PEMROGRAMAN C++

Oleh:

Rosihan Ari Yuana, S.Si, M.Kom
arie@uns.ac.id

BAB I. PENDAHULUAN C++

A. C++ dan C

Berbicara tentang C++ dan C sebagai bahasa pendahulunya, C merupakan termasuk bahasa pemrograma n tingkat menengah. Pencipta C adalah Brian W. Kernighan dan Dennis M. Ritchie pada tahun 1972. C merupakan bahasa pemrograman terstruktur yang membagi program ke dalam sejumlah blok (sub program). Tujuannya adalah untuk memudahkan dalam pembuatan dan pengembangan program. Program yang ditulis dengan C mudah sekali dipindahkan dari satu jenis mesin ke mesin lain. Hal ini karena adanya standarisasi C yaitu ANSI (American National Standards Institute) yang menjadi acuan para pembuat compiler C.

C++ diciptakan satu dekade setelah C. C++ diciptakan oleh Bjarne Stroustroup dari Laboratorium Bell, AT&T pada tahun 1983. Pada awalnya C++ diberi nama "A better C". Nama C++ sendiri diberinama oleh Rick Mascitti. Adapun tanda ++ berasal dari operator increment pada bahasa C.

Keistimewaan C++ adalah karena bahasa ini mendukung OOP (Object Oriented Programming). Tujuan utama pembuatan C++ adalah untuk meningkatkan produktivitas pemrogram dalam membuat aplikasi. Kebanyakan pakar setuju bahwa OOP dan C++ mampu mengurangi kompleksitas terutama program yang terdiri dari 10.000 baris lebih, bahkan dapat meningkatkan produktivitas 2x lipat dari C, Pascal dan Basic.

B. Object Oriented Programming (OOP)

Ide dasar OOP adalah mengkombinasikan data dan fungsi untuk mengakses data menjadi sebuah kesatuan unit. Unit ini dikenal dengan obyek. Sebagai gambaran untuk mempurmudah memahaminya, obyek sebenarnya dapat mencerminkan pola kerja manusia sehari-hari. Sebuah obyek dapat diibaratkan sebagai departemen di dalam sebuah perusahaan bisnis, misalnya departemen

- penjualan
- akunting
- personalia

Pembagian departemen dalam perusahaan merupakan upaya untuk memudahkan pengoperasian perusahaan. Sebagai gambaran, jika Anda seorang manajer penjualan di kantor pusat ingin mengetahui data para salesmen di kantor cabang, apa yang Anda lakukan? Langkah yang Anda tempuh pasti bukan datang ke kantor cabang dan mencari data-data tersebut. Untuk memudahkan tugas Anda cukup Anda menyuruh sekretaris untuk meminta informasi. Masalah bagaimana dan siapa yang mencarikan bukanlah urusan Anda. Analogi dengan hal itu, kalau seseorang bermaksud menggunakan obyek, ia cukup mengirim pesan ke obyek dan obyek itu sendiri yang akan menanganinya.

C. Program C++

Program C++ dapat dibuat menggunakan sebarang editor teks maupun editor sekaligus compilernya. Program utama berekstensi (.CPP). Pada saat kompilasi program utama bersama dengan file header (.h) akan diterjemahkan oleh compiler menjadi file obyek (.OBJ). Selanjutnya file obyek ini bersama-sama dengan file obyek lain dan file library (.LIB) dikaitkan menjadi satu oleh linker. Hasilnya adalah file (.EXE) executable.

D. Compiler C++

Compiler C++ yang telah beredar di pasaran antara lain Microsoft C/C++ dan Visual C++. Keduanya dari Microsoft. Sementara Borland international juga mengeluarkan Turbo C++ dan Borland C++.

BAB II. PENGENALAN PROGRAM C++

A. Hello World

Berikut ini contoh program C++ yang sederhana

```
#include <iostream.h>

void main()
{
      cout << "Hello world.\n";
}
```

Setelah dicompile dan dirun, hasilnya adalah muncul pada layar Hello World.

B. Fungsi main()

Program C++ memang tidak akan pernah lepas dari suatu fungsi/function. Hal ini karena merupakan ciri OOP. Sebuah program C++ minimal memiliki satu fungsi yaitu main(). Fungsi ini merupakan awal program utama. Tulisan main() merupakan nama fungsi, sedangkan bagian yang diapit dengan { dan } disebut blok (tubuh fungsi). Dalam hal ini { merupakan tanda awal blok dan } adalah tanda akhir blok. Seperti halnya dalam Pascal, { dalam Pascal identik dengan BEGIN, sedangkan } identik dengan END. Perintah void bermakna bahwa fungsi main() tidak mengembalikan nilai/value.

Cara penulisan fungsi main() tidak mutlak seperti di atas. Berikut ini cara penulisan yang lain

```
#include <iostream.h>

int main()
{
      cout << "Hello world.\n";
      return 0;
}
```

C. Statement

Perintah `cout << "Hello world.\n";` merupakan salah satu contoh statement. Perintah tersebut digunakan untuk mencetak tulisan pada layar. Setiap statement harus diakhiri dengan ;

Hal yang menjadi catatan penting di sini bahwa program C++ bersifat Case Sensitive, artinya huruf besar dan kecil dianggap beda.

Tanda \n digunakan untuk pindah baris.

D. File Header

Pada contoh di atas, iostream.h disebut file header. File header tersebut diperlukan agar perintah cout bisa dijalankan. Apabila file header tersebut dihapus, maka akan terjadi error. Untuk mengakses file header, digunakan perintah `#include <file header>`, atau `#include "file header"`.
Dalam suatu program bisa jadi melibatkan lebih dari satu file header.

Catatan:

Perintah `cout << "Hello world.\n";`
dapat diganti dengan `printf("Hello world.\n");`

Akan tetapi untuk bisa menggunakan printf diperlukan file header **stdio.h**

E. Menghapus Layar

Dalam C++, perintah clrscr() ; digunakan untuk menghapus/membersihkan layar. Perintah ini akan bisa dijalankan setelah ditambahkan file header conio.h

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

void main()
{
      clrscr();
      cout << "Hello world.\n";
}
```

F. Komentar

Anda dapat menambahkan komentar pada program Anda. Berikut ini style untuk menambah komentar.

```
// ---------------------------------
// ini adalah komentarku yang pertama
// ---------------------------------
```

atau

```
/* ---------------------------------
   ini adalah komentarku yang pertama
   --------------------------------- */
```

G. Latihan

1. Buatlah program seperti di bawah ini

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

void main()
{
      clrscr();
      cout << "It is my first C++ program\n";
      cout << "I am sure that I will be familiar with this";
      cout << "-------------------------\n";
}
```

   Simpan dengan nama file PROGRAM1.CPP

   a. Ubahlah salah satu huruf menjadi huruf besar pada perintah include. Compilelah, apa hasilnya?
   b. Lakukan hal yang sama untuk conio, iostream, void, main, clrscr, cout, \n.
   c. Tarik kesimpulan berdasarkan a dan b.

2. Dengan menggunakan printah cout atau printf, buatlah program C++ untuk menampilkan 3 huruf terdepan nama Anda. Misal nama Anda AGUS, maka buatlah tampilan seperti berikut:

```
AAAAAAAAA    GGGGGGGGGG    UU      UU
AAAAAAAAA    GGGGGGGGGG    UU      UU
AA     AA    GG            UU      UU
AA     AA    GG  GGGGGG    UU      UU
AAAAAAAAA    GG  GGGGGG    UU      UU
AA     AA    GG      GG    UU      UU
AA     AA    GGGGGGGGGG    UUUUUUUUUU
AA     AA    GGGGGGGGGG    UUUUUUUUUU
```

   Berilah sebarang komentar pada setiap baris statement.

   Simpan dengan nama NAMAKU.CPP

BAB III. ELEMEN DASAR C++

A. Identifier (Pengenal)

Pengenal adalah suatu nama yang biasa dipakai dalam pemrograman untuk menyatakan variabel, konstanta, tipe data, dan fungsi.

