PERULANGAN

Fundamen Pengembangan Aplikasi

TOPIK MATERI

1. Perulangan FOR
   a. Perulangan FOR TRAVERSAL
   b. Perulangan FOR EACH
2. Perulangan Bersarang (Nested For)
3. Perulangan WHILE
4. Perulangan DO – WHILE

Perulangan FOR Traversal

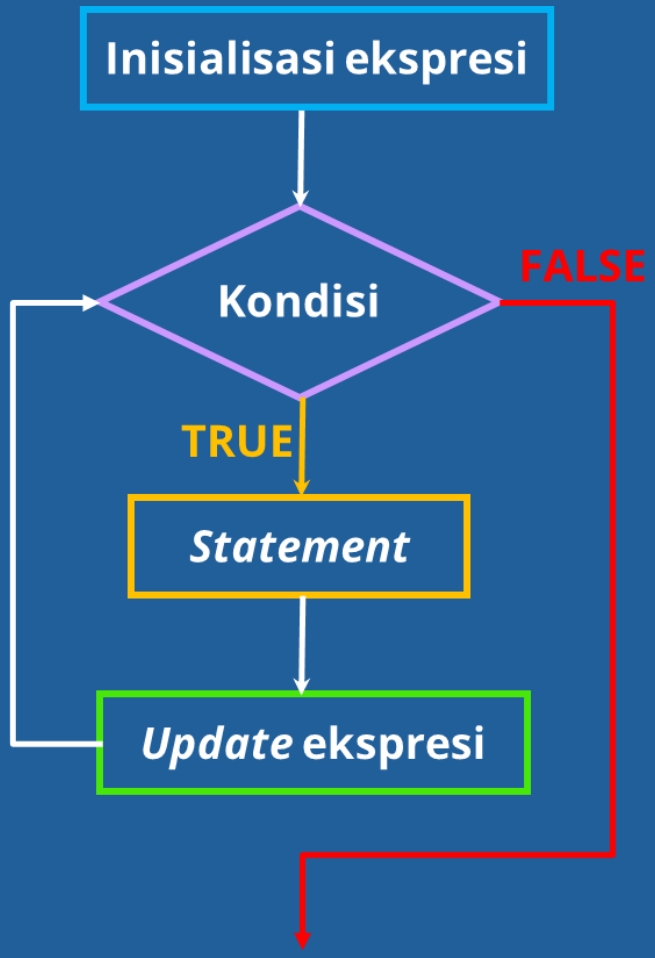
Fundamen Pengembangan Aplikasi

UNIVERSITAS ISLAM INDONESIA

Perulangan: FOR Traversal

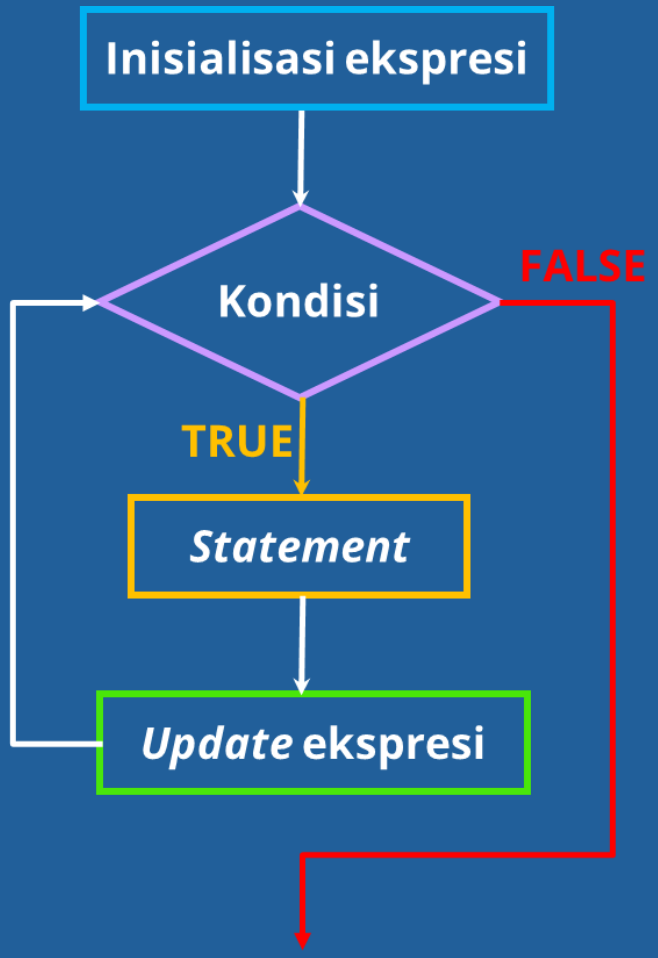
- Ada pernyataan yang harus diulang eksekusinya
- Banyak perulangan sudah diketahui sebelumnya, misal 10 kali, 20 kali
- Banyak perulangan dapat juga berdasarkan nilai variabel
  - Dengan catatan nilai variabel DIKETAHUI sebelum perulangan, baik melalui pernyataan masukan maupun pernyataan pemberian)
- Jadi FOR TRAVERSAL adalah perulangan berdasarkan cacah tertentu
  - Cacah naik
  - Cacah turun

