Módulo Imperativo

Recursión

Autores:
Alejandro Héctor Gonzalez
Silvana Lis Gallo
Mayo 2021

Resumen

En esta clase se aborda el concepto de Recursión, se reaizan varios ejemplos para entender como funciona la recursión y en particular la pila de activación.

Palabras clave

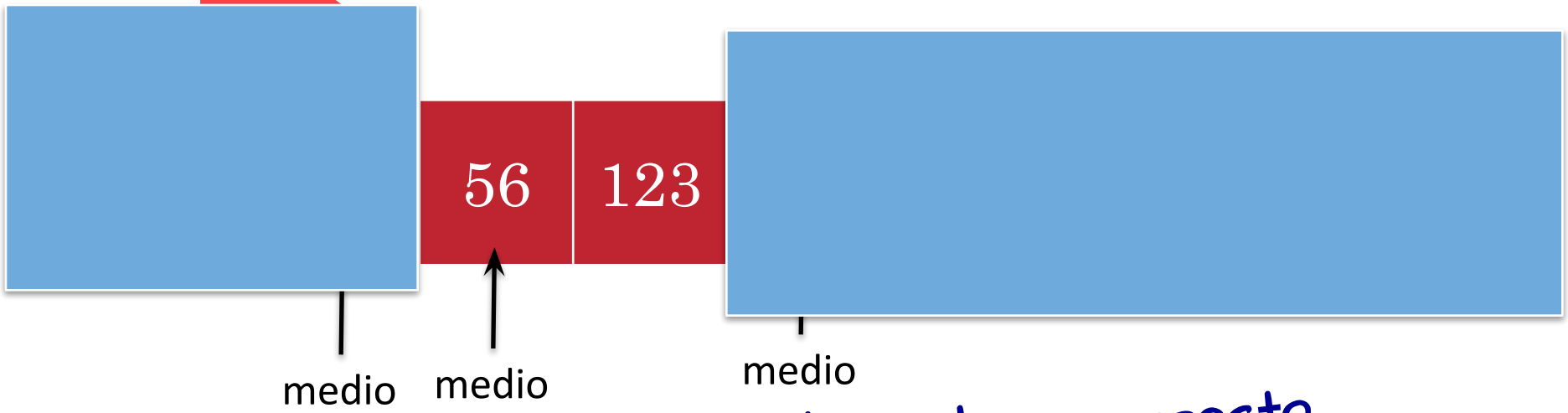
recursión, pila de activación, procedure, function, llamado

Temas de la clase

1. Recursión. Definición y características
2. Ejemplos de Recursión
3. Método de búsqueda dicotómica en vectores. Una aplicación

Motivación
Búsqueda dicotómica

Busco el 56



¿Cómo es el medio con respecto a 56?

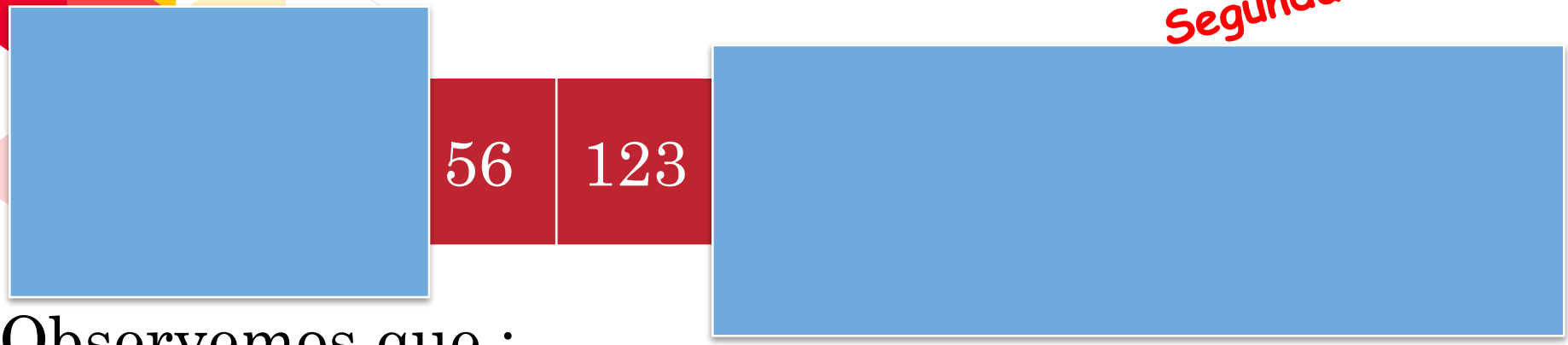
1. Si es = Encontré el 56!
2. Si es > Sigo con la mitad inferior
3. Si es < Sigo con la mitad superior