Aturan untuk penulisan identifier sama dengan aturan dalam pascal, antara lain:
- Tidak boleh dimulai dengan karakter non huruf
- Tidak boleh ada spasi
- Tidak boleh menggunakan karakter-karakter
  ~ ! @ # $ % ^ & * ( ) + ` - = { } [ ] : " ; ' < > ? , . / |
- Tidak boleh menggunakan reserved words yang ada dalam C++.

B. Tipe Data

Berikut ini tipe data yang ada dalam C++

Tipe data bilangan bulat:
- char
- int (integer)
- short (short integer)
- long (long integer)

Tipe data bilangan real:
- float (real)
- double (real double)
- long double

Selain itu terdapat juga tipe data unsigned

Tipe data bilangan bulat:
- unsigned char
- unsigned int (integer)
- unsigned short (short integer)
- unsigned long (long integer)

Tipe data unsigned mirip dengan yang bukan unsigned. Bedanya adalah tipe data unsigned tidak mengenal bilangan negatif (nilainya selalu posisif).

C. Deklarasi Variabel

Seperti halnya Pascal, variabel yang digunakan dalam program harus dideklarasikan terlebih dahulu. Pengertian deklarasi di sini yaitu mengenalkan variabel ke program dan menentukan tipe datanya.

Berikut ini contoh pendeklarasian variabel:

```
int jumlah;
float harga_satuan, variabel1;
char saya, kamu;
```

D. Assignment

Proses assignment adalah proses pemberian nilai kepada suatu variabel yang telah dideklarasikan.
Berikut adalah contoh assignment:

```
Jumlah = 10;
Harga_satuan = 23.456;
Saya = 'B';
Kamu = '2';
```

Berikut ini contoh program yang menggambarkan deklarasi variabel dan assignment.

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

void main()
{
int var1, var2, var3;
char karakter;

var1 = 10;
var2 = 5;
var3 = var1 + var2;

karakter = 'D';

cout << "Nilai var3 = " << var3 << "\n";
cout << "Nilai karakter = " << karakter;
getch();
}
```

contoh berikutnya yang melibatkan tipe data real dan memformat tampilan presisi.

```
#include "conio.h";
#include "iostream.h";
#include "iomanip.h";

void main()
{
clrscr();
double real;

real = 182.2182713674821746;
cout << setprecision(12);
cout << "Nilai real = " << real;
getch();
}
```

E. Konstanta

Untuk pendeklarasian konstanta dalam C++ mirip dengan Pascal. Sintaksnya adalah:

```
const tipe_data nama_konstanta = value;
```

Contoh:

```
const float phi = 3.141592;
```

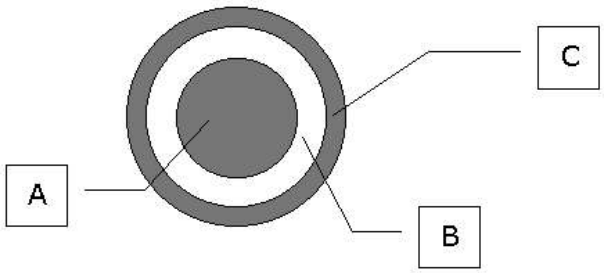
Berikut ini contoh program C++ untuk mencari luas dan keliling lingkaran dengan jari-jari 7.2;

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

void main()
{
const float phi = 3.141592;
float jari_jari, keliling, luas;

jari_jari = 7.2;
luas = phi * jari_jari * jari_jari;
keliling = 2 * phi * jari_jari;

cout << setpresicion(5);
cout << "Luas lingkaran adalah " << luas << "satuan luas \n";
cout << "Keliling lingkaran adalah " << keliling << "satuan
        panjang \n";
getch();
}
```

F. Latihan

1. Buatlah program C++ untuk mencari rata-rata 5 buah bilangan 34, 56, 91, 11, 22!

2. Diketahui 3 buah lingkaran dengan posisi saling menindih seperti pada gambar.



   Jari-jari lingkaran A adalah 10 cm, jari-jari lingkaran B adalah 12 cm, dan C adalah 14 cm. Dengan menggunakan program C++ hitunglah luas bagian yang diarsir.

3. Suatu ember berbentuk tabung dengan tutupnya terbuka berisi air penuh. Jari-jari alas ember adalah 10.5 cm, dan tingginya 5 cm. Kemudian sebuah kerucut dengan jari-jari alas yang berbentuk lingkaran adalah 4 cm dan tingginya 4.7 cm dimasukkan ke dalam ember. Akibatnya sebagian air dalam ember tumpah.
   Dengan menggunakan program C++ hitunglah berapa **liter** air yang tumpah?

BAB IV. OPERATOR DAN STATEMENT

A. Pengantar Operator

Operator merupakan simbol yang biasa dilibatkan dalam program untuk melakukan suatu operasi atau manipulasi, misalnya untuk:
- menjumlahkan dua nilai
- memberikan nilai ke suatu variabel (assignment)
- membandingkan kesamaan dua nilai.

B. Operator Aritmatika

Operator ini digunakan untuk perhitungan dasar aritmatika. Operator ini antara lain

Operator	Keterangan	Contoh
*	Perkalian	2*3
/	Pembagian	7/2
%	Modulo	7%2
+	Penjumlahan	5+4
-	pengurangan	5-4

C. Tingkat Presedensi Operator Aritmatika

Operator yang mempunyai prioritas tinggi akan diutamakan dalam hal pengerjaan dibandingkan dengan operator yang memiliki prioritas lebih rendah. Berikut ini tingkat presedensi operator aritmatika (semakin ke bawah prioritas makin rendah):

- (operator unary negatif)
* / %
+ -

D. Assignment Lanjut

Assignment merupakan proses pemberian nilai pada suatu variabel. Berikut ini contoh-contohnya:

```
a = 1;
a = 2 + b;
a = 2 + (b = 1);
```

Contoh no. 3 di atas prosesnya adalah mula-mula b diberi nilai 1, kemudian variabel a diisi dengan nilai penjumlahan 2 dan 1.

```
a = b = c = d = e = 1;
```

Contoh tersebut identik dengan:

```
e = 1;
d = e;
c = d;
b = c;
a = b;
```

E. Operator Increment dan Decrement

Bahasa C++ menyediakan operator yang disebut increment dan decrement. Operator ini digunakan untuk menaikkan atau menurunkan nilai suatu variabel sebesar 1.

Operator	Keterangan
++	Operator increment
--	Operator decrement

Penempatan operator tersebut dapat di awal variabel atau di belakangnya.
Contoh:

```
x = x + 1;
y = y – 1;
```

dapat ditulis

```
++x;
--y;
```

atau

```
x++;
y--;
```

Secara sekilas tak ada perbedaan antara ++x dan x++ atau --y dan y—
Perhatikan contoh berikut ini:

```
r = 10;
s = 10 + r++;
cout << "Nilai r = "<< r << "\n";
cout << "Nilai s = "<< s << "\n";
```

bandingkan dengan

```
r = 10;
s = 10 + ++r;
cout << "Nilai r = "<< r << "\n";
cout << "Nilai s = "<< s << "\n";
```

F. Operator Majemuk

C++ menyediakan operator yang dimaksudkan untuk memendekkan penulisan operasi assignment, misalnya:

```
x = x + 2;
```

```
y = y * 4;
z = z / 5;
w = w – 8;
```

dapat ditulis

```
x += 2;
y *= 4;
z /= 5;
w -= 8;
```

G. Operator Relasional

Operator ini digunakan untuk membandingkan dua buah nilai. Berikut ini macam-macam operator yang termasuk jenis ini.

Operator	Keterangan
==	Sama dengan (bukan assignment)
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan

Hasil operasi relasi ini dihasilkan nilai benar atau salah.

Contoh:

```
nilai1 = 3 > 2;
nilai2 = 15 == 16;
cout << "Nilai1 = "<< nilai1 << "\n";
cout << "Nilai2  = "<< nilai2 << "\n";
```

Hasil program di atas akan menampilkan nilai1 adalah 1 (benar) dan nilai2 adalah 0 (salah).

