MODUL PRAKTIKUM
PEMROGRAMAN TERSTRUKTUR

Muhammad Sholeh
Erna Kumalasari Nurnawati

ISBN: 978-602-7619-40-1

AKPRIND PRESS

MODUL PRAKTIKUM

PEMROGRAMAN TERSTRUKTUR

ISBN: 978-602-7619-40-1

Penulis : Muhammad Sholeh, Erna Kumalasari Nurnawati
Desain Cover : Erna Kumalasari Nurnawati
Diterbitkan oleh : AKPRIND PRESS

KARTU PRAKTIKUM
PEMROGRAMAN TERSTRUKTUR

Nim : _____

Nama : _____

Minggu ke	Tanggal	Materi	Paraf Asisten dan Cap
1.			
2.			
3.			
4.			
5.			
6.			
7.			
8.			
9.			
10.			
INHAL 1			
INHAL 2			
RESPONSI			

TATA TERTIB PRAKTIKUM

1. Praktikum hadir sebelum praktikum dimulai. Keterlambatan ditoleransi 15 menit. Jika keterlambatan lebih dari 15 menit maka nilai yang diperoleh pada praktikum pada sesi tersebut maksimal 75% dari nilai yang diperoleh
2. Praktikan berpakaian sopan, tidak diperkenankan memamakai kaos oblong, topi serta tidak diijinkan makan di ruang laboratorium (minum boleh tetapi membawa sendiri).
3. **Mengisi daftar hadir dan meminta paraf dan Cap asisten pada kartu praktikum** anda
4. Praktikan tidak diijinkan menggunakan flasdisk
5. Praktikan WAJIB memiliki dan menggunakan modul terbaru. Modul bisa diperoleh di laboratorium dengan mengganti biaya cetak (Kartu praktikum terdapat di dalam modul). **Modul WAJIB dibawa saat praktikum.**
6. Praktikan menjaga kebersihan ruangan dan merapikan komputer, kursi dan peralatan lain setelah praktikum selesai. Matikan komputer dengan prosedur yang benar, serta mematikan stabilizer
7. Demi keamanan, praktikan hendaknya memakai sandal yang disediakan lab. Kembalikan sandal ditempatnya setelah selesai praktikum. Sandal tidak diijinkan digunakan di luar ruangan.
8. Penilaian praktikum dilakukan selama 8 minggu, sejak minggu ke 2 hingga minggu ke 9. Nilai max 90 (karena anda dibimbing asisten dan tidak ada tugas yang 100% anda kerjakan sendiri) dengan bobot masing-masing 10%. Nilai responsi max 100 berbobot 20%. Nilai anda akan diakumulasi dan menjadi nilai tugas ke 4 untuk matakuliah masing-masing. Tidak ada KETIDAK LULUSAN dalam praktikum, tetapi nilai praktikum mempunyai bobot 20% dari nilai matakuliah
9. Kehadiran praktikum harus mencapai 75% (mutual dengan kehadiran kuliah), sehingga apabila kehadiran kurang, maka anda tidak bisa mengikuti ujian akhir. Jika anda tidak hadir, anda diharapkan mengikuti inhal. Mutual dengan perkuliahan, anda diijinkan mengikuti inhal 2x, dan nilai dan kehadiran akan disinkronkan dengan nilai yang anda peroleh. Jika ketidakhadiran lebih dari 2 kali, maka yang bisa diinhal hanya 2x saja, sisanya dianggap tidak hadir dan nilai nol. Nilai inhal max 75.
10. Software dapat di donlod di http//bit.ly/mingw_LAB1
11. Demikian tata tertib praktikum harap mahasiswa menyesuaikan

KATA PENGANTAR

Puji Syukur ke hadirat Allah Tuhan Yanag Maha Esa yang melimpahkan karunianya sehingga perbaikan modul Pemrograman Terstruktur dapat direvisi dengan cukup signifikan.

Modul ini disusun sebagai pegangan pada praktikum Pemrograman Terstruktur dan diharapkan dapat membantu mahasiswa melaksanakan kegiatan praktikum dengan lebih baik. Mengingat kelengkapan modul ini, maka mahasiswa juga dapat menggunakan modul sebagai diktat kuliah Pemrograman Terstruktur di kelas. Dengan menggunakan modul ini diharapkan mahasiswa dapat mengembangkan logika berpikir dan kemampuan melakukan coding dalam bahasa C++.

Modul ini dilengkapi dengan banyak banyak latihan yang harus dikerjakan mahasiswa di labroratorium. Diharapkan dengan banyaknya latihan makin meningkatkan kemampuan pemrograman mahasiswa dan mengasah logika penrograman.

Materi pada modul ini meliputi pengenalan Bahasa C++ sebagai bahasa pemrograman terstruktur, bagaimana struktur pemrograman, mengenal tipe data, operator, identifier dan sintak-sintak bahasa pemrograman. Setelah itu mahasiswa diberikan materi statemen input, output, percabagan dan perulangan. Kemudian dilanjutkan dengan materi larik dimensi satu, larik dimensi dua, function dan procedure dan diakhiri dengan materi record dan array record.

Kami mengucapkan terimakasih atas semua pihak yang berkontribusi atas terselenggaranya revisi modul ini, terutama asisten di Lab Pemrograman Dasar. Jika ada pertanyaan atau perbaikan bias melalui email ernakumala@akprind.ac.id.

Yogyakarta, Februari 2019

Penyusun.

Muhammad Sholeh ST., MT.

Erna Kumalasari Nurnawati ST., MT.

DAFTAR ISI

MODUL 1
PENGENALAN BAHASA C++

Kompetensi: Mahasiswa Mengetahui dasar-dasar pemrograman meliputi struktur dasar bahasa pemrograman, Mengetahui penulisan komentar, pengenal, konstanta dan variabel pada bahasa pemrograman dan mengetahui perintah-perintah dasar untuk menampilkan tampilan pada monitor pada bahasa pemrograman.

A. Pengenalan C++

Berbicara tentang C++ tak lepas dari C, sebagai bahasa pendahulunya. C adalah bahasa pemrograman yang dapat dikatakan berada antara bahasa beraras rendah (bahasa yang berorientasi pada mesin) dan bahasa beraras tinggi (bahasa yang berorientasi pada manusia). Seperti diketahui bahasa tingkat tinggi mempunyai kompatibilitas yang tinggi antar *platform*. Tujuan utama pembuatan C++ adalah untuk meningkatkan produktivitas perogram dalam membuat aplikasi. C++ dapat mengurangi kekompleksitasan, terutam pada program yang besar yang terdiri dari 10.000 baris atau lebih.

Program C++ dapat ditulis menggunakan sembarang editor teks, seperti **EDIT** (milik DOS), **WordStar**, **SideKick**, ataupun menggunakan editor bawaan dari kompiler. Program C++ biasa ditulis dengan nama ektensi **.CPP** (dari kata C Plus Plus). Agar program bisa dijalankan (dieksekusi), program harus dikompilasi terlebih dahulu dengan menggunakan kompiler C++. Untuk praktikum ini kita menggunakan MinGW compiler dan Geany sebagai editor.

Struktur Bahasa C++

Program yang dibuat dengan Bahasa C++ selalu diawali dengan ***Preprocessor directive***, yaitu perintah yang diawali dengan tanda *pound* (#). Khusus untuk bagian awal program, selalu menggunakan ***Directive #include*** yang digunakan untuk mendefinisikan *file header* di dalam kode program. File header (file yang berekstensi **.h**) adalah file yang berisi fungsi-fungsi dan telah dikompilasi sebelumnya. Cara penulisannya adalah:

#include<nama_file> atau **#include "nama_file"**

Contoh :

#include <stdio.h>;

#include <conio.h>;

#include <iostream.h>;

Setelah *preprocessor* didefinisikan, diikuti dengan deklarasi fungsi. Adapun fungsi pertama yang harus ada di dalam program C++ sudah di tentukan namanya, yaitu bernama **fungsi main()** yang ditulis dengan diawali tipe fungsi, misalnya **main(), int main()** atau **char main()**. Fungsi selalu dibuka dengan kurung kurawal buka **'{'** dan diakhiri dengan kurung kurawal tutup **'}'**. Diantara kurung kurawal tersebut dituliskan statemen-statemen program. Penggunaan fungsi yang lain diletakkan di bawah fungsi utama dengan cara yang sama.

Struktur dasar program :

```
Preprosesor directive;
Fungsi Utama()
{
deklarasi variabel;
statement-statement;
}

Fungsi lain()
{
deklarasi variabel;
statement-statement;
}
```

B. Atribut Pemrograman

1. Komentar

Komentar pada suatu baris program merupakan suatu hal yang sangat penting sebagai dokumentasi jika suatu saat dilakukan modifikasi dan tidak ikut dibaca pada saat proses kompilasi.

Komentar menggunakan tanda //, digunakan untuk memberikan komentar atas perintah-perintah yang ada di bawahnya.

Contoh :

// Program ini dibuat pada tanggal 1 Maret 2018
***int luas**; //deklarasi variabel luas bertipe data integer*

Komentar menggunakan tanda /*komentar*/, digunakan untuk menuliskan komentar yang banyaknya satu baris atau lebih, mulai dari /* sampai */. Dengan menggunakan tanda ini dapat digunakan untuk memberi komentar sisipan pada satu baris.

Contoh :

/ Program ini dibuat pada tanggal 1 Maret*
2018 tepat dengan peringatan Serangan Oemoem/*
***int** /*bertipe data integer*/ **luas**;*

2. Pengenal (Identifier)

Pengenal merupakan pengidentifikasi dari nama-nama yang akan dideklarasikan agar *kompiler* dapat mengenalinya, meliputi nama variabel, konstanta, fungsi, kelas, template, maupun *namespace*. Pengenal dalam konstanta dan variabel berfungsi untuk menampung nilai yang digunakan dalam program. Identifikasi ini dilakukan untuk mempermudah proses penanganan data atau nilai. Penggunaan pengenal sebaiknya disesuaikan dengan kebutuhan dan sifat-sifat nilai yang akan ditampungnya dan hindarilah penggunaan pengenal yang mirip antar pengenal. Adapun hal-hal yang berkaitan dengan pengenal antara lain :

1. C++ bersifat case sensitive, sehingga penulisan huruf kecil dan kapital dianggap berbeda. Misalnya variabel **Luas** berbeda dengan **luas**.

Contoh:

int Luas; //variabel Luas dengan ‘L’ huruf kapital berbeda dengan **int luas;** //variabel luas dengan ‘l’ huruf kecil

Tidak boleh diawali dengan angka atau seluruh pengenal berupa angka.