Perulangan: FOR Traversal



- Inisialisasi ekspresi memberikan nilai awal ke variabel pencacah.
- Mengecek nilai variabel pencacah berdasarkan Kondisi.
- Blok Statement dijalankan jika Kondisi bernilai TRUE
- Update ekspresi memperbaharui nilai variabel pencacah
- Jika Kondisi bernilai FALSE, proses perulangan akan berhenti

Perulangan: FOR Traversal



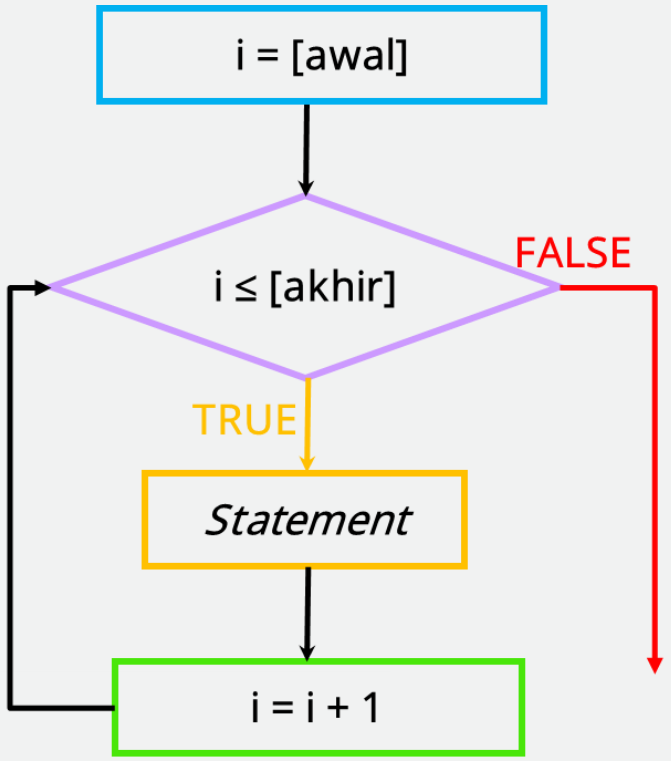
- Syntax di Java

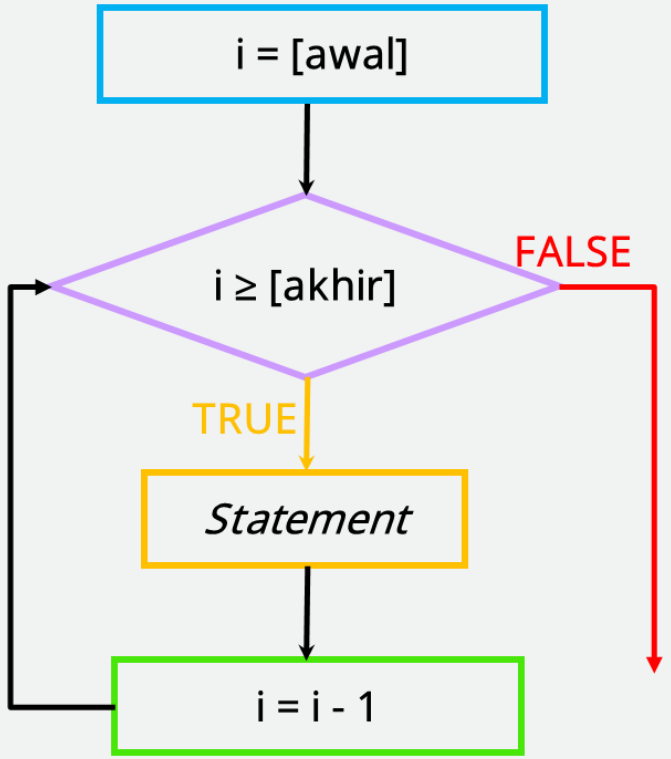
```
for (initialEkspresi; kondisi; updateEkspresi) {

    // statement

}
```

