

# Pertemuan 3

# Struktur Perulangan

# pada Java

Elia Dania Serepina  
PJ AP2A

# Struktur Kontrol Perulangan

**Struktur kontrol pengulangan** adalah berupa pernyataan dari Java yang memungkinkan kita untuk mengeksekusi blok code berulang-ulang sesuai dengan jumlah tertentu yang diinginkan. Ada tiga macam jenis dari struktur kontrol pengulangan yaitu :

1. While
2. Do-while
3. For loops

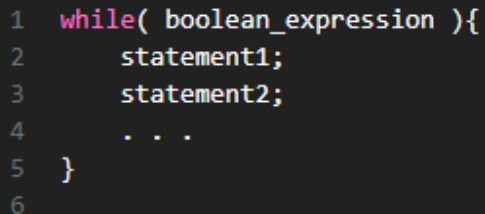
*Loop* secara berulang mengeksekusi sebarisan instruksi yang sama sampai kondisi akhir ditemui. Dengan kata lain, *looping* atau *loop* artinya mengulangi eksekusi blok program tertentu sampai tercapai kondisi untuk menghentikannya (terminasi). Setiap perulangan memiliki 4 bagian yaitu :

- ✓ inisialisasi (*initialization*),
- ✓ badan program (*body*) / *statement*,
- ✓ iterasi (*iteration*), dan
- ✓ *termination*.

# Statement While

Pernyataan **while loop** adalah pernyataan atau blok pernyataan yang diulang-ulang sampai mencapai kondisi yang cocok. Pernyataan di dalam while loop akan dieksekusi berulang-ulang selama kondisi `boolean_expression` bernilai benar (*true*).


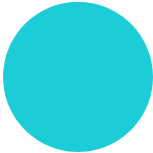
**Bentuk dari pernyataan while :**



```
1  while( boolean_expression ){  
2      statement1;  
3      statement2;  
4      . . .  
5  }  
6
```



## Contoh Kasus Statement While

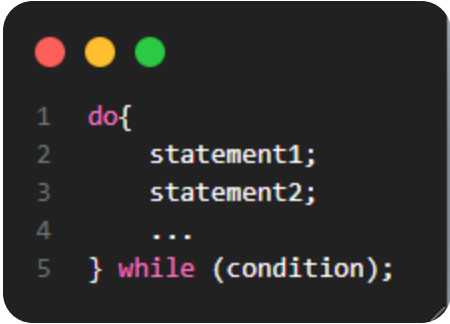


```
1  class contoh_while{  
2      public static void main(String[] args){  
3          int i = 0;  
4          while (i < 10) {  
5              System.out.println("i = " + i);  
6              i++;  
7          }  
8      }  
9  }
```

# Statement Do-While

**Do-while loop** mirip dengan **while-loop**. Pernyataan di dalam do-while loop akan dieksekusi beberapa kali selama kondisi bernilai benar(true). Perbedaan antara while dan do-while loop adalah dimana pernyataan di dalam do-while loop akan dieksekusi sedikitnya satu kali. Pernyataan di dalam do-while loop akan dieksekusi pertama kali, dan akan dievaluasi kondisi dari boolean\_expression. Jika nilai pada boolean\_expression tersebut bernilai true, pernyataan di dalam do-while loop akan dieksekusi lagi.

**Bentuk dari pernyataan do-while:**




```
1  do{  
2      statement1;  
3      statement2;  
4      ...  
5  } while (condition);
```



## Contoh Kasus

# Statement Do-While



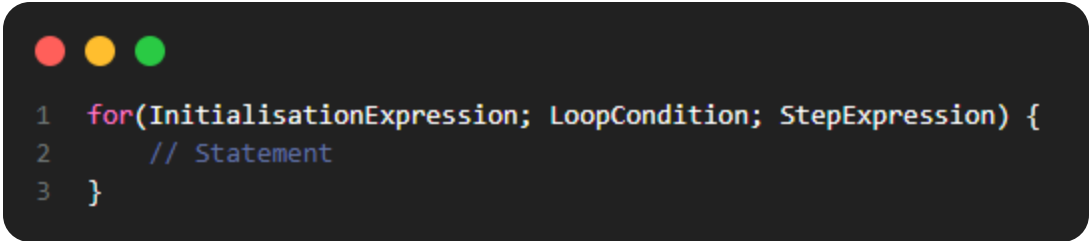
```
1  class contoh_do_while{
2      public static void main(String[] args){
3          int i = 0;
4          do {
5              System.out.println("i = " + i);
6              i++;
7          } while (i < 10);
8      }
9  }
```

# Statement For

Perulangan **for** menyediakan sarana mengulang kode dalam jumlah yang tertentu. Pengulangan ini terstruktur untuk mengulangi kode sampai tercapai batas tertentu.

- ✓ **InitializationExpression**, digunakan untuk inialisasi variabel kendali perulangan.
- ✓ **LoopCondition**, membandingkan variabel kendali perulangan dengan suatu nilai batas.
- ✓ **StepExpression**, menspesifkasikan cara variabel kendali dimodifikasi sebelum iterasi berikutnya dari perulangan.



**Bentuk dari pernyataan for :**



```
1  for(InitialisationExpression; LoopCondition; StepExpression) {  
2      // Statement  
3  }
```



# Contoh Kasus Statement For



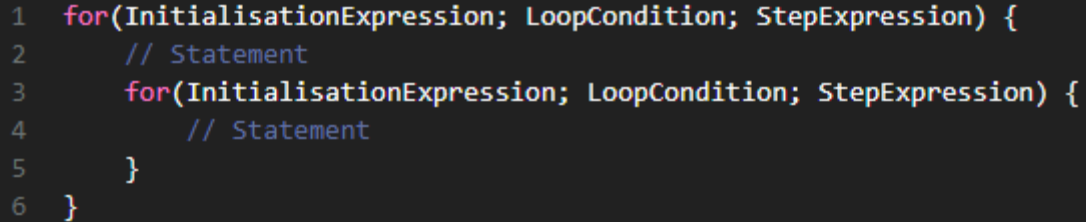
```
1  class contoh_for{  
2      public static void main(String[] args){  
3          for (int i = 0; i < 10; i++) {  
4              System.out.println("i = " + i);  
5          }  
6      }  
7  }
```



# Statement For Bersarang

Java memungkinkan loop yang disarangkan di loop yang lain. Satu loop berada di dalam loop yang lainnya.

**Bentuk dari pernyataan for bersarang :**


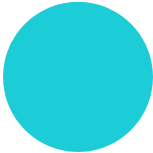


```
1  for(InitialisationExpression; LoopCondition; StepExpression) {  
2      // Statement  
3      for(InitialisationExpression; LoopCondition; StepExpression) {  
4          // Statement  
5      }  
6  }
```



Contoh Kasus

# Statement For Bersarang



```
1  class contoh_for_bersarang{
2      public static void main(String[] args){
3          for (int i = 0; i < 5; i++){
4              for (int j = 0; j <= i; j++){
5                  System.out.print("*");
6              }
7              System.out.println();
8          }
9      }
10 }
```

# Thanks!