H. Operator Logika

Operator ini digunakan untuk menghubungkan 2 atau lebih statement. Biasanya statement yang dihubungkan merupakan operasi relasional. Operator logika juga menghasilkan nilai logika benar atau salah. Macam-macamnya adalah:

Operator	Keterangan
&&	AND
||	OR
!	NOT (negasi)

Contoh:

```
nilai1 = (3 > 2) && (4 < 10);
nilai2 = !(15 == 15);
cout << "Nilai1 = "<< nilai1 << "\n";
cout << "Nilai2  = "<< nilai2 << "\n";
```

Setelah di run, dihasilkan nilai1 adalah 1 (benar) dan nilai2 adalah 0 (salah)

I. Fungsi-fungsi Matematika

C++ menyediakan beberapa fungsi khusus untuk perhitungan matematika. Fungsi-fungsi ini memerlukan file header **math.h**

Fungsi	**Keterangan**
abs(x)	Mencari nilai mutlak
cos(x), sin(x), tan(x)	Mencari nilai cos, sin, tan (x dalam radian)
exp(x)	Mencari nilai e^x
log(x)	Mencari nilai log
pow(x,y)	Mencari nilai x^y
sqrt(x)	Mencari nilai akar kuadrat dari x

BAB V. Input dan Output

A. Cin

Dalam C++, perintah cin digunakan untuk menginput suatu nilai dari suatu piranti masukan (keyboard) untuk selanjutnya diproses oleh program.

Sintaknya adalah:

cin >> variabel;

contohnya:

```
cout << "Masukkan suatu bilangan : ";
cin >> bil;
cout << "Anda memasukkan bilangan " << bil "\n";
```

B. getch()

Perintah getch() berfungsi sama seperti cin (perintah input), akan tetapi getch() khusus untuk input berupa karakter. Disamping itu getch() dapat membaca input berupa spasi atau tab, sedangkan cin tidak bisa.

Sintaksnya:

Variabel = getch();

Fungsi ini juga dapat digunakan apabila tidak diinginkan penekanan ENTER ketika input data karakter.

C. getche()

Perintah getche() kegunaannya sama dengan getch(), bedanya adalah:
- getch() tidak menampilkan karakter yang diiinput
- getche() menampilkan karakter yang diinput

fungsi getch() dan getche() sama-sama membutuhkan file header **conio.h**

Berikut ini contoh penggunaan getch() dan getche()

```
char karakter;

cout << "masukkan sebuah karakter : ";
karakter = getch();
cout << "Anda mengetik karakter : " << karakter;

cout << "masukkan sebuah karakter : ";
karakter = getche();
cout << "Anda mengetik karakter : " << karakter;
```

NB:
Untuk input berupa string, akan dibahas di bab yang lain.

D. cout

Dalam c++, perintah cout digunakan untuk menampilkan suatu informasi ke piranti output (layar). Contoh-contoh penggunaannya telah banyak diberikan di bab-bab sebelumnya. Berikut ini perintah-perintah tambahan untuk mengatur tampilan output.

- endl

  Perintah ini berfungsi sama dengan \n (ganti baris)

  Contoh:

```
cout << "Hallo" << endl;
```

- setw()

  Perintah ini digunakan untuk mengatur lebar tampilan data (rata kanan).

  Contoh:

```
A = 123;
B = 98;
C = 1;

cout << "Nilai A = " << setw(6) << A << endl;
cout << "Nilai B = " << setw(6) << B << endl;
cout << "Nilai C = " << setw(6) << C << endl;
```

  Bandingkan bila tanpa menggunakan setw().

- setfill()

  Perintah setfill digunakan untuk menambahkan suatu karakter tertentu pada field yang kosong pada suatu data. Perhatikan contoh berikut ini

```
A = 123;
B = 98;

cout<< "Nilai A = "<< setw(6) << setfill('*') << A << endl;
cout<< "Nilai B = "<< setw(6) << setfill('.') << B << endl;
```

  Perintah setfill() biasanya digunakan setelah penggunaan setw().

- setprecision()

  Telah dijelaskan di bab sebelumnya.

NB:
Perintah-perintah tambahan pada cout di atas memerlukan file header **iomanip.h** supaya dapat digunakan.

Lab Session

1. Buatlah program menggunakan C++ untuk menentukan harga barang setelah di diskon dengan tampilan sbb:

```
Masukkan harga barang    : Rp. 150000
Masukkan discount (%)    : 12.5
-----------------------------------------------

Harga barang  : Rp.     150000
Besar diskon  : Rp.      18750
                ----------------- -
Harga bersih  : Rp.     131250
```

2. Buatlah program C++ untuk menghitung jarak peluru yang ditembakkan dari suatu lokasi dengan sudut penembakan x^o, dan kecepatan awal peluru V_o. Diketahui besar Pi = 3.141593 dan besar percepatan gravitasi (g) adalah 9.8 m/s^2.

   Input : x (dalam derajad), V_o (dalam m/s).
   Output : jarak peluru (dalam meter)
   Hint: Gunakan rumus $S = 2\ V_o \sin(x) \cos(x)/g$

BAB VI. STATEMENT CONTROL

A. Statement IF

Seperti halnya Pascal, perintah IF dalam C++ juga digunakan untuk menyatakan pernyataan kondisional (bersyarat).

Sintaks sederhana IF adalah

```
if (kondisi)
      statement;
```

Statement pada sintaks di atas akan dilakukan jika kondisinya bernilai TRUE (tidak sama dengan nol).
Apabila statement yang akan dilakukan lebih dari satu, maka sintaksnya menjadi

```
if (kondisi)
      {
      statement1;
      statement2;
      .
      .
      }
```

Contoh sederhana penggunaan IF adalah untuk menentukan boleh tidaknya seseorang melihat film bioskop. Seseorang diperbolehkan menonton jika usianya 17 tahun ke atas. Berikut ini program C++ nya:

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

void main()
{
      int usia;
      clrscr();
      cout << "Berapa usia Anda : ";
      cin >> usia;
      if (usia < 17)
            cout << "Anda tidak boleh menonton bioskop";
}
```

Statement IF juga dapat ditambahkan ELSE sebagai konsekuensi alternatif jika kondisi tidak dipenuhi (FALSE). Sintaksnya:

```
if (kondisi)
{
statement1;
statement2;
.
.
}
else  {
      statement1;
```

```
      statement2;
     }
```

Anda dapat modifikasi program C++ untuk menentukan boleh tidaknya seseorang menonton bioskop seperti di bawah ini:

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

void main()
{
      int usia;
      clrscr();
      cout << "Berapa usia Anda : ";
      cin >> usia;
      if (usia < 17)
            cout << "Anda tidak boleh menonton bioskop";
      else cout << "Anda boleh menonton bioskop";
}
```

Untuk menyatakan kondisi (syarat) yang akan dicek pada IF, Anda dapat menggunakan operator logika dan operator relasional seperti yang telah dijelaskan pada bab sebelumnya. Perhatikan contoh di bawah ini!

```
if ((a >= 2) && (b == 3))
      {
      .
      .
      }
```

Jangan Anda tuliskan

```
if (a >= 2) && (b == 3)
      {
      .
      .
      }
```

atau

```
if ((a >= 2) && (b = 3))
      {
      .
      .
      }
```

Perintah b = 3 merupakan assignment bukan relasional.

Catatan penting:
C++ selalu memperlakukan nilai tidak sama dengan nol sebagai TRUE dan nilai nol sama dengan FALSE. Oleh karena itu, dua perintah di bawah ini adalah identik.

```
if (bil % 2 != 0)
      cout << "Bilangan ganjil";

if (bil % 2)
      cout << "Bilangan ganjil"
```

Selain itu, IF juga dapat berbentuk seperti di bawah ini.

```
if (kondisi1)
      statement1;
else if (kondisi2)
      statement2;
else if (kondisi3)
      statement3;
.
.
else statement;
```

Anda tahu kan maksudnya?

