JOBSHEET #8
MENERAPKAN STRUKTUR KONTROL PERULANGAN DALAM BAHASA PEMROGRAMAN

A. TUJUAN

1. Siswa mampu memahami konsep kontrol Perulangan
2. Siswa mampu memahami For + break dan continue
3. Siswa mampu memahami While
4. Siswa mampu menyajikan Perulangan bersarang
5. Siswa mampu menyajikan deklarasi, aturan penamaan

B. DASAR TEORI

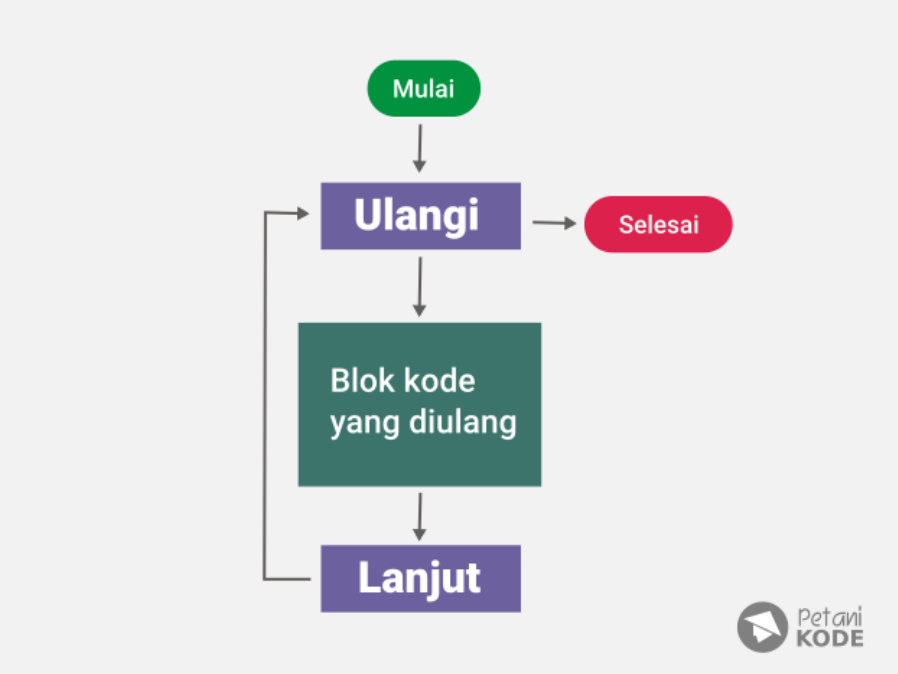
Apa yang akan kita lakukan bila ingin menyuruh komputer mengerjakan perintah yang berulang-ulang?

Misalkan kita ingin menyuruh komputer menampilkan teks Petani Kode sebanyak 5x.

Maka kita bisa menyuruhnya seperti ini:

```
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
```

Tapi… bagaimana kalau sebanyak 1000x, apa kita akan mampu mengetik kode sebanyak itu? Tentunya tidak. Karena itu, kita harus pakai perulangan.



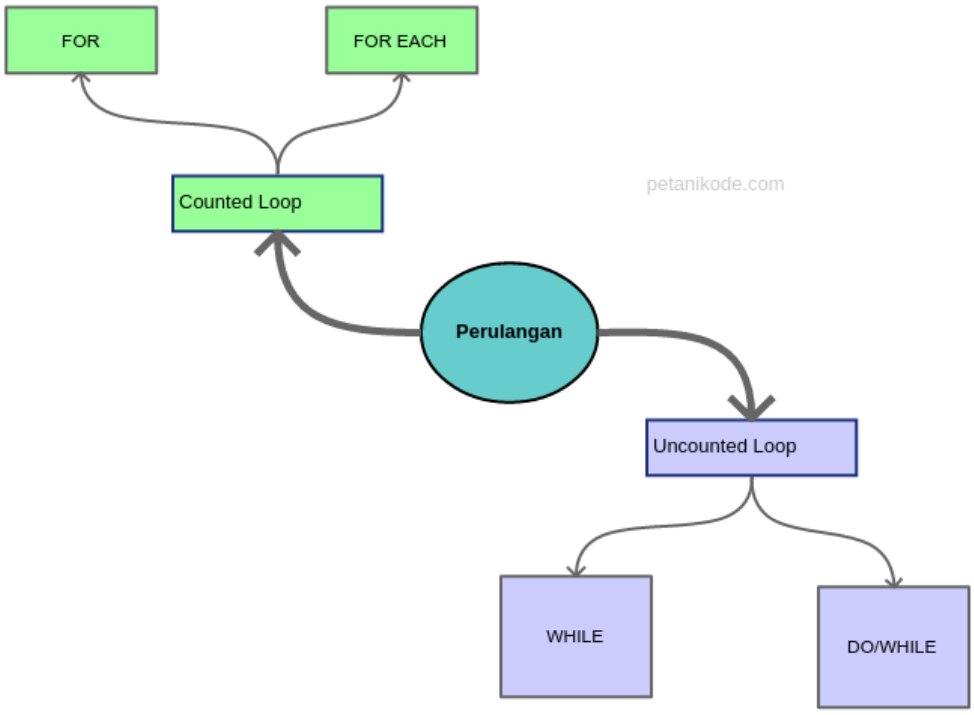
Contoh perulangan:

```
for (int hitungan = 0; hitungan <= 1000; hitungan++) {
    System.out.println("Petani Kode");
}
```

