EL MOTOR
     *  PUEDA GIRAR SIN AYUDA',//,7X, 'LA UTILIZACIÓN DE NEUMÁTICOS NO
     *  PRECONIZADOS PUEDE AUMENTAR EL CONSUMO LO MISMO QUE LA
     *  FALTA DE AIRE'',//,7X, 'EVITE LOS TRAYECTOS CORTOS CON LARGAS
     *  PARADAS, EN LOS QUE EL MOTOR NO ALCANZA NUNCA UNA TEMPE
```

```
*  TURA IDEAL',//,7X, 'EVITE LLENAR EL DEPÓSITO DE GASOLINA AL TOPE
*  Y A QUE ES UNA MANERA DE DESPERDICIAR GASOLINA',//,7X, 'NO RE
*  TIRE NUNCA LA LLAVE DEL CONTACTO SI EL VEHÍCULO SIGUE AÚN
*  EN MOVIMIENTO, PUES CORRE EL RIESGO DE QUE SE QUEDE BLOQUEA
*  DA LA DIRECCIÓN',//,7X, 'EN CASO DE UN DESCENSO ANORMAL EN
*  CUALQUIERA DE LOS NIVELES DE LÍQUIDO, CONSULTE INMEDIATA
*  MENTE SU CONSECIONARIO (NIVEL LÍQUIDO DE FRENOS, DE ACEITE,
*  DE REFRIGERACIÓN)',//,7X, 'NO AGREGUE DETERGENTE AL AGUA DEL
*  LIMPIABRISAS',//,7X, 'NO MEZCLE MARCAS NI CALIDADES DE LÍQUIDO
*  DE FRENOS',//,7X, 'CUANDO CAMBIE UNA BOMBILLA O MÁS, COLOQUE
*  UNA NUEVA DEL MISMO WATIAJE',//,7X, 'SIEMPRE UTILICE LA MISMA
*  MARCA DE ACEITE EN RECAMBIO Y CUANDO SEA NECESARIO COM
*  PLETAR EL NIVEL')
   RETURN
  END
```

EJERCICIO Nro. 4.9

Ejercicio sobre cálculos geométricos relacionados con superficies y volúmenes, mediante la utililización de funciones y subrutinas.

a) Estructura del algoritmo



b) Compilación Fortran

```
      PROGRAM GEOM
      CHARACTER NOS*1
      WRITE(*,1)
    1 FORMAT(12(/),30X, 'PROGRAMA SOBRE CÁLCULOS GEOMÉTRICOS')
    2 WRITE(*,3)
      FORMAT('1',///,29X, 'Cálculos Geométricos',///,15X, '(1) Superficies',//,15X, '(2)
     *Volúmenes',//,15X, '(3) Salida')
      WRITE(*,4)
    4 FORMAT(4(/),20X, 'Seleccione su opción :'\)
      READ(*,5)NUM
    5 FORMAT(I1)
      GOTO(6,7,8),NUM
    6 WRITE(*,9)
    9 FORMAT(4(/),34X, 'SUPERFICIES',///,15X, '(1) Círculo',//,15X, '(2) Rectángulo',
     *//,15X, '(3) Cuadrado',//,15X, '(4) Triángulo',//,15X, '(5) Elipse',//,15X, '(6) Esfe
     *ra',//,15X, '(7) Trapezoide',//,15X, '(8) Lateral de un cilindro circular',//,15X, '(9)
     *Lateral de un cono circular')
      WRITE(*,10)
   10 FORMAT(4(/),20X, 'Seleccione su opción :',\)
      READ(*,11)KUN
   11 FORMAT(I1)
      PHI= 3.141592654
      GOTO(12,13,14,15,16,17,18,19,20),KUN
   12 CALL CIRCULO(PHI,AR)
      GOTO 8
   13 CALL RECTANG(BR)
      GOTO 8
   14 CALL CUADRAD(CR)
      GOTO 8
   15 CALL TRIANGU(DR)
      GOTO 8
   16 CALL ELIPSE(PHI,ER)
      GOTO 8
   17 CALL ESFERA(PHI,FR)
      GOTO 8
   18 CALL TRAPEZ(GR)
      GOTO 8
   19 CALL CILIND(PHI,HR)
      GOTO 8
   20 CALL CONOS(PHI,OR)
      GOTO 8
    7 WRITE(*,21)
```

