# Pertemuan 3

# 3. Struktur Perulangan dalam Bahasa C++

Obyektif Praktikum :

1. Mengerti struktur perulangan dalam C++
2. Dapat menggunakan struktur perulangan berdasarkan penggunaannya

## P.3.1 Struktur perulangan (loops)

*Loops* merupakan perulangan *statement* dengan jumlah tertentu jika kondisi terpenuhi.

**The *while* loop.**
Sintaks :

```
while (expression) statement
```

Fungsi dari statement diatas adalah mengulang *statement* jika *expression* bernilai *true*.

**The *do-while* loop.**
Format:

```
do statement while (condition);
```

Secara fungsional, hampir sama dengan *while* loop, hanya saja *condition* dalam *do-while* dievaluasi setelah eksekusi *statement* , dengan kata lain, sedikitnya satu kali eksekusi *statement* walaupun kondisi tidak terpenuhi.

**The *for* loop.**
Format :

```
for (initialization; condition; increase)
statement;
```

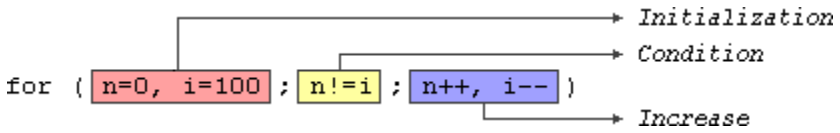
Fungsinya akan mengulang *statement* jika *condition* bernilai benar. Sama seperti *while* loop., hanya saja **for** memungkinkan untuk memberikan instruksi *initialization* dan intruksi *increase*, sehingga dapat menampilkan loop dengan counter.

Algoritma perulangan *for* :
1. *initialization*, digunakan untuk memberikan nilai awal untuk variable counter. Dieksekusi hanya sekali.
2. *condition*, Dievaluasi, jika bernilai **true** maka loop berlanjut, sebaliknya loop berhenti dan *statement* diabaikan
3. *statement*, dieksekusi, bisa berupa instruksi tunggal maupun blok instruksi (dalam tanda **{}** ).
4. *increase*, dieksekusi kemudian algoritma kembali ke step 2.

*Initialization* dan *increase* bersifat optional. Sehingga dapat dituliskan : **for (;n<10;)** untuk for tanpa *initialization* dan *increase*; atau **for (;n<10;n++)** untuk for dengan *increase* tetapi tanpa *initialization*. Dengan operator koma (**,**) kita dapat mendeklarasikan lebih dari satu instruksi pada bagian manapun termasuk dalam loop **for**, contoh :

```
for ( n=0, i=100 ; n!=i ; n++, i-- )
        {
         // whatever here...
        }
```

Loop diatas akan meng-eksekusi sebanyak 50 kali :




nilai awal **n** = **0** dan **i** = **100**, dengan kondisi **(n!=i)** (yaitu **n** tidak sama dengan **i**). Karena **n** mengalami penambahan 1 dan **i** mengalami pengurangan 1, maka kondisi loop akan salah setelah loop yang ke-50, yaitu ketika **n** dan **i** bernilai 50.

**Kontrol Percabangan (Bifurcation) dan Lompatan (jumps)**
**Instruksi *break***
Dengan menggunakan instruksi *break*, program akan keluar dari loop walaupun kondisi untuk berakhirnya loop belum terpenuhi. Dapat digunakan untuk mengakhiri *infinite loop*, atau untuk menyebabkan loop selesai sebelum saatnya

**Instruksi *continue***
Instruksi *continue* menyebabkan program akan melewati instruksi selanjutnya hingga akhir blok dalam loop. Atau dengan kata lain langsung melompat ke iterasi selanjutny

**Instruksi *goto***
Menyebabkan lompatan dalam program. Tujuan dari lompatan diidentifikasikan dengan label, yang berisikan argumen-argumen. penulisan label diikuti dengan tanda colon ( **:** ).

**Struktur Seleksi : *switch*.**
Instruksi *switch* digunakan untuk membandingkan beberapa nilai konstan yang mungkin untuk sebuah ekspresi, hampir sama dengan *if* dan *else if*. Bentuk umumnya :

```
switch (expression) {
  case constant1:
    block of instructions 1
    break;
  case constant2:
    block of instructions 2
    break;
  .
  .
  .
  default:
    default block of instructions
  }
```

**switch** meng-evaluasi *expression* dan memeriksa apakah equivalen dengan *constant1*, jika ya, maka akan meng-eksekusi *block of instructions 1* sampai terbaca keyword **break**, kemudian program akan lompat ke akhir dari stuktur selektif *switch*.

