Bab 6
Perulangan

Dalam kehidupan sehari-hari kita seringkali harus mengulang-ulang sesuatu. Misalnya saya membeli 10 buah buku, lalu ingin menyampulnya dengan plastik. Maka saya mengulang kegiatan "menyampul buku" sebanyak 10 kali. Contoh lain misalnya saya membuat kue kering, setelah membuat adonan, saya mengulang kegiatan "mencetak adonan" hingga adonan tersebut habis.

Kedua contoh diatas sama-sama kegiatan mengulang, namun ada sedikit perbedaan. Pada contoh pertama saya sudah tahu berapa kali saya harus mengulang kegiatan "menyampul buku", yaitu 10 kali. Sedangkan pada contoh kedua, saya tidak tahu berapa kali saya harus mengulang kegiatan "mencetak adonan", tapi saya tahu saya harus berhenti ketika adonan habis.

Pada saat membuat program, kita juga seringkali perlu mengulang-ulang perintah. Pada sebagian kasus kita sudah tahu berapa kali harus mengulang, tapi pada sebagian kasus lainnya kita hanya diberitahu kondisi berhentinya.

Perulangan For

Salah satu jenis perulangan yang sering digunakan dalam bahasa C adalah perintah `for`. Sintaks perintah `for` adalah seperti berikut :

```
for(<inisialisasi> ; <kondisi> ; <langkah selanjutnya>)
{
  <perintah-perintah>
}
```

Pada umumnya for digunakan untuk jenis perulangan dimana kita sudah tahu berapa kali harus mengulang, tapi tidak menutup kemungkinan kita menggunakannya untuk jenis lainnya.

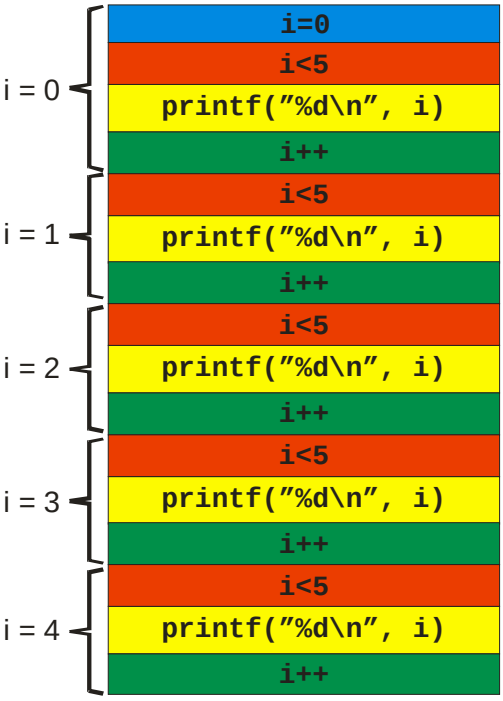
Untuk jenis perulangan pertama, kita gunakan sebuah variable untuk iterasi. Variable ini mula-mula diisi sebuah nilai awal, misalnya 0 (nol). Setiap kali mengulang, variable ini ditambah nilainya, sehingga kita tahu sudah berapa kali mengulang. Misalnya pada suatu saat variable tersebut bernilai 10, maka kita tahu bahwa kita sudah mengulang sebanyak 10 kali.

Supaya program kita berhenti mengulang-ulang, kita harus menambahkan sebuah kondisi. Kondisi ini akan diperiksa setiap kali akan mengulang, jika nilainya true, maka boleh lanjut, tapi jika nilainya false, program akan berhenti. Pada contoh kita menyampul 10 buku, maka kondisinya adalah "variable iterasi bernilai < 10". Kenapa ? Karena ketika variable tersebut mencapai nilai 10, maka kita tahu kesepuluh buku itu sudah tersampul semua. Perhatikan bagan dibawah ini !

`<inisialisasi>` merupakan perintah-perintah yang dikerjakan pertama dan hanya 1 kali saja.

`<kondisi>` merupakan sebuah kondisi yang akan diperiksa saat sebelum `<perintah-perintah>` dilakukan. Jika kondisi bernilai true, maka `<perintah-perintah>` dilakukan.

`<langkah selanjutnya>` merupakan perintah-perintah yang dikerjakan setiap kali `<perintah-perintah>` selesai dilakukan.



Contoh penggunaan for :

```
for(i=0 ; i<5 ; i++)
  printf("%d\n", i);
```

Perintah for diatas akan mencetak angka 0 sampai 4 ke layar. Disini kita menggunakan variable i sebagai variable iterasi. Perintah diatas dapat digambarkan seperti bagan disamping.

Perulangan While

Perulangan while merupakan salah satu alternatif yang banyak digunakan untuk kasus-kasus dimana kita tidak tahu berapa kali harus mengulang, tapi tahu kondisi berhentinya.
Sintaks while adalah seperti berikut :

```
while (<kondisi>)
{
  <perintah-perintah>
}
```

Berbeda dari perintah `for`, perintah ini jauh lebih sederhana. Perulangan `while` tidak menggunakan inisialisasi dan langkah selanjutnya. Namun demikian, sebenarnya prinsip perulangan `while` sama saja dengan `for`. Kita dapat menambahkan inisialisasi sebelum perintah `while` jika memang diperlukan. Kita juga dapat menambahkan langkah selanjutnya didalam blok `while` jika diperlukan.