Búsqueda dicotómica

Medio??

Primera mitad??

Segunda mitad??



Observemos que :

1. La primera vez se trabaja con el vector completo para determinar el punto medio
2. La siguiente vez, el vector se reduce a la mitad
3. La siguiente vez, el vector se reduce a la mitad de la mitad

Búsqueda dicotómica

Buscar (vector, datoABuscar)

Si el vector "no tiene elementos" **entonces**

no lo encontré y termino la búsqueda

sino

Determinar el punto medio del vector

Comparar **datoABuscar** con el contenido del punto medio

Si coincide **entonces** "Lo encontré"

sino

Si datoABuscar < contenido del punto medio **entonces**

Buscar (en la 1era mitad del vector, datoABuscar)

sino

Buscar (en la 2da mitad del vector, datoABuscar)

BÚSQUEDA DICOTÓMICA

```
Buscar (vector, datoABuscar)
  Si el vector "no tiene elementos"  entonces
    no lo encontré y termino la búsqueda
  sino
    Determinar el punto medio del vector
    Comparar datoABuscar con el contenido del punto medio
    Si coincide entonces "Lo encontré"
    sino
      Si datoABuscar < contenido del punto medio entonces
         Buscar (en la 1era mitad del vector, datoABuscar)
         sino
         Buscar (en la 2da mitad del vector, datoABuscar)
```

Observaciones importantes de esta solución :

1) El módulo realiza invocaciones a si mismo. En cada llamada, el tamaño del vector se reduce a la mitad.

1) Existen 2 casos distintos que se resuelven de manera particular y directa:

a) Cuando el vector "no contiene elementos"
b) Cuando encuentro el datoABuscar

Recursión. Definición y Características

Una solución recursiva resuelve un problema por resolución de instancias más pequeñas del mismo problema.

Un algoritmo recursivo involucra:

1. Alguna condición de terminación (implícita/explícita) **CASO BASE**
2. Una ***auto-invocación***. Se debe garantizar que en un nro finito de autoinvocaciones se alcanza la condición de terminación.
3. Se **achica el espacio del problema** en cada llamada

Ejemplo de Recursión

Factorial de un número

Caso base

X!

$$X! = \begin{cases} 1 & \text{si } X <= 1 \\ X * (X-1)! & \text{si } X > 1 \end{cases}$$

Recursión

Ejemplo:

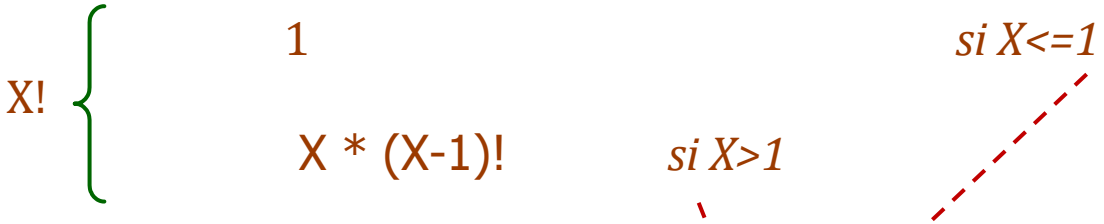
$4! = 4 * 3!$

$= 4 * 3 * 2!$ (3!)