Jika *expression* tidak sama dengan *constant1*, maka akan diperiksa apakah *expression* equivalen dengan *constant2*. jika ya, maka akan dieksekusi *block of instructions 2* sampai terbaca **break**. Begitu seterusnya, jika tidak ada satupun konstanta yang sesuai maka akan mengeksekusi **default:**

contoh :

***switch* example**

```
switch (x) {
  case 1:
    cout << "x is 1";
    break;
  case 2:
    cout << "x is 2";
    break;
  default:
    cout << "value of x unknown";
  }
```

***if-else* equivalent**

```
    if (x == 1) {
    cout << "x is 1";
     }
    else if (x == 2) {
     cout << "x is 2";
   }
  else {
  cout << "value of x unknown";
  }
```

Perintah Switch sering digunakan untuk program yang mengandung menu atau penginputan karakter yang bisa diseleksi.

## P.3.2 Contoh Kasus

- *Countdown using a for loop* :

```
// countdown using a for loop
#include <iostream.h>
int main ()
{
  for (int n=10; n>0; n--) {
    cout << n << ", ";
  }
  cout << "FIRE!";
  return 0;
}
```

Output :

```
10, 9, 8, 7, 6, 5, 4, 3, 2, 1, FIRE!
```

- *Custom countdown using while* :

```
// custom countdown using while
#include <iostream.h>
```

```
int main ()
{
  int n;
  cout << "Enter the starting number > ";
  cin >> n;

  while (n>0) {
    cout << n << ", ";
    --n;
  }
  cout << "FIRE!";
  return 0;
}
```

Output :

```
Enter the starting number > 8
8, 7, 6, 5, 4, 3, 2, 1, FIRE!
```

Algoritma program dimulai dari **`main`** :

**1.** User meng-input nilai untuk **`n`**.

**2.** Instrukti while mengevaluasi apakah **`(n>0)`**. Ada dua kemungkinan :

**true:** meng-eksekusi *statement* (step **3,**)

**false:** melompati *statement*. lanjut ke step **`5.`**.

**3.**Mengeksekusi *statement* :

```
cout << n << ", ";
--n;
```

(Menampilkan **`n`** di layar dan mengurangi **n** dengan 1).

**4.** Akhir dari blok. kembali ke step **2.**

**5.** lanjut menuju program setelah blok. Cetak : **`FIRE!`** dan program berakhir.

- *Number echoer* :

```
// number echoer
#include <iostream.h>
int main ()
{
  unsigned long n;
  do {
    cout << "Enter number (0 to end): ";
    cin >> n;
    cout << "You entered: " << n << "\n";
  } while (n != 0);
  return 0;
}
```

Output :

```
Enter number (0 to end): 12345
You entered: 12345
```

```
Enter number (0 to end): 160277
You entered: 160277
Enter number (0 to end): 0
You entered: 0
```

- *Break loop example* :

```
// break loop example
#include <iostream.h>
int main ()
{
  int n;
  for (n=10; n>0; n--) {
    cout << n << ", ";
    if (n==3)
    {
      cout << "countdown aborted!";
      break;
    }
  }
  return 0;
}
```

Output :

```
10, 9, 8, 7, 6, 5, 4, 3, countdown aborted!
```

## P.3.3 Latihan

**1. Carilah output untuk program di bawah ini :**

```
#include <iostream.h>
int main ()
{
  int n=10;
  loop:
  cout << n << ", ";
  n--;
  if (n>0) goto loop;
  cout << "FIRE!";
  return 0;
}
```

.2. **Carilah output untuk program di bawah ini :**

```
#include <iostream.h>
int main ()
{
  for (int n=10; n>0; n--) {
    if (n==5) continue;
    cout << n << ", ";
  }
  cout << "FIRE!";
  return 0;
}
```

3. Buatlah program dengan output sbb :

MENU

---

1. Deret angka
2. Deret Huruf
3. Keluar

---

Pilihan :

apabila memilih

1.

1 2 3 4 5

A B C D

1 2 3

A B

1

2.

A

B C

D E F

G H I J

11 12 13 14 15

## P. 3.4 Daftar Pustaka

1. Ayuliana, modul pengenalan bahasa C++, Gunadarma Jakarta, February 2004

2. Hari, Konsep Dasar Objek Oriented Programming, FTI budiluhur Jakarta, 2003

3. r.hubbard, John , schaum's outline of theory and problems of programming with C++ second edition, mcgraw-hill, New York 2000

4. http://www.cplusplus.com/

5. http://cs.binghamton.edu/~steflik/

6. http://en.wikipedia.org/wiki/c++