```
21  FORMAT(4(/),35X, 'VOLÚMENES',///,15X, '(1) Esfera',//,15X, '(2) Paralelepípe
   *do',//,15X, '(3) Pirámide',//,15X, '(4) Cono',//,15X, '(5) Cilindro')
    WRITE(*,22)
22  FORMAT(4(/),20X, 'Seleccione su opción  :',\)
    READ(*,23)KAR
23  FORMAT(I1)
    PHI= 3.141592654
    GOTO(24,26,28,30,32),KAR
24  AVOL= ESFER(PHI)
    WRITE(*,25)AVOL
25  FORMAT(//,20X, 'El volumen de la esfera es  :',F9.2)
    GOTO 8
26  BVOL= PARALE(Q)
    WRITE(*,27)BVOL
27  FORMAT(//,20X, 'El volumen del paralelepípedo es  :',F9.2)
    GOTO 8
28  CVOL= PIRAMI(RR)
    WRITE(*,29)CVOL
29  FORMAT(//,20X, 'El volumen de la pirámide es  :',F9.2)
    GOTO 8
30  DVOL= CONO(PHI)
    WRITE(*,31)DVOL
31  FORMAT(//,20X, 'El volumen del cono es  :',F9.2)
    GOTO 8
32  EVOL= CILIN(PHI)
    WRITE(*,33)EVOL
33  FORMAT(//,20X, 'El volumen del cilindro es  :',F9.2)
 8  S=1
    N=2
34  WRITE(*,35)
35  FORMAT(4(/),20X, 'Desea abandonar el programa (S/N):  ',\)
    READ(*,36)NOS
36  FORMAT(A1)
    IF (NOS.EN.'N'.AND.NOS.NE.'n'.AND.NOS.NE.'S'.AND.NOS.EN.'s')GOTO 34
    IF(NOS.EQ.'N'.OR.NOS.EQ.'n') GOTO 2
    STOP
    END

    SUBROUTINE CIRCULO(PHI,AR)
       WRITE(*,37)
37     FORMAT(//,20X, 'Escriba el radio  :  ',\)
       READ(*,*)NRAD
       AR= PHI*NRAD**2
       WRITE(*,38)AR
```

```
38      FORMAT(//,20X, 'El área del círculo es : ',F9,2)
        RETURN
      END

      SUBROUTINE RECTANG(BR)
        WRITE(*,39)
39      FORMAT(//,20X, 'Escriba el lado A:  ',\)
        READ(*,*)LA
        WRITE(*,40)
40      FORMAT(/,20X, 'Escriba el lado B:  ',\)
        READ(*,*)LB
        BR= LA*LB
        WRITE(*,41)BR
41      FORMAT(//,20X, 'El área del rectángulo es:  ',F9.2)
        RETURN
      END

      SUBROUTINE CUADRAD(CR)
        WRITE(*,42)
42      FORMAT(//,20X, 'Escriba el lado:  ',\)
        READ(*,*)LD
        CR= LD**2
        WRITE(*,43)CR
43      FORMAT(//,20X, 'El área del cuadrado es:  ',F9.2)
        RETURN
      END

      SUBROUTINE TRIANGU(DR)
        WRITE(*,44)
44      FORMAT(//,20X, 'Escriba la altura:  ',\)
        READ(*,*)JAL
        WRITE(*,45)
45      FORMAT(/,20X, 'Escriba la base:  ',\)
        READ(*,*)JBA
        DR= (JAL*JBA)/2
        WRITE(*,46)DR
46      FORMAT(//,20X, 'El área del triángulo es:  ',F9.2)
        RETURN
      END