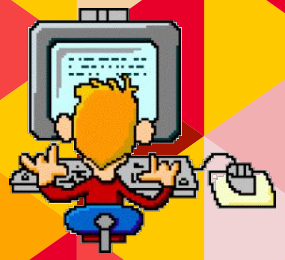
$= 4 * 3 * 2 * 1!$ (2!)

$= 4 * 3 * 2 * 1 = 24$ (1!)

Ejemplo de Recursión




```
Function factorial(x: integer): real;
begin
  if (x <= 1) then
    factorial:= 1
  else
    factorial := x * factorial(x-1)
end;
```



```
Function factorial(x: integer): real;
begin
  if (x <= 1) then
    factorial:= 1;
  else
    factorial := x * factorial(x-1)
end;
```

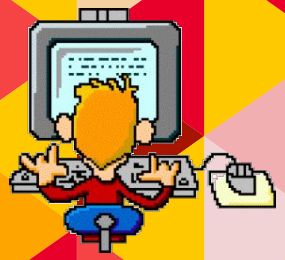
Actividad 1

- **Descargue de Ideas:** **programaCalculoDeFactorial**

a) Implementar el módulo **factorial** como parte del programa **CalculoDeFactorial**

b) Completar el programa **CalculoDeFactorial** para que lea un valor X, invoque a la función **factorial** para calcular X! y muestre el resultado.

c) Compilar y ejecutar



Actividad 2

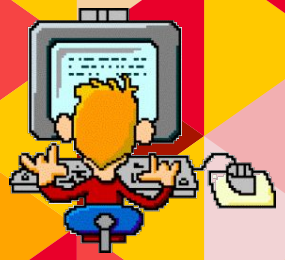
Descargue de Ideas: **programaCalculoDePotencia**

a) Implementar en el programa CalculoDePotencia, la función **potencia1.**

```
Function potencia1 (x,n: integer): real;
begin
  potencia1 := x * potencia1(x,n-1)
end;
```

a) Invocar a la función **potencia1** para calcular 5^3 .

b) Compilar y ejecutar. ¿Qué ocurre? ¿Por qué?



Actividad 3

a) Implementar en el programa CalculoDePotencia, la función **potencia2.**

```
Function potencia2 (x,n: integer):
real;
begin
  if (n = 0) then
    potencia2 := 1;
  else
    potencia2 := x * potencia2(x,n)
end;
```

a) Invocar a la función **potencia2** para calcular 5^3 .

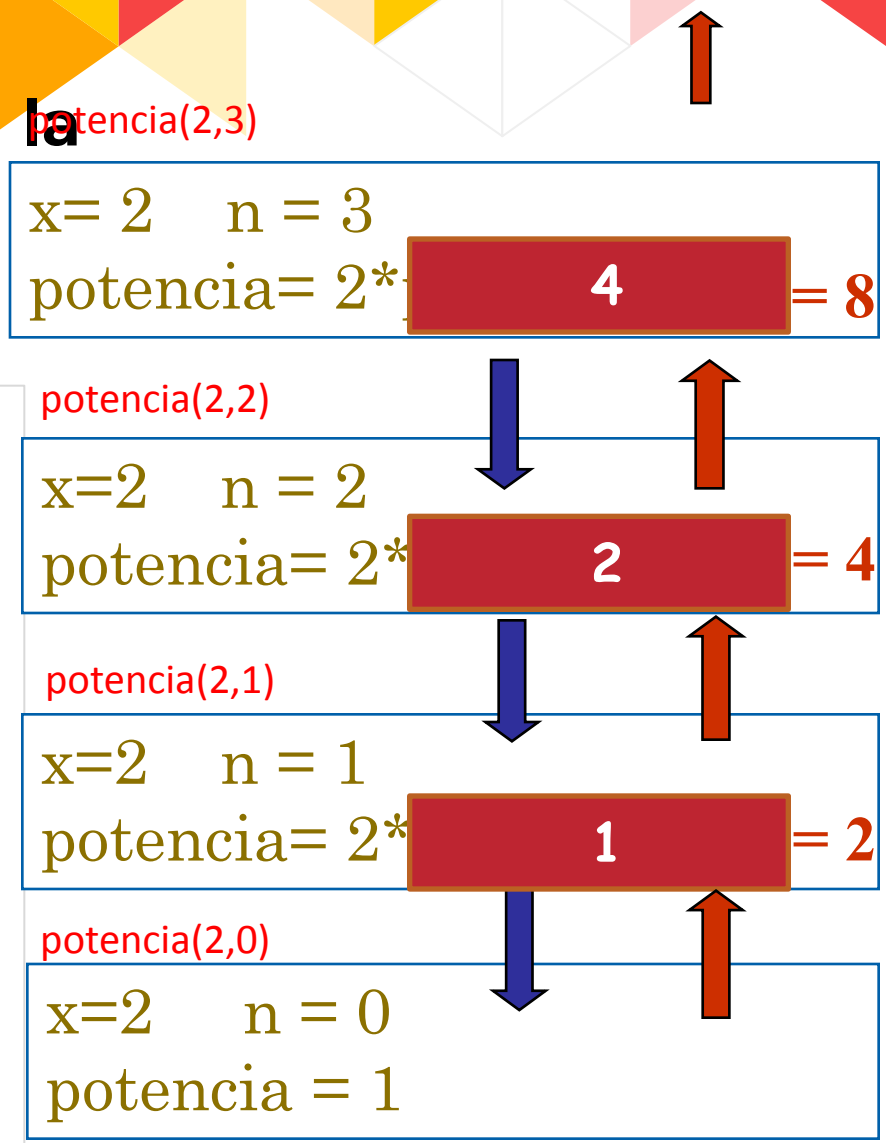
b) Compilar y ejecutar. ¿Qué ocurre? ¿Por qué?

