MODUL 7
Perulangan 1

KOMPETENSI

- Mahasiswa memahami format penulisan program perulangan bagian 1
- Mahasiswa mampu mengimplementasikan flowchart perulangan bagian 1 menggunakan bahasa pemrograman Java

MATERI DASAR

Perulangan adalah suatu blok atau kelompok instruksi yang dilaksanakan secara berulangulang. Perulangan akan membuat efisiensi proses dibandingkan jika dioperasikan secara manual. Kebanyakan aplikasi perangkat lunak melakukan pekerjaan berulang-ulang sampai sebuah kondisi yang diinginkan, oleh karena itu pengulangan merupakan bagian yang penting dalam pemrograman karena dengan adanya pengulangan pembuat program tidak perlu menulis kode program sebanyak pengulangan yang diinginkan.

Perulangan yang dijelaskan pada jobsheet ini adalah :

- Perulangan dengan for
- Perulangan dengan while
- Perulangan dengan do-while

For

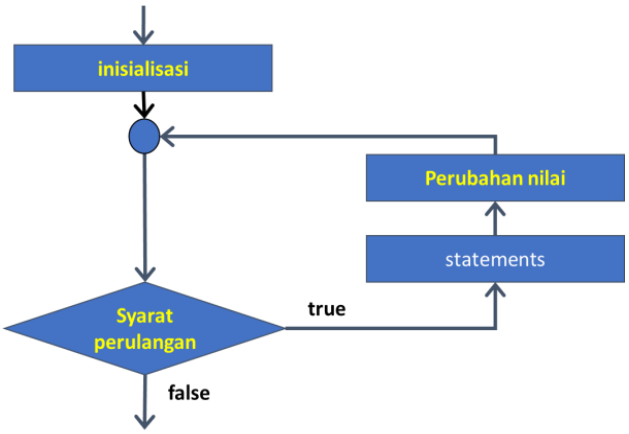
For adalah kode yang digunakan untuk menjalankan serangkaian kode secara berulang-ulang. Pada kode for ini terdapat beberapa komponen yang dicantumkan, antara lain: (1) inisialisasi, (2) kondisi, (3) perubahan nilai, (4) statement yang diulang. Berikut ini format sintaks untuk kode for.

```
for ([inisialisasi]; [kondisi]; [perubahan_nilai]) statement; atau:
for ([inisialisasi]; [kondisi]; [perubahan_nilai]){
        statement1;          statement2;
                ……
}
```

Ekspresi **inisialisasi** adalah suatu ungkapan yang memberikan nilai awal suatu variabel untuk perulangan (misalnya x=0).

Ekspresi **kondisi** adalah suatu ungkapan yang menunjukkan suatu kondisi yang harus dipenuhi agar perulangan dapat terus dilakukan (misalnya x<=20). Ekspresi ini harus diisi karena looping harus dapat berhenti.

Ekspresi **perubahan_nilai** adalah suatu ungkapan yang merubah nilai-nilai variabel pengontrol perulangan setiap saat perulangan dilakukan (misalnya x++). Ekspresi perubahan_nilai dapat menggunakan nilai increment dan decrement. Perubahan nilai juga dapat menggunakan operator yang lain seperti contohnya x+=2, x-=3, x*=4, x/=5 Di bawah ini adalah flowchart perulangan FOR



Berikut ini adalah contoh skrip untuk mencetak tulisan “Hello dasar pemrograman” sebanyak 10 kali.

```
for (int a = 0; a < 10; a++) {
        System.out.println(“Hello dasar pemrograman”);
}
```

Lihat perbandingannya bila kita melakukan pencetakan tulisan “Hello dasar pemrograman” sebanyak 10 kali dan tidak menggunakan perulangan

```
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
System.out.println(“Hello dasar pemrograman”);
```