Tetapi jika huruf awal adalah karakter dan selanjutnya adalah angka diperbolehkan. Contoh:

int gaji; //benar

int gajike13; //benar

int 2006; //salah, karena seluruhnya angka

int 2kali; //salah, karena diawali dengan angka

2. Tidak boleh mengandung spasi. Contoh:

int tgllahir; //benar

int tgl lahir; //salah, karena menggunakan spasi antara 'tgl' dan 'lahir'

3. Tidak boleh menggunakan tanda baca dan simbol-simbol (#, @, ?, !, &, dan lainnya). Contoh:

int tgllahir; //benar

int tgl&lahir; //salah, karena menggunakan simbol '&'

int tgl-lahir; //salah, karena menggunakan tanda baca ' – '

4. Tidak boleh menggunakan kata kunci (*keywords*).

Contoh:

int break; //salah, karena 'break' merupakan keywords

int while; //salah, karena 'while' merupakan keywords **int panjang;** //benar

3. Konstanta

Konstanta adalah jenis pengenal yang bersifat konstan atau tetap, sehingga nilai dari konstanta di dalam program tidak dapat diubah. Konstanta berguna untuk menentukan nilai yang bersifat tetapan, misalnya phi selalu bernilai 22/7 atau 3.142…..

Penggunaan konstanta dapat dilakukan dengan 2 (dua) cara yaitu :

a. Menggunakan ***Preprosessor Directive #define***

Struktur penulisan konstanta menggunakan preprosessor ini adalah:

#define <pengenal> <nilai>;

Contoh:

#define phi 3.14; //mendefinisikan konstanta ‘phi’ dengan nilai 3.14

#define max 1000; //mendefinisikan konstanta ‘max’ dengan nilai 1000

b. Menggunakan kata kunci ***Const***

Struktur penulisan konstanta menggunakan preprosessor ini adalah:

const <tipe_data> <pengenal=nilai>;

Contoh:

const double phi=3.14; // ‘phi’ dengan nilai 3.14 bertipe double **const long max=1000;** // ‘max’ dengan nilai 1000 bertipe long integer **const char lagi=’Y’;** // ‘lagi’ dengan nilai ‘Y’ bertipe karakter

4. Variabel

Variabel adalah pengenal yang mempunyai nilai dinamis sehingga nilai yang disimpan di dalamnya dapat diubah selama program berjalan sesuai dengan kebutuhan. Struktur penulisannya adalah :

<type_data> <pengenal>;

Contoh:

int panjang; // deklarasi variabel ‘panjang’ bertipe integer

int lebar, luas; // deklarasi variabel ‘ le ba r ’ da n ‘ lua s ’ ,ke duan ya be rt ipe integer

Pada deklarasi variabel juga dapat dilakukan inisialisasi (pemberian nilai awal), dengan struktur penulisan sebagai berikut:

<type_data> <pengenal=nilai_awal>;

Contoh:

int panjang=10; //deklarasi variabel ‘panjang’ bertipe integer dengan nilai 10

double tunjangan=1.25;

5. Tipe Data

Tipe data berfungsi untuk merepresentasikan jenis dari sebuah nilai yang terdapat dalam program. Penggunaan tipe data yang benar sangat mendukung efisiensi program karena tipe data juga berpengaruh terhadap kapasitas data.

Dalam C++ terdapat beberapa tipe data dasar yang telah didefinisikan yaitu tipe bilangan bulat (*integer*), bilangan riil (*floating point*), logika (*boolean*), dan teks (*character/string*).

Tipe Bilangan Bulat (*Integer*)

Tipe data ini digunakan untuk data-data angka bilangan bulat (yang tidak mengandung angka di belakang koma), misalnya 21, 100, 1929, dan seterusnya.

Tipe Bilangan Riil (*Floating point*)

Tipe data ini digunakan untuk data-data angka bilangan riil atau pecahan (mengandung angka di belakang koma), misalnya 21.76, 100.567, 1929.156, dan seterusnya.

Tipe Logika (*Boolean*)

Berbeda dengan kedua jenis data di atas, tipe data *boolean* digunakan untuk merepresentasikan data-data yang hanya mengandung dua buah nilai, yaitu nilai *True* (yang direpresentasikan selain 0) dan nilai *false* (direpresentasikan dengan 0).

Teks (*Character*/*String*)

Tipe data ini merepresentasikan data-data yang berupa karakter. Tipe data ini dinyatakan dengan tipe **char**, sedangkan untuk string (=kumpulan karakter) dinyatakan **string** dengan panjang 255 karakter

C. Statement Menampilkan Teks atau Nilai Konstanta dan Variabel

Pada bahasa C standar, untuk menampilkan nilai atau teks menggunakan perintah **printf**. Pada C++ dapat dilakukan menggunakan perintah **cout<<** (perintah *cout* diikuti dengan tanda ‘kurang dari’ sebanyak dua kali) kemudian diikuti dengan string atau teks yang akan ditampilkan diapit dengan tanda kutip ganda.

Contoh :

```
cout << “Belajar C itu mudah”;
```

cout << “Fakultas Teknologi Industri”;

Jika yang akan ditampilkan adalah nilai dari suatu konstanta atau variabel, maka setelah perintah **cout<<** diikuti dengan nama konstanta atau nama variabelnya.

Contoh :

cout << luas; //menampilkan nilai konstanta atau variabel ‘luas’

cout << nama; //menampilkan nilai konstanta atau variabel ‘nama’

D. Statement Input

Suatu program yang dinamis membutuhkan peran aktif dari user untuk memasukkan nilai atau data ke dalam program melalui *keyboard*. Pengguna akan memasukkan data yang akan disimpan atau diolah, kemudian pihak lain atau bahkan pengguna sendiri yang akan membutuhkan informasi dari data yang disimpan atau diolah itu.

Struktur penulisan perintah untuk memasukkan data dari *keyboard* menggunakan perintah **cin>>** (perintah *cin* diikuti dengan tanda ‘lebih dari ‘ sebanyak dua kali) kemudian diikuti dengan nama konstanta atau nama variabel. Syarat yang harus dipenuhi untuk memasukkan data atau nilai ini, konstanta atau variabel harus dideklarasikan tipe datanya terlebih dahulu.

Contoh :

cin >> panjang; //memasukkan isi data variabel ‘panjang’

cin >> nama; //memasukkan isi data variabel ‘nama’

E. Latihan

Latihan 1. Menampilkan Tulisan “Selamat datang…” (1)

Berikut ini adalah program sederhana untuk menampilkan tulisan pada monitor.

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

```
main()
{
cout<<”Hello world, Selamat Datang di Lab Pemrograman”<<endl;
cout<<”Semester ini kita belajar C++ disini”;
}
```

Latihan 2. Menampilkan Nilai Konstanta

Program di bawah ini untuk menampilkan isi nilai dari suatu konstanta. Pada bagian atas dideklarasikan preprosessor directive untuk file header dan mendefinisikan konstanta dengan nama bilangan1. Kemudian pada fungsi utama dideklarasikan dengan cara yang berbeda dengan nama konstanta bilangan2. Baris selanjutnya menampilkan isi konstanta variabel bilangan1 dan bilangan2. Perhatikan cara menampilkan isi dari suatu konstanta bilangan1 dan bilangan2 yang dilakukan dengan cara yang berbeda.

```
#include <iostream>
using namespace std
#define bilangan1=10;
main(){
int bilangan2=5;
cout<<”Berikut ini adalah isi bilangan 1 :
”; cout<<bilangan1;<<endl;cout<<”Berikut ini adalah isi bilangan 2 : “;<<bilangan2;
}
```

Latihan 3. Menghitung Perkalian 2 Bilangan

Program di bawah ini untuk menampilkan isi nilai dari suatu konstanta dimana bilangan1 dan bilangan2 telah ditentukan kemudian dikalikan dan ditampilkan pada baris berikutnya dengan variabel hasil. Perhatikan cara deklarasi suatu konstanta dan proses penghitungan hasil.

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

```
using namespace std;

main(){
int bilangan1; int bilangan2; int hasil;
bilangan1 = 5;
bilangan2 = 3;
hasil = bilangan1 * bilangan2;
cout<<”Bilangan Pertama : “;<<bilangan1;
cout<<”Bilangan Kedua  : “;<<bilangan2;
cout<<”Hasil Perkalian : “;<<hasil;
}
```

Latihan 4. Memasukkan bilangan dan Menampilkan kembali

Program di bawah ini mulai menggunakan variabel. Salah satu bagian dari penggunaan variabel adalah untuk memasukkan nilai dari *keyboard* kemudian ditampilkan kembali. Perhatikan perintah untuk memasukkan data dan perintah untuk menampilkan kembali data yang diinputkan.

```
#include <iostream>
#include <conio.h>
using namespace std
main(){
int bilangan;
cout<<”Masukkan suatu bilangan : “;
cin>>bilangan;<<endl;
cout<<endl;
cout”Bilangan yang ada masukkan adalah : “;<<bilangan;
}
```

MODUL 2
TIPE DATA, OPERATOR DAN STATEMEN

Kompetensi: Mahasiswa mengetahui, mamahami dan menggunakan tipe data sederhana, oprator aritmatika dan menampilkan hasil di monitor pada bahasa pemrograman

A. TIPE DATA

Tipe data berfungsi untuk merepresentasikan jenis dari sebuah nilai yang terdapat dalam program. Kesalahan dalam menyebutkan tipe data program yang dibuat tidak dapat dijalankan atau seandainya jalan akan menghasilkan nilai yang tidak akurat. Sehingga tipe data harus digunakan sesuai dengan kebutuhan program. Dalam C++ terdapat beberapa tipe data dasar yang telah didefinisikan, yaitu yang digolongkan ke dalam tipe bilangan bulat (integer), bilangan riil (floating-point), tipe logika (boolean), tipe karakter/teks (character/string). Tipe- tipe tersebut adalah tipe yang siap digunakan tanpa adanya proses manipulasi terlebih dahulu. Tipe data dalam pemrograman C++ hampir sama dengan pada Pascal. Meliputi:

a. Tipe bulat

Tipe bulat digunakan untuk merepresentasikan bilangan bulat (tanpa koma)

Tipe Data	Ukuran (bit)	Rentang
Int	16 atau 32	-32.768 sampai 32.767
Unsigned int	16 atau 32	0 - 65.535
Signed int	16 atau 32	Sama seperti **int**
Short int	16	-32.768 sampai 32.767
Unsigned short int	16	0 sampai 65.535
Signed short int	16	Sama seperti **short int**
Long int	32	-2.147.483.648 sampai 2.147.483.647
Signed long int	32	Sama seperti **long int**
Unsigned long int	32	0 sampai 4.294.967.295

b. Tipe Real atau floating point

Tipe real digunakan untuk merepresentasikan bilangan decimal (dengan koma), meliputi:

Tipe Data	Ukuran (bit)	Rentang	Presisi
Float	32	1.2E-38 sampai 3.4E+38	6 digit presisi
double	64	2.3E-308 sampai 1.7E+308	15 digit presisi
Long double	80	3.4E-4932 to 1.1E+4932	19 digit presisi

c. Tipe Boolean

Tipe Boolean digunakan untuk merepresentasikan bilangan biner 1 (true) dan 0(false). Tipe ini biasanya digunakan untuk merepresentasikan hasil penyeleksian kondisi apakah bernilai benar (1) atau salah (0)

d. Tipe String

Tipe string adalah tipe data berupa karakter (char) atau deretan karakter (string).

