PETUNJUK PRAKTIKUM DASAR PEMROGRAMAN DENGAN CODE BLOCKS

Disusun oleh:

Alfian Ma'arif, S. T., M. Eng.

LABORATORIUM KOMPUTER

PRAGRAM STUDI TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS AHMAD DAHLAN

2020

DAFTAR ISI

PRAKTIKUM I

PENGANTAR BAHASA C

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan perangkat lunak Code Blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan Praktikum I ini, mahasiswa mampu:

1. Menuliskan *source code* menggunakan editor teks pada Code Blocks
2. Menjelaskan struktur bahasa C
3. Menjelaskan kegunaan direktif **#include**
4. Menjelaskan pustaka/library dasar yang banyak digunakan dalam program-program sederhana menggunakan bahasa C
5. Menyertakan komentar di dalam programnya
6. Menyimpan file program dengan ekstensi **.c**
7. Melakukan proses compile dan eksekusi program
8. Membuat program sederhana menggunakan bahasa C
9. Membedakan fungsi **main ( )** dengan **void main ( )**
10. Menggunakan kode *escape* pada C
11. Membedakan fungsi perinah **puts** dan **printf**

C. DASAR TEORI

1. Bentuk Umum Program

   Secara umum bentuk/struktur program C adalah sebagai berikut:

```
1: #include <file_header>
2: [deklarasi obyek]
3: [void] main ( )
4: {
5:     [deklarasi variabel]
6:             [pernyataan;]
7:       … // isi program
8:     [return nilai;]
9: }
```

Keterangan program:

Baris 1: # include … merupakan pengarah preprosesor untuk memanggil *file header* yang berisi obyek bawaan dari C yang digunakan dalam program.

Baris 2: Tempat untuk mendeklarasikan obyek (fungsi dan data) global.

Baris 3: Merupakan program utama yang berupa fungsi **main ( ),** satu-satunya fungsi yang harus ada dalam program C, *keyword* **void** menunjukkan bahwa fungsi **main** tidak mengembalikan apa-apa (tidak ada *return value*).

Baris 4: {… Adalah awal blok (bisa awal blok program/fungsi atau awal blok pernyataan majemuk).

Baris 5: Deklarasi variabel local, dalam C diperkenankan untuk mendeklarasikan variabel local di sebarang baris, namun disarankan agar lebih mudah dipahami sebaiknya pendeklarasian variabel diletakkan di bawah nama fungsi.

Baris 6,7: Merupakan isi dari program, yang terdiri dari pernyataan-pernyataan C.

Tanda ; (titik koma = semicolon) digunakan untuk mengakhiri sebuah pernyataan. Setiap pernyataan harus diakhiri dengan sebuah tanda semicolon.

Tanda // (*double slash*) merupakan tanda penulisan komentar, selain itu C juga masih mengenal tanda komentar /*……*/

Komentar dipakai untuk memberikan penjelasan kepada pembaca kode, bias berupa nama pembuat kode, tanggal pembuatan, fungdi perintah atau bagian tertentu pada kode.

Baris 8: merupakan nilai pengembalian (*return value*) terhadap fungsi.

Baris 9: …} akhir blok program

Pernyataan-pernyataan dalam C bersifat *case sensitive*, artinya peka/membedakan huruf kecil dan besar, variabel a dengan variabel A adalah dua hal yang berbeda.

Contoh program sangat sederhana dalam C:

```
#include <stdio.h>

void main( )

{ // blok awal program

  puts("Saya sedang belajar Bahasa C ") ;

} // akhir program
```

Menyatakan bahwa fungsi main tidak memiliki nilai kembalian (return value) fungsi yang akan dieksekusi pertama kali oleh C menyatakan bahwa fungsi main tidak memiliki argumen

Setelah program disimpan dalam file dengan ekstensi .cpp, kemudian program dikompile dan dieksekusi, maka hasilnya berupa tulisan pada layar terminal:

Saya sedang belajar Bahasa C

Program C memang tidak pernah lepas dari fungsi. Sebuah program C minimal mengandung sebuah fungsi, yaitu fungsi **main( )**. Fungsi ini menjadi awal dan akhir eksekusi program C.

Instruksi yang diawali dengan symbol # adalah sebuah direktif, bukan instruksi yang akan dijalankan pada saat program yang dibuat dieksekusi, tetapi merupakan perintah pada compiler pada saat mengkompile.

Direktif **# include** digunakan untuk menambahkan pustaka (library). File *.h merupakan *file header* yang berisi definisi variabel, konstanta, dan fungsi untuk keperluan tertentu. Beberapa **file header** yang sering digunakan adalah:

- stdio.h : pustaka standar yang berhubungan input/output
- conio.h : pustaka operasi konsol (layar monitor dan keyboard)
- math.h : pustaka operasi matematis
- string.h : pustaka operasi string
- iostream.h : pustaka operasi stream

#include harus dituliskan sebelum variabel atau konstanta yang dikandungnya digunakan dalam program. Direktif ini biasanya diletakkan di bagian awal program.

Semua program C memiliki sebuah fungsi utama yang akan dijalankan pertama kali ketika program tersebut dieksekusi, yaitu fungsi **main.**

Kode *escape* adalah kode karakter yang penulisannya diawali dengan symbol \ (*back slash*).

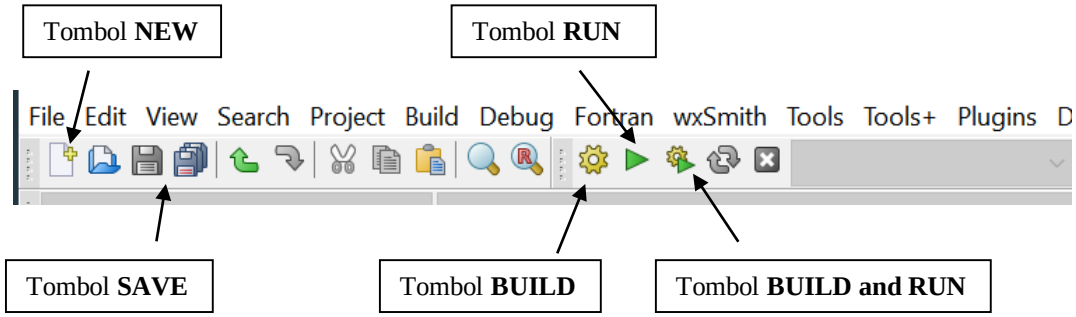
D. TUGAS PENDAHULUAN

1. Jelaskan apa yang dimaksud *case sensitive.*
2. Bagaimana cara menambahkan komentar dalam program C ?
3. Direktif **#include** digunakan untuk apa?
4. Apa yang dimaksud dengan pustaka/*library*?
5. Sebutkan pustaka/*library* yang sering digunakan dan beri penjelasan.

6. Semua program C memiliki sebuah fungsi utama yang akan dijalankan pertama kali ketika program tersebut dieksekusi, yaitu…..
7. Apa beda fungsi **main()** dengan **void main()** ?

LANGKAH PERCOBAAN

1. Jalankan Code Blocks akan terlihat tampilan awal sebagai berikut:



2. Klik Tombol **New**, pilih empty file atau dari menu **File – New – Empty File.**
3. Simpan file dengan menu: **file-save** atau tekan tombol **SAVE** (atau pilih menu **File-Save As**… untuk mengganti nama file) dengan nama **program1_1.c**
4. Ketikkan listing program pada editor teks.
5. Kompile dan jalankan program sekaligus dengan memilih menu: Build – **Build and Run** (Tekan tombol F9 atau tekan tombol **Build and Run**)
6. Jika ada kesalahan muncul jendela message, perbaiki program sesuai dengan peran kesalahan yang muncul.

Ketiklah listing program berikut ini, kemudian jalankan.

Program1_1.c

Program1_1.c

main( )
{
puts(“Hallo kawan!“)
}

Akan muncul *Error message*. Apa saja?

Tambahkan baris berikut pada bagian paling atas

```
#include <stdio.h>
```

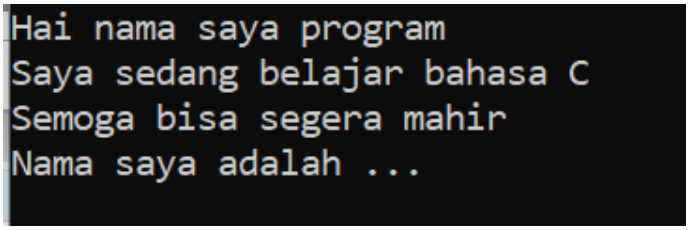
Sebelum tanda } tambahkan:

```
return 0;
```

Perbaiki pula *Error* yang lain jika ada.