Perhatikan contoh `while` dibawah ini :

```
i = 0;
while(i<5)
{
  printf("%d\n", i);
  i++;
}
```

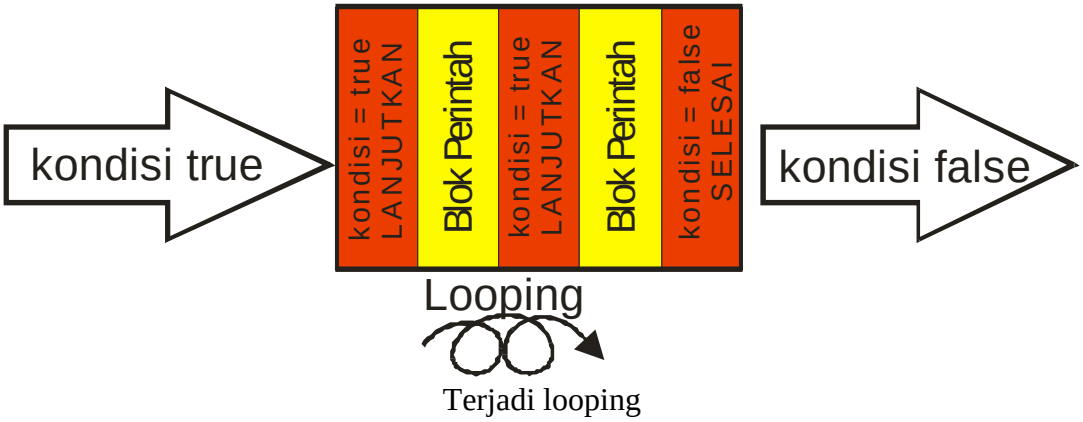
Potongan program diatas melakukan perintah-perintah yang sama dengan contoh `for` sebelumnya. Pada kasus ini kita butuh memberi nilai awal pada `i`, sehingga kita tambahkan perintah `i = 0`; sebelum baris while.

Perhatikan bahwa di dalam blok harus ada perintah yang dapat membuat kondisi menjadi *false,* supaya pada suatu saat perulangan kita dapat berakhir. Pada contoh diatas, kondisi adalah `i<5` , mula-mula `i` diisi 0, sehingga kondisi bernilai *true*. Di dalam blok `while` ada perintah `i++`, maka suatu saat kondisi `i<5` akan menjadi *false*.

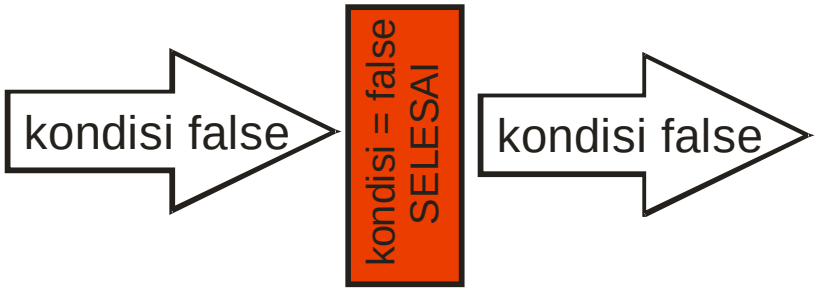
Perulangan Do-While

Pada kedua jenis perulangan diatas, kita memiliki sebuah kondisi yang diperiksa setiap kali akan menjalankan blok perintah. Jika kondisi *true*, barulah blok perintah dijalankan, tapi jika *false*, blok perintah tidak dijalankan dan perulangan berakhir.

Pada umumnya saat masuk ke perulangan kondisi bernilai *true*, lalu kemudian di dalam blok ada perintah yang membuat kondisi tersebut berubah menjadi *false*, barulah saat itu perulangan berakhir.
Tapi ada kalanya kita membuat sebuah perulangan yang kondisinya berkaitan dengan perintah-perintah sebelumnya, sehingga mungkin saja pada saat awal kondisinya sudah *false*. Pada kasus ini perulangan sama sekali tidak terjadi.

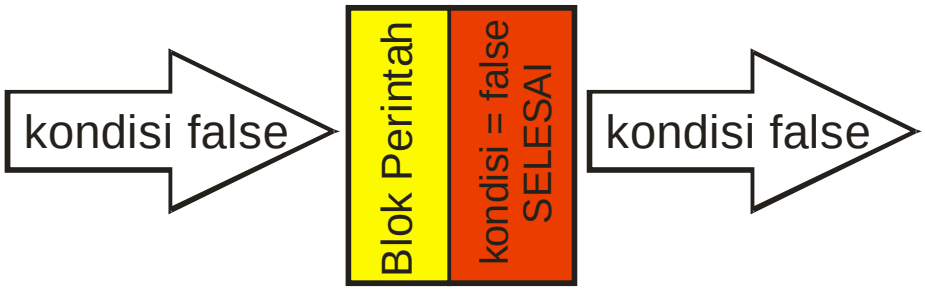


Terjadi looping



Tidak terjadi looping

Perulangan Do-While sebenarnya mirip dengan perulangan While, tapi disini pemeriksaan kondisi dilakukan di akhir. Hal ini menyebabkan blok program pasti dijalankan minimal 1 kali.



Sintaks Do-While adalah seperti berikut :

```
do
{
  <perintah-perintah>
}while(<kondisi>);
```

Contoh penggunaan Do-While :

```
i = 0;
do
{
  printf("%d\n",i);
  i++;
}while(i<5);
```