Perulangan: FOR Traversal






```
for (int i=[awal]; i<=[akhir]; i++) {
    statement;
}
```

```
for (int i=[awal]; i>=[akhir]; i--) {
    statement;
}
```

Perulangan: FOR Traversal

```
for (int i = 1; i<=5; i++) {
    System.out.println(i);
}
```

```
PS D:\CODING JAVA\00 - Kuliah FPA\P4\P4>
1
2
3
4
5
```

Perulangan: FOR Traversal

Bagaimana menampilkan nilai variabel ekspresi dari 15 sampai 1 yang bernilai ganjil, menggunakan FOR-LOOP.

```
PS D:\CODING JAVA\00 - Kuliah FPA\P4\P4>
15
13
11
9
7
5
3
1
```

Perulangan FOR EACH

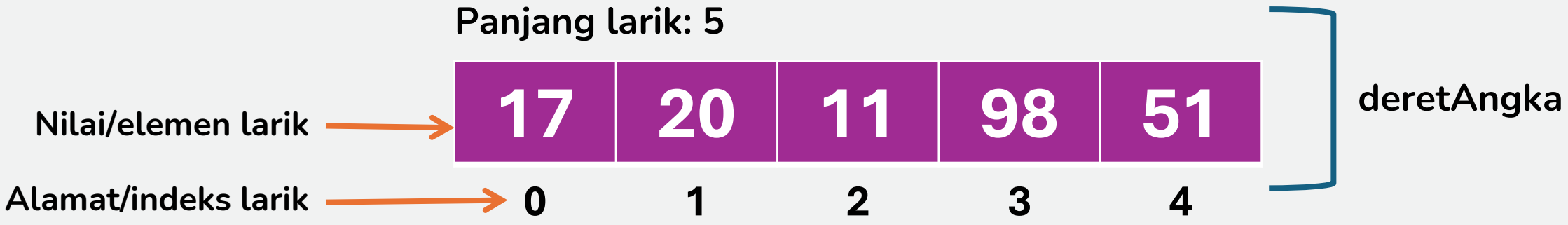
Fundamen Pengembangan Aplikasi

UNIVERSITAS ISLAM INDONESIA

Perulangan: FOR EACH

- Mengapa disebut FOR EACH?
- Perulangan untuk memproses SETIAP HIMPUNAN DATA untuk diproses, contoh:
  - Perulangan pada list (memproses setiap elemen list)
  - Perulangan pada larik (memproses setiap elemen larik)
- Dalam pendekatan pemrograman prosedural, data adalah nilai variabel.
- Dalam bahasa Java yang menggunakan pendekatan pemrograman berorientasi objek, data adalah bagian dari objek (atribut objek).
  - Perulangan FOR EACH merupakan perulangan untuk memproses himpunan objek

Pengantar Larik/Array



- Larik (array) adalah suatu kumpulan data yang memiliki tipe data yang sama.
- Pada gambar di atas terdapat larik bernama deretAngka yang berisi 5 elemen bertipe data integer.
- Alamat atau index larik dimulai dari 0 bukan 1.

Deklarasi Larik/Array

1

Syntax:

```
tipeData[] namaLarik;
```

Contoh:

```
int[] deretAngka;
```

2

Syntax:

```
tipeData []namaLarik;
```

Contoh:

```
int []deretAngka;
```

3

Syntax:

```
tipeData namaLarik[];
```

Contoh:

```
int deretAngka[];
```

*Sebelum menggunakan larik, Anda harus **mendeklarasikan** variabel dengan **tipe acuan** yang mengacu ke larik*

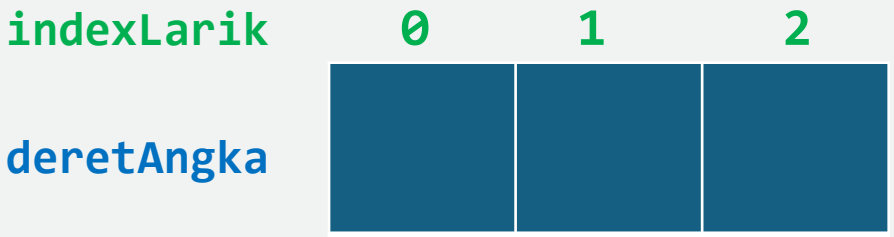
Instansiasi Larik/Array

Sintaks:

```
namaLarik = new tipeData[banyakElemen];
```

Contoh

```
deretAngka = new int[3];
```



- Instansiasi artinya membuat objek dari sebuah class/collection.
- Contoh di atas menunjukkan bahwa larik `deretAngka` yang telah dideklarasikan di slide sebelumnya, diinstansiasi dengan tipe data `int` dan memiliki `3` elemen.
- Panjang atau ukuran larik sifatnya TETAP setelah dilakukannya instansiasi.