Program1_2.c

Buatlah program menggunakan **puts** untuk menampilkan tulisan berikut di layar. (isi titik-titik dengan nama anda). Hasil eksekusi program adalah sbb:



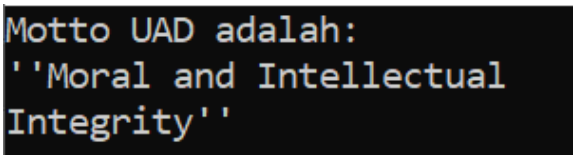
Tuliskan listing **Program1_2.c**

Program1_3.c

Gantilah fungsi **puts** pada program sebelumnya menjadi **printf** (tanpa mengubah parameternya). Apa yang terjadi? Apa perbedaan antara **puts** dan **printf**?

Program1_4.c

Buatlah program menggunakan fungsi **puts** dan **printf** untuk menampilkan tulisan berikut di layar. Hasil eksekusi program adalah sebagai berikut:



Tuliskan listing **Program1_4.c**

E. TUGAS

Perhatikan hasil eksekusi ketiga program berikut. Bagaimana efek karakter *escape* **/t**, **/n**, **/r** dan **/a**? Apa Fungsi perintah **/t**, **/n**, **/r** dan **/a**?

Program menggunakan kode *escape*.

Program1_5_1.c

```
#include <stdio.h>
void main()
{
  printf (“1 \t2 \t3 \t4 \t5 \t6 \t7 \t8 \n”) ;
  printf (“Program \tKomputer \tBahasa \tC++”) ;
  printf (“\t di Lab \t ini \n”) ;
  printf (“Saya sedang mempelajari \r”) ;
}
```

Program1_5_2.c

```
#include <stdio.h>
void main()
{
  printf (“kode escape \n”) ;
}
```

Program1_5_3.c

```
#include <stdio.h>
void main()
{
  puts(“suara apakah ini? \a”) ;
}
```

PRAKTIKUM II

VARIABEL, OPERATOR, dan TIPE DATA

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu Code Blocks.

B. TUJUAN

Setelah menyelesaikan Praktikum II ini, mahasiswa mampu:

1. Menjelaskan definisi dan aturan variabel
2. Menggunakan variabel dalam membuat program
3. Menerapkan keyword gets
4. Menerapkan keyword scanf
5. Menerapkan berbagai macam operator
6. Menjelaskan tipe data
7. Menggunakan berbagai tipe data dalam membuat program
8. Menjelaskan kode-kode penentu format

C. DASAR TEORI VARIABEL

VARIABEL

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variabel bisa diubah-ubah sesuai kebutuhan. Nama dari suatu variabel dapat ditentukan sendiri oleh pemrograman dengan aturan sebagai berikut:

- Terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Karena bersifat case-sensitive sehingga antara nim, NIM dan Nim dianggap berbeda
- Tidak boleh mengandung spasi

- Tidak boleh mengandung simbol-simbol khusus, kecuali garis bawah (underscore).

  Yang termasuk simbol khusus yang tidak di perbolehkan antara lain:

  $,?,%,#,!,%,*,(,),-,+,= , dsb

- Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai

Contoh penamaan variabel yang benar:

NIM , a, x, nama_mhs, f3098, f4, nilai, budi, dsb

Contoh penamaan variabel yang salah:

%nilai_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!, dsb

OPERATOR ARITMATIKA

C++ menyediakan lima operator aritmetika, yaitu:

*	untuk perkalian
/	untuk pembagian
%	untuk sisa pembagian (modulus)
+	untuk pertambahan
-	Untuk pengurangan

OPERATOR PEMBANDING

<	Kurang dari
<=	Kurang dari sama dengan
>	Lebih dari
>=	Lebih dari sama dengan
==	Sama dengan
!=	Tidak sama dengan

OPERATOR LOGIKA

&& Logika AND (DAN)

|| Logika OR (ATAU)

! Logika NOT (INGKARAN)

TIPE DATA

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh komputer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2 namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif

Tipe data yang banyak digunakan dalam bahasa C++ yaitu:

Tipe	**Ukuran**	**Range**	**Keterangan**
Char	1 byte	Signed: -128 ... 1127 Unsignet: 0 ... 255	Character
Short int (short)	2 byte	Signed: -32768 ... 32767 Unsignet: 0 ... 65535	Short integer
Int	4 byte	Signed: -2147483648 ... 2147483647 Unsigned: 0 ... 4294967295	Integer / bilangan bulat
Long int (long)	4 byte	Signed: -2147483648 ... 2147483647 Unsigned: 0 ... 4294967295	Log integer

Bool	1 byte	True atau false	Boolean
Float	4 byte	+3.4e-38 ... +3.4e-38 -3.4e-38 ... -3.4e-38	Float / bilanagan
Double	8 byte	1.7e - 308 ... 1.7e308 -1.7e - 308 ... -1.7e308	Pecahan presisi

KODE PENENTU FORMAT

%c : Membaca sebuah karakter

%s : Membaca sebuah string

%i,%d : Membaca sebuah bilangan bulat (integer)

%f,%e : Membaca sebuah bilangan pecahan (float)

%g : Membaca sebuah bilangan pecahan, dalam notasi %e atau %f

%o : Membaca sebuah bilangan oktal

%x : Membaca sebuah bilangan heksadesimal

%u : Membaca sebuah bilangan tak bertanda

Fungsi gets dan scanf

Kedua fungsi ini digunakan untuk mengambil input dari keyboard dan memasukkannya ke dalam suatu variabel, hanya saja parameter keduanya berbeda. Fungsi gets hanya dapat mengambil input bertipe string, sedangkan scanf dapat mengambil input bertipe string maupun yang lain, misal float dan double.

D. TUGAS PENDAHULUAN

1. Jelaskan definisi dan aturan variabel!

2. Mengapa tipe data sangat penting?
3. Sebutkan operator yang banyak dipakai dalam C++!
4. Sebutkan tipe data yang banyak dipakai dalam C++!
5. Sebutkan kode-kode penentu format!

E. LANGKAH PERCOBAAN

Program mengisi input ke variabel string

Program_2_1.c

```
#include <stdio.h>
main( )
{
        char nama[20];
        printf("Masukkan namamu: ");
        gets (nama);
        printf("Hallo ");
        puts(nama);
        printf("Betul kan, namamu %s?\n", nama);
        return 0;
}
```

(1) Apa arti angka 20 pada nama [20] di atas?

(2) Apa kegunaan fungsi **gets**?

Ganti:

```
gets (nama);
```

menjadi:

```
scanf ("%s" , nama);
```

(3) Apa kegunaan fungsi **scanf**?

Program matematika bilangan bulat & riil : pembagian

Program_2_2.c

```
#include <stdio.h>
main ( )
{
        int a = 10, c;
        float b = 3.5, d;

        c = a/b; d = a/b;
        printf (“a = %d\n”,a);
        printf (“b = %f\n”,b);
        printf (“c = %d\n”,c);
        printf (“d = %f\n”,d);
        return 0;
```

Perhatikan cara menginisialisasi nilai variabel a dan b.

(4) Berapakah nilai c dan d?

(5) Mengapa nilai keduanya berbeda meskipun operasinya sama?

Program input ke variabel bilangan : menghitung akar

Program_2_3.c

```
#include <stdio.h>
main ( )
{
        int a ;
        float b ;
        printf (“Masukkan nilai a = ”) ;
        scanf (“%d” ,a) ;
        b = sqrt (a) ;
        printf (“Akar dari a = %f ” ,b) ;
        return 0;
}
```

(6) Terjadi error waktu program di atas dikompile, mengapa?

Tambahkan **#include <math.h>** pada bagian atas, lalu compile lagi. Jika berhasil, jalankan.

(7) *Runtime-Error* apa yang muncul?

Ganti:

```
scanf (“%d”,a);
```

menjadi:

```
scanf (‘%d”,&a);
```

Parameter **scanf** adalah *pass by reference,* sehingga harus diberi alamat variabel . Notasi “&” digunakan untuk menunjukkan alamat variabel “a” atau pointer ke variabel “a”

Jalankan program yang sudah diperbaiki. Isi nilai a dengan 9, 16, 25, dll.

Coba masukkan nilai a = 100.5

(8) Bagaimana hasilnya? (9) Mengapa demikian?

Ganti:

```
scanf (“%d”,&a);
```

menjadi:

```
scanf (“%f,&a);
```

(10) Apa yang terjadi? (11) Mengapa harus %d, dan bukannya %f?