B. OPERATOR

Operator digunakan untuk menghubungkan atau mengekspresikan perintah dalam bahasa pemrograman. Suatu statemen biasanya terdiri dari variable, konstanta, operator dan syntax. Operator dalam C++ meliputi:

a. Operator pemberian nilai (assignment)

Dalam C++ operator pemberian nilai disimbolkan dengan "=". Pemberian nilai bias dilakukan secara langsung, maupun melalui suatu ekspresi aritmatika atau logika. Sebagai contoh pemberian nilai pada variable berikut:

```
Int a=10;
Float b=5.2010;
String nama="Bambang Gentolet";
```

b. Operator Aritmatika

Operator aritmatika adalah operator-operator yang digunakan dalam ekspresi matematika. Operator aritmatika bisa diterapkan pada tipe data numeric dan non numeric. Adapun operator aritmatika dalam C++ meliputi:

Operator	Jenis Operasi	Contoh
+	Penjumlahan	2+3=5
-	Pengurangan	5-3=2
*	Perkalian	2*3=6
/	Pembagian	10.0/3.0=3.3333
%	Sisa bagi (modulus)	10%3=1

Untuk operator % (modulus) harus menggunakan tipe bulat. Sedangkan pada operator pembagian (%) apabila tipe data yang digunakan adalah bulat maka hasilnya akan bulat, dengan membuang sisanya.

c. Operator Logika

Operator logika digunakan untuk menguji hasil kebenaran suatu ekspresi dan menghasilkan nilai true atau false (bit 1 atau 0). Adapun operator logika dalam C++ meliputi:

Operator	Jenis Operasi	Contoh
&&	AND (dan)	1 && 1 = 1
||	OR (atau)	1 || 0 = 1
!	NOT (negasi)	!0 = 1

d. Operator relasional

Operator relasional digunakan untuk mengetahui hasil relasi atau hubungan diantara dua ekspresi atau dua operand. Hasil dari statemen relasional adalah nilai true atau false. Adapun operator relasional meliputi:

Operator	Jenis Operasi	Contoh
>	Lebih besar	(5>2)=1
<	Lebih kecil	(5<2)=0
>=	Lebih besar atau sama dengan	(5>=5)=1
<=	Lebih kecil atau sama dengan	(5<=2)=0
==	Sama dengan	(5==2)=0
!=	Tidak sama dengan	(5!=2)=1

e. Operator Unary

Operator unary adalah operator yang digunakan terhadap satu variable saja. Meliputi :

Operator	Jenis Operasi	Contoh
+	Membuat nilai positif	+7
-	Membuat nilai negatif	-7
++	Increment	C++
--	Decrement	C--

Penggunaan operator ini hanya membutuhkan satu operand saja.

C. STATEMEN INPUT DAN OUTPUT

a. Statemen Output

Statemen output adalah statemen untuk menampilkan hasil proses, komentar maupun ekspresi ke layar monitor. Perintahnya adalah :

cout<<" komentar"<<var<<var<<"komentar " ;

jika diingankan ganti baris setelah mencetak maka dapat diberikan perintah sbb:

cout<<" komentar"<<var<<var<<"komentar " <<endl;

atau

cout<<" komentar"<<var<<var<<"komentar\n " ;

jika diinginkan ada beep saat mencetak maka bisa diberikan perintah sbb:

cout << "Pilihan Anda salah !\a\n";

Jika anda ingin memberikan lebar dalam tampilan data maka dapat digunakan setw(). Contoh:

```
cout << "Barang 1 = " << setw(4) << jumbar1 << endl;
cout << "Barang2 = "  << setw(4)  << jumbar2 << endl;
```

jika ingin membersihkan layar sebelum mencetak maka dapat dilakukan dengan perintah clrscr(). Jika anda ingin menggunakan beberapa perintah pengaturan tampilan di atas maka harus memanggil fungsi #include<conio.h>.

Contoh :

```
//*-------------------------------------------------------------------------*
//*  Program Manampilkan 3 buah barang dengan                               *
//*                menggunakan manipulator setw()  dan clrscr()     *
//*-------------------------------------------------------------------------*
#include <iostream.h>
#include <iomaip.h>   // Untuk manipulator setw()
#include <conio.h>
using namespace std


  main()
{
      int jumbar1 = 150;
       jumbar2 = 23;     jumbar3 = 1401;

      clrscr();  // Hapus layar
```

```
        cout << "Barang 1 = " << setw(4) << jumbar1 << endl;
        cout << "Barang2 = "  << setw(4)  << jumbar2 << endl;
        cout << "Barang3 = "  << setw(4)  << jumbar3 << endl;
}
```

Hasil eksekusi :

```
Barang 1 =    150
Barang 2 =     23
Barang 3 =   1401
```

b. Statemen Input

Dalam pemrograman c++ statemen input digunakan untuk memasukkan nilai masukan ke variable melalui keyboard. Adapun statemen yang digunakan adalah :

cin>> var;

Contoh program yang menunjukkan penmakaian **cin** untuk membaca data bertipe **int** dan **float**.

```
//*-------------------------------------------------------*
//* Program  Membaca data bertipe int dan     *
//*              float dari keyboard                  *
//*-------------------------------------------------------*
#include <iostream.h>
  main ()
{
        int bil_x ;  // Definisi bilangan bulat
        float bil_y ; // Definisi bilangan pecahan
        cout << "Masukkan sebuah bilangan bulat  = " ;
        cin  >> bil_x ;
        cout << "Masukkan sebuah bilangan pecahan = " ;
        cin >> bil_y;

        cout << " Bilangan Bulat = " << bil_x << endl;
        cout  << " Bilangan Pecahan = " << bil_y << endl;
 }
```

Hasil eksekusi :

```
Masukkan sebuah bilangan bulat = 123
Masukkan sebua  bilangan pecahan = 23.1
Bilangan Bulat = 123
Bilangan Pecahan 23.1
```

D. LATIHAN:

1. Tulis program berikut :

```
#include <iostream>
using namespace std;
int main()
{   int a,b;
 a=10;
 b=20;
 c=a+b;
 cout<<"isi c = "<<c;}
```

Apa hasilnya dan kenapa hasil seperti itu ? bagaimana supaya benar

```
#include <iostream>
using namespace std;
int main()
{   int a,b,c;
 a=40000;
 b=20;
 c=a+b;
 cout<<"isi c = "<<c;
}
```

Apa hasilnya dan kenapa hasil seperti itu ? bagaimana supaya benar

```
#include <iostream>
using namespace std;
int main()
{   int a,b,c;
 a=15;
 b=2;
 c=a/b;
 cout<<"isi c = "<<c;
}
```

Apa hasilnya dan kenapa hasil seperti itu ? bagaimana supaya benar

```
#include <iostream>
using namespace std;
int main()
{   int a,b;
 float c;
 a=15;
 b=2;
 c=a/b;
```

```
 cout<<"isi c = "<<c;
}
```

Apa hasilnya dan kenapa hasil seperti itu ? bagaimana supaya benar silahkan ditelaah,

2. Dari program-propgram di atas sekarang gunakan perintah INPUT

```
Int main()
{   float a,b;
   float c;
   cout<<"masukan nilai a "; cin>>a;
   cout<<"masukan nilai b "; cin>>b;
   c=a+b;
   cout<<"isi c = "<<c;
}
```

Apa hasilnya dan kenapa hasil seperti itu ?

```
Int main()
{   float a,b;
   float c;
   cout<<"masukan nilai a "; cin>>a>>b;
   c=a+b;
   cout<<"isi c = "<<c;
}
```

Kenapa dan bagaimana program yang benar silahkan ditelaah.

E. Tugas :

1. Buatlah program untuk konversi suhu dari Celcius ke Reamur, Fahrenheit dan Kalvin dengan rumus sebagai berikut:

F = 9/5*C + 32

R = 4/5 * C

K = 273+C

2. Buatlah program untuk mengkonversi satuan panjang dengan ketentuan sebagai berikut:

Konversi satuan panjang dari yard, kaki dan inchi ke meter menggunakan standar berikut ini :

1 yard = 3 kaki = 0,9144 meter

1 kaki = 12 inchi = 30,48 centimeter = 0,3048 meter

1 inchi = 25,4 milimeter = 0,0254 meter

MODUL KE-3
PENGGUNAAN OPERATOR LANJUTAN

Kompetensi: Mahasiswa mengetahui, mamahami dan menggunakan tipe data sederhana, operator aritmatika dan menampilkan hasil di monitor pada bahasa pemrograman

A. Pendahuluan

Dari teori dan praktek yang telah dipelajari di modul pertama dan kedua, maka latihan berikut ini adalah penggunaan operator dan variable secara lebih mendalam.

B. Latihan

Latihan 1. Konversi satuan panjang dari yard, kaki dan inchi ke meter menggunakan standar berikut ini :

1 yard = 3 kaki = 0,9144 meter

1 kaki = 12 inchi = 30,48 centimeter = 0,3048 meter

1 inchi = 25,4 milimeter = 0,0254 meter

Algoritma dari program ini adalah sebagai berikut :

1. Masukkan satuan panjang meter
2. meter 0.9144 * yard + 0.3048 * kaki + 0.0254 * inchi
3. Tampilkan meter

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

int main()
{
 //Deklarasi variabel
 double yard, kaki, inchi, meter; cout<<”Masukkan
 satuan yard : “; cin>>yard; cout<<”Masukkan
 satuan kaki : “; cin>>kaki; cout<<”Masukkan
```

```
satuan inchi : “; cin>>inchi;

//menghitung konversi
meter = 0.9144 * yard + 0.3048 * kaki + 0.0254 *
inchi; cout<<endl;

//menampilkan hasilnya
cout<<yard<<” Yard”<<kaki<<” Kaki”<<inchi<<” Inchi
setara dengan “<<meter<<” meter”;

return 0; //mengembalikan nilai 0 sebab fungsi main bertipe int

}
```

Jika sudah selesai lakukan running dan pelajarilah hasilnya.

Latihan 2. Menggunakan fungsi sqrt() dan pow(a,b). Setelah selesai mengerjakan latihan pertama, kerjakan latihan ke dua berikut ini.