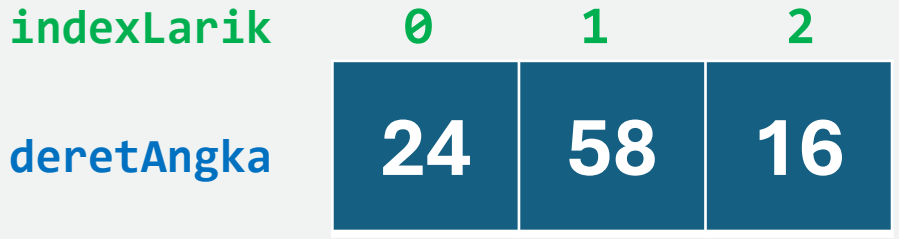
Inisialisasi Larik/Array

Sintaks:

```
namaLarik[indexLarik] = nilai;
```

Contoh

```
deretAngka[0] = 24;
deretAngka[1] = 58;
deretAngka[2] = 16;
```



- Inisialisasi artinya pemberian nilai/data awal sebuah larik.
- Contoh di atas menunjukkan bahwa larik `deretAngka` yang telah dideklarasi dan diinstansiasi di slide sebelumnya, diisi setiap elemen dengan suatu nilai.

Inisialisasi Larik/Array

- Suatu larik dapat juga dideklarasi, diinstansiasi, dan diinisialisasi sekaligus, dengan cara sebagai berikut:

Sintaks:

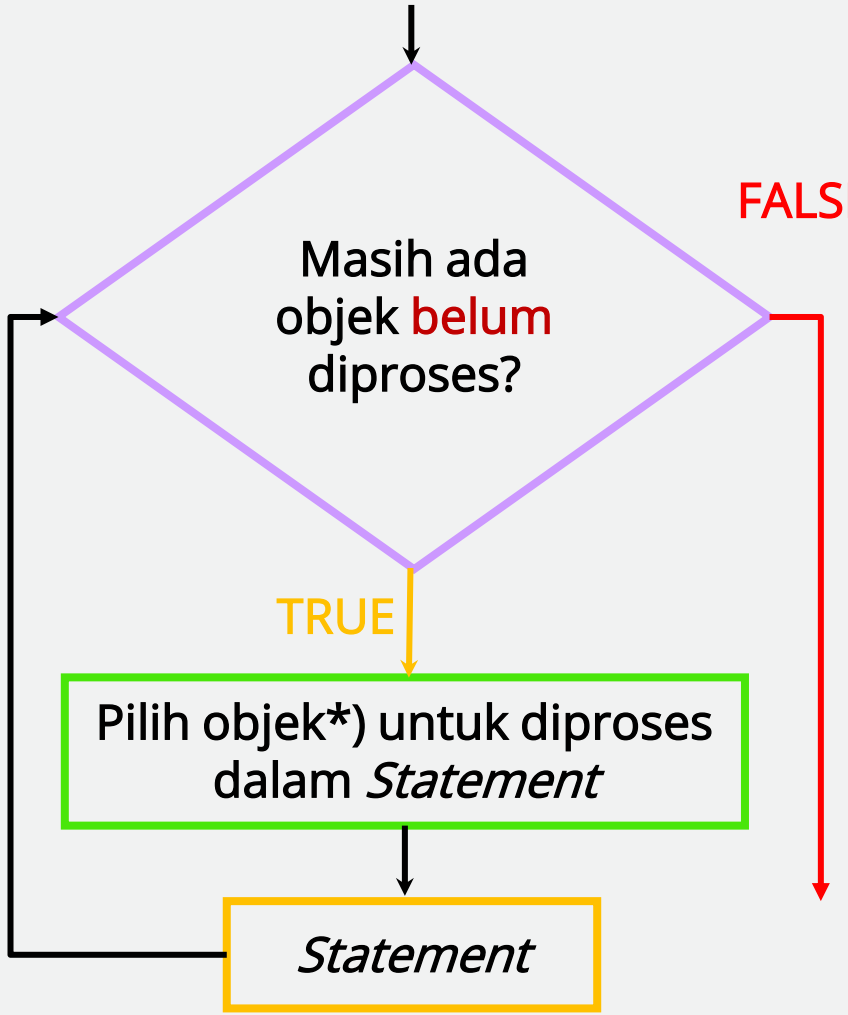
```
tipeData namaLarik[] = {nilai1, nilai2, …, nilai-n};
```