Ganti:

```
printf (“Akar dari a = %f”,b);
```

menjadi:

```
printf (“akar dari a = %d”,b);
```

(12) Apa yang terjadi? (13) Mengapa harus %f, dan bukannya %d?

Program dengan beberapa input : menghitung rerata

Program_2_4.c

```
#include <stdio.h>
main ( )
{
        float a, b, c, rerata;

        printf (“Masukkan nilai a, b, dan c = ”);
        scanf (“%f %f %f , &a, &b, &c);
        rerata = (a+b+c) \ 3;
        printf (“Rerata = %f”, rerata);
```

Jalankan program di atas dengan mengisikan tiga buah bilangan dengan diselingi spasi (contoh: 10 20 45). Coba pula dengan diselingi Enter.

Ubahlah tipe variabel a, b, dan c menjadi tipe bilangan bulat.

(14) Apalagi yang harus diubah agar tidak terjadi error?

Program menghitung invers

Program_2_5.c

```
#include <stdio.h>
main ()
{
        int a;
        float b, c;

        printf  ("Masukkan nilai a = ");
        scanf   ("%d", &a);
        b = 1/a;
        printf  ("b = 1/%d = %f\n", a, b);
        c = 1.0/a;
        printf  ("c = 1/%d = %f\n", a, c);
        return 0;
```

Compile dan jalankan program di atas.

(15) Mengapa nilai b dan c berbeda meskipun operasinya sama?

D. TUGAS

1. Tingkatkan presisi bilangan riil

Compile dan jalankan program berikut. Meskipun nilai b dan c memiliki tipe data yang berbeda, perhatikan hasil program pada terminal meskipun operasinya sama?

Program_2_6.c

```
#include <stdio.h>
main()
{
        float a=3, b;
        double c;
        b = 1000/a;
        c = 1000/a;
        printf  ("a =   %f\n", a);
        printf  ("b =   %f\n", b);
        printf  ("c =   %f\n", c);
```

2. Peraturan tampilan

Compile dan jalankan program berikut. Perhatikan tampilan nilai a dan b untuk setiap format *specifier* yang berbeda. Tuliskan secara singkat maksud dari masing-masing format tadi (%f, %7.2f, %+7.4f, %g, %e, %5.2e).

Program_2_7.c

```
#include <stdio.h>
main()
{
        double a=3.0, b;
        b = 1000.0/a;
        printf  ("a =   %f\t\t b = %f\n", a, b);
        printf ("a =    %7.2f\t\t b = %7.2f\n", a, b);
        printf  ("a =   %+7.4f\t\t b = %+7.4f\n", a, b);
        printf  ("a =   %g\t\t b = %g\n", a, b);
        printf  ("a =   %e\t b = %e\n", a, b);
        printf  ("a =   %5.2e\t\t b = %5.2e\n", a, b);
```

PRAKTIKUM III
INPUT & OUTPUT

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu Code Blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan Praktikum III ini, mahasiswa mampu:

1. Menjelaskan konsep *stream* dalam pemrograman C++.
2. Menjelaskan cara menuliskan perintah untuk menampilkan data ke layar monitor
3. Menjelaskan cara menuliskan perintah untuk mengambil masukan dari keyboard
4. Menerapkan *keyword cout*
5. Menerapkan *keyword cin*
6. Membuat program kombinasi input – output
7. Menjelaskan listing program yang telah dibuat

C. DASAR TEORI

Pustaka **iostream** adalah pustaka/library yang sering digunakan dalam pemograman C++. Program yang memakai pustaka iostream harus menyertakan (*include*) **file iostrream.h.**

Pustaka **iostream** menyediakan sejumlah operasi untuk menangani baca dan tulis tipe-tipe data baku. Pustaka **iostream** merupakan pustaka berbasis objek yang menyediakan fungsi-fungsi input dan output menggunakan stream.

Stream adalah sebuah abstraksi yang mempersentasikan media yang digunakan pada operasi-operasi input dan output. Sebuah stream pada dasarnya dapat direpresentasikan sebagai sebuah sumber atau tujuan karakter dengan panjang yang tidak terbatas. Stream, secara umum diasosiasikan dengan perangkat fisik yang menjadi sumber atau tujuan karakter-karakter tersebut, misal disk, keyboard, atau layar monitor, sehingga karakter yang diperoleh atau dituliskan dari/ke abstraksi yang disebut stream ini, secara fisik merupakan output/input dari/ke perangkat fisik. Sebagai contoh, file stream adalah objek C++ untuk memanipulasi dan berinteraksi dengan file.

Ketika *file stream* digunakan unutk membuka file, maka operasi-operasi input/output pada stream tersebut secara fisik terlihat pada file tersebut. Pada level paling bawah, suatu file diinterpretasikan sebagai suatu barisan atau stream dari bytes. Pada level ini konsep tipe data tidak ada. Sedangkan pada level user, suatu file terdiri dari suatu barisan data dari satu atau lebih tipe karakter, nilai-nilai numeric, dan obyek-obyek kelas.

Pernyataan/perintah keluaran/output adalah pernyataan yang dipakai untuk menampilkan suatu data ke perangkat output, misal layar monitor. Argument dapat berupa data string atau variabel yang sudah dideklerasikan. Perintah yang digunakan adalah **cout**. Operasi output dilakukan oleh operator *leftshift* atau operator insertion (<<). Sintaksis yang digunakan pada perintah keluaran tersebut adalah :

cout << daftar_keluaran

Pernyataan/perintah masukan/input adalah pernyataan yang dipakai untuk memasukan suatu harga ke dalam variabel tertentu dari perangkat input, misal keyboard. Perintah yang digunakan adalah **cin.** operasi input dilakukan oleh operator *rightshift* atau operator *extraction* (>>). Sintaksis yang digunakan untuk perintah masukan adalah sebagai berikut :

cin >> daftar_masukan

D. TUGAS PPENDAHULUAN

1. Jelaskan apa yang dimaksud dengan:
    a. Stream
    b. Extraction eperator
    c. Insertion operator
2. Jelaskan fungsi perintah cin dan perintah cout
3. Tuliskan sintaksis yang digunakan pada perintah masukan dan keluaran !

E. LANGKAH PERCOBAAN

Output menggunakan *stream*: Salam

Program_3_1.cpp

```
#include<iostream.h>
main()
```

```
{
cout<<”Assalamualaikum…!\nSaya    sedang    belajar
bahasa C++”
}
```

Jalankan. Ganti isi programnya menjadi:

Program _3_2.cpp

```
#include<iostream.h>
main()
{
Cout<< “Assalamualaikum…!” <<end1
<<  “Saya  sedang  belajar  bahasa
C++”;
}
```

Apakah hasilnya sama dengan program sebelumnya? (1)

Apakah fungsi dari **endl** ? (2)

Buatlah program menggunakan cout untuk menampilkan tulisan yang sama dengan **program_l_2** pada soal no (2) pada pratikum unit 1 (3). Beri nama **Program_3_3.cpp**

Input string menggunakan stream : Nama

Program_3_4.cpp

```
#include<iostream.h>
main()
{
char nama[80];

Cout<<"Masukkan nama: ";
Cin  >> nama;
Cout << "Hallo " << nama << end1
<< "Betul kan, namamu " << nama;
}
```

Bandingkan **Program_3_4** dengan program yang serupa pada Pratikum II **(program_2_1)** (4)**.** Menggantikan fungsi apakah `cin` dan `cout(5)`**.**

Program input ke variabel bilangan : Menghitung akar

Program_3_5.cpp

```
#include <iostream.h>
main()
{
int a;
float b;

cout << Masukkan nilai  a = ";
cin >> a;
b = sqrt (a);
cout << "akar dari a = " << b;
}
```

Bandingkan dengan program serupa pada Pratikum unit 1 yang mengisikan variabel dengan fungsi **scanf**. Apakah perbedaan perlakuan terhadap variabel **`a`** untuk input menggunakan **`cin`** dalam **program_3_5.cpp** dengan program menggunakan **`scanf`** (6)? Ganti tipe variabel **`a`** menjadi bilangan riil. Periksalah apakah ada bagian lain yang perlu dimodifikasi akibat pergantian tipe data tersebut ? (7)

Program dengan beberapa input : Menghitung rerata

Program_3_6.cpp

```
#include <iostream.h>
main()
{
float a, b, c, rerata;

cout << "Masukkan nilai a, b dan c = ";
cin << a << b << c;
rerata = (a+b+c) /3;
cout << "Rerata = " rerata;
}
```

Apakah kesalahan dalam program tersebut (8)? Bagaimanakah yang benar (9)?
Jelaskan program di atas dengan mengisikan tiga buah bilangan dengan diselingi spasi (contoh: 10 20 45).