Latihan:

Buatlah program dengan C++ untuk menentukan hari menggunakan IF. Desain tampilannya sbb:

PROGRAM MENENTUKAN NAMA HARI

Pilihan:
A = SENIN C = RABU E = JUM'AT G = AHAD
B = SELASA D = KAMIS F = SABTU

Masukkan Kode Hari (A..G) :

Skenario:
Apabila kode hari yang dimasukkan A s/d G, maka selanjutnya akan tampil nama hari sesuai kodenya. Tapi apabila selain 1 s/d 7, maka akan tampil "MAAF KODE HARINYA SALAH". Kode hari harus bisa juga membaca huruf kecil, artinya jika kode harinya 'a', maka akan tampil SENIN, dst.

B. Statement SWITCH

Statement SWITCH juga berfungsi sama dengan IF. Perintah SWITCH sama dengan perintah CASE OF dalam PASCAL.

Sintaks:

```
switch (variabel)
{
case value1 : statement1;
              break;
case value2 : statement2;
              break;
.
.
```

```
default     : statement;    /* optional */
              break;
}
```

contoh penggunaan:

```
#include <iostream.h>
#include <conio.h>
void main()
{
  int bil;
  clrscr();
  cout << "Masukkan bilangan : ";
  cin >> bil
  switch (bil)
  {
  case 1 : cout << "Anda memasukkan bil. satu";
           break;
  case 2 : cout << "Anda memasukkan bil. dua";
           break;
  case 3 : cout << "Anda memasukkan bil. tiga";
           break;
  default: cout << "Anda memasukkan bil selain 1, 2, dan 3";
           break;
  }
}
```

Selanjutnya coba kalian hapus semua break program di atas dan kalian jalankan. Apa yang terjadi?? Keanehan akan muncul. Why ??

Latihan:

Dengan soal yang sama dengan latihan sebelumnya, kalian coba buat program C++ nya menggunakan SWITCH.

C. Statement FOR

Statement FOR digunakan untuk menyatakan perulangan (seperti PASCAL). Sintaksnya:

```
for (ungkapan1; ungkapan2; ungkapan3)
  {
    .
    .
  }
```

- Ungkapan1 merupakan statement awal (inisialisasi)
- Ungkapan2 merupakan kondisi/syarat perulangan dilakukan
- Ungkapan3 merupakan statement control untuk perulangan

Contoh:

```
for (a = 1; a <= 5; a++)
      {
      cout << "Hello world \n";
      }
```

NB: tipe data variabel a adalah integer

Perintah di atas akan menampilkan teks Hello World sebanyak 5 buah. Perhatikan tanda **a++**. Apa maksudnya?

Selain berupa angka, pencacah perulangan juga dapat berupa karakter. Contoh

```
for (huruf = 'Z'; huruf >= 'A'; huruf--)
      {
      cout << "Huruf abjad= " << huruf << "\n";
      }
```

Perintah di atas akan menampilkan teks Huruf abjad = ... mulai dari Z sampai dengan A. Perhatikan perintah **huruf--**

```
for (angka = 1; angka <= 6; angka+=2)
      {
      cout << "Isi dari angka = " << angka << endl;
      }
```

Perintah di atas akan menampilkan angka 1, 3, 5. Kok bisa? Perhatikan perintah **angka+=2**.

Di bawah ini adalah program untuk mencetak bilangan genap yang kurang dari n (n diperoleh dari input).

```
#include <iostream.h>
#include <conio.h>
void main()
{
      int bil, n;
      cout << "Masukkan n = ";
      cin >> n;
      for (bil = 0; bil < n; bil++)
            {
            if (bil % 2 == 0) cout << bil << " ";
            }
}
```

Latihan:

Buatlah program C++ untuk membuat tampilan segitiga seperti di bawah ini menggunakan FOR.

```
*
* *
* * *
* * * *
* * * * *
```

.
.
dst

Inputnya merupakan tinggi segitiga.

D. Statement WHILE

Statement WHILE juga digunakan untuk menyatakan perulangan. Penggunaannya mirip pada PASCAL. Sintaksnya:

```
while (kondisi)
      {
      .
      .
      }
```

contoh:

Dua perintah di bawah ini adalah identik.

```
for (a = 1; a <= 5; a++)
      {
      cout << "Hello world \n";
      }
```

dengan

```
a = 1;
while (a <= 5)
   {
      cout << "Hello world \n";
      a++;
   }
```

Penting!!!
Jika Anda menggunakan WHILE, pastikan bahwa suatu saat bagian kondisi sampai bernilai FALSE. Apabila tidak, proses perulangan akan terus berjalan selamanya.

Contoh program di bawah ini digunakan untuk menjumlahkan sejumlah data angka. Angka yang akan dijumlahkan diinputkan satu-persatu. Proses pemasukan data angka akan berhenti ketika dimasukkan angka –1. Setelah itu tampil hasil penjumlahannya.

```
#include <iostream.h>
#include <conio.h>
void main()
{
      int data, jumlah,cacah;
      jumlah = 0;
      data = 0;
      cacah = 0;
      while (data != -1)
```

```
      {
            cout << "Masukkan data angka : ";
            cin >> data;
            jumlah += data;
            cacah++;
      }
      cout << "Jumlah data adalah : " << jumlah" << endl;
      cout << "Rata-rata : " << jumlah/cacah;
}
```

Kondisi dalam WHILE juga dapat menggunakan operator logika. Misalnya

```
while ((kondisi1) && (kondisi2))
      {
      .
      .
      }
```

Latihan:

Buatlah simulasi menu program dengan tampilan di bawah ini menggunakan WHILE.

MENU PILIHAN
1. Baca Data
2. Ubah Data
3. Hapus Data
4. Exit

Pilihan Anda (1/2/3/4) ? ...

Apabila dipilih menu no 1, maka akan tampil teks "Anda memilih menu 1". Demikian pula untuk menu 2 dan 3. Kemudian setelah itu muncul teks "Tekan ENTER untuk kembali ke menu utama". Artinya begitu kita tekan ENTER menu pilihan akan muncul kembali, dst. Akan tetapi bila yang dipilih menu 4 (EXIT), program langsung berhenti.

E. Statement DO ... WHILE

Perintah DO ... WHILE hampir sama dengan WHILE sebelumnya. Sintaknya:

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

Perbedaan dengan WHILE sebelumnya yaitu bahwa pada DO WHILE statement perulangannya dilakukan terlebih dahulu baru kemudian di cek kondisinya. Sedangkan WHILE kondisi dicek dulu baru kemudia statement perulangannya dijalankan. Akibat dari hal ini adalah dalam DO WHILE minimal terdapat 1x

perulangan. Sedangkan WHILE dimungkinkan perulangan tidak pernah terjadi yaitu ketika kondisinya langsung bernilai FALSE.

Contoh:

```
a = 1;
do
   {
      cout << "Hello world \n";
      a++;
   }
while(a==0)
```

Perintah di atas akan muncul satu buah Hello World. Bandingkan dengan yang berikut ini:

```
a = 1;
while(a==0)
  {
      cout << "Hello world \n";
      a++;
  }
```

Perintah di atas sama sekali tidak menampilkan Hello World, karena kondisinya langsung FALSE.

TUGAS:

1. Buatlah program dengan C++ untuk menampilkan semua penyelesaian dari persamaan x + y + z = 20.
   Dengan x, y, z bilangan bulat >= 0.

   Contoh tampilan outputnya:

   ```
   ----------------------------
   x            y        z
   ----------------------------
   0            0        20
   0            1        19
   0            2        18
   ```

   dst

2. Seseorang punya rekening tabungan di bank sebesar Rp. 10.000,- (saldo awal). Selanjutnya ia dapat menyetor atau mengambil tabungannya. Buatlah program dengan C++ untuk keperluan transaksi di bank tsb. Tampilan menu utamanya sbb:

   PT. BANK SYARIAH BENERAN

   Saldo : ....