Contoh

```
int deretAngka[] = {24, 58, 16};
```

Gunakan kurung kurawal

Perulangan: FOR EACH



`namaVariabel` akan menyimpan SETIAP objek/elemen yang ada di `namaArray`

Syntax di Java

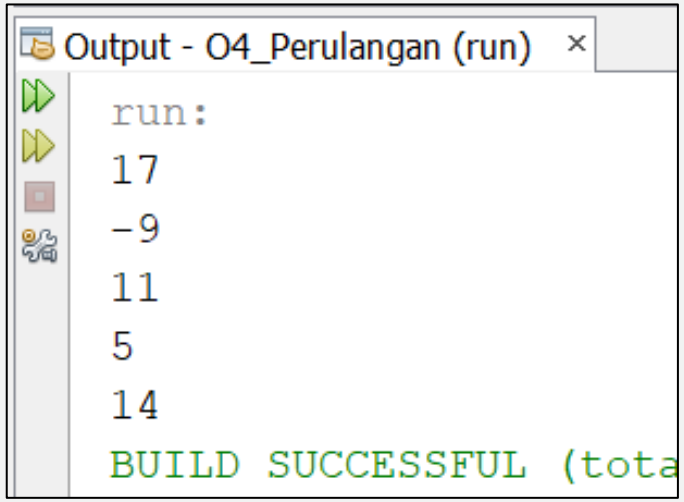
```
for ([tipeData] [namaVariabel]: [namaArray]) {
   // statement
}
```

*) Setiap elemen larik HANYA dapat dipilih/dieksekusi SATU KALI saja

Perulangan: FOR EACH

- Contoh
  - Menampilkan isi dari suatu larik bernama angka terdiri elemen-elemen bertipe integer.
  - Larik tersebut berisi 5 data yaitu 17, -9, 11, 5, 14. Elemen larik akan diproses sebagai nilai dari variabel tampunganNilai
- Code program

```
int angka[] = {17, -9, 11, 5, 14};
for (int tampunganNilai: angka) {
    System.out.println(tampunganNilai);
}
```



PERULANGAN NESTED FOR

Fundamen Pengembangan Aplikasi

UNIVERSITAS ISLAM INDONESIA

Nested FOR

- Ada perulangan di dalam perulangan.
- *Outer loop* akan mengeksekusi seluruh perulangan di *Inner loop* sampai selesai, setelah itu *outer loop* melanjutkan prosesnya
- Kode program:

```
for (i = [awalOut]; i<= [batasOut]; i++) {
    for (j = [awalIn]; j<= [batasIn]; j++) {
        // statement
    }
}
```

Nested FOR: Contoh

- **Contoh:**
  - **Menampilkan nilai variabel ekspresi**

```
for (int i = 1; i<=3; i++) {
    for (int j = 10; j <= 12; j++) {
     System.out.println(i+" "+j);
    }
}
```

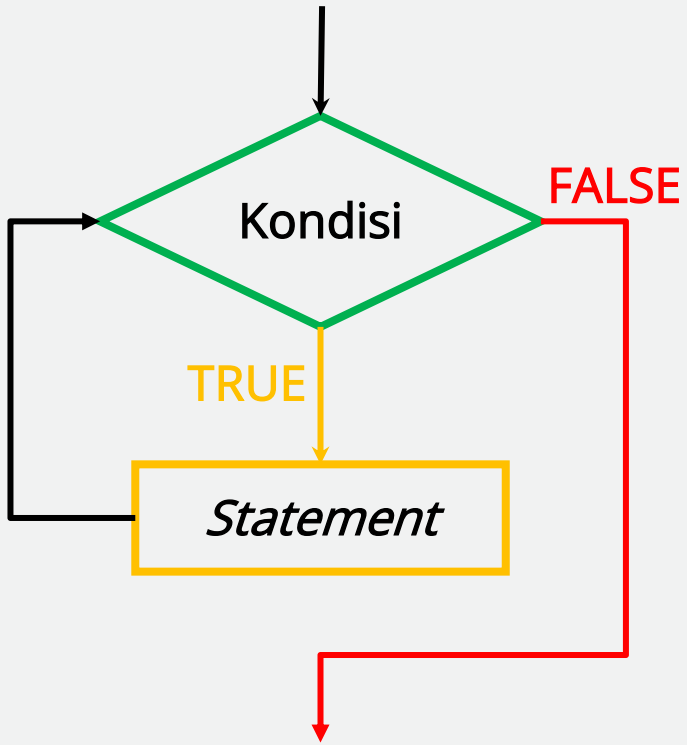
```
1 10
1 11
1 12
2 10
2 11
2 12
3 10
3 11
3 12
PS D:\CODING JAVA\00 - Kuliah FPA\P4\P4>
```