Program kombinasi input-output: Data diri

Dalam membuat program, usahakan agar mudah dimengerti, yaitu dengan memberi nama variabel yang mempunyai arti, memberi keterangan, dan membuat program terstruktur dengan baik.

F. TUGAS

Buatlah program untuk mengisikan data diri melalui keyboard dan menampilkan di layar berikut ini; kemudian lakukan modifikasi agar tampilan data tampak lebih rapi. Beri penjelasan untuk tiap baris instruksi.

Program_3_7.cpp

#include <iostream.h>	(1)
main()	(2)
{	(3)
CharNama[50],Alamat	(4)
[40],progStudi[15];	(5)
int angkatan;	(6)
// Mengisikan data	(7)
cout<<Nama : ";	(8)
cin>>Nama;	(9)
cout<<"Alamat: ";	(10)
cin>>Alamat;	(11)
cout<<"program studi : ";	(12)
cin>>progStudi;	(13)
cout<<"Angkatan : ";	(14)
cin>>angkatan;	(15)
// Menampilkan data	(16)
cout<<"\nData anda adalah	(17)
:\n";	(18)
cout<<"Nama: "<<Nama;	(19)
cout<<"Alamat: "<<Alamat;	(20)
cout<<"program Studi:	(21)
"<<progStudi;	(22)
cout<<"Angkatan:	(23)
"<<angkatan;	(24)
}	(25)

PRAKTIKUM IV
PERCABANGAN (1)

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep bahasa pemrograman dengan alat bantu Code Blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan Pratikum IV ini, mahasiswa mampu:

1. Menyebutkan dua perintah dasar struktur kondisional dalam C++
2. Menjelaskan struktur sintaksis **if, if-else, if-else if**
3. Menyusun program menggunakan struktur kondisional **if, if-else, if-else if.**
4. Menyusun *flowchart* untuk struktur kondisional

C. DASAR TEORI

Dalam bahasa C++, struktur kondisional didukung oleh dua perintah dasar yaitu : **if** dan **switch**. Bentuk perintah **if** mempunyai beberapa variasi sebagai berikut :

a. Bentuk if pertama

Struktur sintaksis **if** pertama mempunyai bentuk sebagai berikut :

```
if (kondisi)
{
Pernyataan
}
```

Bentuk pertama ini dibaca

JIKA kondisi benar, Maka kerjakan pernyataan.

Bentuk pertama ini merupakan bentuk perintah **if** yang paling sederhana. Dalam bentuk ini, pernyataan akan dikerjakan jika kondisi bernilai benar. Nilai benar dalam bahasa C++ dinyatakan dengan nilai 1 (satu) sedang nilai yang salah dinyatakan dengan nilai 0 (nol).

b. Bentuk if kedua

Struktur sintaksis **if** kedua mempunyai bentuk sebagai berikut :

```
if (kondisi)
{
Pernyataan
}
else
{
Pernyataan
lain
}
```

Bentuk kedua ini dibaca :

JIKA kondisi benar, MAKA kerjakan pernyataan
JIKA TIDAK, kerjakan pernyataan lain.

Dalam bentuk kedua, jika kondisi bernilai benar, maka mengerjakan pernyataan. Tetapi jika kondisi bernilai salah, maka yang dikerjakan pernyataan lainnya.

c. Bentuk if ketiga (bertumpuk)

Struktur sintaksis **if** ketiga yaitu **if** bertumpuk mempunyai bentuk sebagai berikut :

```
if (kondisi 1)
{
pernyataan 1;
}
else if (kodisi 2)
{
Pernyataan 2
}
...
Else
{
Pernyataan lain ;
}
```

Bentuk ketiga **if** tersebut dibaca :

JIKA kondisi 1 benar, MAKA kerjakan pernyataan 1.
JIKA TIDAK, kerjakan pernyataan lain.
JIKA kondisi 2 benar, MAKA kerjakan pernyataan 2.
dst . . .

Bentuk ketiga ini merupakan perluasan bentuk kedua **if** yang mana dalam pernyataan **else** terdapat **if** lagi. Bentuk ini digunakan untuk menyatakan pilihan yang lebih dari dua. Dalam bentuk ini terdapat beberapa kondisi yang akan diuji oleh perintah **if**. Jika salah kondisi benar, maka pernyataan yang bersesuaian dengan kondisi tersebut akan dikerjakan. Jika seluruh kondisi tidak ada yang benar, maka akan dikerjakan pernyataan yang lain.

D. TUGAS PENDAHULUAN

1. Dua perintah dasar struktur kondisional dalam C++ adalah....dan....
2. Jelaskan struktur sintaksis **if, if-else, if-else if**
3. Buat program sederhana yang menggunakan struktur kondisional
   a. **if**
   b. **if-else**
   c. **if-else if**

E. LANGKAH PERCOBAAN

Percabangan tunggal

Program_4_1.cpp

```
#include <iostream.h>
using namespace std;
int main()
{
  int umur;
  cout << "masukkan umurmu = ";
  cin >> umur;
  if umur >= 60;
  cout << " Hallo Mbah…!! " << endl;
  cout << " Jadi umurmu " << umur << " tahun";
}
```

Compile program tersebut. *Error message* apakah yang muncul (1). Bagaimana yang benar? (2). Perbaiki program tersebut dan jalankan beberapa kali dengan mengisikan nilai umur yang bervariasi.

Program_4_2.cpp

Tambahkan baris berikut :

```
cout << " Salam buat cucumu ya " << endl ;
```

Setelah baris

```
  cout << " Hallo mbah…!! " << endl ;
```

Untuk memberi komentar tambahan jika umur lebih dari 60 tahun.

Jalankan dengan mengisikan umur 80 tahun, lalu jalankan lagi untuk umur 20 tahun. Mengapa komentar tambahan selalu muncul untuk umur berapapun (tidak sesuai dengan yang diinginkan)? (3), Bagaimana yang benar? (4).

Program_4_3.cpp

Buatlah program sesuai contoh berikut :

```
#include <iostream.h>
using namespace std;
int main()
{
  int tahun, umur ;
  cout << "masukkan tahun kelahiranmu = ";
  cin >> tahun;
  Umur = 2009-tahun;
  Cout << "Umurmu\t" << umur << "tahun\n";
  if (umur < 17);
    cout << "kamu belum sweet seventeen\n";
    cout << "belum cukup umur\n";
}
```

Compile program tersebut. Jalankan dan isikan tahun kelahiranmu, misal 1994, 1975, atau yang lain. (5) Kesalahan apakah yang terjadi? (6) Bagaimanakah yang benar ?

Program_4_4.cpp

Percabangan IF-ELSE

```
#include <iostream.h>
using namespace std;
int main()
{
  int N;
  cout << "masukkan nilai (0 s/d 100)= " ;
  cin >> N;
  if (N >= 50)
     cout << "lulus"
  else (N < 50)
     cout << "Tidak lulus";
}
```

(7) *Error* apakah yang muncul? (8) Bagaimanakah yang benar? Perbaiki dan jalankan dengan mengisikan beberapa nilai N.

Program_4_5.cpp

Percabangan bertingkat : IF-ELSE IF

```
#include<iostream.h>
using namespace std;
int main()
{
  int N;

  cout << "masukkan nilai= ";
  cin >> N;
  cout << "nilai huruf = ";
  if (N >= 80)
    Cout << "A";
  if (N >= 60)
    cout << "B";
  if (N >= 40)
    cout << "c";
  if (N < 40)
    cout << "D";
)
```

Jalankan dan isikan nilai 15, 25, 50, 75 dan 100. Hasilnya tidak sesuai dengan yang seharusnya. (9) Jelaskan mengapa bisa terjadi seperti itu? Perbaiki program di atas dengan struktur percabangan bertingkat : **if**. . . **else if**. . .dst.

Buat *flowchart* program tersebut (10).

Program_4_6.cpp

Kondisi kombinatorial

Perbaiki **Program_4_5.cpp** dengan kondisi yang merupakan kombinasi logika, misalnya jika N lebih dari atau sama dengan 60 tetapi kurang dari 80 maka …, tanpa menggunakan percabangan bertingkat. Buat *flowchart* dari program tersebut (11).

