**LOOPING**

***Loop atau perulangan*** adalah suatu bentuk kegiatan mengulang suatu statement sampai batas yang diinginkan. Dalam pemograman Loop atau perulangan sangat berguna sekali. Kegunaan sederhana dalam perulangan terlukis seperti ini: Jika kita ingin menuliskan kata “saya ganteng” sebanyak seribu kali maka kita harus membuat seribu baris kata “saya ganteng” dalam kode pemrogaman kita. Tapi dengan adanya perulangan maka kita hanya cukup menuliskan 5 sampai 7 baris kode saja. Dalam praktek sebenarnya, tentu saja perulangan sangat membantu sekali. Misal untuk memuat seluruh isi table dari sebuah database dimana datanya ingin kita tampilkan.  
Loop atau perulangan dalam Java dapat dilakukan dengan 3 teknik:

1. ***Statement for to do*** merupakan perulangan dengan perhitungan dari nilai kecil ke besar, dan akan melakukan perulangan mulai dari awal sampai nilai akhir terpenuhi.

Perulangan dengan teknik ini dikontrol oleh tiga bagian yang ada dalam tanda kurung dan masing-masing bagian ini dipisahkan oleh titik-koma. Pada bagian pertama(inisialisasi ekspresi), sebuah variabel akan di deklarasikan sebagai sebuah titik awal dari perulangan, biasanya variable ini mempunyai tipe data integer atau float. Sementara pada bagian kedua disinilah perulangan akan diperiksa apakah masih memenuhi syarat atau tidak, jika masih memenuhi syarat maka statement dibawahnya akan di eksekusi. Sedangkan bagian ketiga adalah bagian dimana jika bagian kedua masih memenuhi syarat maka nilai variabel akan ditambahkan sesuai dengan syarat yang dituliskan. Bagian ketiga ini secara otomatis akan tidak dibaca oleh program jika kondisi pada bagian ke-dua sudah tidak lagi memenuhi syarat, dan perulangan pun menjadi terhenti.

Contoh:

Program fortodo;

Uses crt;

Var

I:integer;

Begin

Clrscr;

For i:= 1 to 4 do

Write(i+i);

Readln;

End.

1. ***Statement for down to*** merupakan perulangan dengan perhitungan dari nilai besar ke   kecil, dan akan melakukan perulangan mulai dari awal sampai nilai akhir terpenuhi.

Contoh:

Program fordowntodo;

Uses crt;

Var

i: integer;

begin

clrscr;

for i:=4 downto 1 do

write(i+i);

readln;

end.

1. ***Statement while do*** merupakan perulangan yang akan di eksekusi selama kondisi while terpenuhi.

Perulangan dengan menggunakan teknik while ini sebenarnya adalah suatu bentuk perulangan yang memodifikasi teknik pencabangan (branching) secara kasar. Pernyataan yang ada didalam blok perulangan akan dieksekusi dengan cara memeriksa ekspresi yang ada, sepanjang ekspresi bernilai true maka statement akan terus di eksekusi. Variabel sebagai kontrol perulangan bentuk ini diinisialisai di luar blok perulangan ini. Dan penambahan atau increment nilai variabel berada didalam blok perulangan ini. Kelebihan perulangan dengan bentuk ini adalah variabel yang dideklarasikan tidak hanya bertipe integer atau float saja namun bisa juga bertipe boolean atau string.

Contoh:

Program whiledo;

Uses crt;

Var

i:integer;

begin

clrscr;

i:=2;

while i<=8 do

begin

write(i);

i:=i+2;

end;

readln;

end.

1. ***Statement repeat until*** merupakan kondisi dimana perulangan diperiksa setelah pengeksekusian pernyataan yang ada dalam pernyataan repeat sampai kondisi until terpenuhi.

Repeat berarti ulangi dan until berarti sampai. Jadi, repeat-until adalah struktur pengulangandimana aksi dilakukan hingga kondisi (persyaratan) berhenti terpenuhi.

Kondisi dalam Pengulangan :

1.  Repeat n (Aksi) Times

     \*berarti pernyataan dimana kondisi awalnya sudah diketahui\*

2.  Repeat n (Kondisi) Until

     \*berarti pernyataan dimana kondisi awalnya tidak diketahui\*

Contoh:

Program repeatuntil;

Uses crt;

Var

i:byte;

begin

clrscr;

i:=0;

begin

repeat

i:=i+2;

write(i);

until i=8

end;

readln;

end.

Jadi inti dalam sequential programming adalah bahwa sebuah program itu dikerjakan dengan jalan berurutan langkah-langkahnya. setahap demi setahap. secara umum diawali dengan inputan kemudian pengolahan input setelah itu hasil.

**PERCABANGAN / BRANCHING**

Percabangan / branching adalah struktur pengambilan keputusan yang digunakan untuk memilih salah satu alternatif jawaban yang tepat dari pilihan yang ada. Dalam bahasa ruby menyediakan perintah IF dan perintah CASE – WHEN.

Perintah percabangan / branching berguna untuk memilih salah satu dari 2 atau lebih alternatif jawaban yang tersedia. Setiap perintah IF harus di akhiri dengan END untuk yang menandakan akhir dari blok pernyataan IF.

1.     Perintah IF ...

Jika kondisi bernilai benar maka pernyataan di bawahnya akan dikerjakan.

2.     Perintah IF ... ELSE ...

Jika kondisi bernilai benar maka pernyataan 1 s/d m akan dikerjakan dan jika kondisi bernilai salah maka pernyataan 4 s/d n yang akan dikerjakan.

3.     Perintah IF ... ELSIF ... ELSE ... (IF bertingkat)

jika kondisi bernilai benar maka pernyataan 1 yang akan dikerjakan dan apabila kondisi 1 tidak benar maka kondisi 2 yang akan dilihat.Jika kondisi 2 bernilai benar maka pernyataan 2 yang akan dikerjakan dan apabila kondisi 2 tidak benar maka kondisi 3 yang akan dilihat sampai pernyataan terakhir.

4.     Perintah IF bersarang (IF dalam IF)

Jika kondisi 1 bernilai benar maka dilanjutkan pengecekan ke kondisi 2. Tetapi jika kondisi 1 bernilai salah maka akan dilanjutkan ke bagian else dan kemudian dilakukan pengecekan pada kondisi 3.

5.     Perintah CASE

pernyataan yang dirancang untuk menangani pengambilan keputusan yang melibatkan srjumlah pilihan alternatif yang diantaranya untuk menggantikan pernyataan IF bertingkat