PERULANGAN FOR BAHASA C++

- Perulangan akan membantu kita mengeksekusi kode yang berulang-ulang, berapapun yang kita mau.
- Ada empat macam bentuk perulangan pada C++.
- Secara umum, dibagi menjadi dua kelompok.
- Yaitu: *counted loop* dan *uncounted loop*.
- Perbedaanya:
- **Counted Loop** merupakan perulangan yang jelas dan sudah tentu banyak kali perulangannya.
- Sedangkan **Uncounted Loop**, merupakan perulangan yang tidak jelas berapa kali ia harus mengulang.

PENGERTIAN STRUKTUR PERULANGAN FOR BAHASA C++

```
for (start; condition; increment)
{
   // kode program
   // kode program
}
```

Start adalah kondisi pada saat awal perulangan. Biasanya kondisi awal ini berisi perintah untuk memberikan nilai kepada **variabel counter**. *Variabel counter* sendiri adalah sebuah variabel yang menentukan berapa banyak perulangan dilakukan. Kebanyakan programmer menggunakan variabel **i** sebagai variabel counter (ini tidak harus, boleh juga memakai variabel lain).

Condition adalah kondisi yang harus dipenuhi agar perulangan berjalan. Selama kondisi ini terpenuhi, maka *compiler* bahasa C++ akan terus melakukan perulangan. Misalnya *condition* ini berisi perintah **i < 7**, maka selama *variabel counter* **i** berisi angka yang kurang dari 7, terus lakukan perulangan.

Increment adalah bagian yang dipakai untuk memproses variabel counter agar bisa memenuhi kondisi akhir perulangan. Bagian ini akan selalu di eksekusi di setiap perulangan.

CONTOH KODE PROGRAM PERULANGAN FOR BAHASA C++

Sebagai contoh pertama, saya ingin menampilkan teks "**Hello World**" sebanyak 5 kali. Berikut kode programnya:

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    int i;
8    for (i = 1; i < 5; i++) {
9      cout << "Hello World " << endl;
10   }
11
12   return 0;
13 }
```

Di baris 7 saya membuat sebuah variabel **i** yang di set dengan tipe data **integer**. Variabel ini nantinya akan dipakai sebagai **variabel counter**, yakni variabel yang menentukan kondisi akhir perulangan.

Perintah di baris 8, yakni **for (i = 1; i < 5; i++)**, bisa dibaca:

*"Jalankan perulangan, mulai dari variabel **i = 1** sampai **i < 5**. Dalam setiap iterasi, naikkan nilai variabel i sebanyak 1 angka menggunakan perintah **i++**".*

DI DALAM PERULANGAN, KITA JUGA BISA MENGAKSES *VARIABEL COUNTER* SEPERTI CONTOH BERIKUT:

Sekarang setelah teks "**Hello World**", tampil angka yang berasal dari nilai variabel i. Karena dalam setiap iterasi variabel counter **i** akan naik 1 angka (proses *increment*), maka ketika ditampilkan juga akan naik 1 angka untuk setiap iterasi.

Variabel counter i juga tidak harus di increment, tapi juga bisa di *decrement* untuk membuat perulangan menurun. Berikut contohnya:

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    int i;
8    for (i = 5; i >= 1; i--) {
9      cout << "Hello World " << i << endl;
10   }
11
12   return 0;
13 }
```

Hasil kode program:

```
Hello World 5
Hello World 4
Hello World 3
Hello World 2
Hello World 1
```

Sebagai contoh terakhir, bisakah anda membuat perulangan untuk menampilkan angka kelipatan 3 sebanyak 10 kali? Hasil akhir yang kita inginkan adalah sebagai berikut:

```
3 6 9 12 15 18 21 24 27 30
```

```cpp
#include <iostream>

using namespace std;

int main()
{
  int i;
  for (i = 1; i <= 10; i++) {
    cout << i*3 << " ";
  }

  return 0;
}
```

Cara kedua adalah memodifikasi proses *increment* dari variabel counter:

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    int i;
8    for (i = 3; i <= 30; i = i + 3) {
9      cout << i << " ";
10   }
11
12   return 0;
13 }
```

Perhatikan perintah perulangan for di baris 8. Perintah **for (i = 3; i <= 30; i = i + 3)** bisa dibaca:

*"Jalankan perulangan, mulai dari variabel **i = 3** sampai **i <= 30**. Dalam setiap iterasi, naikkan nilai variabel i sebanyak 3 angka menggunakan perintah **i = i + 3**".*

Teknik ini agak jarang dipakai, tapi itu bisa dilakukan.

LATIHAN

- Buatlah program untuk menentukan bilangan genap mulai dari 1 hingga data ke n.