F. TUGAS

Menu : Program Konversi Suhu

Program_4_7.cpp

Buatlah program untuk menghitung konversi suhu dari C ke F dan sebaliknya. Program dimulai dengan mengisikan suhu yang akan dikonversi, kemudian manampilkan pilihan (menu) sebagai berikut :

```
Pilih konversi berikut :
  a. Celcius ke Fahrenheit
  b. Fahrenheit ke Celcius
```

Setelah dipilih, maka program melakukan perhitungan konversi sesuai dengan yang dipilih dan menampilkan hasilnya. Gunakan struktur percabangan **if** untuk program tersebut. Buat *flowchart*-nya terlebih dahulu, kemudian buat programnya.

PRAKTIKUM V
PERCABANGAN (2)

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu Code Blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan Praktikum V ini, mahasiswa mampu

1. Menjelaskan struktur kondisional **switch**.
2. Menjelaskan struktur sintaksis **switch**.
3. Menjelaskan fungsi perintah **break**, **label**, **goto**.
4. Menjelaskan keterbatasan struktur **switch**.
5. Menyusun program menggunakan struktur kondisional **switch**.
6. Menyusun *flowchart* untuk struktur kondisional

C. DASAR TEORI

Kasus yang terjadi pada **if** bertumpuk, kadang dapat membosankan dalam menulis program, karena menulis beberapa perintah yang hampir sama secara berulang – ulang. Hal demikian akan semakin terlihat jika ekspresi yang harus diuji oleh perintah **if** semakin banyak. Dalam kasus seperti ini C++ menyediakan perintah khusus yaitu **switch**. Perintah **switch** memungkinkan untuk melakukan sejumlah tindakan berbeda terhadap sejumlah kemungkinan nilai. Bentuk umum perintah **switch** adalah sebagai berikut.

```
switch (ekspresi)
{
case label1: pernyataan1; break;
case label1: pernyataan1; break;
…
default: pernyataan lain;
}
```

Bentuk perintah tersebut dibaca:

```
Jika ekspresi sama dengan label1, maka kerjakan pernyataan1.

Jika ekspresi sama dengan label2, maka kerjakan pernyataan2.

…

Dan seterusnya

Jika ekspresi tidak sesuai dengan semua label, maka kerjakan
pernyataan lain.
```

Jadi misalnya ekspresi sama dengan **label1**, maka pernyataan1 akan dieksekusi hingga ditemukan statemen **break**, kemudian program akan melompat ke bagian akhir struktur **switch**.

Pernyataan **switch** penggunaannya terbatas, yaitu bahwa label hanya dapat berupa konstanta bertipe **char** atau **int**.

D. TUGAS PENDAHULUAN

1. Jelaskan penggunaan struktur kondisional switch.
2. Jelaskan struktur sintaksis switch.
3. Jelaskan fungsi perintah break, label, goto.
4. Jelaskan keterbatasan struktur switch.

E. LANGKAH PERCOBAAN

1.1. Percabangan switch

1.2. Program_5_1.cpp

Jalankan program berikut dan isikan kategori A, B, C, D, atau E. Bagaimana keluarannya? **(1)**.

Buat modifikasi agar menjadi benar **(2)**.

```
#include<iostream>
using namespace std;

char kategori;
float diskon;

int main()
{
      cout << ”Kategori pelanggan (A/ B/ C/ D/ E) = ”;
      cin >> kategori;
      switch(kategori)
      {
            case ’A’:
            diskon=40;
            case ’B’:
            diskon=25;
            case ’C’:
            case ’D’:
            diskon=10;
            default:
            diskon=0;
      }
      cout << ”Diskon = ” << diskon << “%”;
}
```

1.3. Program_5_2.cpp

Jalankan program berikut dan isikan pilih = 1, 2, 3, dan 4. Bagaimana keluarannya? **(3).**

```
#include<iostream>
using namespace std;
int pilih;
int main()
{
	cout << “Pilihan” << endl;
	cout << “1. Jakarta ” << endl;
	cout << “2. Bandung ” << endl;
	cout << “3. Surabaya ” << endl;
	cout << endl;
	cout << “Pilihan = ”;
	cin >> (pilih);
	switch(pilih)
	{
		case 1:
			cout << “Pilihan ke Jakarta” << endl;
		break;
		case 2:
			cout << “Pilihan ke Bandung” << endl;
		break;
		case 3:
			cout << “Pilihan ke Surabaya” << endl;
		break;
		default:
			cout << “Pilihan Anda Salah” << endl;
		break;
	}
}
```

Modifikasilah Program_5_2.cpp sehingga akan keluar tampilan sebagai berikut **(4)**.

```
Konsentrasi yang dipilih
    1. Teknik Kendali
    2. Teknik Elektronika dan Telekomunikasi
    3. Teknik Komputer dan Teknologi Informasi

Pilihan = 1
Teknik Kendali
```

1.4. Lompatan: label dan goto

Buatlah Program_5_3.cpp, kemudian jalankan dan tuliskan tampilan yang muncul pada terminal **(5)**. Jelaskan fungsi **label** dan **goto**.

1.5. Program_5_3.cpp

```
#include<iostream>
using namespace std;

int main()
{
```

```
      cout << ”Ini langkah pertama” << endl;
      goto LABEL2;
LABEL1:
      cout << “Ini langkah kedua” << endl;
      goto LABEL3;
LABEL2:
      cout << “Ini langkah ketiga” << endl;
      goto LABEL1;
LABEL3:
      cout << “Ini langkah keempat” << endl;
}
```

F. TUGAS

Buatlah program untuk menghitung konversi suhu dari C ke F atau R dan sebaliknya. Program dimulai dengan mengisikan suhu yang akan dikonversi, kemudian menampilkan pilihan (menu) sebagai berikut.

Pilih konversi berikut

A. Celcius ke Fahrenheit
B. Fahrenheit ke Celcius
C. Celcius ke Reamur
D. Reamur ke Celcius
E. Fahrenheit ke Reamur
F. Reamur ke Fahrenheit

Setelah dipilih, maka program melakukan perhitungan konversi sesuai dengan yang dipilih dan menampilkan hasilnya. Pergunakan struktur percabangan switch. Buat flowchart terlebih dahulu, kemudian susun programnya.

PRAKTIKUM VI
PERULANGAN

G. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu code blocks.

H. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan praktikum ini, mahasiswa mampu:

1. Menjelaskan struktur perulangan **while**, **do**…**while**, **for**.
2. Menjelaskan struktur siktaksis **while**, **do**…**while**, **for**.
3. Menyusun dan menjelaskan program menggunakan struktur perulangan **while**, **do**…**while**, **for**.
4. Menyusun *flowchart*/diagram alir untuk struktur perulangan.

I. DASAR TEORI

Perulangan adalah suatu tindakan untuk melakukan hal yang serupa berkali-kali, misalnya menampilkan tulisan "Teknik Elektro UAD" seratus kali.

1.6. Perulangan Bersyarat "Periksa-Jalankan": WHILE

Struktur **while** mempunyai bentuk sebagai berikut:

```
while (kondisi)
{
  Pernyataan;
}
```

1.7. Perulangan Bersyarat "Jalankan-Periksa": DO-WHILE

Struktur perulangan **do-while** mempunyai bentuk sebagai berikut:

```
do
{
  Pernyataan;
}
while (kondisi);
```

Bentuk perintah tersebut dibaca:

Kerjakan pernyataan selama kondisi bernilai benar (true)

Dalam struktur ini pernyataan paling sedikit akan dikerjakan satu kali. Pernyataan pasti dikerjakan karena langkah pertama struktur perulangan **do-while** mengerjakan pernyataan, kemudian baru diikuti pengujian terhadap kondisi. Struktur ini sangat cocok untuk program yang tidak memerlukan pengujian terlebih dahulu sebelum mengerjakan pernyataan.

1.8. Perulangan dengan FOR

Struktur perulangan **for** mempunyai bentuk sebagai berikut:

```
for (inisialisasi; konidis; perubahan)
```

```
{
  Pernyataan;
}
```

Struktur for digunakan untuk perulangan dengan jumlah pengulangan sudah dipastikan.

Keterangan

Inisialisasi berfungsi untuk memberi nilai awal pada variabel kendali

Kondisi berfungsi untuk mengendalikan perulangan, dilanjutkan/ diakhiri

Perubahan berfungsi untuk menyatakan perubahan nilai variabel kendali yaitu penambahan atau pengurangan

J. TUGAS PENDAHULUAN

1. Bagaimana struktur sintaksis **while**, **do**…**while**, dan **for**, kemudian beri penjelasan
2. Buat *flowchart* untuk struktur perulangan **while**, **do**…**while**, dan **for**.