```
#include<iostream>
#include<conio.h>
#include<math.h>
using namespace std;
int a,t;
float m,L,K;
int main()
{
cout<<"Menghitung Luas dan Keliling Segitiga Siku-siku"<<endl;
cout<<"masukkan alas   : ";cin>>a;
cout<<"masukkan tinggi : ";cin>>t;
m=sqrt(pow(a,2)+ pow(t,2));
L=0.5*a*t;
K=a+m+t;
cout<<"Segitiga dengan alas = "<<a<<" dan tinggi = "<<t<<endl;
cout<<"Luas                 = "<<L<<endl;
cout<<"Keliling    = "<<K<<endl;
}
```

Fungsi sqrt() adalah fungsi untuk mencari akar dari suatu variable atau ekspresi. Sedangkan fungsi pow(a,b) adalah fungsi untuk mencari pangkat. Dalam hal ini adalah a pangkat b. Kedua fungsi di atas tersedia dalam library math.h sehingga harus melakukan #include<math.h> jika akan menggunakan fungsi tersebut.

Latihan 3. Menggunakan operator modulus. Operator modulus (%) digunakan untuk

mencari sisa hasil pembagian. Misal 8 % 2 = 0, atau 10 % 3 = 1 dan sebagainya. Penggunaan operator ini banyak digunakan untuk menentukan factor dari suatu bilangan, sisa pembagian dan sebagainya. Ketiklah contoh program berikut:

```
#include<iostream>
using namespace std;
int a,b,c;

main()
{
  cout<<"masukkan nilai a: ";
  cin>>a;
  cout<<endl<<"masukkan nilai b: ";
  cin>>b;
  c=a%b;
  cout<<endl<<"hasil "<<a<<" modulo "<<b<<" = "<<c<<endl;
}
```

C. Tugas

Dari 3 latihan di atas, kerjakan soal berikut:

1. Buatlah program untuk mengkonversi satuan panjang dari meter ke yard, kaki dan inchi. Masukan dari program ini adalah panjang dalam satuan meter. Keluarannya adalah panjang dalam satuan yard, kaki dan inchi.

2. Buatlah program untuk menentukan berat badan ideal seseorang jika diketahui tinggi badannya. Sebagai contoh jika tinggi badannya 180 centimeter maka berat badan yang ideal adalah 72 kilogram. Karena berat badan ideal dihitung dari tinggi badan dikurangi dengan 100 kemudian dikurangi lagi dengan 10% dari sisanya. Jadi 180 cm dikurangi 100 cm menjadi 80 cm. Dari 80 cm dikurangi 10% (8 cm) menjadi 72.

MODUL KE-4
STATEMEN KONDISIONAL

Kompetensi: Mahasiswa mengetahui, mamahami dan menggunakan dasar percabangan dalam statemen kondisional, mampu memilih dan menerapkan dalam penyelesaian masalah pada bahasa pemrograman

A. Pendahuluan

Percabangan (condisional) adalah kondisi pada pemrograman dimana alur logika dihadapkan pada pilihan. Statemen kondisional adalah statemen yang menggunakan percabangan dalam alur programnya. Statemen akan dieksekusi berdasarkan kondisi yang memenuhi, apakah kondisi benar (true) atau salah (false). Statemen kondisional selalu menghasilkan ekspresi logika true atau false dan tidak mungkin keduanya. Dalam C++ percabangan terdiri dari percabangan tunggal dan kondisi ganda dengan IF dan SWITCH.

Struktur Kondisional adalah sebagai berikut:

```
// jika terdapat lebih dari satu statemen if (kondisi)
{
Statemen1;
Statemen2;
....
}

//Jika hanya satu statemen, dapat ditulis seperti di bawah if
(kondisi) Statemen;
```

Statemen 1 dan 2 akan dieksekusi jika kondisi bernilai true, dan akan diabaikan jika kondisi bernilai false. Struktur kondisi juga bisa menggunakan ELSE sebagai berikut:

```
// jika terdapat lebih dari satu statemen if (kondisi)
{
Statemen1;
Statemen2;
....
}
else
{
```

```
Statemen3;
Statemen4;
....
}
```

Statemen1 dan statemen 2 akan dieksekusi jika kondisi bernilai true, dan akan mengerjakan statemen3 dan statemen4 jika kondisi bernilai false. Struktur kondisi juga bisa menggunakan nested IF sebagai berikut:

```
if (kondisi1)
   pernyataan1;
else if (kondisi2)
   pernyataan2;
else if (kondisi3)
   pernyataan3;
if (kondisiM)
   pernyataanM;
else                    /*Opsional*/
   pernyataanN;         /*Opsional*/
```

3. Kondisional menggunakan SWITCH

Switch adalah pernyataan yang digunakan untuk menjalankan salah satu pernyataan dari beberapa kemungkinan pernyataan, berdasarkan nilai dari sebuah ungkapan dan nilai penyeleksi. Kaidah umum pernyataan switch :

```
switch (ungkapan)
{
    case ungkapan1;
         pernyataan_1;
         break;
    case ungkapan2;
         pernyataan_2;
         break;
     …….
     default :                  /*Opsinal*/
         pernyataan_x;          /*Opsinal*/
}
```

B. Latihan

Latihan. 1.

Program berikut adalah menentukan apakah bilangan yang diinputkan adalah bilangan positif atau negative

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

int main()
{
    cout<<"masukkan nilai a: ";
    cin>>a;
    if (a>0)
    {cout<<a<<" adalah bilangan positif"<<endl;}
    else
    {cout<<a<<" adalah bilangan negatif"<<endl;}
}
```

Latihan 2. Menggunakan IF NESTED

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

int main()
{
    cout<<"masukkan nilai a: ";   cin>>a;
    if (a==0)
            cout<<" anda memasukkan bilangan nol"<<endl;
            else if (a>0)
            cout<<a<<" adalah bilangan positif"<<endl;
            else
                    cout<<a<<" adalah bilangan negatif"<<endl;
}
```

Latihan 3. Menggunakan IF dengan Operator relasional

```
#include <iostream>
using namespace std;
int main()
{  int i,j,k; i=30;
    if (i<=20)
            cout<<"A"<<endl;
    else
            cout<<"B"<<endl;
    cout<<"c"<<endl;
}
```

Latihan 4. Menggunakan SWITCH

```
include<stdio.h>
#include<conio.h>
#include<iostream.h>
main()
{
    char kode;
    cout<<"Masukkan Kode Barang [A..C] : ";
    cin>>kode;
    switch(kode)
    {
        case 'A' :
            cout<<"Alat Olah Raga";
            break;
        case 'B' :
            cout<<"Alat Elelktronik";
            break;
        case 'C' :
            cout<<"Alat Masak";
            break;
        default:
            cout<<"Anda Salah Memasukan kode";
        break;
    }
    getch();}
```

C. TUGAS .

1. Untuk menilai seseorang termasuk terlalu gemuk atau kurus, seseorang tersebut dikontrol dengan criteria sbb : bila tinggi badan - berat badan >120 artinya kegemukan
   bila tinggi badan - berat badan antara 111 s.d.120 artinya ideal
   bila tinggi badan - berat badan antara 100 s.d.110 artinya kurus
   bila tinggi badan - berat badan < 100 artinya sangat kurus

2. Buatlah program dengan ketentuan :

   - Membaca *input* berupa skor (angka bilangan bulat) dari seorang mahasiswa
   - Menampilkan nilai (dalam huruf) dari mahasiswa tsb.

   Adapun syarat-syarat pemberian nilai adalah sebagai berikut:

   - Jika skor > 80 nilai: **A**
   - Jika 60 <= skor <= 79 nilai: **B**
   - Jika 40 <= skor <= 59 nilai: **C**
   - Jika 20 <= skor <= 39 nilai: **D**
   - Jika <20 nilai: **E**

MODUL KE-5
PERULANGAN (LOOP)

Kompetensi: Mahasiswa mengetahui, mamahami dan menggunakan statemen perulangan sederhana dengan jumlah tertentu pada bahasa pemrograman. Pada modul ke 5 ini dibahas perulangan dengan menggunakan FOR.

A. Teori

Perulangan adalah suatu mekanisme dalam bahasa pemrograman agar melakukan eksekusi terhadap suatu statemen berulangkali. Statemen dalam blok yang berada dalam perulangan akan diulang sesuai jumlah yang ditetapkan atau akan diulang sesuai kondisi yang berlaku. Dalam pemrograman, perulangan terdiri dari dua kategori, yaitu perulanagn tertentu, dimana jumlah perulangan statemen sudah ditentukan, dan perulangan tak tentu, dimana jumlah perulangan ditentukan oleh kondisi. Ada perulangan yang dikerjakan jika kondisi tertentu di penuhi, ada pula perulangan yang dilakukan terus menerus hingga kondisi tertentu terpenuhi baru perulangan dihentikan.

Struktur pengulangan jenis ini digunakan untuk melakukan perulangan yang telah diketahui banyaknya statemen yang diulang. Untuk melakukan perulangan dengan menggunakan struktur ini, kita harus memiliki sebuah variabel sebagai indeksnya. Namun perlu sekali untuk diperhatikan bahwa tipe data dari variabel yang akan digunakan sebagai indeks haruslah tipe data yang mempunyai urutan yang teratur, misalnya tipe data **integer** (0,1,2,….) atau **char** (‘a’, ‘b’, ‘c’,…..).

Dalam statemen perulangan dikenal suatu variable index sebagai pencacah jumlah perulangan yang dilakukan. Jika perulangan terindeks naik maka nilai awal harus lebih kecil dari nilai akhir. Sebaliknya jika perulangan terindeks menurun maka nilai awal harus lebih besar dari nilai akhir.

Dalam C++ perulangan tertentu diberikan dengan perintah FOR dengan struktur sebagai berikut:

```
// Untuk pengulangan yang sifatnya menaik (increment)
      for (variabel=nilai_awal; kondisi; variabel ++) {
       Statemen_yang_akan_diulang;
}
// Untuk pengulangan yang sifatnya menurun (decrement)
      for (variabel=nilai_awal; kondisi; variabel --) {
       Statemen_yang_akan_diulang;
}
```

B. Latihan

Contoh program latihan 1. Index menaik

```
#include<iostream>
#include<conio.h>
using namespace std;
int a,b;

int main()
{   b=100;
    for(a=5;a<=10;a++)
    {cout<<"a = "<<a<<"  b= "<<b<<endl;
     b=b+5;          }
    }
```

Contoh program latihan 2. Index menurun

```
#include<iostream>
#include<conio.h>
using namespace std;
int a,b;
int main()
{       b=100;
        for(a=10;a>=5;a--)
        {cout<<"a = "<<a<<"  b= "<<b<<endl;
         b=b+5;          }
```

```
}
```

Contoh latihan 3. Index diinputkan

```
#include <iostream>
using namespace std;
int main()
{  int a,b,c;
        cout<<"masukan nilai awal ";cin>>a;
        cout<<"masukan nilai akhir ";cin>>b;
        for(c=a;c<=b;c++)
                cout<<c<<endl;
        cout<<"selesai";
}
```

C.Tugas.

1. Buatlah program sederhana untuk menampilkan bilangan ganjil diantara dua buah bilangan
2. Buatlah program sederhana untuk mencetak kelipatan deret aritmatika ke 1 sampai dengan ke n dengan nilai awal a. Deret berikutnya adalah ditambah 3 jika. Hitung seluruh deret yang telah dibuat

MODUL KE-6
PERULANGAN (LOOP) LANJUTAN

Kompetensi: Mahasiswa mengetahui, mamahami dan menggunakan statemen perulangan dengan jumlah tak tentu untuk menyelesaikan masalah dalam menyelesaikan kasus dalam bahasa pemrograman. Pada modul ke enam ini dibahas perulangan berdasar kondisi yang diuji dengan while

A. Pendahuluan

Perulangan dalam jumlah tak tentu banyak digunakan dalam kasus pemrograman dikarenakan pengulangan statemen tidak ditentukan dengan pasti berapa jumlahnya, Misal dalam input data, maka selama operator mau menginput maka input bisa dilakukan. Inputan akan berhenti jika operator memasukkan angka 0. Atau dalam kasus lain, missal seseorang yang dihitung langkahnya maka akan terus melangkah hingga tujuan tercapai. Dalam C++ perulangan kondisional dideklarasikan dengan struktur sebagai berikut:

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

Atau sebagai berikut:

```
Do
{       Statemen_statemen_yang_akan_diulang;
} while (kondisi)
```

Pada while (kondisi) maka kondisi akan diuji terlebih dahulu, jika bernilai true maka statemen-statmen akan diulang. Dalam hal ini ada kemungkinan statemen tidak akan dikerjakan sama sekali jika pada pengecekan pertama kali kondisi langsung bernilai false. Sementara pada do {statemen}while (kondisi), statemen akan dikerjakan terlebih dahulu baru kondisi dicek. Hal ini sama dengan repeat…until(kondisi) dalam Pascal sehingga jika menggunakan statemen ini, maka statemen paling tidak akan dikerjakan satu kali.

B. Latihan

Latihan 1. Cobalah program di bawah ini dan pelajari keluarannya

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

int main()
{ a=10;
  while(a<=50)
  {       cout<<"a = "<<a<<endl;
          a=a+5;
  }
  }
```

Latihan 2. Cobalah program di bawah ini dan pelajari keluarannya. Bandingkan dengan program sebelumnya.

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

int main()
{   a=10;
    do
    {       cout<<"a = "<<a<<endl;
            a=a+5;
    }
    while(a<=50);
}
```

Latihan 3.

```
#include<iostream>
```

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

int main()
{
int num=0;
cout<<" Angka   Angka kuadrat"<<endl;
cout<<"==============="<<endl;
while (num++ < 10)
cout<<num<<"     "<<(num*num)<<endl;
return 0;
}
```

C. Tugas

1. Buatlah program untuk menginputkan sejumlah angka (diakhiri jika angka yang dimasukkan >=20. Kemudian hitung cacah angka yang diimputkan dan jumlah data keseluruhan. Cari rata-ratanya. Cetak jumlah dan rata-ratanya. Gunakan while do dan do while

2. Buatlah program untuk menampilkan bilangan kelipatan 5 antara 125 sampai dengan 200, menggunakan **while** dan **do..while**

3. Buatlah program untuk menghitung jumlah pantulan bola jika bola dijatuhkan dari ketinggian h cm, pantulan pertama x cm. Bola dianggap berhenti apabila pantulan mendekati nol (x<=1 cm).

MODUL KE-7
LARIK DIMENSI SATU

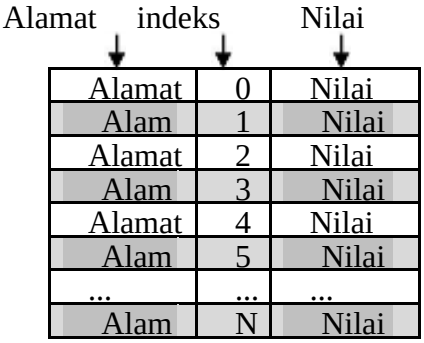
Kompetensi: Mahasiswa mampu mengenali larik dimensi satu, mampu mendeklarasikan dan menggunakan larik dimensi satu dalam menyelesaikan masalah dalam pemrograman

A. Pengertian Larik Dimensi Satu

Tipe data yang digunakan sebelumnya adalah suatu tipe data tunggal, dimana masing-masing variable akan dapat menyimpan sebuah nilai pada satu kesempatan. Namun ada kalannya kita harus menyimpan data dalam jumlah banyak dengan tipe yang sama. Misal jika ingin menyimpan data nama mahasiswa dalam satu kelas dengan jumlah mahasiswa maksimum 50, maka tidak efisien jika kita menyimpanya dalam 50 variabel. Untuk menyelesaikan masalah tersebut, maka dapat digunakan tipe data array dimensi satu.

Jadi Array adalah variabel penyimpan sekumpulan data yang memiliki tipe sama. Setiap data menempati lokasi atau alamat memori yang berbeda-beda dan selanjutnya disebut dengan elemen array. Elemen array itu kemudian dapat diakses melalui indeks yang terdapat di dalamnya. Berbeda dengan Bahasa Pascal yang memulai indeks dari 1 (satu), indeks array pada Bahasa C dimulai dari 0 (nol).

Berikut ini adalah gambaran sederhana tentang array :



Sebelum digunakan, array harus dideklarasikan terlebih dahulu menggunakan tanda **[ ]** (*bracket*). Struktur penulisannya adalah sebagai berikut :

Tipe data nama_array[jumlah array] ;

Contoh:

int angka[10];

float angka2[10];

char nilai[5];

Pada deklarasi di atas variable angka bertipe integer dengan jumlah elemen maksimum 10, sedangkan variable angka2 bertipe float dengan jumlah elemen maksimum juga 10, sedangkan variable nilai bertipe char dengan jumlah elemen maksimum 5. Pada variable bertipe array digunakan index untuk menyatakan elemen yang dituju. Misal angka[1] menyatakan variable angka elemen pertama, sedangkan angka[10] menyatakan elemen ke-10.

2. Memasukkan nilai ke dalam Larik dan mencetak hasilnya

Untuk memasukkan nilai ke dalam elemen-elemen larik, dapat dilakukan dapat dilakukan langsung untuk setiap elemen jika diketahui alamat indeksnya. Misal jika akan memasukkan nilai 60 ke dalam array **nilai** pada indeks ke 7 maka perintahnya :

nilai[7] =”A”;

Namun cara ini tidak direkomendasikan karena tidak efisien dan bersifat statis. Karena indeks pada array bernilai urutan data secara teratur dari 0, 1, 2, 3, 4, dan seterusnya maka lebih umum dan banyak digunakan oleh para programmer untuk mengisikan nilai ke dalam elemen-elemen array dengan perulangan (*looping*). Cara ini akan lebih cepat dan bersifat dinamis.

Struktur penulisan array dengan menggunakan perulangan adalah :

```
Tipe data nama_array[jumlah array]
For (index awal;index akhir;increment index)
 {  pengisian larik
    Manipulasi larik
   Pencetakan larik
 }
```

Contoh:

```
#include<iostream>
#include<conio.h>
using namespace std;
int a[10];
int main()
{
  int i=0;
//input data
 for(i=0;i<=4;i++)
 {cout<<"masukkan bilangan bulat ke "<<i<<" = ";
  cin>>a[i];
 }
 cout<<endl;
 //mencetak hasil
 for(i=0;i<=4;i++)
 {cout<<" bilangan ke : "<<i<<" = "<<a[i]<<endl;}
}
```

Pada program di atas dapat dilihat bagaimana cara menginputkan dan mencetak data yang telah diisikan ke dalam elemen-elemen larik.

B. Latihan

1. Ketikkan program berikut untuk mempelajari cara kerja dan efek penggunaan index pada larik

```
#include<iostream>
#include<conio.h>
using namespace std;
int a[10];

int main()
{
  int i=0;
//input data
 for(i=0;i<=4;i++)
 {cout<<"masukkan bilangan bulat ke "<<i<<" = ";
```

```
 cin>>a[i];
 }
 cout<<endl;
 //mencetak hasil
 for(i=0;i<=4;i++)
 {cout<<" bilangan ke : "<<i<<" = "<<a[i]<<endl;}
}
```

2. Program dibawah adalah program pengimputan n buah data kemudian dihitung jumlahnya dan dicari nilai rata-ratanya

```
#include<iostream>
#include<conio.h>
using namespace std;
int a[10];
int n,jum=0;
float mean;
main()
{
        int i=0;
        //input data
        cout<<"berapa data yang akan diinputkan : ";cin>>n;
        for(i=0;i<n;i++)
        {       cout<<"masukkan bilangan bulat ke "<<i+1<<" = ";
                cin>>a[i];
                jum=jum+a[i];
        }
         mean=jum/n;
        cout<<endl;
        //mencetak hasil
         for(i=0;i<n;i++)
        {       cout<<" bilangan ke : "<<i+1<<" = "<<a[i]<<endl;}
                 cout<<endl<<" rata-rata dari bilangan di atas ="<<mean;
        }
```

Pelajari mengapa hasilnya seperti itu? Perbaikilah program tersebut.