   Menu Transaksi

   1. Setor Tabungan
   2. Ambil Tabungan
   3. Exit

   Pilihan menu (1/2/3) ? ...

   Ketentuan:
   Bank membuat kebijakan bahwa saldo minimum yang harus disisakan di rekening adalah Rp. 10.000,-

BAB VII. FUNCTION

Dalam pemrograman, string merupakan kumpulan dari beberapa karakter-karakter. Untuk membedakan string dengan karakter, dalam C++ dibedakan penulisannya. Suatu nilai merupakan string apabila diapit dengan tanda petik ganda "...", misalnya "SAYA". Sedangkan karakter (char) diapit dengan tanda petik tunggal, misal 's'.

A. Pengantar Function

Sebuah function berisi sejumlah pernyataan yang dikemas dalam sebuah nama. Nama ini selanjutnya dapat dipanggil beberapa kali di beberapa tempat dalam program.

Tujuannya:
1. memudahkan dalam mengembangkan program. Program dibagi menjadi beberapa subprogram kecil, sehingga hal ini menjadi kunci dalam pembuatan program terstruktur.
2. menghemat ukuran program, karena beberapa perintah yang sama dan dijalankan beberapa kali dalam program dapat dijadikan satu kali saja dalam suatu function, kemudian function tersebut dapat dipanggil berulang kali.

Contoh Function I:

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

void garis();          // prototype function

void main()            // main function
{
      clrscr();
      garis();         // panggil function
      cout << "NIM               NAMA MAHASISWA" << endl;
      garis();         // panggil function
      cout << "M0197001          AMIR HAMZAH " << endl;
      cout << "M0197002          PAIMAN" << endl;
      garis();         // panggil function
}

void garis()           // detail function
{
      int i;
      for(i=0;i<=40;i++)
      {
            cout << "-";
      }
      cout << endl;
}
```

Contoh di atas menggambarkan bagaimana membuat function untuk membuat garis. Nama functionnya adalah **garis**. Untuk membuat suatu function, diperlukan suatu prototype dari function tersebut. Prototype function memiliki sintaks sbb:

returned_value_data_type nama_function(argumen);

Seperti halnya dalam Pascal, suatu function dapat mengembalikan (return) suatu nilai (value) yang tergantung tipe datanya. Tipe data value yang dikembalikan inilah yang dimaksud dengan returned_value_data_type.
Sedangkan argumen merupakan parameter-parameter yang akan diolah dalam function tersebut. Argumen boleh ada boleh tidak, sesuai kebutuhan. Apabila parameter argumennya lebih dari satu, cara penulisannya sbb:

tipe_data param1, tipe_data param2, ...

Contoh penulisan prototype function:

- double kuadrat(int x);
- float luas_segitiga(float alas, float tinggi);
- int jumlah_bil(int x, int y, int z);

Apabila suatu function tidak mengembalikan nilai, maka
returned_value_data_type nya diisi **void**.

Setelah prototype function dibuat, selanjutnya membuat function tersebut secara detail. Suatu function disebut juga subprogram, oleh karena itu strukturnya juga sama dengan struktur program utama. Pada contoh function garis() di atas, detail dari function tersebut adalah:

```
void garis()          // detail function
{
      int i;
      for(i=0;i<=40;i++)
      {
            cout << "-";
      }
      cout << endl;
}
```

Kalau diperhatikan, strukturnya sama dengan program utama **main()**.

Contoh Function II:

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

float luas(float alas, float tinggi);
void main()
{
      clrscr();
      a = 10.5;
      t = 11;
      cout << "HITUNG LUAS SEGITIGA" << endl;
      cout << "Panjang alas : " << a << endl;
      cout << "Tinggi       : " << t << endl;
      cout << "Luasnya           : " << luas(a,t) << endl;
}

float luas(float alas, float tinggi)
{
```

```
      float luas_segitiga;
      luas_segitiga = alas * tinggi * 0.5;
      return luas_segitiga;
}
```

Detail function luas di atas dapat ditulis sbb:

```
float luas(float alas, float tinggi)
{
      return (alas * tinggi * 0.5);
}
```

Perintah return adalah untuk mengembalikan hasil operasi di sebelah kanannya ke perintah pemanggilan function.

B. Variabel Global dan Lokal

Setiap kali kita deklarasikan suatu variabel, belum tentu variabel tersebut dikenal di setiap function yang kita buat. Contoh:

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

void cetak();

void main()
{
      int a;
      a = 10;
      cout << "Nilai a = " << a << endl;
      cetak();
}

void cetak()
{
      a++;
      cout << "Nilai a = " << a << endl;
}
```

Ketika program di atas dicompile, akan terdapat error yaitu variabel a dalam function cetak() undefined. Artinya bahwa variabel a tidak dikenal dalam cetak(). Variabel a hanya dikenal dalam program utama/ function main() saja. Maka dalam hal ini variabel a disebut variabel lokal (hanya dikenal dalam function yang di dalamnya didefinisikan a tersebut).

Selanjutnya program di atas diubah sbb:

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

int a;

void cetak();
void main()
{
```

```
      a = 10;
      cout << "Nilai a = " << a << endl;
      cetak();
}

void cetak()
{
      a++;
      cout << "Nilai a = " << a << endl;
}
```

Apabila program di atas dijalankan maka akan tampil:
Nilai a = 10
Nilai a = 11

Pada program di atas, variabel a disebut variabel global karena variabel tersebut dapat dikenali di setiap function yang ada.

Bagaimana dengan yang ini???

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

void cetak();
void main()
{
      int a;
      a = 10;
      cout << "Nilai a = " << a << endl;
      cetak();
}

void cetak()
{
      int a;
      cout << "Nilai a = " << a << endl;
}
```

Apabila program di atas dijalankan, hasilnya adalah:

Nilai a = 10
Nilai a = 747

Hasil di atas menunjukkan bahwa meskipun nama variabelnya sama-sama a, tapi kedua variabel a tersebut berbeda. Setiap variabel a tersebut hanya dikenali di functionnya masing-masing (tidak terkait satu dengan yang lain).

Latihan:

1. Buatlah function yang akan memberikan nilai 1 jika nilai parameter yang dimasukkan huruf 'a', dan akan memberikan nilai 0 jika nilai parameter yang dimasukkan selain huruf 'a'.

2. Buatlah function untuk mencari nilai fungsi $f(x) = 2x^2 - 3x + 1$;
   Parameter functionnya adalah nilai x (tipe data float/double). Return valuenya adalah nilai f(x).

3. Buatlah function untuk membuat tampilan seperti di bawah ini:

   ```
   *
   * *
   * * *
   * *
   *
   ```

   Tampilan di atas muncul apabila dimasukkan n = 3.

   ```
   *
   * *
   * * *
   * * * *
   * * *
   * *
   *
   ```

   Tampilan di atas muncul apabila dimasukkan n = 4.

   Parameter functionnya adalah n (tipe datanya integer).

BAB VIII. STRINGS

Dalam pemrograman, string merupakan kumpulan dari beberapa karakter-karakter. Untuk membedakan string dengan karakter, dalam C++ dibedakan penulisannya. Suatu nilai merupakan string apabila diapit dengan tanda petik ganda "...", misalnya "SAYA". Sedangkan karakter (char) diapit dengan tanda petik tunggal, misal 's'.

Lantas bagaimana dengan "s"?? Dalam hal ini "s" juga merupakan string, meskipun karakter penyusunnya terlihat hanya satu. Akan tetapi pada kenyataannya, "s" disusun tidak hanya karakter 's' saja, melainkan terdapat pula karakter NULL atau '\0', yang berfungsi sebagai tanda akhir dari string.

Untuk mendeklarasikan suatu variabel merupakan string, maka perintahnya:

```
char variabel[maks karakter];
```

contoh:

```
char teks[20];
```

Perintah di atas bermakna bahwa teks merupakan variabel string dengan jumlah karakter yang dapat disimpan maksimal adalah 20 (sudah termasuk karakter NULL).

A. Inisialisasi String

Misalkan suatu variabel string katakanlah kalimat[30] akan diberi nilai "SAYA BELAJAR C++", maka perintahnya:

```
char kalimat[30] = "SAYA BELAJAR C++";
```

Program lengkapnya sbb:

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

void main()
{
      int a;
      a = 20;

      char kalimat[30] = "SAYA BELAJAR C++";

      cout << "Nilai a = " <<  a << endl;
      cout << "Nilai kalimat = " << kalimat << endl;
}
```

Tidak boleh seperti di bawah ini!!

```
void main()
{
      int a;
      char kalimat[30];
      a = 20;

      kalimat = "SAYA BELAJAR C++";     // atau
      kalimat[30] = "SAYA BELAJAR C++";

      cout << "Nilai a = " <<  a << endl;
      cout << "Nilai kalimat = " << kalimat << endl;
}
```

C. Membaca String dari Keyboard

Selanjutnya bagaimana cara membaca string yang berasal dari keyboard?