PERULANGAN WHILE

Fundamen Pengembangan Aplikasi

UNIVERSITAS ISLAM INDONESIA

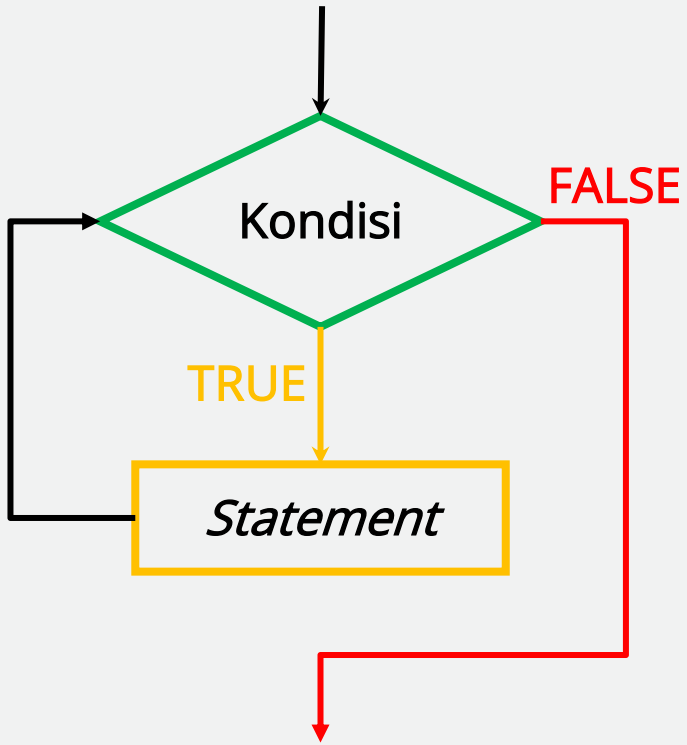
Perulangan WHILE



Perulangan ini akan menjalankan bagian program berulang kali NAMUN jumlah pengulangan belum diketahui, yang diketahui hanya KAPAN perulangan tersebut harus BERHENTI.

- Pengecekan nilai variabel berdasarkan Kondisi.
- Blok *Statement* dijalankan jika Kondisi bernilai TRUE.
- Jika Kondisi bernilai FALSE, proses perulangan akan berhenti.

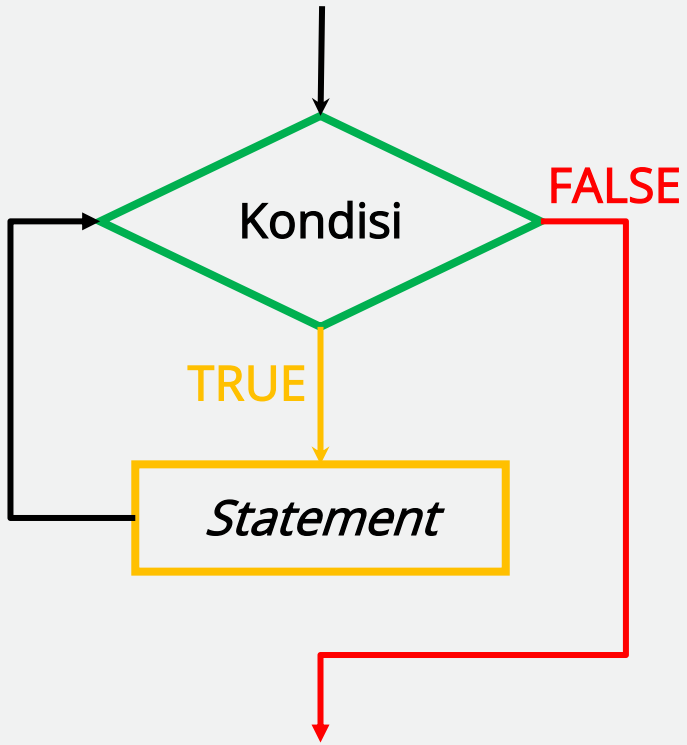
Perulangan WHILE



Sintaks di JAVA:

```
while (kondisi) {
    //statement
}
```

Perulangan WHILE



Contoh:
Menampilkan nilai yang nilainya kurang dari sama dengan 5

Kode program:

```
int i = 1;
while (i<=5) {
    System.out.println(i);
    i++;
}
```