C. Tugas.

1. Buatlah program untuk menginputkan sejumlah bilangan, kemudian lakukan pencarian terhadap suatu bilangan, jika ketemu beri tahu posisi bilangan tersebut, jika tidak ketemu juga beri komentar. Anda bisa menggunakan variable bertipe boolen untuk menyimpan hasil pencarian.

2. Buatlah program untuk menentukan nilai tertinggi dan terendah dari data yang telah diinputkan ke dalam suatu array

MODUL KE-8
LARIK DIMENSI DUA

Kompetensi: Mahasiswa mampu mengenali larik dimensi dua, mampu mendeklarasikan dan menggunakan larik dimensi dua dalam menyelesaikan masalah dalam pemrograman

A. Pengertian Larik Dimensi Dua

Larik dimensi dua dapat dipahami sebagai larik dengan elemen baris dan kolom. Pemanfaatan array tidak hanya dapat digunakan untuk menyimpan data dalam bentuk satu dimensi tetapi juga dapat digunakan untuk menyimpan data dalam bentuk 2 dimensi. Misal ada data dalam bentuk representasi sebagai berikut:

Tahun/Prodi	2012	2013	2014	2015	2016
Informatika	60	65	78	100	98
Kimia	100	97	102	76	94
Geologi	120	130	132	142	140
Mesin	200	180	210	187	176

Pada table di atas, maka index baris digunakan untuk menyimpan data prodi dan index kolom digunakan untuk menyimpan data tahun. Misal a[3][2] artinya jumlah mahasiswa pada prodi geologi pada tahun 2013. Atau dalam merepresentasikan matrix berorde mxn sehingga index baris menggunakan m dan index kolom menggunakan n. Pada pengelolaan larik berdimensi dua digunakan dua index, yaitu index baris dan index kolom.

Cara mendeklarasikan larik dimensi dua adalah sebagai berikut:

Tipe data nama_variabel[jumlah baris][jumlah kolom]

Contoh:

int mhs[5][4];

float matrix[10][10];

Pada deklarasi di atas maka variable mhs mempunyai dimensi dua dengan maksimum baris 5 dan maksimum kolom 4. Sedangkan variable matrix memiliki maksimum orde baris = 10 dan kolom=10.

B. Cara memberikan nilai pada larik Dimensi Dua dan mencetaknya

Seperti halnya pada larik berdimensi satu, maka pengisian atau pencetakan data pada larik dimensi dua juga menggunakan loop. Akan tetapi karena index ada dua, yaitu baris dan kolom maka loop yang digunakan juga harus berupa loop bersarang (nested loop).

Struktur penulisan array dengan menggunakan perulangan adalah :

```
    Tipe data nama_array[jumlah array]
 //index untuk baris
    For (index awal;index akhir;increment index)
    {
//index untuk kolom nested di dalam index baris
     For (index awal;index akhir;increment index)
     {
          pengisian larik
          Manipulasi larik
          Pencetakan larik
      }
      }
```

C.Latihan

Latihan 1.

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

using namespace std;
int mat[3][3];


int main()
{
```

```
        int m,n;
        cout<<"masukkan baris matrix max 3: ";cin>>m;
        cout<<"masukkan kolom matrix max 3: ";cin>>n;
        //input elemen matrix
        for(int i=1;i<=m;i++)
        {
                for(int j=1;j<=n;j++)
                {
                cout<<"masukkan data matriks ke ["<<i<<","<<j<<"]= ";
                cin>>mat[i][j];
                }
        }
        //mencetak hasilnya
        cout<<"hasil matriks adalah sbb : "<<endl;
        for(int i=1;i<=m;i++)
        {
                for(int j=1;j<=n;j++)
                {
                cout<<mat[i][j]<<"   ";;
                }
                cout<<endl;
        }

}
```

Latihan 2. Jika ingin membuat validasi terhadap masukan orde untuk m dan m dapat ditambahkan sebagai berikut:

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

using namespace std;
int mat[3][3];


int main()
{
        int m,n;
        cout<<"masukkan baris matrix max 3: ";cin>>m;
        cout<<"masukkan kolom matrix max 3: ";cin>>n;
        if((m==3) || (n==3))
                {cout<<"orde tidak diijinkan"<<endl;}
        else
        {
        //input elemen matrix
                for(int i=1;i<=m;i++)
                {
                        for(int j=1;j<=n;j++)
```

```
                {
                cout<<"masukkan data matriks ke ["<<i<<","<<j<<"]= ";
                cin>>mat[i][j];
                }
        }
    //mencetak hasilnya
        cout<<"hasil matriks adalah sbb : "<<endl;
        for(int i=1;i<=m;i++)
        {
            for(int j=1;j<=n;j++)
            {
            cout<<mat[i][j]<<"   ";;
            }
            cout<<endl;
        }
    }
}
```

D. Tugas

1. Program diatas silahkan dilanjutkan dengan mendeklarasikan 3 buah matrix dan menghitung penjumlahan dan pengurangan matriks dan tampilkan hasilnya di layar. Lakukan validasi terhadap orde seperti pada contoh tetapi ditambahkan validasi bahwa m dan n tidak boleh melebihi dimensi yang ditetapkan.

2. Buatlah program untuk menginputkan data mahasiswa seperti pada tabel di atas

MODUL KE-9
FUNGSI DAN PROCEDURE

Kompetensi: Mahasiswa mampu mengenali dan membedakan fungsi dengan procedure. Mahasiswa mengetahui, mamahami dan menggunakan fungsi dan procedure dalam menyelesaikan masalah untuk mengoptimalkan pemrograman modular.

A. Pendahuluan

Pemrograman terstruktur mempunyai kekuatan untuk mengoptimalisasikan program yang dibangun dengan konsep modularitas. Modul adalah sub program yang dapat dipanggil sewaktu-waktu, baik oleh program utama main() maupun oleh modul-modul lainnya dengan hasil yang berbeda-beda sesuai parameter yang dikirimkan kepadanya. Misal kita memiliki modul input_data(x) dimana x menerima data inputan berupa string, maka akan melalukan simpana terhadap inputan data sembarang yang dimasukkan oleh user. Atau misalnya kita memiliki modul untuk menghitung luas lingkaran dengan parameter jari-jari, maka modul akan mengeluarkan hasil sesuai inputan jari-jari yang diberikan oleh user. Dalam pemrograman terstruktur, apabila akan menggunakan modul maka harus memahami konsep procedure, function, parameter dan berbagai jenis pengiriman parameter. Struktur modul dalam C++ secara umum adalah sebagai berikut:

```
Void NamaModul (DaftarParameter){
   /*Code atau Badan modul*/
}
```

Semua modul yang akan digunakan harus dibuat dan didefinisikan dalam program.

B. Procedure

Prosedur merupakan suatu program terpisah dalam blok sendiri yang berfungsi sebagai subprogram (program bagian). Prosedur biasanya bersifat suatu aktifitas seperti mencari

bilangan prima dari sekumpulan bilangan atau mencari bilangan genap dari sekumpulan bilangan ,dsb. Prosedur biasanya digunakan pada program yang terstruktur karena:

a. Merupakan penerapan konsep program modular, yaitu memecah-mecah program yang rumit menjadi program-program bagian yang lebih sederhana dalam bentuk prosedur-prosedur.
b. Untuk hal-hal yang sering dilakukan / dipakai berulang-ulang, cukup dituliskan sekali saja dalam bentuk prosedur dan dapat dipergunakan atau dipanggil berulangkali jika diperlukan.
c. Membuat kode program lebih mudah dibaca / dimengerti terutama oleh programmer lain.
d. Dapat digunakan untuk menyembunyikan detail program

Ciri-ciri Prosedur yang baik adalah sebagai berikut:

a. Hanya memiliki satu fungsi tujuan (logical inherent). Sebuah prosedur sebaiknya hanya mempunyai satu fungsi tujuan / hanya memecahkan sebuah masalah dalam program dan tidak bercampur dengan tujuan lain. Hal tersebut agar prosedur lebih focus sehingga tujuan dari sebuah prosedur lebih mundah dipahami.
b. Tidak tergantung pada prosedur lain (independent). Sebuah prosedur sebaiknya bersifat mandiri, artinya sebuah prosedur dapat dijalankan dan diuji tanpa menunggu bagian lainnya (prosedur / fungsi lain) selesai. Selain itu variable yang digunakan dalam prosedur tidak mempengaruhi variabel yang digunakan pada bagian lain di keseluruhan program (misalnya dengan menggunakan variabel lokal).
c. Berukuran kecil (small size). Yang dimaksud ukuran disini adalah panjang kode program atau panjang algoritma pada sebuah prosedur. Ukuran kecil pada sebuah prosedur agar prosedur lebih mudah dibaca, dipahami maupun diperbaiki jika terdapat kesalahan pada sebuah prosedur.

Deklarasi procedure dalam C++ adalah sebagai berikut:

```
Void NamaProcedure (DaftarParameter)
{
    /*Code atau Badan procedure*/
}
```

C. Fungsi

Fungsi merupakan suatu program terpisah dalam blok sendiri yang berfungsi sebagai subprogram (program bagian). Sama seperti halnya dengan prosedur, namun tetap ada perbedaannya yaitu fungsi mempunyai pengembalian nilai / mengembalikan sebuah nilai (memiliki return value) dari tipe tertentu (tipe dasar atau tipe bentukan).

Adapun cara mendeklarasikan fungsi dalam C++ adalah sebagai berikut:

```
TipeData NamaFungsi (DaftarParameter)
{
    /*Code atau Badan Fungsi*/
    return nilaireturn;
}
```

Jika diperhatikan maka dalam pendeklarasian fungsi terdapat tipe data fungsi dan nilai kembalian yang harus diberikan bergantung pada parameter yang dikirimkan. Kembalian fungsi adalah nama fungsi tersebut.

D. Perbedaan Fungsi dan Procedure

Perbedaan fungsi dengan prosedur

- Pada fungsi, tipe data nilai yang dikirimkan balik (return value) terdapat pada nama fungsinya, sedangkan prosedur tidak memiliki return value (misal return luas).
- Karena nilai balik berada di nama fungsi tersebut, maka fungsi tersebut dapat langsung digunakan untuk dicetak hasilnya. Atau nilai fungsi tersebut dapat juga langsung dipindahkan ke pengenal variable yang lainnya (misal luas1 = ContohFungsi(panjang)).