      SUBROUTINE ELIPSE(PHI,ER)
        WRITE(*,47)
47      FORMAT(//,20X, 'Escriba el eje A:  ',\)
        READ(*,*)JEA
```

```
      WRITE(*,48)
48    FORMAT(/,20X, 'Escriba el eje B:  ',\)
      READ(*,*)JEB
      ER= PHI*JEA*JEB
      WRITE(*,49)ER
49    FORMAT(//,20X, 'El área de la elipse es:  ',F9.2)
      RETURN
    END

    SUBROUTINE ESFERA(PHI,FR)
      WRITE(*,50)
50    FORMAT(//,20X, 'Escriba el radio:  ',\)
      READ(*,*)NRA
      FR= 4*PHI*NRA**2
      WRITE(*,51)FR
51    FORMAT(//,20X, 'La superficie de la esfera es:  ',F9.2)
      RETURN
    END

    SUBROUTINE TRAPEZ(GR)
      WRITE(*,52)
52    FORMAT(//,20X, 'Escriba la base mayor:  ',\)
      READ(*,*)NBM
      WRITE(*,53)
53    FORMAT(/,20X, 'Escriba la base menor:  ',\)
      READ(*,*)NBN
      WRITE(*,54)
54    FORMAT(//,20X, 'Escriba la altura:  ',\)
      READ(*,*)NH.
      GR= (NBM+NBN)*NH/2
      WRITE(*,55)GR
55    FORMAT(//,20X, 'El área del trapezoide es:  ',F9.2)
      RETURN
    END

    SUBROUTINE CILIND(PHI,HR)
      WRITE(*,56)
56    FORMAT(//,20X, 'Escriba el radio:  ',\)
      READ(*,*)NR
      WRITE(*,57)
57    FORMAT(/,20X, 'Escriba la altura:  ',\)
      READ(*,*)NLT
      HR= 2*PHI*NR*NLT
      WRITE(*,58)HR
```

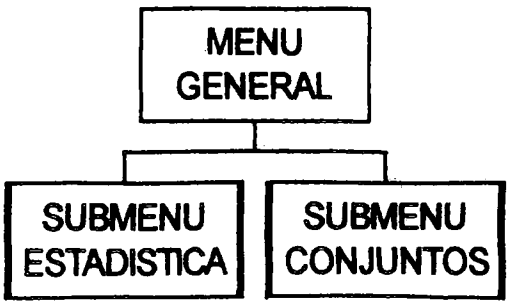
```
58      FORMAT(//,20X, 'La superficie del cilindro es:  ',F9.2)
        RETURN
      END

      SUBROUTINE CONOS(PHI,OR)
        WRITE(*,59)
59      FORMAT(//,20X, 'Escriba el radio:  ',\)
        READ(*,*)NRD
        WRITE(*,60)
60      FORMAT(/,20X, 'Escriba la altura:  ',\)
        READ(*,*)NLA
        OR= PHI*NRD*NLA
        WRITE(*,61)OR
61      FORMAT(//,20X, 'La superficie del cono es:  ',F9.2)
        RETURN
      END

      FUNCTION ESFER(PHI)
        WRITE(*,62)
62      FORMAT(//,20X, 'Escriba el radio:  ',\)
        READ(*,*)LRA
        ESFER= (4*PHI*LRA**3)/3
        RETURN
      END

      FUNCTION PARALE(Q)
        WRITE(*,63)
63      FORMAT(//,20X, 'Escriba el lado A:  ',\)
        READ(*,*)ILA
        WRITE(*,64)
64      FORMAT(/,20X, 'Escriba el lado B:  ',\)
        READ(*,*)ILB
        WRITE(*,65)
65      FORMAT(//,20X, 'Escriba el lado C:  ',\)
        READ(*,*)ILC
        PARALE= ILA*ILB*ILC
        RETURN
      END