Berikut ini contohnya:

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

void main()
{
      char nama[20];
      char alamat[30];

      cout << "Masukkan nama Anda : ";
      cin.getline(nama, sizeof(nama));

      cout << "Masukkan alamat Anda : ";
      cin.getline(alamat, sizeof(alamat));

      cout << "Nama Anda : " << nama << endl;
      cout << "Alamat Anda : " << alamat << endl;
}
```

D. Mengcopy String

Kemudian bagaimana untuk mengassign suatu string dari variabel ke variabel lain? Misalnya **kata1** diberi string "HALLO". Selanjutnya **kata2** akan diberi string dari **kata1**. Untuk melakukan hal ini, **Anda tidak bisa memberikan perintah**

```
kata2 = kata1;
```

Perintah yang digunakan untuk keperluan di atas adalah dengan:

```
strcpy(kata2, kata1);       // mengcopy isi dari kata1 ke kata2
```

Contoh:

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

void main()
{
      char kata1[20] = "HALLO";
      char kata2[20];

      strcpy(kata2, kata1);
      cout << "Kata1: " << kata1 << endl;
      cout << "Kata2: " << kata2 << endl;
}
```

E. Function untuk Operasi String

Function-function berikut ini dapat digunakan untuk memanipulasi string. Sebelum function digunakan, tambahkan file header **string.h** pada include.

- **Mengetahui panjang string dengan `strlen()`**

  sintaks:

  ```
  strlen(string)
  ```

  akan mereturn bilangan bulat yang menyatakan panjang string.

  Contoh:

  ```
  int panjangteks;
  char kalimat[30] = "BELAJAR C++ TIDAKLAH SULIT";

  panjangteks = strlen(kalimat);

  cout << "Panjang string : " << panjangteks;
  ```

- **Menggabungkan string dengan `strcat()`**

  Sintaks:

  ```
  strcat(string1, string2)
  ```

  menambahkan string2 ke string1.

  Contoh:

  ```
  char kata1[5] = "SATU ";
  char kata2[5] = "DUA";

  strcat(kata1, kata2);    // nilai kata1 menjadi "SATU DUA"
  ```

- **Mengkonversi ke huruf kapital dengan `strupr()`**

  Sintaks:

  ```
  strupr(string)
  ```

  Mengubah huruf kecil dari string ke huruf kapital.

  Contoh;

  ```
  char string1[30] = "aBcDefgHIJKLmno";

  strupr(string1);   //nilai string1 menjadi "ABCDEFGHIJKLMNO"
  ```

- **Mengkonversi ke huruf kecil dengan `strlwr()`**

  Sintaks:

  ```
  strlwr(string)
  ```

  Function ini kebalikan dari strupr().

- **Mencari Substring dengan `strstr()`**

  Misalkan diberikan suatu string "JAKARTA KOTA METROPOLITAN". Apakah string "METRO" terdapat dalam string tersebut?

  Untuk mengetahui hal ini dengan C++, kita dapat menggunakan function strstr().

  Sintaks:

  ```
  strstr(string1, string2);
  ```

  Function tersebut akan mereturn nilai 1 jika string2 merupakan substring dari string1, dan akan mereturn 0 jika tidak.

  Contoh:

  ```
  if (strstr("JAKARTA KOTA METROPOLITAN", "METRO") == 1)
     cout << "Merupakan substring";
  else cout << "Bukan merupakan substring";
  ```

- **Membalik string dengan `strrev()`**

  Bagaimana cara membalik string "C++" supaya diperoleh "++C"?
  Berikut ini perintah dalam C++,

  sintaks:

  ```
  strrev(string);
  ```

Contoh:

```
char kata[10] = "C++";

strrev(kata);
cout << kata;
```

BAB IX. ARRAY

Array adalah kumpulan data yang bertipe sama yang menggunakan nama yang sama. Dengan menggunakan array, sejumlah variabel dapat memakai nama yang sama. Antara satu variabel dengan variabel lain di dalam array dibedakan berdasarkan nomor elemen (subscript).

Contoh 1. Penggunaan array dalam C++:

Program di bawah ini untuk membaca data kemudian menampilkannya.

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

void main()
{
        int data[10];           // array dengan 10 elemen bertipe integer
        int elemen;
        clrscr();

        // entri 10 data
        for (elemen=0;elemen <= 9;elemen++)
        {
                cout << "Data ke – " << elemen << ": ";
                cin >> data[elemen];
        }

        // tampilkan data setelah entri
        for (elemen=0;elemen <= 9;elemen++)
        {
                cout << "Data ke – " << elemen << ": " << data[elemen];
        }
}
```

NB: Dalam C/C++ elemen array dimulai dari 0.

Contoh 2. Program untuk menampilkan data array dari hasil inisialisasi:

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

void main()
{
        int data[5] = {4, 1, 0, -9, 8};
        int elemen;
        clrscr();

        // tampilkan data
        for (elemen=0;elemen <= 4;elemen++)
        {
                cout << "Data ke – " << elemen << ": " << data[elemen];
        }
}
```

Contoh 3. Program untuk mencari data dari array, dan menampilkan nomor elemennya.

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

void main()
{
	int data[10] = {4, 1, 0, -9, 8, 5, -1, 2, 3, -7};
	int elemen, ketemu;

	cout << "Data yang dicari : ";
	cin >> x;

	ketemu = 0;
	for(elemen=0; elemen<= 9; elemen++)
	{
		if (data[elemen] == x)
		{
		ketemu = ! ketemu;
		break;
		}
	}

	if (ketemu == 0) cout << "Data tidak ditemukan ";
	else cout << "Data ada di elemen : " << elemen;
}
```

Contoh 4. Program untuk menampilkan data terbesar (maks) dari suatu array.

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

void main()
{
	int data[10] = {4, 1, 0, -9, 8, 5, -1, 2, 3, -7};
	int elemen, max;

	max = data[0];

	for(elemen=0; elemen<= 9; elemen++)
	{
		if (data[elemen]>max) max = data[elemen];
		else max = max;
	}

	cout << "Nilai maksimum adalah : " << max;
}
```

Array di atas adalah array dimensi satu. Bagaimana dengan array dimensi dua?