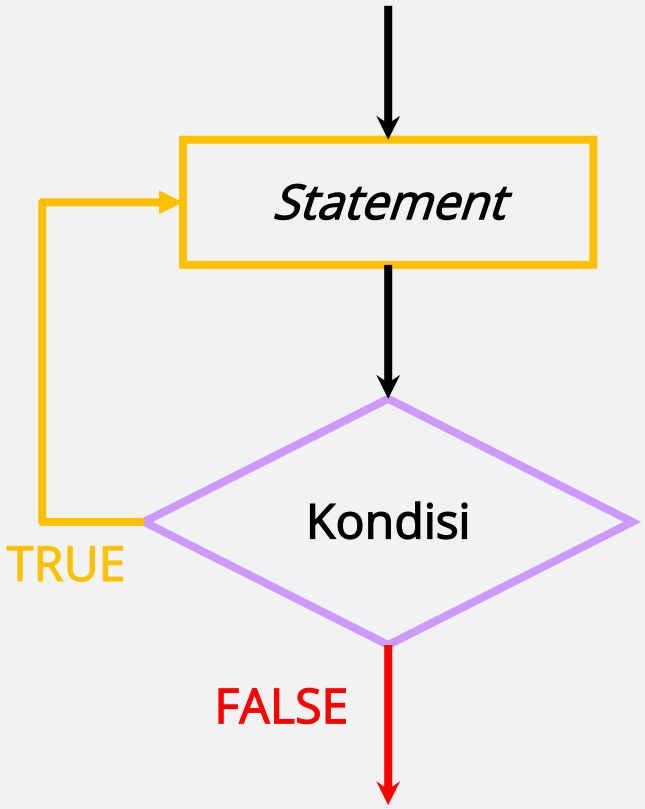
```
1
2
3
4
5
PS D:\CODING JAVA\00
```

PERULANGAN DO-WHILE

Fundamen Pengembangan Aplikasi

UNIVERSITAS ISLAM INDONESIA

Perulangan DO-WHILE



Perulangan DO-WHILE sama dengan perulangan WHILE-DO.

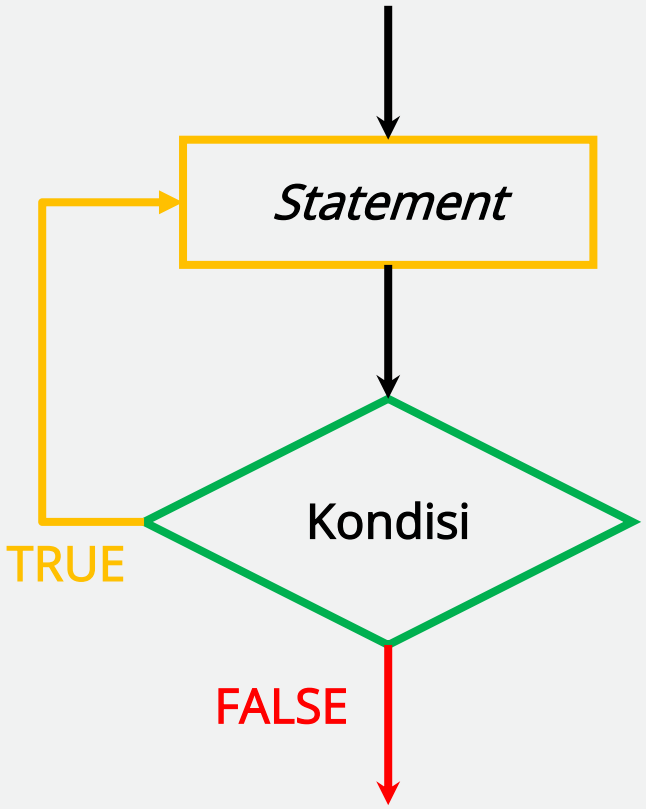
Bedanya :

- *Statement* pada perulangan DO-WHILE pasti dieksekusi minimal satu kali.