      FUNCTION PIRAMI(RR)
        WRITE(*,66)
66      FORMAT(//,20X, 'Escriba el área de la base:  ',\)
        READ(*,*)JAR
        WRITE(*,67)
```

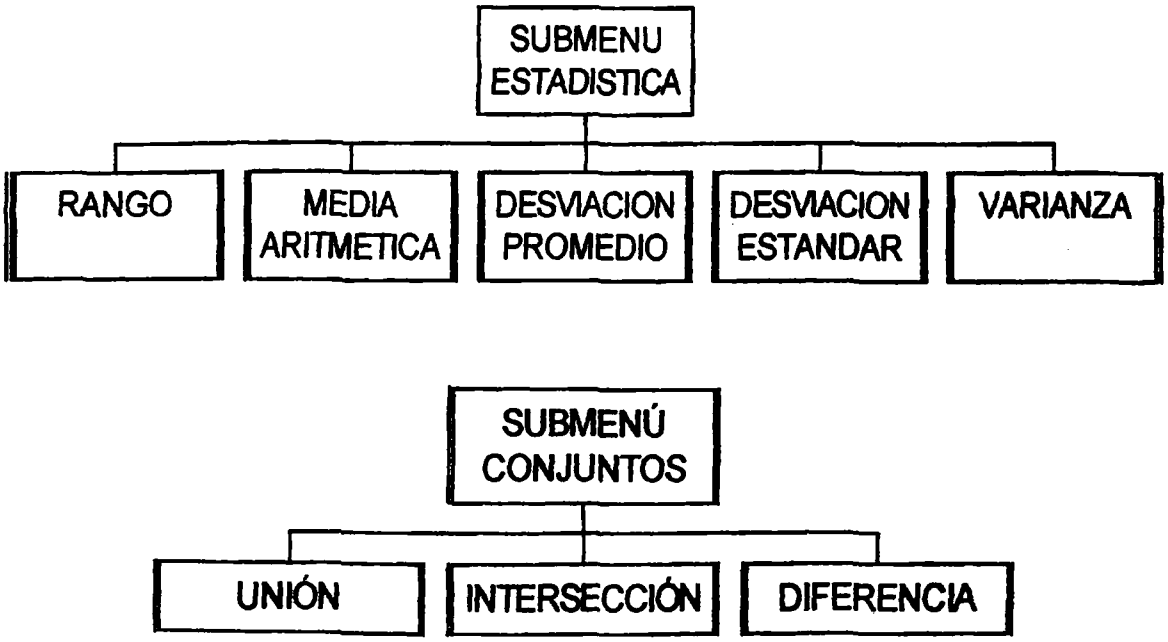
```
67      FORMAT(/,20X, 'Escriba la altura:  ',\)
        READ(*,*)IAL
        PIRAMI= (JAR*IAL)/3
        RETURN
      END

      FUNCTION CONO(PHI)
        WRITE(*,68)
68      FORMAT(//,20X, 'Escriba el radio de la base:  ',\)
        READ(*,*)JRB
        WRITE(*,69)
69      FORMAT(/,20X, 'Escriba la altura:  ',\)
        READ(*,*)IH
        CONO= (PHI*JRB**2*IH)/3
        RETURN
      END

      FUNCTION CILIN(PHI)
        WRITE(*,70)
70      FORMAT(//,20X, 'Escriba el radio de la base:  ',\)
        READ(*,*)JRC
        WRITE(*,71)
71      FORMAT(/,20X, 'Escriba la altura:  ',\)
        READ(*,*)LH
        CILIN= PHI*JRC*2*LH
        RETURN
      END
```

EJERCICIO Nro. 4.10

Programa que realiza cálculos estadísticos y operaciones con conjuntos mediante el encadenamiento de menús.

a) Estructura del algoritmo





b) Compilación Pascal

```
PROGRAM PRACTICA;
USES CRT;
{Programa que calcula algunas medidas estadísticas y realiza operaciones con conjuntos}
TYPE
    CONJLETR     = SET OF CHAR;
VAR
    RES                          :STRING[1];
    X                            :ARRAY[1..100] OF INTEGER;
    N,J,R,MAYOR,MENOR            :INTEGER;
    SUM,DP,S,XMED,DESV           :REAL;
    OK                           :BOOLEAN;
    OPCION,OPCION1,OPCION2       :BYTE;
    LETRA                        :CHAR;
    CAD1,CAD2,CAD3               :STRING[30];
    CONJ1,CONJ2,CONJ3            :CONJLETR;