Berikut ini contoh penggunaan array dua dimensi:

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

void main()
{
	int j, k;
	int data[5][3] =
		{
			{3, 4, -1},
			{2, 3, 0},
			{1, 1, 2},
			{5, 9, -4},
			{6, 6, 2}
		};

	for (j = 0; j<=4; j++)
	{
		for (k = 0; k<=2; k++)
		cout << "data[" << j << "][" << k << "] = " << data[j][k] << endl;
	}
}
```

Latihan :

1. Diberikan suatu array bertipe integer yang berukuran 10 ruang/elemen yang setiap elemen sudah ada nilainya yang sudah terurut. Buatlah program untuk mencari jangkauan (range) dari nilai dalam array tersebut.

2. Diberikan suatu n buah data statistik yang diisikan ke dalam larik bertipe real. Standard deviasi dari suatu data statistik didefinisikan sebagai

$$d = \sqrt{\frac{\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2}{n-1}}$$

BAB X. STRUKTUR

Struktur bermanfaat untuk mengelompokkan sejumlah data dengan tipe yang berlainan. Perhatikan contoh struktur berikut ini:

```
struct data_tanggal
{
	int tahun;
	int bulan;
	int tanggal;
};
```

Struktur di atas bernama data_tanggal, yang tersusun atas 3 unit penyusunnya yaitu tahun, tanggal, dan bulan. Setelah dibuat struktur tanggal, selanjutnya struktur tersebut dapat digunakan sebagai tipe data suatu variabel, dalam hal ini disebut tipe data abstrak.

Berikut contoh programnya.

Contoh 1.

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

void main()
{
        struct data_tanggal
        {
                int tanggal;
                int bulan;
                int tahun;
        };
data_tanggal  tanggal_lahir;

tanggal_lahir.tanggal = 1;
tanggal_lahir.bulan = 9;
tanggal_lahir.tahun = 1979;

cout    << tanggal_lahir.tanggal << '/' << tanggal_lahir.bulan << '/'
        << tanggal_lahir.tahun;

}
```

Misalkan terdapat 2 buah variabel tgl1 dan tgl2 sama-sama bertipe data_tanggal. Selanjutnya diinginkan mengcopy isi dari tgl1 ke tgl2, maka dapat dilakukan dengan perintah assignment biasa. Contoh:

Contoh 2.

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

void main()
{
        struct data_tanggal
        {
                int tanggal;
                int bulan;
                int tahun;
        };
data_tanggal  tgl1, tgl2;

tgl1.tanggal = 1;
tgl1.bulan = 9;
tgl1.tahun = 1979;

tgl2 = tgl1;    // atau

tgl2.tanggal = tgl1.tanggal;
tgl2.bulan = tgl1.bulan;
tgl2.tahun = tgl2.tahun;

cout    << tgl1.tanggal << '/' << tgl1.bulan << '/' << tgl1.tahun << endl;
```

```
cout    << tgl2.tanggal << '/' << tgl2.bulan << '/' << tgl2.tahun << endl;
}
```

Untuk membandingkan 2 buah struktur, masing-masing unit harus dibandingkan sendiri-sendiri, contoh:

```
if ((tgl1.tanggal == tgl2.tanggal) && (tgl1.bulan == tgl2.bulan) &&
  (tgl1.tahun == tgl2.tahun))
cout << "Isi strukturnya sama";
else cout << "Isi struktur tak sama";
```

Suatu struktur juga dapat digunakan untuk argumen/parameter suatu function.

Contoh 3.

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

struct data
{
        int x;
        int y;
};

void tampilkan(data nilai)

void main()
{
        data nilaiku;
        nilaiku.x = 10;
        nilaiku.y = 16;

        tampilkan(nilaiku);
}

void tampilkan(data nilai)
{
        cout << "Nilai x = " << nilai.x << endl;
        cout << "Nilai y = " << nilai.y << endl;
}
```

Latihan:

1. Buatlah program menggunakan function untuk menentukan selisih antara dua waktu (jam). Misalkan selisih antara pukul 4.30 dengan 6.00 adalah 90 menit. Terdapat 2 argumen function yaitu jam pertama dan jam kedua (lebih besar dari jam pertama).

   hint: buat struktur jam (dengan unit jam, dan menit).

2. Bilangan kompleks memiliki format a+b i, dengan a dan b adalah bilangan real. Notasi a disebut juga bagian real, dan b disebut juga bagian imajiner. Buatlah program menggunakan struktur untuk menjumlahkan, mengurangkan, dan mengalikan 2 buah bilangan kompleks.

   Contoh:

$(2+3\,i) + (-1 + 3i) = 1+6\,i$
$(2+3\,i) - (-1 + 6i) = 3 - 3i$
$(2+3\,i) \,.\, (-1 + 3i) = -2 - 3i + 6i + 9\,i^2 = -2 - 3i + 6i - 9 = -11 + 3i$

BAB XI. Operasi File

Operasi dasar file pada prinsipnya terbagi menjadi 3 tahap, yaitu:
- membuka atau mengaktifkan file
- melaksanakan pemrosesan file
- menutup dile

A. Membuka file

Sebelum suatu file dapat diproses, file harus dibuka terlebih dahulu. Sebelum file dibuka, terlebih dahulu obyek file harus didefinisikan. Sintaksnya:

```
ofstream nama_obyek;
```

perintah ofstream dapat dijalankan dengan menyertakan file header **fstream.h**

Setelah itu, suatu file dapat dibuka dengan perintah

```
nama_obyek.open("nama file dan path");
```

B. Menulis ke File

Salah satu jenis pemrosesan pada file adalah menulis atau merekam data ke file. Sintaknya:

```
nama_obyek << ... ;
```

C. Menutup File

Setelah pemrosesan file selesai, file dapat ditutup menggunakan perintah

```
nama_obyek.close();
```

Contoh 1.
Program berikut ini untuk menulis teks ke dalam file

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

void main()
{
 ofstream fileteks;
 fileteks.open("C:/algo.txt");

 fileteks << "Untuk mencapai tujuan yg besar, maka tujuan itu"
          << endl;
 fileteks << "harus dibagi-bagi menjadi tujuan kecil"<< endl;
 fileteks << "sampai tujuan itu merupakan tujuan yg dapat "
          << "dicapai" << endl;
 fileteks << "berdasarkan kondisi dan potensi yg dimiliki saat "
          << itu " << endl;

 fileteks.close();
}
```

perintah fileteks.open("C:/algo.txt"); akan membuka file algo.txt yang ada di C:\ Apabila file tersebut belum ada maka akan dibuat secara otomatis, dan apabila sudah ada isi file algo.txt akan terhapus.

D. Menambah Data pada File

Suatu file yang sudah ada sebelumnya dapat ditambah data yang baru (tidak menghapus data lama). Caranya dengan menambahkan perintah **ios::app** pada open().

```
nama_obyek.open("nama file", ios::app);
```

Contoh 2.

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

void main()
{
 ofstream fileteks;
 fileteks.open("C:/algo.txt", ios::app);

 fileteks << endl;
 fileteks << "Oleh: Al Khowarizmi << endl;

 fileteks.close();
}
```

E. Memeriksa Keberhasilan Operasi File

Tidak selamanya jalan yang mulus ditemui. Ada kemungkinan terjadi saat file dibuka, ternyata file tidak ada. Dalam C++ tersedia function untuk memeriksa kondisi-kondisi pada operasi file, sehingga kesalahan saat eksekusi dapat dikendalikan.

Function yang dimaksud adalah **fail()**.

Contoh 3:

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

void main()
{
 ifstream fileteks; { ifstream digunakan u/ membaca file }
 fileteks.open("C:/algo.txt");
 if (fileteks.fail()) cout << "Maaf file takdapat dibuka/"
                           << "tidak ditemukan";
 fileteks.close();
}
```

F. Operasi Berbasis Karakter

Operasi file dapat dilakukan dalam bentuk karakter. Misalnya proses penyimpanan data ke file dilakukan setiap karakter, atau membaca data file karakter per karakter. Operasi ini didukung oleh function **put()** dan **get()**.

Contoh 4:

Program untuk menyimpan data karakter per karakter ke dalam file.

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

void main()
{
 ofstream fileteks;
 fileteks.open("C:/contoh.txt");
 fileteks.put('A');
 fileteks.put('B');
 fileteks.put('C');
 fileteks.close();
}
```