Ekspresi **kondisi** dapat diisi dengan sebuah variabel Boolean. Seperti contoh di bawah ini

```
Scanner input = new Scanner(System.in);
int angka, i;
boolean stop=false;

for (i=0; !stop; i++){
    System.out.print("Masukkan angka : ");
    angka = input.nextInt();
    System.out.println("Angka yang anda masukkan: "+angka);
    if (angka == 0)
    stop = true;
}
System.out.println("Selesai.");
```

Akan menghasilkan keluaran:

```
Masukkan angka : 16
Angka yang anda masukkan: 16
Masukkan angka : 10
Angka yang anda masukkan: 10
Masukkan angka : 2020
Angka yang anda masukkan: 2020
Masukkan angka : 0
Angka yang anda masukkan: 0
Selesai.
```

Ekspresi **inisialisasi** dan ekspresi **perubahan_nilai** dapat dikosongi sesuai dengan kebutuhan. Mengingat ekspresi kondisi dapat berisi variabel boolean, sehingga terkadang tidak perlu ada inisialisasi dan perubahan nilai pada loop nya. Seperti contoh di bawah ini:

```
Scanner input = new Scanner(System.in);
int angka, i;
boolean stop=false;

for (; !stop;){
    System.out.print("Masukkan angka : ");
    angka = input.nextInt();
    System.out.println("Angka yang anda masukkan: "+angka);
    if (angka == 0)
    stop = true;
}
System.out.println("Selesai.");
```

Akan menghasikan keluaran:

```
Masukkan angka : 10
Angka yang anda masukkan: 10
Masukkan angka : 25
Angka yang anda masukkan: 25
Masukkan angka : 0
Angka yang anda masukkan: 0
Selesai.
```

Ekspresi **inisialisasi** dan ekspresi **perubahan_nilai** juga dapat terdiri dari beberapa ekspresi yang dipisahkan dengan koma. Seperti contoh di bawah ini:

```
int i,j;
for(i=1, j=30; i<j; i++, j--)
    System.out.printf("%04d -- %04d\n",i,j);
```

Akan menghasilkan keluaran

```
0001 -- 0030
0002 -- 0029
0003 -- 0028
0004 -- 0027
0005 -- 0026
0006 -- 0025
0007 -- 0024
0008 -- 0023
0009 -- 0022
0010 -- 0021
0011 -- 0020
0012 -- 0019
0013 -- 0018
0014 -- 0017
0015 -- 0016
```

While

Kode while merupakan kode alternatif untuk melakukan perulangan selain for. Perulangan while biasanya digunakan jika jumlah perulangan tidak diketahui atau memiliki kemungkinan dapat dilakukan kurang dari batas perulangan yang telah ditentukan. Perulangan while hanya akan melakukan perulangan selama kondisi perulangan terpenuhi. Perintah-perintah akan dilaksanakan apabila ekspresi boolean dalam keadaan true. Di dalam loop ada nilai yang mengontrol loop dan nilainya harus berubah, sehingga pada akhir program akan keluar dari loop.

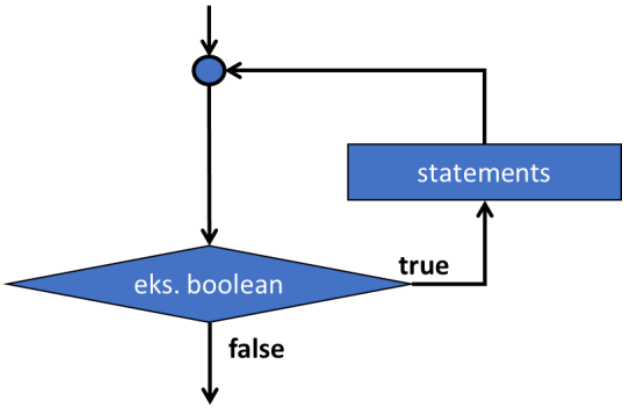
Cara kerjanya sama, namun sintaks (aturan penulisan) yang berbeda. Berikut sintaks while

```
while(kondisi) statement;
atau
while(kondisi) {   statement;   perubahan nilai;
```