PROCEDURE PANTA;
{Presentación inicial del programa}
BEGIN
   CLRSCR;
   GOTOXY(20,10);
```

```
    WRITE('UNIVERSIDAD NACIONAL DE COLOMBIA');
    GOTOXY(32,6);
    WRITE('SEDE MANIZALES');
    GOTOXY(26,12),
    WRITE('PRÁCTICA DE PROGRAMACIÓN');
    GOTOXY(26,14);
    WRITE('AUTOR : ');
    GOTOXY(40,20);
    WRITE('ALONSO TAMAYO ALZATE');
    GOTOXY(26,24);
    WRITE('MANIZALES, OCTUBRE DE 1995');
    DELAY(4000);
END; {PANTA}

PROCEDURE ERROR;
{Produce mensaje de error}
BEGIN
    GOTOXY(1,24);
          WRITE('ERROR EN DATOS, PRESIONE LA TECLA ENTER PARA
CONTINUAR');
    GOTOXY(60,24);
    READ(RES);
    GOTOXY(1,24);CLREOL;
END;

PROCEDURE MENUGENERAL;
{Crea el menú general}
BEGIN
    CLRSCR;
    GOTOXY(1,4);
WRITE('*************************************************************************');
    GOTOXY(1,8);
WRITE('*************************************************************************');
    GOTOXY(30,6);
    WRITE('MENÚ GENERAL');
    GOTOXY(10,14);
    WRITE('1- MENÚ ESTADÍSTICAS');
    GOTOXY(10,16);
    WRITE('2- MENÚ CONJUNTOS');
    GOTOXY(10,18);
    WRITE('3- FIN DE LA SESIÓN');
    GOTOXY(5,24);
    WRITE('Teclee la opción seleccionada');
    REPEAT
```

```
        GOTOXY(50,24);
        READ(OPCION);
        OK:= (OPCION>=1) AND (OPCION<=3);
        IF NOT OK THEN
           ERROR;
    UNTIL OK;
END; {MENUGENERAL}

PROCEDURE MENUESTA;
{Crea el menú de estadística}
BEGIN
    CLRSCR;
    GOTOXY(33,6);
    WRITE('MENÚ ESTADÍSTICA');
    GOTOXY(10,10);
    WRITE('1- RANGO');
    GOTOXY(10,12);
    WRITE('2- MEDIA ARITMÉTICA');
    GOTOXY(10,14);
    WRITE('3- DESVIACIÓN PROMEDIO');
    GOTOXY(10,16);
    WRITE('4- DESVIACIÓN ESTÁNDAR');
    GOTOXY(10,18);
    WRITE('5- VARIANZA');
    GOTOXY(10,20);
    WRITE('6- RETORNO AL MENÚ GENERAL');
    GOTOXY(5,24);
    WRITE('Teclee la opción seleccionada');
    REPEAT
       GOTOXY(50,24);
       READ(OPCION1);
       OK:= (OPCION1>=1) AND (OPCION1<=6);
       IF NOT OK THEN
          ERROR;
    UNTIL OK;
END; {MENUESTA}

PROCEDURE MENUCON;
{Crea el menú de conjuntos}
BEGIN
    CLRSCR;
    GOTOXY(35,6);
    WRITE('MENÚ CONJUNTOS');
    GOTOXY(10,10);
```

```
    WRITE('1- UNIÓN');
    GOTOXY(10,12);
    WRITE('2- INTERSECCIÓN');
    GOTOXY(10,14);
    WRITE('3- DIFERENCIA');
    GOTOXY(10,16);
    WRITE('4- RETORNO AL MENÚ GENERAL');
    GOTOXY(5,24);
    WRITE('Teclee la opción seleccionada');
    REPEAT
       GOTOXY(50,24);
       READ(OPCION2);
       OK:= (OPCION2>=1) AND (OPCION2<=4);
       IF NOT OK THEN
          ERROR;
    UNTIL OK;
END; {MENUCON}

