**PRAKTIKUM III**

**SELECTION & REPETITION**

Dalam sebuah proses program, biasanya terdapat kode penyeleksian kondisi, kode pengulangan program, atau kode untuk pengambilan keputusan. Untuk tujuan tersebut, C++ memberikan berbagai kemudahan dalam sintaknya. Terdapat sebuah konsep, yakni Blok Instruksi. Sebuah blok dari instruksi merupakan sekelompok instruksi yang dipisahkan dengan tanda semicolon (;) dan berada diantara tanda { dan }. Untuk Blok Instruksi, penggunaan tanda { dan } boleh ditiadakan. Dengan syarat, hanya pernyataan tunggal yang akan dilaksanakan oleh blok instruksi. Apabila pernyataan yang dijalankan lebih dari satu, maka tanda { dan } wajib disertakan.

1. **SELEKSI KONDISIONAL ( if …else…)**

Format penulisannya :

if (kondisi) pernyataan;

kondisi adalah ekspresi yang akan dibandingkan. Jika kondisi bernilai benar, maka pernyataan akan dijalankan. Namun, jika kondisi bernilai salah, maka pernyataan akan diabaikan. Contoh pernyataan berikut akan menampilkan tulisan x adalah 100 apabila x bernilai 100:

if (x==100)

cout << “x adalah 100”;

Jika menginginkan lebih dari sebuah pernyataan yang dijalankan, ketika kondisi terpenuhi maka blok instruksi harus menyertakan tanda { dan }.

if (x==100)

{

cout << “x adalah ”;

cout << x;

}

Bila menginginkan sesuatu dijalankan ketika kondisi tidak terpenuhi, dapat ditambahkan keyword ***else***. Sintaknya adalah :

**if** (kondisi)

pernyataan1;

**else**

pernyataan2;

Contoh :

if (x==100)

cout <<”x adalah 100”;

else

cout <<”x bukan 100”;

Pernyataan if…else… dapat terdiri dari beberapa else. Pada contoh berikut,

program akan memberikan jawaban terhadap inputan data, apakah berupa nilai

positif, negative atau nol :

if (x>0)

cout<<”positive”;

else if (x<0)

cout<<”negative”;

else

cout<<”x adalah 0”;

1. **PERULANGAN (loops)**

Sebuah atau beberapa pernyataan akan dijalankan secara berulang-ulang, selama kondisi terpenuhi.

* Perulangan dengan **while**

Sintaknya adalah :

while (kondisi) pernyataan;

pernyataan akan dijalankan selama ekspresi bernilai **true**. Contoh :

//hitungan mundur menggunakan while

#include<iostream.h>

int main()

{

int n;

cout<<”Masukkan angka untuk mulai ”;

cin>>n;

while (n>0) {

cout << n << “, “;

--n; }

cout<<”STOP!”;

return 0;

}

Di layar akan tampil :

**Masukkan angka untuk mulai : 4 4,**

**3, 2, 1, STOP!**

Algoritma untuk perulangan di atas adalah sebagai berikut :

1. User menginputkan sebuah nilai ke variable n.

2. Pernyataan while akan melakukan pengecekan apakah (n>0) ?.

Dalam kondisi ini, terdapat 2 kemungkinan :

a. **true** : lakukan pernyataan (langkah **3**)

b. **false** : lompati pernyataan (lanjutkan ke langkah **5..**)

3. Lakukan perintah :

cout << n << “, “; --n;

(cetak n ke layar, dan turunkan 1 nilai n)

4. Akhiri blok. Kembali lagi ke langkah **2..**

5. Lanjutkan program setelah blok while. Cetak STOP! Dan akhiri program.

* Perulangan dengan **do…while**

Sintaknya :

**do** *pernyataan* **while** (*kondisi*);

Konsep do...while mirip dengan while. Namun pernyataan akan dijalankan terlebih dahulu sebelum pengecekan kondisi. Untuk lebih jelasnya, perhatikan contoh berikut: 

Akan tampil :

**Masukkan nomor (tekan 0 untuk mengakhiri) : 67**

**Anda memasukkan angka : 67**

**Masukkan nomor (tekan 0 untuk mengakhiri) : 12**

**Anda memasukkan angka : 12**

**Masukkan nomor (tekan 0 untuk mengakhiri) : 0**

**Anda memasukkan angka : 0**

* Perulangan dengan **for**

Sintaknya :

**for** (inisialisasi; kondisi; counter) pernyataan;

Pernyataan akan diulangi jika kondisi bernilai true. Sama seperti perulangan dengan while. Namun for menetapkan inisialisasi dan penaikan berada dalam ( dan ).

Penjelasannya adalah sebagai berikut:

1. *Inisialisasi:* akan dieksekusi. Biasanya merupakan variable yang akan dipakai sebagai counter atau pencacah. Bagian ini akan dieksekusi hanya sekali.
2. *Kondisi:* akan diperiksa, jika bernilai true maka perulangan akan dilanjutkan dan jika bernilai false maka perulangan akan dilewati.
3. *Pernyataan:* akan dieksekusi. Biasanya dapat terdiri dari sebuah instruksi atau blok instruksi yang berada di antara { dan }.
4. Terakhir, apapun perintah dalam *counter* akan dijalankan dan kemudian perulangan kembali lagi ke langkah 2.