Sebelum masuk ke pembahasan lebih dalam, ada hal yang harus kalian ketahui terlebih dahulu.

Perulangan dalam pemrograman dibagi manjadi dua jenis:

1. **Counted loop**: Perulangan yang jumlah pengulangannya terhitung atau tentu.
2. **Uncounted loop**: Perulangan yang jumlah pengulangannya tidak terhitung atau tidak tentu.



Counted loop terdiri dari perulangan *For* dan *For each*. Sedangkan *Uncounted loop* terdiri dari perulangan *While* dan *Do/While*

1. Counted Loop

Perulangan *For*

Format penulisan perulangan *For* di java adalah sebagai berikut:

```
for( int hitungan = 0; hitungan <= 10; hitungan++ ){
```

```
    // blok kode yang akan diulang
}
```

Penjelasan:

- variabel hitungan tugasnya untuk menyimpan hitungan pengulangan.
- hitungan <= 10 artinya selama nilai hitungannya lebih kecil atau sama dengan 10, maka pengulangan akan terus dilakukan. Dengan kata lain, perualangan ini akan mengulang sebanyak 10 kali.
- hitungan++ fungsinya untuk menambah satu (+1) nilai hitungan peda setiap pengulangan.
- Blok kode *For* dimulai dengan tanda '{' dan diakhiri dengan '}'.

Perulangan *For Each*

Perulangan ini sebenarnya digunakan untuk menampilkan isi dari *array*.

Apa itu *array*?

Singkatnya, *array* itu variabel yang menyimpan lebih dari satu nilai dan memiliki indeks.

Contoh Program For Each

Buat sebuah class baru bernama PerulanganForeach, kemudian ikuti kode berikut.

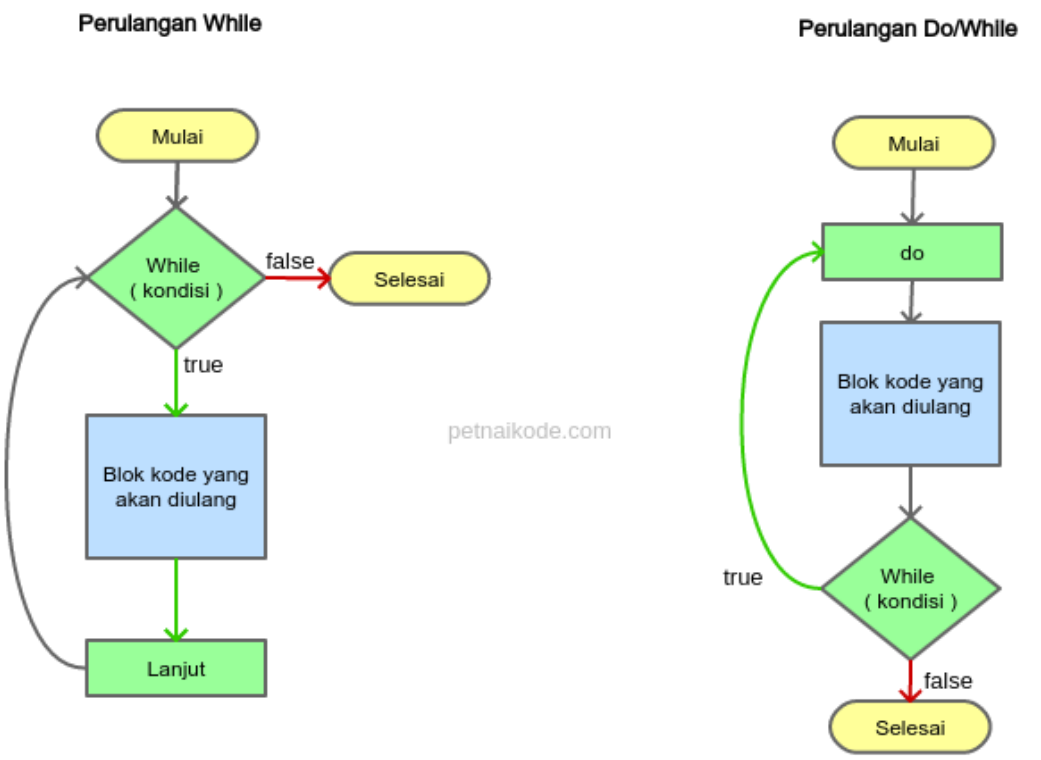
```
public class PerulanganForeach {
    public static void main(String[] args) {

        // membuat array
        int angka[] = {3,1,42,24,12};

        // menggunakan perulangan For each untuk
menampilkan angka
        for( int x : angka ){
            System.out.print(x + " ");
        }

    }
}
```