Berikut ini adalah contoh skrip untuk mencetak tulisan “Hello dasar pemrograman” sebanyak 10 kali.

```
int a = 0; while (a <
10) {
        System.out.println(“Hello dasar pemrograman”);
}
```

Flowchart perulangan while ditunjukkan pada gambar di bawah ini:



Do - While

Kode do-while merupakan kode while-do dengan sintaks yang berbeda. Cara kerja do-while relatif sama dengan while. Perintah do-while() akan mengulang statement miliknya selama syarat pengulangannya terpenuhi. Hanya saja, perintah do-while() menjalankan statementnya terlebih dahulu, setelah itu baru memeriksa syaratnya. Sedangkan perintah while() memeriksa syarat terlebih dahulu. Oleh karena itu, perintah do-while() akan menjalankan statementnya paling tidak sebanyak satu kali, meskipun syarat pengulangan tidak terpenuhi.

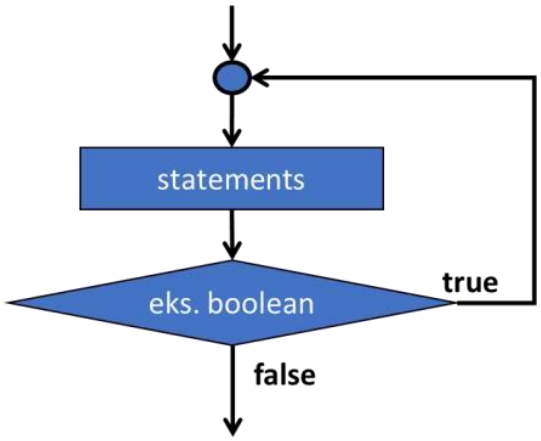
Berikut sintaks untuk do-while.

```
do    {                    statement;
perubahan_nilai;
} while (kondisi);
```

Berikut ini adalah contoh skrip untuk mencetak tulisan “Hello dasar pemrograman” sebanyak 10 kali menggunakan do-while.

```
int a = 0; do {
        System.out.println(“Hello”); a++;
} while (a < 10);
```

Berikut ini gambaran flowchart perulangan do-while:



Ketiga jenis loop tersebut sama-sama memiliki kondisi yang merupakan batasan suatu perulangan dilakukan. Batasan yang menjadi kondisi suatu perulangan didefinisikan dulu di awal, kemudian dilanjutkan dengan statement yang harus di-looping. Berbeda dengan for dan while, pada perulangan do-while, setelah inisialisasi dilakukan, akan di proses dulu statement yang harus dijalankan, baru kemudian dilakukan pembatasan looping dalam penulisan kondisi.

Mengontrol Loop

Perulangan atau loop dalam program dapat dikontrol dengan menggunakan:

- Sentinel
- Pertanyaan

Sentinel ini disebut juga dengan signal value atau dummy value atau flag value, dengan tujuan untuk memberitahu bahwa penggunaan sentinel tersebut adalah “akhir dari penginputan data”. Cara kerja sentinel ini, user akan memasukkan data sesuai dengan kebutuhannya, dan setelah data tersebut semuanya diinput, kemudian user akan memasukkan nilai sentinel untuk mengindikasikan

bahwa tidak ada lagi data yang dimasukkan. Pengontrolan loop dengan sentinel disebut juga dengan loop yang tidak terhingga, karena jumlah perulangan tidak diketahui sebelum loop dieksekusi.

Contoh : Cara 'sentinel' pada konstruksi do-while dengan memakai nilai 0 pada variabel panjang dan variabel lebar.

```
int p, l, luas;
Scanner input=new Scanner(System.in);
System.out.println("PROGRAM MENGHITUNG LUAS PERSEGI PANJANG\n");
do{
    System.out.print("Panjang [0=selesai] : ");
    p=input.nextInt();
    System.out.print("Lebar [0=selesai] : ");
    l=input.nextInt();
    luas=p*l;
    System.out.printf("Luas = %d x %d = %d\n\n",p,l,luas);
}while((p!=0)&&(l!=0));
System.out.print("===============selesai==============");
```