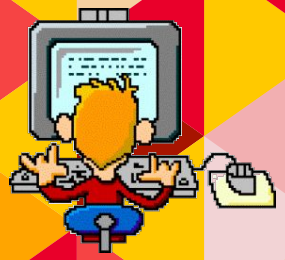
¿Cómo funciona la pila de activación?

```
Program ejemplo

Function potencia (x,n:integer): real;
begin
  if (n = 0) then
    potencia:= 1;
  else
    potencia := x * potencia(x,n-1)
end;

Begin
  read (x,n);
  write(potencia(x,n))
End.
```





Actividad 4

- **Descargar de Ideas:** `programaRecursion`

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Compilar y ejecutar.

b) Responder:

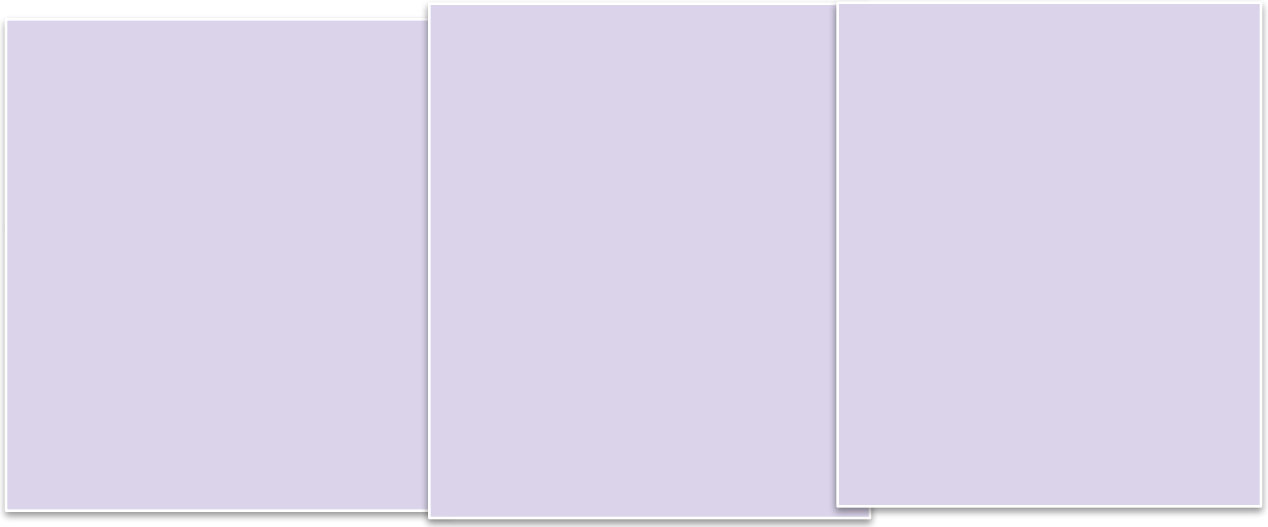
- ¿Cuál es el caso base en el procedimiento `digitoMaximo`?
- ¿Cómo se acerca al caso base?