2. Uncounted Loop



Perulangan While

While bisa kita artikan *selama*.

Cara kerja perulangan ini seperti percabangan, ia akan melakukan perulangan selama kondisinya bernilai true.

Struktur penulisan perulangan while:

```
while ( kondisi ) {
    // blok kode yang akan diulang
}
```

Penjelasan:

- *kondisi* bisa kita isi dengan perbandingan maupun variabel boolean. *Kondisi* ini hanya memiliki nilai true dan flase.
- Perulangan while akan berhenti sampai *kondisi* bernilai false.

Untuk lebih jelasnya, mari kita coba membuat program…

Contoh Program dengan Perulangan While

Program ini akan melakukan perulangan selama jawabannya tidak.

```
import java.util.Scanner;

public class PerulanganWhile {
    public static void main(String[] args) {

        // membuat variabel dan scanner
        boolean running = true;
        int counter = 0;
        String jawab;
        Scanner scan = new Scanner(System.in);

        while( running ) {
            System.out.println("Apakah anda ingin keluar?");
            System.out.print("Jawab [ya/tidak]> ");

            jawab = scan.nextLine();

            // cek jawabnnya, kalau ya maka berhenti mengulang
            if( jawab.equalsIgnoreCase("ya") ){
                running = false;
            }

            counter++;
        }

        System.out.println("Anda sudah melakukan perulangan sebanyak
" + counter + " kali");

    }
}
```

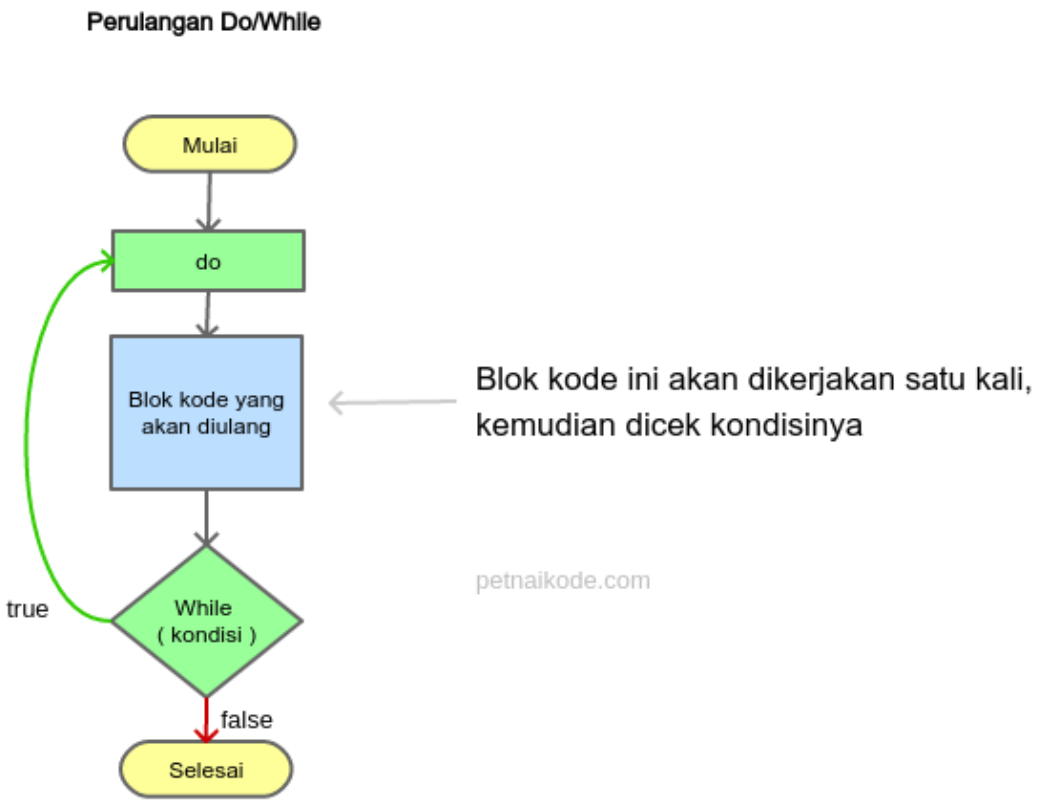
Hasil outputnya:

```
run:
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> ya
Anda sudah melakukan perulangan sebanyak 4 kali
BUILD SUCCESSFUL (total time: 9 seconds)
```

Di sana telah dilakukan perulangan sebanyak 4 kali. Bisa saja terjadi 10 kali. Itu targantung dari kondisinya. Kalau nilai variabel running bernilai false, maka perulangan berhenti.