- Pada prosedur, nama prosedur tidak bisa digunakan lagsung tidak seperti pada Nama Fungsi, yang bisa langsung digunakan dari sebuah prosedur adalah parameternya yang mengandung nilai balik.
- Pada dasarnya tidak ada perbedaan yang signifikan antara fungsi dan prosedur pada Bahasa C / C++, hanya dibedakan dari return value. Jika suatu sub program tersebut memiliki nilai balik (return value) maka sub program tersebut disebut Fungsi (Function) namun jika tidak mempunyai nilai balik maka sub program tersebut adalah prosedur (Procedure).

Adapun alasan menggunakan fungsi dan procedure antara lain:

- Meningkatkan kemampuan untuk menganalisis kesalahan, jika terjadi suatu kesalahan kita tinggal mencari fungsi atau prosedur yang bersangkutan saja dan tak perlu di seluruh program.
- Modifikasi program dapat dilakukan pada suatu fungsi atau prosedur tertentu saja tanpa mengganggu program keseluruhan (fungsi / prosedur lain)
- Memecahkan program yang rumit dan besar menjadi program-program yang lebih sederhana atau kecil sehingga program lebih mudah dibaca dan mudah dipahami
- Dengan memecah program menjadi subprogram-subprogram yang lebih kecil, Program dapat dikerjakan oleh beberapa orang dengan pembagian beberapa subprogram (fungsi / prosedur) untuk tiap orang nya sehingga program cepat selesai dengan koordinasi yang mudah.
- Untuk aktivitas yang dilakukan lebih dari satu kali / sering dilakukan berulang-ulang. fungsi dan prosedur dapat digunakan untuk menghindari penulisan program yang sama yang ditulis secara berulang.
- Mempermudah dokumentasi.

D. Latihan

Llatihan 1 :

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

// Deklarasi Prosedur / Prototype Prosedur
```

```
// int a merupakan parameter formal
void ContohProsedur(int a);
void ContohProsedur2(int a);

// Fungsi Utama
int main(){
    int panjang = 5;

    cout<<"============================";
    cout<<"\n== Program Contoh Prosedur ==\n";
    cout<<"============================\n";

    // memanggil prosedur "ContohProsedur" dan "ContohProsedur2"
    // panjang merupakan parameter aktual
    ContohProsedur(panjang);
    ContohProsedur2(panjang);

    return 0;
}

// Contoh Prosedur
// dimisalkan int panjang sebagai parameter input


void ContohProsedur(int panjang)
{
    int lebar, luas;

    cout<<"\n\nMasukkan Lebar Persegi Panjang keI : ";cin>>lebar;

    luas=panjang*lebar;

    cout<<"Luas Persegi Panjang keI = "<<panjang<<" x "<<lebar<<" =
"<<luas<<endl;
}

// Contoh Prosedur
// dimisalkan int panjang sebagai parameter input
void ContohProsedur2(int panjang){
    int lebar, keliling;

    cout<<"\n\nMasukkan Lebar Persegi Panjang keII : ";cin>>lebar;

    keliling=(panjang+lebar)*2;

    cout<<"Keliling Persegi Panjang keII = ("<<panjang<<" + "<<lebar<<") x 2 =
"<<keliling<<endl;
}
```

Berikut adalah latihan untuk memperlihatkan contoh fungsi **(latihan2)** :

```
#include <iostream>

using namespace std;

// Deklarasi Fungsi / Prototype Fungsi
// int a merupakan parameter formal
int ContohFungsi(int a);

// Fungsi Utama
int main(){
    int luas1, luas2, totalluas;
    int panjang = 5;

    cout<<"============================";
    cout<<"\n== Program Contoh Fungsi ==\n";
    cout<<"============================\n";

    // memanggil fungsi ContohFungsi
    // panjang merupakan parameter aktual
    luas1 = ContohFungsi(panjang);
    luas2 = ContohFungsi(panjang);
    totalluas = luas1 + luas2;

    cout<<"\n\nLuas Gabungan Kedua Persegi Panjang adalah =
"<<totalluas<<endl;

    return 0;
}

// Contoh Fungsi
// dimisalkan int panjang sebagai parameter input
int ContohFungsi(int panjang){
    int lebar, luas;

    cout<<"\n\nMasukkan Lebar Persegi Panjang : ";cin>>lebar;

    luas=panjang*lebar;

    cout<<"Luas Persegi Panjang adalah "<<panjang<<" x "<<lebar<<" = "<<luas;

    return luas;
}
```

Dari kedua contoh di atas dapat dilihat bahwa procedure tidak mempunyai nilai kembalian yang berupa return, sementara fungsi mengembalikan satu keluaran, yaitu luas sedangkan apabila fungsi tidak menghasilkan suatu keluaran maka disebutkan return 0.

E. Parameter

Parameter adalah variable yang dikirimkan oleh pemanggil dan diterima oleh procedure atau fungsi.

Terdapat dua macam para parameter fungsi, yaitu :

1. Parameter formal adalah variabel yang ada pada daftar parameter dalam definisi fungsi.
2. Parameter Aktual adalah variabel yang dipakai dalam pemanggilan fungsi.

Bentuk penulisan Parameter Formal dan Parameter Aktual.

1. Pemanggilan dengan nilai (Call by Value)

   Pemanggilan dengan nilai merupakan cara yang dipakai untuk seluruh fungsi buatan yang telah dibahas didepan. Pada pemanggilan dengan nilai, nilai dari parameter aktual akan ditulis keparameter formal. Dengan cara ini nilai parameter aktual tidak bisa berubah, walaupun nilai parameter formal berubah.

   Contoh **(Latihan 3):**

```
/* ———————————— */
/* Penggunaan Call By Value */
/* Program Pertukaran Nilai */
/* ———————————— */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
tukar(int x, int y);
main()
{
int a, b;
a = 88; b = 77;
clrscr();
cout<<”Nilai Sebelum Pemanggilan Fungsi”;
cout<<”\na = “<<a<<” b = “<<b;
tukar(a,b);
```

```
cout<<"\nNilai Setelah Pemanggilan Fungsi";
cout<<"\na = "<<a<<" b = "<<b;
getch();
} tukar(int x, int y)
{
int z;
z = x; x = y; y = z;
cout<<"\n\nNilai di dalam Fungsi Tukar()";
cout<<"\nx = "<<x<<" y = "<<y;
cout<<endl;
}
```

1. Pemanggilan dengan Referensi (Call by Reference)

   Pemanggilan dengan reference merupakan upaya untuk melewatkan alamat dari suatu variabel kedalam fungsi. Cara ini dapat dipakai untuk mengubah isi suatu variabel diluar fungsi dengan melaksanakan pengubahan dilakukan didalam fungsi.

Contoh **(latihan 4)** :

```
/* Penggunaan Call By Reference */
/* Program Pertukaran Nilai */
/* ———————————- */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
tukar(int *x, int *y);
main()
{
int a, b;
a = 88; b = 77;
clrscr();
cout<<"Nilai Sebelum Pemanggilan Fungsi";
cout<<"\na = "<<a<<" b = "<<b;
tukar(&a,&b);
```

```
cout<<endl;
cout<<"\nNilai Setelah Pemanggilan Fungsi";
cout<<"\na = "<<a<<" b = "<<b;
getch();
}
tukar(int *x, int *y)
{
int z;
z = *x; *x = *y; *y = z;
cout<<endl;
cout<<"\nNilai di Akhir Fungsi Tukar()";
cout<<"\nx = "<<*x<<" y = "<<*y;
}
```

E. Pengiriman Data Ke Fungsi

1. Pengiriman Data Konstanta Ke Fungsi

Mengirimkan suatu nilai data konstanta ke suatu fungsi yang lain dapat dilakukan dengan cara yang mudah, dapat dilihat dari program berikut :

Contoh **(latihan 5):**

```
/* Pengriman data Konstanta */
/* ——————————— */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
luas(float sisi);
main()
{
float luas_bs;
clrscr();
luas_bs = luas(4.25);
cout<<"\nLuas Bujur Sangkar = "<<luas_bs;
getch();
```

```
} luas(float sisi)
{
return(sisi*sisi);
}
```

1. Pengiriman Data Variabel Ke Fungsi

Bentuk pengiriman data Variabel, sama seperti halnya pengiriman suatu nilai data konstanta ke suatu fungsi, hanya saja nilai yang dikirimkan tersebut senantiasa dapat berubah-ubah. Bentuk pengiriman tersebut dapat dilihat dari program berikut:

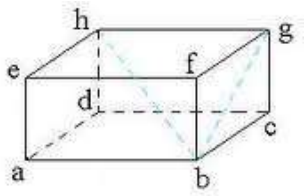
Contoh **(latihan 6)**