K. LANGKAH PERCOBAAN

1.9. Perulangan bersyarat "Periksa-Jalankan": WHILE

Jalankan Program_6_1.cpp dan beri penjelasan program baris demi baris (1-14) **(1)**.

1.10. Program_6_1.cpp

```
#include<iostream>                                  (1)
#include<conio.h>                                  (2)
using namespace std;                               (3)
char nama[80];                                     (4)
int tombol='Y';                                    (5)

int main()                                         (6)
{
  while ((tombol=='Y')||(tombol=='y'))             (7)
  {
    cout<<"Masukkan nama Anda = ";                 (8)
    cin>>nama;                                     (9)
    cout<<"Halo " << nama << endl << endl;         (10)
    cout<<"Apakah mau mengulangi (Y/T)? ";         (11)
    tombol=getch();                                (12)
    cout<<endl<<endl;                              (13)
  }
cout<<"Selesai";                                   (14)
}
```

1.11. Perulangan Bersyarat "Jalankan-Periksa": DO-WHILE

Apa fungsi perintah cacah++? **(2)**. Jalankan program_6_2.cpp dan masukkan beberapa nilai untuk dihitung reratanya, akhiri masukkan dengan memasukkan bilangan negatif. Beri penjelasan program baris demi baris **(3)**.

1.12. Program_6_2.cpp

```
#include<iostream>
using namespace std;

int cacah;
float nilai, jumlah=0, rerata

int main()
{
      cout<<” MENGHITUNG RERATA NILAI\n”;
      cout<<” Masukkan nilai” <<” (masukkan bilangan negatif untuk
mengakhiri \n\n”;
      do
      {
            cacah++;
            cout<<” Data ke-” << cacah << “ = ”;
      }
      while (nilai>=0);
      rerata = jumlah/cacah;
      cout<<” \n Banyakknya Data =” << cacah;
      cout<<” \n Jumlah =” << jumlah;
      cout<<” \n Rerata =” << rerata;
}
```

1.13. Memutus Perulangan: BREAK & CONTINUE

Jalankan program_6_3.cpp dan catat tampilan hasil eksekusi di terminal. Apa fungsi simbol //? **(4)**. Hilangkan tanda // tersebut dan jalankan. Apa fungsi perintah **break**? **(5)**. Gantilah **break** dengan **continue** dan jalankan. Apa fungsi perintah **continue**? **(6)**. Beri penjelasan program baris demi baris **(7)**.

1.14. Program_6_3.cpp

```
#include<iostream>
using namespace std;
int i=0;
int main()
{
      cout<<i;
      do{
            i++;
            cout<<” - “;
            //if (i==4)
            //break
            cout<<I;
      } while(i<10)
      cout<<”\nSelesai\n”;
}
```

1.15. FOR untuk Perulangan

Jalankan program_6_4.cpp dan perhatikan hasilnya. Ubah baris instruksi **for** berikut

```
for(i=1; i<=20; i++)
```

Menjadi

```
for(i=10; i<=20; i++)

for(i=1; i<=10; i++)

for(i=1; i<=20; i=i+2)
```

Jelaskan pengaruh yang terjadi untuk ketiga macam perubahan tersebut **(8)**.

1.16. Program_6_4.cpp

```
#include<iostream>
using namespace std;
int i=0;
int main()
{
      for(i=1; i<=20; i++)
      {
            cout<<”Kalang ke-” <<i<<endl;
      }
}
```

1.17. Perulangan FOR bertingkat: Tabel Perkalian

Jalankan program_6_5.cpp dan perhatikan hasil program di terminal. Gantilah angka 5 dalam kurung pada fungsi **setw()** menjadi 3 dan 8, kemudian perhatikan hasilnya. Apa fungsi **setw()** tersebut? **(9)**.

1.18. Program_6_5.cpp

```
#include<iostream>
#include<iomanip.h>

using namespace std;

int main()
{
      for(int i=1; i<=10; i++)
      {
            for(int j=1; j<=10; j++)
            {
                  cout<<setw(5)<<i*j;
            }
            cout<<endl;
      }
}
```

L. TUGAS

1.19. Menghitung Jumlah dan Rerata

Buatlah program untuk menghitung jumlah dan rerata nilai, tetapi dengan memasukkan dahulu berapa banyaknya nilai yang akan dimasukkan. Tampilannya adalah sebagai berikut. (Huruf tebal adalah nilai yang diisikan melalui *keyboard*).

Contoh tampilan:

1.20. Program_6_6.cpp

```
Banyaknya nilai = 3
Data ke-1 = 10
Data ke-2 = 9
Data ke-3 = 8
Jumlah = 27
Rerata = 9
```

Buatlah dalam 2 versi, pertama menggunakan struktur perulangan **WHILE** dan kedua menggunakan struktur perulangan **DO-WHILE**. Buat flowchart-nya terlebih dahulu, kemudian susun programnya.

1.21. Tabel Konversi Suhu

Dengan struktur yang serupa (tapi tak sama) memakai **DO-WHILE**, buatlah tabel konversi suhu dari Celsius ke Fahrenheit, Reamur, dan Kelvin, mulai dari suhu 0^{o}C sampai 100^{o}C dengan selisih 5^{o}C. Tampilannya kurang lebih adalah sebagai berikut:

1.22. Program_6_7.cpp

```
        Tabel Konversi Suhu
------------------------------------------------
        C       F       R       K
        0       32      0       273
        5       41      4       278
        10      50      8       283
        …
        95      203     76      368
        100     212     80      373
```

PRAKTIKUM VII
ARRAY (LARIK)

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu code blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

setelah menyelesaikan Praktikum VII ini, mahasiswa mampu :

1. Menjelaskan struktur penyimpanan **array**
2. Menjelaskan cara deklarasi **array**
3. Menjelaskan cara mengakses elemen **array**
4. Menyusun dan menjelaskan program menggunakan struktur penyimpanan **array**
5. Menjelaskan berbagai perintah untuk operasi pada variabel dengan tipe **string**

C. DASAR TEORI

Larik (array) merupakan struktur penyimpanan sekumpulan data secara beruntun didalam memori, tiap elemen data diacu menggunakan indeks. Indeks menyatakan posisi data relatif dalam kumpulannya. *Array* (larik) juga merupakan elemen data dengan tipe sama.

Contoh :

Larik **nilai suhu** dengan 7 elemen yang tiap elemen berisi data suhu dengan tipe integer.

indeks	nilai
0	32
1	31
2	30
3	32
4	33
5	32
6	31

Deklarasi array:

```
tipeData namaArray
```

Larik tersebut dapat dideklarasikan sebagai berikut:

```
int nilaiSuhu[7];
```

Artinya: array dengan nama **nilaiSuhu** memiliki 7 buah elemen dengan tipe data **int.**

Jumlah elemen larik harus sudah diketahui sebelum program dieksekusi, jumlah elemen larik tidak dapat diubah selama program dieksekusi.

Mengakses elemen array

Notasi untuk mengakses elemen array:

```
namaArray[indeks]
```

Misal: mengakses elemen larik A: A[1], A[2], A[3]

Mengakses elemen larik nilaiSuhu: **nilaiSuhu[0] = 35;**

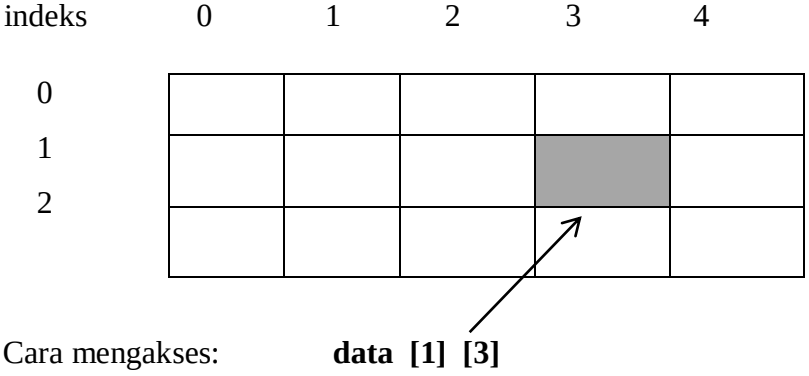
Artinya akan membuat elemen pertama (indeks 0) array **nilaiSuhu** diisi dengan nilai 35.

Array multidimensi

Array multidimensi juga disebut "arrays of arrays". Misalnya array dua dimensi bisa digambarkan sebagai tabel dua dimensi, yang berisi elemen dengan tipe data yang sama.

Deklarasi: **int data [7] [5]**

Artinya array dua dimensi dengan 3 baris dan 5 kolom.