Keluaran Program:

```
PROGRAM MENGHITUNG LUAS PERSEGI PANJANG

Panjang [0=selesai] : 20
Lebar [0=selesai] : 13
Luas = 20 x 13 = 260

Panjang [0=selesai] : 4
Lebar [0=selesai] : 0
Luas = 4 x 0 = 0

===============selesai==============
```

Penghentian looping juga dapat menggunakan pertanyaan yang diberikan sebelum mengakhiri satu step loop. Pada saat itu, user diberikan pertanyaan apakah mau menghentikan / melanjutkan looping. Kemudian user menjawab pertanyaan dengan inputan user dengan kriteria yang ditentukan. Berikut ini contoh program yang menggunakan pertanyaan untuk mengontrol loop:

```
int p, l, luas;
String s;
Scanner input=new Scanner(System.in);
System.out.println("PROGRAM MENGHITUNG LUAS PERSEGI PANJANG\n");
do{
    System.out.print("Panjang : ");
    p=input.nextInt();
    System.out.print("Lebar : ");
    l=input.nextInt();
    luas=p*l;
    System.out.printf("Luas = %d x %d = %d\n\n",p,l,luas);
    System.out.println("Apakah anda ingin menghitung kembali? <Y/T>");
    s=input.next();
}while((s.charAt(0)=='Y')||(s.charAt(0)=='y'));
System.out.print("===============selesai==============");
```

Akan menghasilkan keluaran:

```
PROGRAM MENGHITUNG LUAS PERSEGI PANJANG

Panjang : 14
Lebar : 10
Luas = 14 x 10 = 140

Apakah anda ingin menghitung kembali? <Y/T>
y
Panjang : 22
Lebar : 9
Luas = 22 x 9 = 198

Apakah anda ingin menghitung kembali? <Y/T>
t
===============selesai==============
```

Break dan Continue

Break dan continue tergolong ke dalam keyword di bahasa pemrograman java, yang keduanya digunakan pada suatu kondisi tertentu , pada perulangan seperti while ,do while dan for. Jika fungsi break digunakan untuk menghentikan suatu pernyataan (statement), dan jika fungsi continue digunakan untuk mengabaikan ,lalu melanjutkan suatu pernyataan pada perulangan. Keyword break dan continue juga biasa digunakan, bersamaan dengan Control Flow seperti if else, dan switch case di dalam program java

Contoh penggunaan statement break pada loop yang menyebabkan program keluar dari loop tersebut:

```
int x = 1;
while (x<=10) {
      System.out.printf( "%d\n", x );
      x++;
      if (x>5) break;
}
```

Keluar dari loop

Berikut ini contoh penggunaan continue untuk keluar dari step looping dan melanjutkan

```
int x;
for(x=1; x<=10; x++) {        if (x == 5) continue;
      System.out.print(x);
}
```

Output : 1 2 3 4 6 7 8 9 10

Percobaan 1

Menghitung nilai factorial menggunakan perulangan for, while dan do-while

1. Tulis ulang program untuk melakukan perulangan sebagai berikut :

a) Perulangan dengan **for**

```
public static void main(String[] args) {
  int angka, fac, i;
  System.out.println ("=====PROGRAM MENGHITUNG NILAI FAKTORIAL DENGAN FOR=====");
  System.out.print("Masukkan bilangan : ");

  Scanner input = new Scanner (System.in);
  angka = input.nextInt();
  fac =1;
  for (i=1; i<=angka;i++)
  {
    fac = fac*i;
  }
  System.out.printf("Nilai faktorial bilangan tersebut adalah : %d \n", fac);
}
}
```

b) Perulangan dengan **while**

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, fac, i;
  System.out.println ("=====PROGRAM MENGHITUNG NILAI FAKTORIAL DENGAN WHILE=====")
  System.out.print("Masukkan bilangan : ");