PROCEDURE CAPTURA;
{Procedure captura de datos}
BEGIN
    CLRSCR;
    GOTOXY(10,14);
    WRITE('Son datos nuevos para procesar ? (S/N)');
    GOTOXY(50,4);
    READ(RES);
    IF (RES='S') OR (RES='s') THEN
       BEGIN
          GOTOXY(10,8);
          WRITE('Teclee el total de datos a procesar ');
          REPEAT
             GOTOXY(50,8);
             READ(N);
             OK:= (N>000) AND (N<=999);
             IF NOT OK THEN
                ERROR;
             GOTOXY(50,8);
          UNTIL OK;
          FOR J:=1 TO N DO
             X[J]:=0;
          FOR J:=1 TO N DO
             BEGIN
                REPEAT
                   GOTOXY(10,12);
```

```
                    WRITE('Teclee cada dato y presione la tecla Enter');
                    GOTOXY(30,16);
                    READ(X[J]);
                    OK:= (X[J]>= 000) AND (X[J]<= 999);
                    IF NOT OK THEN
                       ERROR;
                    GOTOXY(30,16);CLREOL;
                 UNTIL OK;
              END;
        END;
     CLRSCR;
END; {CAPTURA}

PROCEDURE CONJUNDATOS;
{Procedure captura de conjuntos}
BEGIN
    CLRSCR;
    GOTOXY(10,4);
    WRITE('Son nuevos conjuntos de datos ? (S/N)');
    GOTOXY(50,4);
    READ(RES);
    IF (RES='S') OR (RES='s') THEN
       BEGIN
          REPEAT
             GOTOXY(10,8);
             WRITE('Teclee el primer conjunto de caracteres');
             GOTOXY(10,10);
             CLREOL;
             READ(CAD1);
             CONJ1:=[ ];
             FOR J:=1 TO LENGTH(CAD1) DO
                CONJ1:=CONJ1+[CAD1[J]];
             OK:= CONJ1<= ['1'.. 'Z'];
             IF NOT OK THEN
                ERROR;
          UNTIL OK;
          REPEAT
             GOTOXY(10,16);
             WRITE('Teclee el segundo conjunto de caracteres');
             GOTOXY(10,18);
             READ(CAD2);
             CONJ2:=[ ];
             FOR J:=1 TO LENGTH(CAD2) DO
```

```
                CONJ2:=CONJ2+[CAD2[J]];
            OK:= CONJ2<= ['1'.. 'Z'];
            IF NOT OK THEN
                ERROR;
        UNTIL OK;
    END;
END; {CONJUNDATOS}

PROCEDURE MED;
{Procedimiento que calcula la media de un conjunto de datos')
BEGIN
    SUM:= 0.0;
    FOR J:= 1 TO N DO
        SUM:= SUM+X[J];
    XMED:= SUM/N;
END; {MED}

PROCEDURE RANGO;
{Procedimiento que calcula el rango de un conjunto de datos}
BEGIN
    CAPTURA;
    MAYOR:= X[1];
    MENOR:= X[1];
    FOR J:= 2 TO N DO
        BEGIN
            IF (X[J]>MAYOR) THEN
                MAYOR:= X[J]
            ELSE
                IF (X[J]<MENOR) THEN
                    MENOR:= X[J];
        END;
    R:= MAYOR-MENOR;
    CLRSCR;
    GOTOXY(20,4);
    WRITE('Los datos analizados son  : ');
    GOTOXY(1,6);
    FOR J:= 1 TO N DO
        WRITE(X[J]:5);
    GOTOXY(10,16);
    WRITE('El elemento mayor es   ',MAYOR:5);
    GOTOXY(10,18);
    WRITE('El elemento menor es   ',MENOR:5);
    GOTOXY(10,20);
```

```
    WRITE('El rango es ',R:5);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {RANGO}