Cara mengakses: **data [1] [3]**

Artinya mengakses elemen dengan indeks baris =1, dan indeks kolom = 3.

Literal String

String berarti deretan atau kombinasi sejumlah karakter, yang ditulis di antara tanda petik ganda (“).

Contoh:

“universitas Ahmad Dahlan”

“Jl. Prof. Dr. Soepomo”

“” ⟶ disebut string kosong

D. TUGAS PENDAHULUAN

1. Jelaskan apa yang dimaksud dengan struktur penyimpanan **array** (larik)
2. Jelaskan cara deklarasi **array**
3. Tuliskan cara deklarasi array dua dimensi 3 kolom 100 baris dengan nama **DataNilai** tipe char.
4. Jelaskan cara mengakses elemen array
5. Sebutkan beberapa perintah untuk operasi pada variabel dengan tipe string

E. LANGKAH PERCOBAAN

Array pada perhitungan nilai rerata

Ingat kembali program menghitung rerata nilai pada praktikum II. sebagai ganti menggunakan 3 buah variabel (a, b, dan c), program tersebut dapat dimodifikasi menggunakan sebuah variabel larik dengan nama **a**, seperti pada **program_7_1.cpp** Buat program tersebut, beri penjelasan skrip program baris 1-5 **(1)**, kemudian jalankan. *Error* apa yang terjadi **(2)** dan bagaimana yang benar **(3)**?

Program_7_1.cpp

```
#include <iostream>
using namespace std;
int main ()
{
     float rerata, a[3];                    (1)

     cout << "Masukkan 3 buah nilai = ";         (2)
     cin >> a[0] >> a[1] >> a [2];              (3)
     rerata = (a(0)+a(1)+a(2))/3;               (4)
     cout << "Rerata = " << rerata;         (5)
}
```

Program menghitung rerata secara umum dapat berupa program seperti berikut

Program_7_2.cpp

```
#include <iostream>
using namespace std;
int main ()
{
      int N;
      float jumlah, rerata, a[5];
      cout << "Banyaknya nilai = ";
      cin >> N;
      jumlah = 0;
      for (int i=0; i<N; i++)
      {
            cout << "Nilai ke-" << (i+1) << " = ";
            cin >> a[i];
            jumlah = jumlah+a[i];
      }
      rerata = jumlah/N;
      cout << "Rerata = " << rerata;
}
```

Beri penjelasan skrip program tersebut baris demi baris **(4)**? isikan banyaknya nilai=4 dan isikan nilai-nilainya. Program berjalan dengan lancer. Lalu coba lagi dengan banyaknya nilai = 15 dan isikan nilai nilainya. Apa yang terjaadi dan mengapa begitu, serta bagaimana memperbaikinya **(5)**?

Pengulangan FOR untuk mengisi dan menampilkan array dua dimensi (matriks)

Program_7_3.cpp adalah sebuah contoh program yang digunakan untuk mengisi nilai-nilai elemen pada matriks ukuran 3x3.

Program_7_3.cpp

```
#include <iostream>
using namespace std;
int main ()
{
      int N = 3;
      float A[5][5];
      for (int i=0; i<N; i++)
      {
            for (int j=0; j<N; j++)
            {
                  cout << "A(" << i << "," << j << ") = ";
                  cin >> A[i][j];
            }
      }
      cout << "Matriks A =\n";
      for (i=0; i<N; i++)
      {
            for (int j=0; j<N; j++)
            {
                  cout << "\t" << A [i][j];
            }
            cout << endl ;
      }
}
```

Modifikasi **Program_7_3.cpp** (dengan menambah atau mengubah sekitar 3 baris saja) program tersebut agar dapat digunakan untuk mengisi dan menampilkan matriks yang berukuran 4 baris x 5 kolom **(6)**.

F. TUGAS

Operasi pada string: Utak-atik nama

Program_7_4.cpp

```
#include <stdio.h>
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
      char nama[80], nama2[80], tulisan[80];
      int panjang, posisi;
      cout << “Masukkan nama = “;
      gets(nama);
      strcpy(tulisan,”Halo “);
      cout << strcat(tulisan,nama) << endl;

      if (strcmp(nama,”Miko”) == 0){
            cout << “Namamu Miko kan” << endl;
      }
      else
      {
            cout << “Namamu bukan Miko” << endl;
      }

      panjang = strlen(nama);
      cout << “panjang namamu “ << panjang << endl;
      posisi = strcspn(nama,”z”);

      if (posisi < panjang){
            cout << “Huruf z pada posisi “ << posisi + 1 << endl;
      }
      else
      {
            cout << “Tidak ada huruf z nya” << endl;
      }

      strcpy(nama2,nama);
      cout << strupr(nama2) << endl;
      cout << strlwr(nama2) << endl;
      cout << strrev(nama2) << endl;
      cout << strset(nama2,’x’) << endl;
}
```

Jalankan program7_4.cpp, amati dan tuliskan tampilan hasil eksekusi pada terminal ketika diberi masukan:

1. Miko **(7)**
2. Miko **(8)**
3. Ahmad Dahlan **(9)**
4. Zainal **(10)**
5. Nama masing-masing **(11)**

Ganti baris:

```
cout << strset(nama2,'x') <<
```

Menjadi:

```
cout << strset(nama2,'a') <<
```

Jalankan program, pengaruh yang terjadi apa? **(12)**

Dari pengamatan hasil eksekusi tersebut, jelaskan kegunaan fungsi-fungsi berikut ini: `Strlen(), strcpy(),strcat(), strcmp(), strlen(), strcspn(), strupr(), strlwr(), strrev(), dan strset().`

PRAKTIKUM VIII

FUNGSI

A. KOMPETENSI DASAR

Setelah menyelesaikan praktikum ini, mahasiswa dapat memahami implementasi konsep dasar pemrograman dengan alat bantu Code Blocks.

B. INDIKATOR PENCAPAIAN KOMPETENSI

Setelah menyelesaikan Praktikum ini, mahasiswa mampu:

1. Menjelaskan konsep **fungsi**
2. Menjelaskan struktur **fungsi** dan bagian – bagiannya.
3. Menjelaskan cara memanggil **fungsi** dalam **fungsi main ( )**
4. Menyusun dan menjelaskan program menggunakan struktur **fungsi**
5. Menjelaskan fungsi tanpa nilai balik (**tipe void**)

C. DASAR TEORI

Sebuah fungsi pada C++ berisi sejumlah pernyataan C++ yang dikemas dengan sebuah nama. Selanjutnya nama ini dapat dipanggil beberapa kali dalam suatu kode C++. Salah satu fungsi yang sering dilibatkan dalam aplikasi berbasis konsol yaitu **main ( ).** Biasanya di dalam fungsi tersebut dituliskan sejumlah pernyataan.

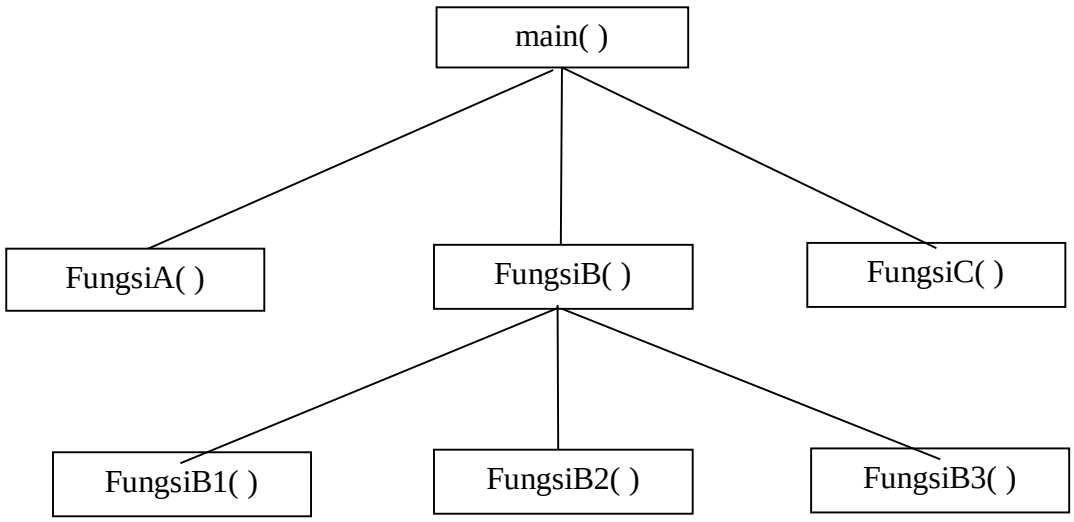
Tujuan utama pembuatan fungsi adalah untuk membuat suatu aplikasi agar dapat dipecah menjadi sejumlah bagian yang dapat dikelola dengan lebih mudah oleh pemrogram dari pada kalau aplikasi hanya mengandung sebuah fungsi, misalnya **fungsiA ( ), fungsiB ( ),** dan **fungsiC ( ).** Kemudian di dalam fungsi **main ( ),** terdapat pemanggilan ketiga fungsi tersebut sebagai berikut:

```
int main()
{
      fungsiA();
      fungsiB();
      fungsiC();
      return 0;
```

```
}
```

Dalam implementasinya, pemrogram dapat berkonsentrasi untuk menuliskan **fungsiA ( )** secara detail terlebih dahulu, kemudian ke fungsi – fungsi berikutnya sehingga seluruh kode dituliskan.