Sedangkan

- *Statement* pada perulangan WHILE-DO hanya akan dieksekusi jika Kondisi bernilai TRUE.

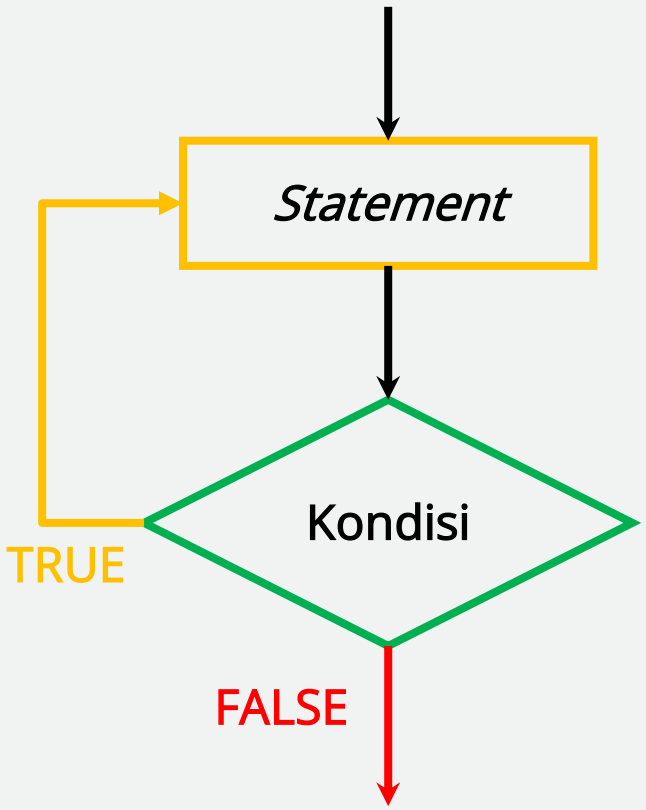
Perulangan DO-WHILE



Sintaks di JAVA:

```
do{
    // statement
}while(kondisi)
```

Perulangan DO-WHILE



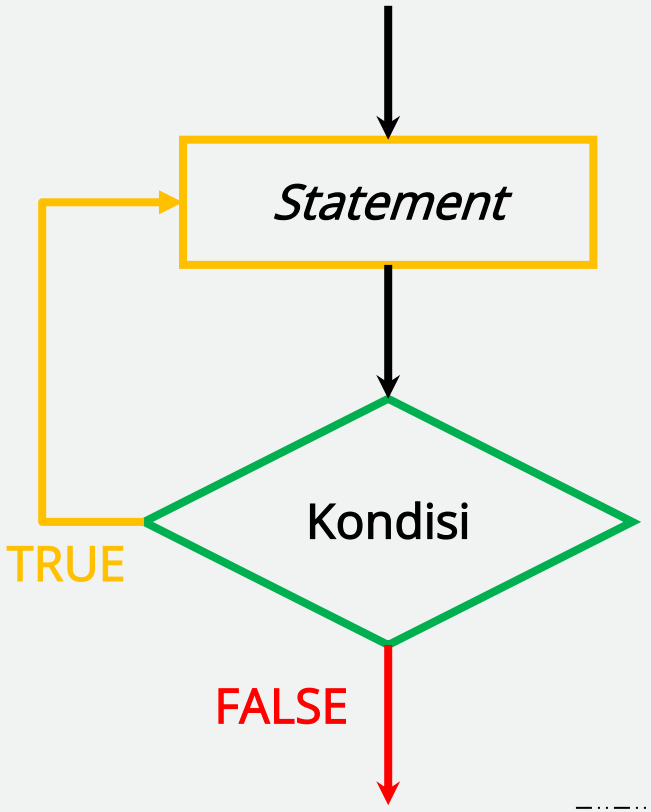
Contoh:

Menampilkan nilai yang nilainya kurang dari sama dengan 5.

Kode Program:

```
int i = 1;
do{
    System.out.println(i);
    i++;
}while (i<=5);
```

Perulangan DO-WHILE



Contoh 2:

Kode Program:

```
int i = 1;
do{
    System.out.println(i);
    i++;
}while (i<=1);
```

```
Output - O4_Perulangan (run)
run:
1
BUILD SUCCESSFUL (total
```

Perhatikan: *statement tetap dieksekusi walaupun nilai tidak kurang dari satu*

TERIMA KASIH