Java Programlama Dilinde Veri Tipleri

Doç. Dr. Aybars UĞUR

Metoda Temel Veri Tipi Gönderme

Java'daki 8 Temel veri tipi (boolean, char, byte, short, int, long, float, double).

```
public class OrnekTemelTip
{
   public static void main(String[] args)
   {
      int sayi = 5;
      System.out.println("Metottan Once = " + sayi);
      degerArttir(sayi);
      System.out.println("Metottan Sonra = " + sayi);
   }

   public static void degerArttir(int deger)
   {
      deger += 1;
   }
}
```

```
Metottan Once = 5

Metottan Sonra = 5
```

Metoda Nesne Gönderme

```
public class OrnekReferansTip
{
   public static void main(String[] args)
   {
      TamsayiSinifi d = new TamsayiSinifi();
      d.sayi = 5;
      System.out.println("Metottan Once = " + d.sayi);
      degerArttir(d);
      System.out.println("Metottan Sonra = " + d.sayi);
   }
   public static void degerArttir(TamsayiSinifi d)
   {
      d.sayi += 1;
   }
}

class TamsayiSinifi // sayi temel veri tipi TamsayiSinifi tarafından sarmalanmaktadır.
{
      public int sayi;
}
```

Metottan Once = 5

Metottan Sonra = 6

Nesne elemanlarının değerleri değişmektedir

Metoda Dizi Gönderme

```
public class OrnekDizi
{
   public static void main(String[] args)
   {
      double dizi[] = { 5,5,5,5 };
      System.out.print("\nMetottan Once = ");
      for(int i=0; i<dizi.length; ++i)
        System.out.print(dizi[i]+" ");
      degerArttir(dizi,2);
      System.out.print("\nMetottan Sonra = ");
      for(int i=0; i<dizi.length; ++i)
        System.out.print(dizi[i]+" ");
   }
   public static void degerArttir(double[] dizi, int indis)
   {
      dizi[indis] += 5;
   }
}
```

Metottan Once = 5.0 5.0 5.0 5.0

Metottan Sonra = 5.0 5.0 10.0 5.0

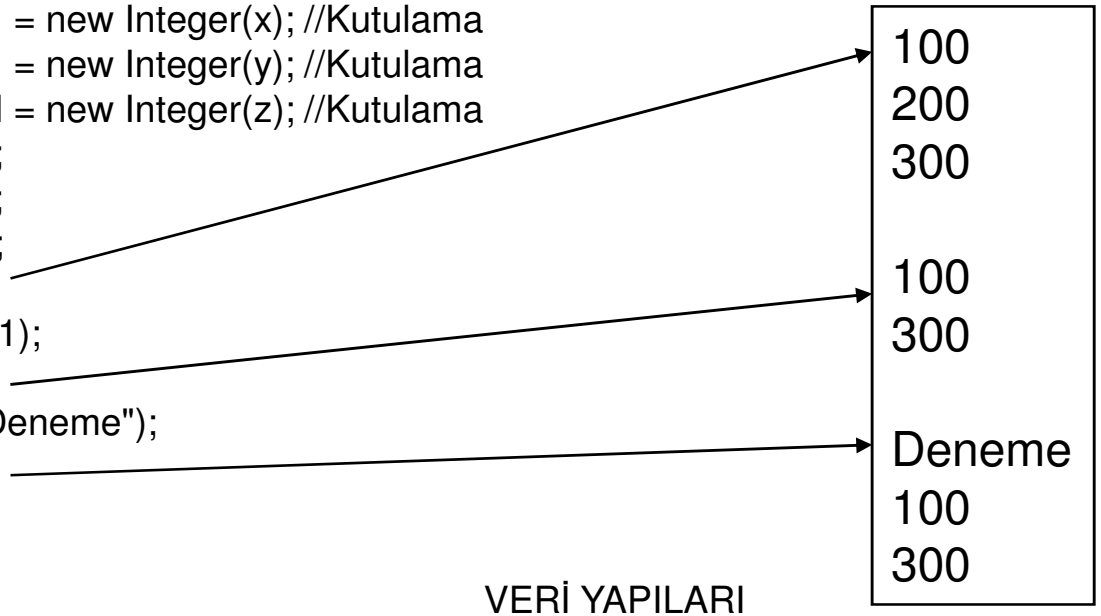
Dizi elemanlarının değerleri değişmektedir

Vektörler

```
import java.util.Vector;
public class Vektor
{
   public static void main(String[] args)
   {
      Vector v = new Vector();
      int x = 100;
      int y = 200;
      int z = 300;
      Integer xN = new Integer(x); //Kutulama
      Integer yN = new Integer(y); //Kutulama
      Integer zN = new Integer(z); //Kutulama
      v.add(xN);
      v.add(yN);
      v.add(zN);
      yazdir(v);
      v.remove(1);
      yazdir(v);
      v.add(0,"Deneme");
      yazdir(v);
   }
```

```
   public static void yazdir(Vector v)
   {
         System.out.println();
         for(int i=0; i<v.size();++i)
        System.out.println(v.elementAt(i));
   }

} // Vektor Sınıfı
```



Sarmalayıcı (Wrapper) Sınıflar

Primitive type	Wrapper class
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

```
int sayi = 100;

// Kutulama (Boxing)
Integer kutuSayi = new Integer (sayi);

// Kutudan Çıkarma (Unboxing)
int y = kutuSayi.intValue();
```

Kutulama : Bir temel veri tipini sarmalayıcı sınıf içine koyma işlemi

Kutudan Çıkarma : Sarmalayıcı sınıf içerisinden ilkel tipi geri alma işlemi

…

```
Integer d = new Integer(100);


System.out.println("Metottan Once = " + d);
degerArttir(d);
System.out.println(“Metottan Sonra = " + d);
```

…

Wrapper sınıflar değişmez (immutable) özelliğe sahiptirler, sıradan nesneler gibi davranmazlar. d değeri değişmez

Java 5.0 (ve sonrası)
Kutulama ve Kutudan Çıkarma Özellikleri

```
import java.util.Vector;
public class Java50
{
   public static void main(String[] args)
   {
      Integer i = new Integer(100);
      i++; //Java 5.0'da dogru

      Integer x = new Integer(100);
      Integer y = new Integer(200);
      Integer z = x * y; //Java 5.0'da dogru

      int a = new Integer(100); //Java 5.0'da dogru : Kutudan çıkarma
      Integer b = 100; //Java 5.0'da dogru : Otomatik Kutulama
      int c = 100;

      Vector v = new Vector();
      v.add(c); //Java 5.0'da dogru : Otomatik Kutulama

      int k = 100;
      Integer l = new Integer(200);
      int m = k + l; //Java 5.0'da dogru
   }
}
```

Immutable Objects : String

A classic example of an immutable object is an instance of the Java `String class.`

```
String s = "ABC";

s.toLowerCase();
```

The method `toLowerCase() will not change the data "ABC" that s contains. Instead, a new String object is instantiated and given the data "abc" during its construction. A reference to this String object is returned by the toLowerCase() method. To make the String s contain the data "abc", a different approach is needed.`

```
s = s.toLowerCase();
```

`(from Wikipedia :` Immutable object)