Contoh kode *while* di atas dapat kita baca seperti ini: “Lakukan perulangan selama nilai running bernilai true.” Tidak menutup kemungkinan juga, perulangan ini dapat melakukan *counted loop*. Perulangan *Do/While*

Cara kerja perulangan *Do/While* sebenarnya sama seperti perulangan *While*.
Bedanya, *Do/While* melakukan satu kali perulangan dulu. Kemudian mengecek kondisinya.



Struktur penulisannya seperti ini:

```
do {
    // blok kode yang akan diulang
```

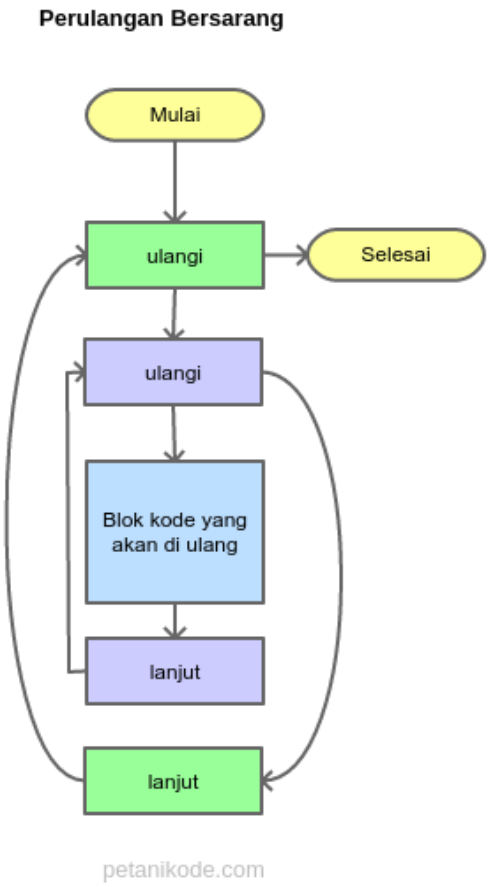
```
} while (kondisi);
```

Jadi kerjakan dulu (Do), baru di cek kondisinya while(kondisi).
Kalau kondisibernilai ture, maka lanjutkan perulangan.

Parulangan Bersarang *(Nested Loop)*

Perulangan juga dapat bersarang. Perulangan bersarang maksudnya, perulangan dalam perulangan atau disebut juga *nested loop*.

Contoh bentuk flow chart-nya seperti ini:



C. LATIHAN

- Buatlah contoh program menggunakan struktur perulangan
- Perhatikan kode berikut.

```
int i = 0;
for (  ; i <= 10  ;  )
{
```

```
        i++;
    }
```

Apa yang terjadi jika kode tersebut dijalankan? Apakah terjadi error?

- Perhatikan kode berikut.

int a = 1;
while (a < 0)
{
System.out.println(a);
a++;
}
	int a = 1;
	do
	{
	System.out.println(a);
	a++;
	}
	while (a < 0)

Apa keluaran dari setiap kode ketika dijalankan?

Apa perbedaan antara while dan do-while secara umum?

D. TUGAS PRAKTIKUM

Buat java class baru dengan nama Tugas01, kemudian ketikkan kode berikut.

```
public static void main(String[] args)
{
    int i;
    for(i = 1; i <= 10; i++)
    {
        if(i == 5)
        {
            break;
        }
        else
        {
            System.out.print(i + " ");
        }
    }
}
```

Simpan dan jalankan program. Tambahkan screenshoot dari hasil eksekusi program.

1. Berdasarkan hasil eksekusi, jelaskan fungsi dari kode break.

Buat java class baru dengan nama Tugas02, kemudian ketikkan kode berikut.

```
public static void main(String[] args)
{
    int i;
    for(i = 1; i <= 10; i++)
    {
        if(i == 5)
        {
            continue;
        }
        else
        {
            System.out.print(i + " ");
        }
    }
}
```

Simpan dan jalankan program. Tambahkan screenshoot dari hasil eksekusi program.

2. Berdasarkan hasil eksekusi, jelaskan fungsi dari kode continue.