Contoh :

*// hitungan mundur dengan for*

#include<iostream.h>

int main()

{

**for**(int n=10; n>0;n--)

{ cout<<n<<”, “;

}

cout<<”STOP!”;

return 0;

}

Hasilnya adalah :

**10, 9, 8, 7, 6, 5, 4, 3, 2, 1, STOP!**

*Inisialisasi* dan *Counter* adalah optional atau dapat ditiadakan. Namun tidak demikian dengan tanda semicolon (;). Misalnya kita dapat menuliskan: for(;n<10;) jika tanpa *inisialisasi* dan tanpa *penaikan*, atau for(;n<10;n++) jika tanpa *inisialisasi* namun tetap menggunakan *penaikan*.

**Nested Loops (Perulangan Bertumpuk)**

Perulangan bertumpuk secara sederhana dapat diartikan : terdapat satu atau lebih loop di dalam sebuah loop. Banyaknya tingkatan perulangan, tergantung dari kebutuhan. Biasanya, nested loops digunakan untuk membuat aplikasi matematika yang menggunakan baris dan kolom. Loop luar, biasanya digunakan untuk mendefinisikan baris. Sedangkan loop dalam, digunakan untuk mendefinisikan kolom. Contoh:

**for**(**int** baris = 1; baris <= 4; baris++)

{

**for** (**int** kolom = 1; kolom <= 5; kolom++) {

cout<<kolom<<" ";

}

cout<<endl;

}

*Penjelasan program:*

Perulangan akan menghasilkan nilai sebagai berikut :

**bans =1** ; kolom =l; cetakl

kolom = 2; cetak 2

kolom = 3; cetak 3

kolom = 4; cetak 4

kolom = 5 ; cetak 5

ganti baris !

**bans =2** ; kolom = 1; cetakl

kolom = 2; cetak 2

kolom = 3; cetak 3

kolom = 4; cetak 4

kolom = 5 ; cetak 5

ganti baris !

**bans =3** ; kolom =l; cetak 1

kolom = 2; cetak 2

kolom = 3; cetak 3

kolom = 4; cetak 4

kolom = 5 ; cetak 5

ganti baris !

**bans =4** ; kolom = 1; cetak 1

kolom = 2; cetak 2

kolom = 3; cetak 3

kolom = 4; cetak 4

kolom = 5 ; cetak 5

ganti baris !

selesai.

Dan di layar akan muncul hasil dengan bentuk matrik sebagai berikut: 1

2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5

*Tambahan :*

* + perintah **break**

break berfungsi untuk keluar dari loop, walaupun kondisinya belum seluruhnya terpenuhi. Biasanya, perintah ini digunakan untuk memaksa program keluar dari loop. Contoh berikut akan berhenti menghitung sebelum terhenti secara normal.

for (int n=10; n>0;n--) {

cout<<n<<", ";

if (n==3) {

cout<<"penghitungan dihentikan !"; break; }

}

dan di layar akan tampak hasil sebagai berikut :

10, 9, 8, 7, 6, 5, 4, 3, penghitungan dihentikan !

* + perintah **continue**

Perintah ini akan melewati satu iterasi yang sesuai dengan syarat tertentu, dan melanjutkan ke iterasi berikutnya. Contoh:

for (int n=10; n>0;n--) {

if (n==5) continue;

cout<<n<<", "; } cout<<"STOP !";

dan di layar akan muncul :

10, 9, 8, 7, 6, 4, 3, 2, 1, STOP !

Struktur Selektif dengan **switch**

Logika menggunakan switch sama dengan menggunakan perintah if yang telah dijelaskan sebelumnya. Sintaknya adalah :

**switch** (*pilihan*)

{

**case** nilai1 :

blok pernyataan 1

**break;**

**case** nilai2 :

blok pernyataan 2

**break;**

**default :**

blok pernyataan default }

Cara kerjanya:

1. **switch** akan mengevaluasi *pilihan* dan apabila isinya sama dengan *nilai1*, maka *blok pernyataan 1* akan dijalankan sampai menemukan perintah **break** untuk kemudian keluar dari blok switch.
2. Bila pilihan tidak sama isinya dengan nilai1, maka akan dicocokkan lagi dengan *nilai2*. dan apabila isinya sama dengan *nilai2*, maka *blok pernyataan 2* akan dijalankan sampai menemukan perintah **break** untuk kemudian keluar dari blok switch.
3. Terakhir, apabila isi pilihan tidak sesuai dengan *nilai1*, *nilai2* dan seterusnya maka secara otomatis yang dijalankan adalah *blok pernyataan default*. Contoh (*kedua penggalan program memiliki arti yang sama*) :

contoh **switch** contoh **if**

switch (x) { if (x==1) {

case 1: cout<<”x adalah 1”; cout<<”x adalah 1”; }

break; else if(x==2) {

case 2: cout<<”x adalah 2”; cout<<”x adalah 2”; }

break; else {

default: cout<<”tidak teridentifikasi”;

cout<<”tidak teridentifikasi”;

} }

Sedangkan untuk program yang memiliki beberapa nilai pilihan, maka dapat ditulis seperti contoh berikut:

switch (x) {

case 1: case

2: case 3:

cout<<”x = 1, 2 atau 3”;

break;

default:

cout<<”x tidak sama dengan 1, 2 atau 3 ”;}