PROCEDURE MEDIA;
{Cálculo de la media aritmética}
BEGIN
    CAPTURA;
    MED;
    CLRSCR;
    GOTOXY(20,4);
    WRITE('Los datos analizados son  : ');
    GOTOXY(1,6);
    FOR J:=1 TO N DO
        WRITE(X[J]:5);
    GOTOXY(10,16);
    WRITE('La media aritmética es ',XMED:6:2);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {MEDIA}

PROCEDURE DESPROM;
{Cálculo de la desviación promedio}
BEGIN
    CAPTURA;
    MED;
    SUM:= 0.0;
    FOR J:=1 TO N DO
        SUM:= SUM+ABS(X[J]-XMED);
    DP:= SUM/N;
    CLRSCR;
    GOTOXY(20,4);
    WRITE('Los datos analizados son  : ');
    GOTOXY(1,6);
    FOR J:=1 TO N DO
        WRITE(X[J]:5);
    GOTOXY(10,16);
    WRITE('La desviación promedio es ',DP:6:2);
    GOTOXY(10,23);
```

```
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {DESPROM}

PROCEDURE EST;
{Cálculo de la sumatoria de las desviaciones}
BEGIN
    CAPTURA;
    MED;
    SUM:= 0.0;
    FOR J:=1 TO N DO
        SUM:= SUM+SQR(X[J]-XMED);
END; {EST}

PROCEDURE ESTANDAR;
{Cálculo de la desviación estándar}
BEGIN
    EST;
    DESV:= SQRT(SUM/N);
    CLRSCR;
    GOTOXY(20,4);
    WRITE('Los datos analizados son  : ');
    GOTOXY(1,6);
    FOR J:=1 TO N DO
        WRITE(X[J]:5);
    GOTOXY(10,16);
    WRITE('La desviación estándar es  ',DESV:6:2);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {ESTANDAR}

PROCEDURE VARIANZA;
{Cálculo de la varianza}
BEGIN
    EST;
    S:= SUM/N;
    CLRSCR;
    GOTOXY(20,4);
    WRITE('Los datos analizados son  : ');
    GOTOXY(1,6);
    FOR J:=1 TO N DO
```

```
        WRITE(X[J]:5);
    GOTOXY(10,16);
    WRITE('La varianza es ',S:6:2);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {VARIANZA}

PROCEDURE IMPRESION;
{Procedimiento de impresión o escritura}
VAR
     LETRA  :CHAR;
BEGIN
    CAD3:= '';
    FOR LETRA:= 'A' TO 'Z' DO
        IF LETRA IN CONJ3 THEN
            CAD3:= CONCAT(CAD3,LETRA);
    CLRSCR;
    GOTOXY(10,6);
    WRITE('El primer conjunto es : ',CAD1);
    GOTOXY(10,12);
    WRITE('El segundo conjunto es : ',CAD2);
END; {IMPRESION}

PROCEDURE UNION
{Unión de conjuntos}
BEGIN
    CLRSCR;
    CONJUNDATOS;
    CLRSCR;
    CONJ3:= CONJ1+CONJ2;
    IMPRESION;
    GOTOXY(10,18);
    WRITE('La unión de los conjuntos es : ',CAD3);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {UNION}

PROCEDURE INTER;
{Intersección de conjuntos}
BEGIN
```

```
    CLRSCR;
    CONJUNDATOS;
    CLRSCR;
    CONJ3:= CONJ1*CONJ2;
    IMPRESION;
    GOTOXY(10,18);
    WRITE('La intersección de los conjuntos es : ',CAD3);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {INTER}

PROCEDURE DIFER;
{Diferencia de conjuntos}
BEGIN
    CLRSCR;
    CONJUNDATOS;
    CLRSCR;
    CONJ3:= CONJ1-CONJ2;
    IMPRESION;
    GOTOXY(10,18);
    WRITE('La diferencia de conjuntos es : ',CAD3);
    GOTOXY(10,23);
    WRITE('PRESIONE LA TECLA ENTER PARA CONTINUAR');
    GOTOXY(55,23);
    READ(RES);
END; {DIFER}