Contoh 5.
Program untuk membaca file karakter per karakter

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

void main()
{
  char karakter;
  ifstream fileteks; {}
  fileteks.open("C:/contoh.txt");

  while(!fileteks.eof())
  {
    fileteks.get(karakter);
    cout << karakter;
  }

  fileteks.close();
}
```

Latihan.

1. Buatlah program C++ untuk menghitung jumlah karakter dalam suatu file. Inputnya adalah nama file dan pathnya.
2. Buatlah program C++ untuk menghitung jumlah karakter tertentu, misalnya karakter 'A'. Input berupa nama file dan karakter yang akan dihitung.
3. Misalkan suatu file teks berisi listing program C++. Buatlah program untuk menghitung pasangan kurung kurawal yang ada pada file teks tersebut.
4. Buatlah program C++ untuk melakukan enkripsi shift chiper suatu file teks (dengan asumsi semua karakter huruf adalah huruf kapital). Inputnya adalah file teks yang akan dienkripsi dan besar pergeseran (integer). Outputnya adalah file teks hasil enkripsi.

Hint:
Ide dasar shift chiper adalah mengubah setiap karakter huruf ke karakter huruf lain. Misalkan pergeserannya 2, maka berikut ini karakter hasil enkripsi

awal	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
hasil	C D E F G H I J K L M N O P Q R S T U V W X Y Z A B

Sehingga misal diberikan suatu teks C++ IS EASY, maka hasil enkripsinya adalah E++ KU GCUA

BAB XII. CLASS

Class merupakan struktur data dari obyek. Untuk menjelaskan tentang class, lihat perbandingannya dengan struktur berikut ini:

Perhatikan program di bawah ini:

Contoh 1.

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

struct mahasiswa
{
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

Setelah program di atas dicompile, error tidak ada.
Berikutnya struktur di atas kita ganti dengan class, menjadi

Contoh 2.

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

class mahasiswa
{
      char nim[8]; char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

setelah program di atas di compile, ternyata error muncul. Error tersebut muncul karena class tidak dikenal dalam main(). Kesalahan ini sekaligus menunjukkan perbedaan dengan struktur.

Penggunaan PUBLIC

Agar program di atas dapat dicompile, ditambahkan perintah public diikuti dengan tanda titik dua (:), sehingga programnya menjadi

Contoh 3.

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

class mahasiswa
{
   public:
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

Perintah PUBLIC menyatakan bahwa perintah-perintah yang ada di bawahnya dapat diakses diluar class. Perintah PUBLIC merupakan termasuk a*ccess specifier* (penentu akses). Selain PUBLIC, terdapat perintah lain yang termasuk access specifier, yaitu PRIVATE.

Contoh 4.

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

class mahasiswa
{
 private:
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
```

```
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

Setelah dicompile, error yang sama dengan sebelumnya muncul yaitu class tidak dapat diakses di main().
Perintah PRIVATE memberi pengertian bahwa perintah yang ada dibawahnya hanya dapat diakses dalam class tersebut, yang dalam hal ini adalah class mahasiswa.

variabel nim, nama, dan umur dalam class mahasiswa disebut data anggota. Selain data anggota, kita juga dapat menambahkan fungsi anggota.

Contoh 5.

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

class mahasiswa
{
      private :
            char nim[8];
            char nama[20];
            int umur;

      public :
      void inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS)
      {
            strcpy(nim, NIMMHS);
            strcpy(nama, NAMAMHS);
            umur = UMURMHS;
      }

      void tampilkan()
      {
            cout << nim << endl;
            cout << nama << endl;
            cout << umur << endl;
      }

};

void main()
{
      mahasiswa mhsd3;
      mhsd3.inisialisasi("M0197001", "Burhanudin Harahap", 20);
      mhsd3.tampilkan();
}
```

Pada program di atas, fungsi inisialisasi() dan tampilkan() merupakan fungsi anggota dari class mahasiswa. Keduanya dibuat public karena akan diakses dari luar class, sedangkan data anggotanya (nim, nama, umur) dibuat private.

Mungkin Anda berpikir mengapa program terakhir (contoh 5) terlalu panjang, padahal akan diperoleh hasil yang sama dengan program sebelumnya (contoh 1). Anda memang benar, tapi contoh 5 ini merupakan cara pemrograman berorientasi obyek (PBO).

Perhatikan program contoh 5, khususnya pada kedua fungsi anggota. Kedua fungsi tidak punya prototype. Kita juga dapat memberikan prototype yang merupakan cara kedua dalam penulisan.

Contoh 6.

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

class mhs
{
  private :
      char nim[8];
      char nama[20];
      int umur;

  public :
      void inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS);
      void tampilkan();
};

void main()
{
      mhs mhsd3;
      mhsd3.inisialisasi("M0197001", "Burhanudin Harahap", 20);
      mhsd3.tampilkan();
}

void mhs::inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS)
{
      strcpy(nim, NIMMHS);
      strcpy(nama, NAMAMHS);
      umur = UMURMHS;
}

void mhs::tampilkan()
{
      cout << nim << endl;
      cout << nama << endl;
      cout << umur << endl;
}
```

Cara kedua inilah yang sering dipilih oleh para programmer C++.

Berikut ini contoh-contoh program yang memanfaatkan class

Contoh 7.
Program untuk menyimpan data n data mahasiswa kemudian menampilkannya.

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

```
class mahasiswa
{
   public:
      char nim[20];
      char nama[50];
      int  umur;

      void tampilkan(char *NIMMHS, char *NAMAMHS, int UMURMHS)
           {
                cout << "NIM MHS  : " << NIMMHS << endl;
                cout << "NAMA MHS : " << NAMAMHS << endl;
                cout << "UMUR     : " << UMURMHS << endl;
           }
};

void main()
{
   mahasiswa mhsd3[50]; { tipe data array }
   char temp[10]; int n, i;
   clrscr();
   cout << "<< ENTRI DATA MAHASISWA D3 " << endl;
   cout << endl;
   cout << "Jumlah mahasiswa : ";
   cin.getline(temp, sizeof(temp));
   n = atoi(temp);
   for (i=0;i<=n-1;i++)
   {
      cout << "DATA - " << i+1 << endl;
      cout << "NIM MAHASISWA  : " ;
      cin.getline(mhsd3[i].nim, sizeof(mhsd3[i].nim));
      cout << "NAMA MAHASISWA : " ;
      cin.getline(mhsd3[i].nama, sizeof(mhsd3[i].nama));
      cout << "UMUR           : ";
      cin.getline(temp, sizeof(temp));
      mhsd3[i].umur = atoi(temp);
      cout << endl;
   }

   // tampilkan semua data
   cout << "-------------------------" << endl;
   cout << "DATA YANG MASUK" << endl;
   cout << "-------------------------" << endl;
   for (i=0;i<=n-1;i++)
   {
     cout << "DATA MAHASISWA " << i+1 << endl;
     mhsd3[i].tampilkan(mhsd3[i].nim, mhsd3[i].nama, mhsd3[i].umur);
     cout << endl;
   }
   getch();
}
```

Contoh 8.
Program untuk menjumlahkan dan mengurangkan 2 buah bilangan kompleks.

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

class kompleks
{
       private:
            float real;
            float imajiner;

       public:
            void tambah(float real1, float imajiner1, float
                 real2, float imajiner2)
                 {
                    real = real1 + real2;
                    imajiner = imajiner1 + imajiner2;
                 }

                  void kurangi(float real1, float imajiner1,
               float
                 real2, float imajiner2)
                 {
                    real = real1 - real2;
                    imajiner = imajiner1 - imajiner2;
            }

            void tampilkan()
            {
              cout << "Hasilnya adalah : " << real << "+ "
                   << imajiner << 'i' << endl;
            }
};

void main()
{
      clrscr(); kompleks bilkompleks;
     float el_real1, el_real2, el_imaj1, el_imaj2;
      cout << "Bilangan Kompleks pertama" << endl;
      cout << "Elemen real     : ";
      cin  >> el_real1;
      cout << "Elemen imajiner : ";
      cin  >> el_imaj1;
      cout << "Bilangan Kompleks kedua" << endl;
      cout << "Elemen real     : ";
      cin  >> el_real2;
      cout << "Elemen imajiner : ";
      cin  >> el_imaj2;
      bilkompleks.tambah(el_real1,el_imaj1,el_real2,el_imaj2);
      bilkompleks.tampilkan();

      bilkompleks.kurangi(el_real1,el_imaj1,el_real2,el_imaj2);
      bilkompleks.tampilkan();
      getch();
}
```