¿Cómo funciona el llamado?

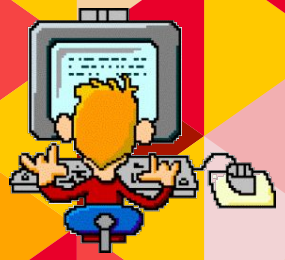
Prog. ppal

Max = **3**

digitoMaximo(132, max)



```
procedure digitoMaximo(n: integer; var max: integer);
var
  dig: integer;
begin
  dig:= n mod 10;
  if ( dig > max ) then
    max:= dig;
  n:= n div 10;
  if (n <> 0) then
    digitoMaximo(n, max);
end;
```



Actividad 5

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Modificar el procedimiento digitoMaximo. Debe colocarse la instrucción **`writeln ('max: ', max);`** como última instrucción del procedimiento.

b) Compilar y ejecutar.

c) Responder:

- ¿Qué valor se muestra antes de finalizar cada módulo?
- ¿Qué valor se muestra en el programa principal?

¿Cómo funciona?

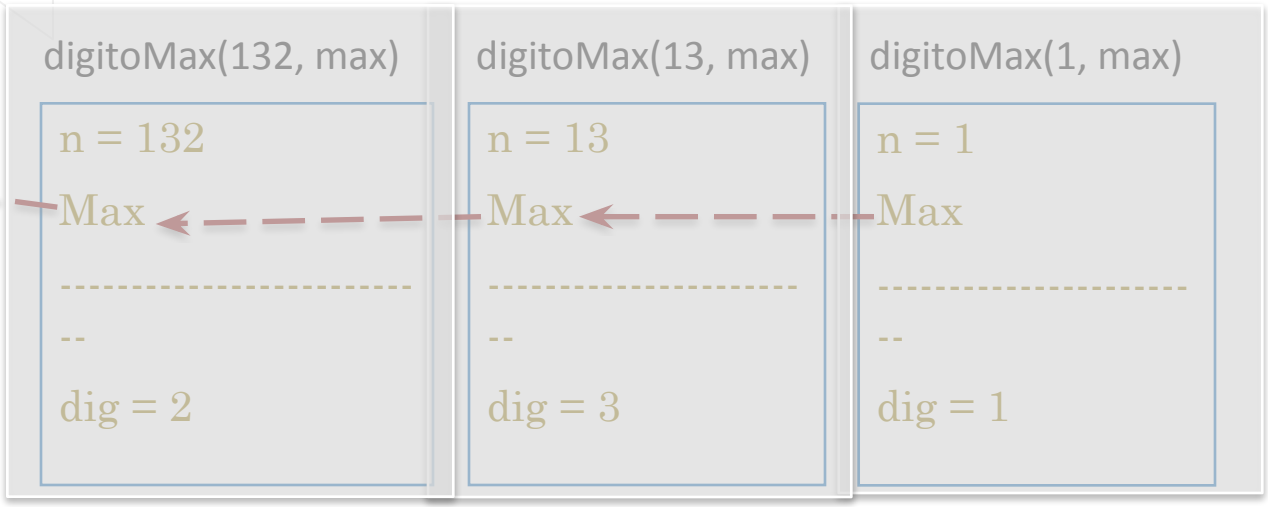
Prog. ppal

Max = **3**

digitoMaximo(132, max)

Write (max)