Dalam praktiknya fungsiA( ) juga bisa tersusun atas sejumlah fungsi. Berikut adalah contoh kerangka pemakaian fungsi dalam sebuah aplikasi.



Struktur fungsi

```
tipe_nilai_balik nama_fungsi (tipe_parameterA, tipe_parameterB, . . .)
{
        pernyataan_1;
        . . .
        pernyataan_n;
        return nilai_balik;
}
```

Penjelasan:

tipe_nilai_balik nama_fungsi (tipe_parameterA, tipe_parameterB, . . .) disebut judul fungsi, yang terdiri dari tiga bagian:

1. tipe_nilai_balik : menetukan tipe nilai yang diberikan oleh fungsi ketika fungsi dipanggil. Nilai balik ditentukan melalui pernyataan return.
2. nama_fungsi
3. parameter : digunakan untuk melewatkan nilai ke fungsi.

Antar parameter dipisahkan oleh tanda koma (,). Jika tak ada parameter, judul fungsi berupa:

```
tipe_nilai_balik nama_fungsi()
```

dalam definisi fungsi, tanda titik koma sesudah tanda } tidak diperlukan.

Pada bagian pernyataan fungsi terdapat pernyataan return yang digunakan untuk mengakhiri eksekusi fungsi dan memberikan nilai balik, yaitu nilai yang diberikan oleh fungsi ketika fungsi dipanggil.

Contoh:

```
Long kuadrat(long x)
{
       long hasil = x * x;
       return hasil;
}
```

Artinya didefinisikan fungsi dengan nama kuadrat ( ) yang digunakan untuk menghitung nilai kuadrat dari sebuah bilangan. Nama parameter adalah x, dengan tipe long dan memberikan nilai balik berupa perkalian x dengan x dan bertipe long.

Fungsi tanpa nilai balik

Fungsi dengan tipe void berarti tidak memiliki nilai balik, sehingga tidak dapat berkedudukan sebagai ekspresi.

Contoh:

```
Void info ( )
{
      cout << “Tidak ada nilai balik”;
}
```

Bisa ditambahkan pernyataan return:

```
Void info ( )
{
      cout << “Tidak ada nilai balik”;
      return
}
```

Cara pemanggilan fungsi di atas adalah **Info ( ) ;**

D. TUGAS PENDAHULUAN

1. Jelaskan apa yang dimaksud dengan fungsi
2. Jelaskan cara deklarasi fungsi dan jelaskan juga bagian-bagiannya
3. Jelaskan cara memanggil fungsi dalam fungsi main()

4. Jelaskan apa yang dimaksud dengan fungsi tipe void dan beri contohnya

E. LANGKAH PERCOBAAN

Fungsi pada perhitungan nilai rerata

Program menghitung rerata yang telah pernah dipelajari pada praktikum sebelumnya, dapat dibuat menjadi beberapa fungsi. Jalankan program_8_1.cpp dan berikan penjelasan baris demi baris **(1)**.

Program_8_1.cpp

```
#include <iostream>
using namespace std;
int N;
float jumlah, rerata, data[100];

void Masukan_Data()
{
      cout << “Banyaknya nilai = “;
      cin >> N;
      for (int i=0; i<N; i++)
      {
            cout << “Nilai ke-” << (i+1) << “ = “;
            cin >> data[i];
      }
}

void Hitung_Rerata()
{
      jumlah = 0;
      for (int i=0; i<N; i++)
      {
            jumlah = jumlah+data[i];
      }
      rerata = jumlah/N;
}

void Tampilkan_Hasil()
{
      cout << “Jumlah = “ << jumlah << endl;
```

```
      cout << “Rerata = “ << rerata << endl;
}

int main()
{
      Masukan_Data();
      Hitung_Rerata();
      Tampilkan_Hasil();
}
```

Jika diketahui rumus untuk menampilkan standar deviasi adalah sebagai berikut:

$$\alpha^2 = \frac{\sum(X-u)^2}{N}$$

$$u = \frac{\sum X}{N}$$

Tambahkan sebuah fungsi untuk perhitungan setandar deviasi, serta tambahkan perintah untuk menampilkan hasil perhitungan tersebut pada fungsi void **tampilkan_hasil ( ) (2)**.

Fungsi untuk mengisi dan menampilkan matriks

Program_8_2.cpp

Program_8_2.cpp sama dengan program Praktikum VII, namun instruksi untuk mengisi dan menampilkan matriks dijadikan fungsi yang dipanggil pada main program. Jalankan program berikut, beri penjelasan baris demi baris **(3)**.

```
#include <iostream>
using namespace std;
int N = 2;

void MengisiMatriks(float X[5][5]))
{
      for (int i=0; i<N; i++)
      {
            for (int j=0; j<N; j++)
            {
                cout << “elemen” << i << “,” << j << “ =
“;
```

```
                cin >> X[i][j];
            }
      }
}

void MenampilkanMatriks(float X[5][5])
{
      for (int i=0; i<N; i++)
      {
            for (int j=0; j<N; j++)
            {
                  cout << "\t" << X[i][j];
            }
            cout << endl;
      }
}

int main()
{
      float A[5][5];
      cout << "Masukan elemen matriks A\n";
      MengisiMatriks(A);
      cout << "Matriks A = \n";
      MenampilkanMatriks(A);
}
```

Selanjutnya, modifikasi program_8_2.cpp untuk dapat mengisi dan menampilkan matriks B dengan ukuran yang sama, serta matriks C yang merupakan penjumlahan antara A dan B **(4)**.

Program Konversi Suhu

Program_8_3.cpp

Program_8_3.cpp adalah program untuk mengonversi suhu dalam Celsius ke Fahrenheit menggunakan fungsi.

```
#include <iostream.h>
using namespace std;
```

```
float Suhu_C, Suhu_F;            //definisikan variabel

float C_ke_F(float C)
{
      float F;
      F = 1.8*C+32.0;
      return F;
}

void Mengisi_Input()
{
      cout << “Isikan nilai Suhu C = “; // tampilkan
tulisan
      cin >> Suhu_C;                //isikan nilai Suhu C
}
void Mengkonversi()
{
      Suhu_F = C_ke_F(Suhu_C);   //hitung nilai Suhu F
}

void Menampilkan_Hasil()
{
      cout << “Temperatur” << Suhu_C << “C = “ << Suhu_F
<< “F” << endl;
}
int main()
{
      Mengisi_Input();
      Mengkonversi();
      Menampilkan_Hasil();
}
```

Jalankan program Program_8_3.cpp dan lihat hasilnya. Pindahkan keempat fungsi yang ada ke bagian bawah setelah akhir dari program utama main ( ) lalu *compile. Error* apa yang muncul **(5)**? Di atas fungsi main ( ) tambah pendefinisian keempat fungsi tadi seperti berikut:

```
float C_ke_f(float C);
void Mengisi_Input ();
void Mengkonversi ();
Void Menampilkan_hasil ();
```

Jalankan. Apa guna pendefinisian fungsi tersebut **(6)**.

F. TUGAS

Buatlah **Program_8_4.cpp** untuk menghitung konversi suhu dari C ke F atau R dan sebaliknya. Program dimulai dengan menampilkan pilihan (menu) sebagai berikut:

```
Pilih konversi berikut:
  A. Celcius ke Fahrenheit
  B. Fahrenheit ke Celcius
  C. Celcius ke Reamur
  D. Reamur ke Celcius
  E. Fahrenheit ke Reamur
  F. Reamur ke Fahrenheit
```

Setelah menu dipilih, isikan suhu yang akan dikonversi, kemudian program melakukan perhitungan konversi sesuai menu yang dipilih dan menampilkan hasilnya. Gunakan struktur pemrograman fungsi untuk masing-masing jenis konversi **(7)**.