BEGIN {PROGRAMA PRINCIPAL}
    PANTA;
    REPEAT
       MENUGENERAL;
       CASE OPCION OF
          1: BEGIN
              REPEAT
                 MENUESTA;
                    CASE OPCION1 OF
                       1: RANGO;
                       2: MEDIA;
                       3: DESPROM;
                       4: ESTANDAR;
                       5: VARIANZA;
                       6: MENUGENERAL;
```

```
                END;
            UNTIL OPCION1=6
          END;
        2: BEGIN
            CLRSCR;
            REPEAT
              MENUCON;
              CASE OPCION2 OF
                1: BEGIN
                     CLRSCR;
                     IMPRESION;
                     UNION;
                   END;
                2: BEGIN
                     CLRSCR;
                     IMPRESION;
                     INTER;
                   END;
                3: BEGIN
                     CLRSCR;
                     IMPRESION;
                     DIFER;
                   END;
                4: BEGIN
                     CLRSCR;
                     MENUGENERAL;
                   END;
              END;
            UNTIL OPCION2=4;
          END;
        3: CLRSCR;
      END;
  UNTIL OPCION=3;
  CLRSCR;
END. {PRACTICA}
```

BIBLIOGRAFÍA

PRIETO Alberto. Introducción a la informática. Mc. Graw Hill. 1989

LEVINE GUTIÉRREZ Guillermo. Introducción a la computación. Mc Graw Hill. 1985

GOLDSCHLAGER Andrew Lister. Introducción moderna a la ciencia de la computación. Un enfoque algorítmico. Prentice Hall. 1986

TREMBLAY Jean Paul. Introducción a las ciencias de los computadores. Un enfoque algorítmico. Mc. Graw Hill. 1987

AHO V. Alfred. Estructura de datos y algoritmos. Addison Wesley. 1988

WIRTH Niklaus. Algoritmos y estructura de datos. Prentice Hall. 1987

ABELLANAS M. Análisis de algoritmos y teoría de grafos. Macrobit RA-MA. 1991

HAMMOND H. Robert. Introducción al Fortran 77. Mc Graw Hill. 1988

FRIEDMAN L. Frank. Fortran. Introducción al lenguaje y resolución de problemas con programación estructurada. Ed. Fondo Educativo Interamericano. 1984

LIPSCHUTZ Seymour. Programación con Fortran. Mc Graw Hill. 1982

DAVIS B. Gordon. Fortran 77. Un estilo estructurado. Mc Graw Hill. 1990

LEE Hans. The design and complementation of programs in Fortran 77. Prentice Hall. 1990

SCHILDT HERBERT. Programación y técnicas Turbo Pascal Avanzado. Mc Graw Hill. 1988

HAREL David. Algorithmics. Addison Wesley. 1987

BECERRA SATAMARÍA César A. Turbo Pascal. Ed. Por Computador. 1987

WIRTH NIKLAUS. Algoritmos y estructuras de datos. Prentice Hall. 1987

PRATT TERRENCE W. Lenguajes de programación. Prentice Hall. 1984

TUCKER. Lenguajes de programación. Mc. Graw Hill. 1986

CARROL W. David. Programación en Turbo Pascal. Mc Graw Hill. 1988

SETHI Ravi. Lenguajes de Programación: Conceptos y constructores. Addison Wesley. 1991

NUNCIO LIMON Reynaldo. Historia y perspectivas de la programación: Fundamentos de informática. Ed. Trillas. 1991

ROA MACKENZIE Mauricio. Curso básico de programación. Me. Graw Hill. 1991

FOLEY RICHARD W. Introducción a la programación con Turbo Pascal. Addison Wesley. 1993

DALE Nell. Pascal y estructura de datos. Mc Graw Hill. 1992

JOYANES AGUILAR Luis. Programación en Turbo Pascal. Mc Graw Hill. 1990

JOYANES AGUILAR Luis. Pascal y Turbo Pascal. Mc. Graw Hill. 1994