Imprime max: 3

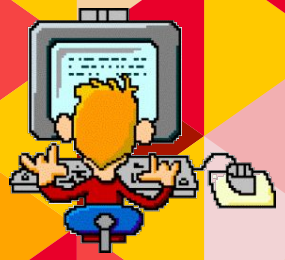


Imprime max: 3

Imprime max: 3

Imprime max: 3

```
procedure digitoMaximo(n: integer; var max: integer);
var
  dig: integer;
begin
  dig:= n mod 10;
  if ( dig > max ) then
    max:= dig;
  n:= n div 10;
  if (n <> 0) then
    digitoMaximo(n, max);
  writeln ('max: ', max);
```



Actividad 6

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Modificar el procedimiento `digitoMaximo`. Debe pasarse el parámetro `max` por valor.

b) Compilar y ejecutar.

c) Responder:

- ¿Qué valor se muestra antes de finalizar cada módulo?
- ¿Qué valor se muestra en el programa principal?

¿Cómo funciona?

Prog. ppal

```
Max = -1
digitoMaximo(132, max)
Write (max)
```

Imprime
max: -1

digitoMax(132, max)

```
n = 132
Max = 2
dig = 2
digitoMax(13, max)
```

Imprime
max: 2

digitoMax(13, max)

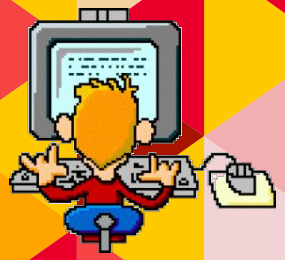
```
n = 13
Max = 3
dig = 3
digitoMax(1,max)
```

Imprime
max: 3

digitoMax(1, max)

```
n = 1
Max = 3
dig = 1
```

Imprime
max: 3



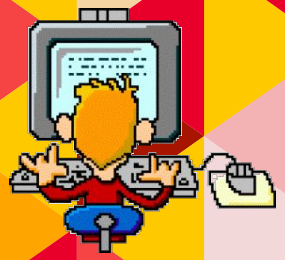
Actividad 7

Utilizando **`ProgramaRecursion`** realice las siguientes actividades:

a) Escribir el procedimiento `digitoMaximo` como una función.

b) En el programa, leer un número, invocar a la función y mostrar el resultado.

c) Compilar y ejecutar.



Actividad 8

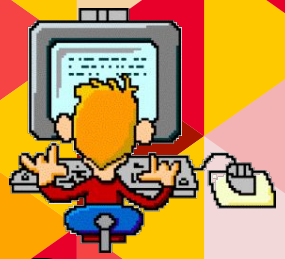
- **Descargar de Ideas:** **`programaVectorOrdenado`**

Utilizando **`programaVectorOrdenado`** realice las siguientes actividades:

a) Compilar y ejecutar.

b) Implementar el método de búsqueda dicotómica.

c) Utilizar el método implementado para buscar un valor que se lee de teclado y mostrar el resultado.



Actividad 9

Descargar de Ideas: `programaVectores`

Utilizando `programaVectores` realice las siguientes actividades

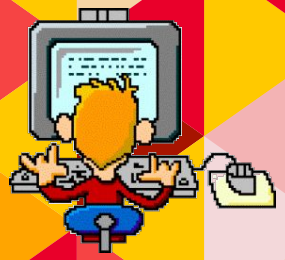
a) Compilar y ejecutar

b) Implementar un módulo recursivo **`Máximo`** que devuelva el máximo valor del vector.

c) Implementar un módulo recursivo **`Suma`** que devuelva la suma de los valores contenidos en el vector

d) Utilizar los módulos implementados para mostrar el máximo y la suma.

e) Compilar y ejecutar



Actividad 10

Descargar de Ideas: `programaLista`

Utilizando `programaLista` realice las siguientes actividades:

a) Compilar y ejecutar

b) Implementar un módulo recursivo **`Mínimo`** que devuelva el mínimo valor de la lista.

c) Implementar un módulo recursivo **`Imprimir`** que imprima los valores contenidos en la lista.

d) Utilizar los módulos implementados para mostrar el mínimo y la impresión.

e) Compilar y ejecutar