```
/* ——————————— */
/* Pengriman data Konstanta */
/* ——————————— */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
luas(float sisi);
main()
{
float luas_bs, sisi_bs;
clrscr();
cout<<"\nMenghitung Luas Bujur Sangkar"<<endl;
cout<<"\nMasukan Nilai Sisi Bujur Sangkar : ";
cin>>sisi_bs;
luas_bs = luas(sisi_bs);
cout<<"\nLuas Bujur Sangkar = "<<luas_bs<<" Cm";
getch();
} luas(float sisi)
{
return(sisi*sisi);
}
```

F. Tugas

1. Buatlah Program dengan menggunakan procedure untuk kasus berikut:

Balok adalah bangun ruang yang dibatasi oleh enam bidang yang berbentuk persegi panjang dan sepasang-sepasang kongruen.



Keterangan :
p = panjang balok
l =lebar balok
t = tinggi balok

Luas balok:

$L = 2 (p.l + p.t + l.t)$

Volume balok:

$V = p \times l \times t$

Buat program mencari luas dan volume dengan menggunakan fungsi, nilai p.l dan t harus dimasukan lewat keyboard dan dari program di atas, modifikasi agar program bisa melakukan perkalian antara Luas balok dengan Volume balok (proses perkalian di lakukan di program utama (main)

2. Buatlah program penghitung sederhana yang terdiri dari beberapa fungsi dan procedure yang bias mengelola proses: 1. Mencari Luas dan Keliling Lingkaran 2. Menghitung luas dan keliling segitiga siku-siku 3. Menghitung luas dan Volume balok 4. Menghitung Volume dan Luas permukaan silinder. Gunakan SWITCH dan fungsi atau procedure

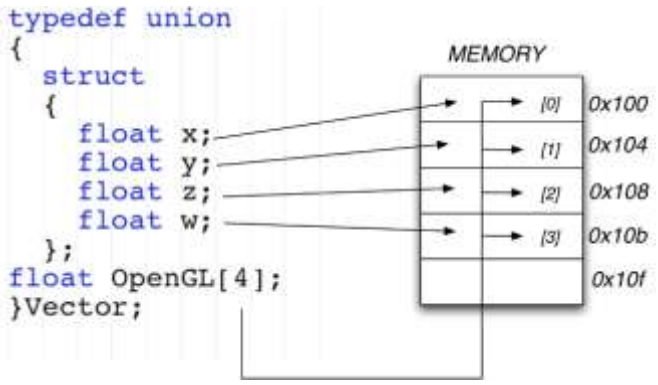
MODUL KE-10
RECORD dengan STRUCT

Kompetensi: Mahasiswa mampu mengenali record dan record array Mahasiswa mengetahui, mamahami dan menggunakan record dan record array dalam menyelesaikan masalah untuk mengoptimalkan pemrograman modular sederhana.

A. Record

Dalam kehidupan sehati-hari penggunaan tunggal missal nama, alamat, program studi, umur dan lain-lain jarang ditemukan dalam kondisi berdiri sendiri. Sebagai gambaran, jika akan menyimpan data mahasiswa maka akan disimpan nim, nama, prodi, alamat, semester, sehingga semua atribut tersebut merupakan satu kesatuan. Atau missal pada pendataan kependudukan maka nik, nama, alamat, status merupakan satu kesatuan. Dalam bahasa pemrograman konsep tersebut disebut record atau rekaman. Dalam bahasa C++ disebut struct. Variabel yang memiliki tipe struct akan memiliki beberapa elemen dengan tipe yang berbeda-beda. Apabila data rekaman berjumlah lebih dari satu maka akan menjadi larik record.

Struct atau record adalah kumpulan data yang memiliki tipe data yang berbeda. Secara pendeklarasian, struct sangat berbeda dengan array yang hanya memiliki satu buah tipe data untuk setiap kumpulannya. Struct digunakan apabila data yang ingin dikelompokkan memiliki tipe data yang berbeda.



Pendeklarasian struct sebagai berikut:

```
struct nama_struktur {
elemen1;
elemen2;
};
```

atau

```
typedef struct {
elemen1;
elemen2;
} nama_struct;
```

Elemen1, elemen2 dan seterusnya menyatakan nama atribut, dan tipenya. Contoh pendeklarasian:

```
stuct data_mahasiswa
{
  long int nim;
  char nama[100];
  char fakultas[100];
};
data_mahasiswa mahasiswa1, mahasiswa2;
```

Dari contoh deklarasi maka type struct data_mahasiswa mempunyai atribut nim, nama dan fakultas. Variabel mahasiswa1 dan mahasiswa2 mempunyai tipe data_mahasiswa. Untuk dapat menggunakan tipe data tersebut sebuah variabel harus dideklarasikan menggunakan nama struct nya. Bentuk umum pendeklarasian variabel struct nya adalah sebagai berikut:

struct data_mahasiswa mahasiswa1, mahasiswa2;

Deklarasi dua variabel di atas ada dua variabel bernama Mahasiswa1 dan Mahasiswa2 setiap variabel tersebut mempunyai field sesuai dengan data_mahasiswa.

Selain dekalrasi variabel ada hal yang harus diperhatikan yaitu cara untuk mengisi dan memanggil nilai yang ada di dalam sebuah struct, yaitu sebagai berikut:

Latihan 1. Berikut adalah program untuk mengisi dan mencetak data mahasiswa dalam record data_mahasiswa

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

using namespace std;

//Deklarasi struct
struct data_mahasiswa
{
   long int nim;
   char nama[100];
   char fakultas[100];
};

//Deklarasi variabel struct
data_mahasiswa mahasiswa1, mahasiswa2;

int main()
{
   //Input struct data mahasiswa
   cout<<" Data Mahasiswa Pertama\n";
   cout<<"----------------------\n";
   cout<<" NIM      : "; cin>>mahasiswa1.nim;
   cout<<" Nama     : "; fflush(stdin); gets(mahasiswa1.nama);
   cout<<" Fakultas : "; fflush(stdin); gets(mahasiswa1.fakultas);
   cout<<"\n\n";
   cout<<" Data Mahasiswa Kedua\n";
   cout<<"----------------------\n";
   cout<<" NIM      : "; cin>>mahasiswa2.nim;
   cout<<" Nama     : "; fflush(stdin); gets(mahasiswa2.nama);
   cout<<" Fakultas : "; fflush(stdin); gets(mahasiswa2.fakultas);
   cout<<"\n\n";

   //Output struct data mahasiswa
   cout<<" Data Mahasiswa Pertama\n";
   cout<<"----------------------\n";
```

```
    cout<<" NIM      : "<<mahasiswa1.nim<<endl;
    cout<<" Nama     : "<<mahasiswa1.nama<<endl;
    cout<<" Fakultas : "<<mahasiswa1.fakultas<<endl;
    cout<<"\n\n";
    cout<<" Data Mahasiswa Kedua\n";
    cout<<"----------------------\n";
    cout<<" NIM      : "<<mahasiswa2.nim<<endl;
    cout<<" Nama     : "<<mahasiswa2.nama<<endl;
    cout<<" Fakultas : "<<mahasiswa2.fakultas<<endl;
    cout<<"\n\n";
    getch();
}
```

B. Larik Record

Larik record adalah record-record yang sama yang memiliki atribut yang sama dengan isian yang berbeda-beda. Jika kita memilki banyak mahasiswa maka harus digunakan larik struct mahasiswa. Pada contoh di atas kita mempunyai 2 mahasiswa maka dideklarasikan variable mahasiswa1 dan mahasiswa2. Bagaimana jika mahasiswa teris bertambah? Maka larik record adalah solusinya.

Deklarasi larik record dalam C++ adalah sebagai berikut:

```
stuct data_mahasiswa
{
  long int nim;
  char nama[100];
  char fakultas[100];
};
data_mahasiswa mahasiswa[10];
```

maka kita akan memiliki variable mahasiswa dengan tipe data_mahasiswa sebanyak maksimum 10 elemen yang memiliki atribut sama. Cara manipulasi larik struct pada C++ sama dengan cara mengelola array dengan memberikan index pada variable dan diikuti dengan atributnya. Penginputan, proses dan pencetakan dapat dilakukan dengan loop.

C. Latihan 2.

Membuat program struct untuk memasukkan 3 user lalu menyimpan data loginnya. Yang harus disimpan adalah waktu login (tanggal, bulan, tahun, jam, menit dan detik).

Code :

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

using namespace std;
struct Date{
        int dd;
        int mm;
        int yyyy;
       };
struct Time{
        int h;
        int m;
        int s;
       };
struct Login{
        int ID;
        Date tglLogin;
        Time waktuLogin;
       };

int main()
{
   Login user[3];

   for(int i=0;i<3;i++)
   {
   cout<<"\nUser ke-"<<i+1<<endl;
   cout<<"ID : ";cin>>user[i].ID;
   cout<<"\nTanggal login\n";
   cout<<"Hari  : ";cin>>user[i].tglLogin.dd;
   cout<<"Bulan : ";cin>>user[i].tglLogin.mm;
   cout<<"Tahun : ";cin>>user[i].tglLogin.yyyy;
   cout<<"\nWaktu Login\n";
   cout<<"Jam   : ";cin>>user[i].waktuLogin.h;
```

```
    cout<<"Menit : ";cin>>user[i].waktuLogin.m;
    cout<<"Detik : ";cin>>user[i].waktuLogin.s;
    cout<<"\nTerimakasih Atas Pengisiannya\n";

    cout<<"\nData User ke-"<<i+1<<endl;
    cout<<"Login ID   : "<<user[i].ID<<endl;
    cout<<"Login Date : "<<user[i].tglLogin.dd<<"-"<<user[i].tglLogin.mm<<"-
"<<user[i].tglLogin.yyyy<<endl;
    cout<<"Login Time : "<<user[i].waktuLogin.h<<"-
"<<user[i].waktuLogin.m<<"-"<<user[i].waktuLogin.s<<endl;

    }
    getch();
   return 0;
}
```

Latihan 3.

Program Sederhana :

```
#include <iostream>
#include <iomanip>

using namespace std;

struct barang {
string kode;
string nama_brg;
long harga_jual;
int stock;
    };

struct penjualan {
    string nama_pmbeli;
    string kode_brg;
    string nama_brg;
    long jumlah_bayar;
};

int main() {
barang brg[5];
penjualan jual;
```

```
int n,jumlah;
string nama,nama_brg;
char ya='y';
n=1;
cout<<"data pengelolaan"<<endl;
while (ya=='y')
{
    cout<<"Masukkan Kode barang   : ";cin>>brg[n].kode;
    cout<<"Masukkan Nama barang   : ";cin>>brg[n].nama_brg;
    cout<<"Masukkan harga barang  : ";cin>>brg[n].harga_jual;
    cout<<"Masukkan stock barang  : ";cin>>brg[n].stock;
    cout<<"mau memasukkan data lagi : ";cin>>ya;
    n++;
}
    cout<<endl;
    cout<<"Penjualan barang"<<endl;
    cout<<"masukkan nama pembeli : ";cin>>nama;
    cout<<"masukkan nama barang  : ";cin>>nama_brg;
    for (int i=1;i<n;i++)
    {
    if (brg[i].nama_brg==nama_brg)
    {
    cout<<"Masukkan jumlah barang yang dibeli : ";cin>>jumlah;
    jual.nama_pmbeli=nama;
    jual.nama_brg=brg[i].nama_brg;
    jual.jumlah_bayar=brg[i].harga_jual*jumlah;
    }

    }
    cout<<endl;
    cout<<"Nama pembeli       : "<<jual.nama_pmbeli<<endl;
    cout<<"Nama barang        : "<<jual.nama_brg<<endl;
    cout<<"Jumlah yang dibeli : "<<jual.jumlah_bayar<<endl;

    }
```

D. Tugas

Buat program yang dapat menghitung kelulusan mahasiswa
Dengan seperti dibawah. Nama dan nilai diinputkan sedangkan keterangan ditentukan dengan syarat keluluran minimum nilai 70.

No	Nama	nilai	keterangan
1	Dodi	90	lulus
2	Agus	90	lulus
3	Ardi	80	lulus
4	Ramul	45	gagal

Ketentuan :

Input dimasukan dari keyboard
Syarat lulus ika nilai di atas 70
Gunakan record dan array untuk menyimpan no, nama, nilai dan keterangan

Catatan

Catatan

ISBN 978-602-7619-40-1