  angka = input.nextInt();
  fac =1;
  i = 1;
  while (i<=angka)
  {
    fac = fac*i;
    i++;
  }
  System.out.printf("Nilai faktorial bilangan tersebut adalah : %d \n", fac);
}
```

c) Perulangan dengan **do-while**

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, fac, i;
  System.out.println ("=====PROGRAM MENGHITUNG NILAI FAKTORIAL DENGAN DO-WHILE=====")
  System.out.print("Masukkan bilangan : ");

  angka = input.nextInt();
  fac =1;
  i = 1;
  do
  {
    fac = fac*i;
    i++;
  }
  while (i<=angka);
  System.out.printf("Nilai faktorial bilangan tersebut adalah : %d \n", fac);
}
```

2. Cocokkan hasil *running* program yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?

```
=====PROGRAM MENGHITUNG NILAI FAKTORIAL=====
Masukkan bilangan : 5
Nilai faktorial bilangan tersebut adalah : 120
```

Percobaan 2:

Keluar dari perulangan menggunakan break

1. Salinlah program perulangan dengan menggunakan *break* berikut :

a) Perulangan dengan **for**

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, b;
  System.out.println ("=====PROGRAM LOOP DENGAN BREAK=====");
  for (b=0; true;){
    System.out.print("Masukkan bilangan : ");
    angka = input.nextInt();
    b += angka;

    if (b>50) break;
  }
  System.out.printf("Angka berhenti pada jumlah angka : %d \n", b);
}
```

b) Perulangan dengan **while**

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, b;
  System.out.println ("=====PROGRAM LOOP DENGAN BREAK=====");
  b=0;
  while (true){
    System.out.print("Masukkan bilangan : ");
    angka = input.nextInt();
    b += angka;
    if (b>50) break;
  }
  System.out.printf("Angka berhenti pada angka : %d \n", b);
}
```

b) Perulangan dengan **do-while**

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, b;
  System.out.println ("=====PROGRAM LOOP DENGAN BREAK=====");
  b=0;
  do
  {
    System.out.print("Masukkan bilangan : ");
    angka = input.nextInt();
    b += angka;
    if (b>50) break;
  }
  while (true);
  System.out.printf("Angka berhenti pada angka : %d \n", b);
}
```

2. Cocokkan hasil *running* program *looping* menggunakan *break* yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?

```
=====PROGRAM LOOP DENGAN BREAK=====
Masukkan bilangan : 2
Masukkan bilangan : 12
Masukkan bilangan : 22
Masukkan bilangan : 32
Angka berhenti pada angka : 68
```

Percobaan 3

Keluar dari step perulangan menggunakan continue

1. Salinlah program perulangan dengan menggunakan *continue* berikut :

```
public static void main(String[] args) {
  Scanner input = new Scanner (System.in);
  int angka, b, i, count;
  double avg, total;
  System.out.println ("=====PROGRAM LOOP DENGAN CONTINUE=====");
  b=0;
  count=0;

  for (i=0;i<5;i++){
    System.out.print("Masukkan bilangan : ");
    angka = input.nextInt();
    if (angka>=50) continue;
    b += angka;
    count ++;
  }
  total =(double)b;
  System.out.printf("Jumlah angka kurang dari 50: %.2f \n", total);
  avg = (double)b/count;
  System.out.printf("Rata-rata angka kurang dari 50: %.2f \n", avg);
}
```

2. Cocokkan hasil *running* program *looping* menggunakan *continue* yang sudah Anda buat apakah sudah sesuai dengan tampilan berikut ini?

```
=====PROGRAM LOOP DENGAN CONTINUE=====
Masukkan bilangan : 30
Masukkan bilangan : 40
Masukkan bilangan : 50
Masukkan bilangan : 60
Masukkan bilangan : 20
Jumlah angka kurang dari 50: 90.00
Rata-rata angka kurang dari 50: 30.00
```

TUGAS

1. Lakukan percobaan diatas. Pahami seluruh percobaan yang telah anda lakukan. Capture seluruh listing program dan hasil running program. Tampilkan hasil capture tersebut dalam laporan praktikum anda.