PART 7

Java Döngüleri

Bazı durumlarda, bir kod bloğunu birkaç kere çalıştırmamız gerekir ve bu çoğu zaman döngü olarak adlandırılır. Java, çok esnek üç döngü mekanizmasına sahiptir. Aşağıdaki üç döngüden birini kullanabilirsiniz:

- while
- do...while
- for

Java 5 itibariyle ***geliştirilmiş for döngüsü (enhanced for loop)*** tanıtıldı. Bu döngü daha çok diziler için kullanılmaktadır.

7.1 while Döngüsü

While döngüsü, bir görevi belirli sayıda tekrarlarlamamız için olanak sağlayan bir kontrol yapısıdır.

```
while(Boolean_expression)
{
   //Statements
}
```

Çalıştırırken, *boolean_expression* ifadesinin sonucu true ise , döngü içindeki işlemler yürütülecektir. Bu işlem , ifadenin değeri true oldukça devam edecektir.

Burada while döngüsünün kilit noktası, döngünün hiç çalışmayabilir olmasıdır. İfade test edilir ve sonuç false olursa, döngü gövdesi geçilecektir.

Örnek:

```
public class Test {

   public static void main(String args[]) {
      int x = 10;

      while( x < 20 ) {
         System.out.print("value of x : " + x );
         x++;
         System.out.print("\n");
      }
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
value of x : 15
value of x : 16
value of x : 17
value of x : 18
value of x : 19
```

7.2 do…while döngüsü

Bir do…while döngüsü, en az bir sefer çalışmayı garanti etmesi haricinde, while döngüsü ile birbirine benzerdir.

```
do
{
   //Statements
}while(Boolean_expression);
```

Dikkat ederseniz, Boolean ifade döngünün sonunda görülmektedir, bu yüzden döngünün içindeki komutlar Boolean test edilmeden önce bir kere çalışır.Boolean ifade true ise, kontrol akışı geri atlama yapar ve döngü içindeki komutlar tekrar çalışır. Bu işlem, Boolean ifade false olana kadar tekrarlanır.

Örnek:

```
public class Test {

   public static void main(String args[]){
      int x = 10;

      do{
         System.out.print("value of x : " + x );
         x++;
         System.out.print("\n");
      }while( x < 20 );
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
value of x : 15
value of x : 16
value of x : 17
value of x : 18
value of x : 19
```

7.3 for Döngüsü

For döngüsü, belirli bir sayıda yürütülmesi gereken döngüyü, verimli bir şekilde yazmanıza olanak sağlayan bir tekrarlı kontrol yapısıdır. For döngüsü, bir görevin kaç defa tekrarlanacağını bildiğiniz zaman yararlıdır.

```
for(initialization; Boolean_expression; update)
{
   //Statements
}
```

Örnek:

```
public class Test {

   public static void main(String args[]) {

      for(int x = 10; x < 20; x = x+1) {
         System.out.print("value of x : " + x );
         System.out.print("\n");
      }
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
value of x : 10
value of x : 11
value of x : 12
value of x : 13
value of x : 14
value of x : 15
value of x : 16
value of x : 17
value of x : 18
value of x : 19
```

7.4 Geliştirilmiş(Enhanced) for Döngüsü

Java 5'ten itibaren ***geliştirilmiş for döngüsü(enhanced for loop)*** adlı yeni bir döngü yapısı kullanılmaya başlanmıştır. Bu döngü daha çok diziler için kullanılmaktadır.

Geliştirilmiş for döngüsünün syntax'i:

```
for(declaration : expression)
{
   //Statements
}
```

- **declaration.** Eriştiğiniz dizi elemanlarının türü ile uyumlu, yeni deklare edilmiş blok değişkeni.

- **expression.** Döngü boyunca ihtiyacınız olan dizi. İfade, bir dizi değişkeni veya bir dizi döndüren metot çağrısı olabilir.

Örnek:

```
public class Test {

   public static void main(String args[]){
      int [] numbers = {10, 20, 30, 40, 50};

      for(int x : numbers ){
         System.out.print( x );
         System.out.print(",");
      }
      System.out.print("\n");
      String [] names ={"James", "Larry", "Tom", "Lacy"};
      for( String name : names ) {
         System.out.print( name );
         System.out.print(",");
      }
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
10,20,30,40,50,
James,Larry,Tom,Lacy,
```

7.5 "break" anahtar sözcüğü

Break anahtar sözcüğü, tüm döngüyü durdurmak için kullanılır. Break anahtar sözcüğü, herhangi bir döngü içinde veya bir switch komutu içinde kullanılmalıdır.

Örnek

```
public class Test {

   public static void main(String args[]) {
      int [] numbers = {10, 20, 30, 40, 50};

      for(int x : numbers ) {
         if( x == 30 ) {
              break;
         }
         System.out.print( x );
         System.out.print("\n");
      }
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
10
20
```

7.6 “continue” anahtar sözcüğü

Continue anahtar sözcüğü, döngü kontrol yapılarından herhangi birinin içinde kullanılabilir. Döngünün derhal sonraki iterasyonuna atlamasına sebep olur.

Örnek:

```
public class Test {

   public static void main(String args[]) {
      int [] numbers = {10, 20, 30, 40, 50};

      for(int x : numbers ) {
         if( x == 30 ) {
              continue;
         }
         System.out.print( x );
         System.out.print("\n");
      }
   }
}
```

Bu aşağıdaki sonucu üretecektir:

```
10
20
40
50
```