

Görsel Programlama

DERS 04

Diziler ve Koleksiyonlar(Collections)

Diziler aynı tipli değişkenleri tutmak için kullanılan veri yapılarıdır. Diziler sabit uzunluktadır. Birkez oluşturulduktan sonra boyutları değiştirilemez.

Java da diziler de bir nesnedir.



c.length=4 dir.

Her dizi de bir nesne olduğu için alanları bulunmaktadır, bunların biride dizi uzunluğunu veren “**length**” dir.

Diziler ve Koleksiyonlar(Collections)

Bir diziyi oluşturmak için ;

A-) `int[] c=new int[4];`

B-) `int c[] = new int[4];`

C-) `int[] c;`

`c=new int[4]; c[0]=-45;c[1]=6;c[2]=0;c[3]=72;`



Dizi ilk oluşturulduğunda elemanların hepsine varsayılan değer ataması yapılır.

Sayısal değerler için **0**, nesne dizisinde ise **null** atanır.

Diziler ve Koleksiyonlar(Collections)

```
DiziOrnek.java X
package com.comu.gorsel_programlama.ders04;

import javax.swing.JOptionPane;
import javax.swing.JTextArea;

public class DiziOrnek {
    public static void main(String[] args) {
        int[] dizi;
        dizi = new int[10];
        String cikti = "indis\tDeger\n";
        for (int i = 0; i < dizi.length; i++) {
            cikti += i + "\t" + dizi[i] + "\n";
        }
        JTextArea ciktiAlani = new JTextArea();
        ciktiAlani.setText(cikti);
        JOptionPane.showMessageDialog(null, ciktiAlani,
            "Dizinin elemanlarına ilk deger atanması",
            JOptionPane.INFORMATION_MESSAGE);
        System.exit(0);
    }
}
```

Diziler ve Koleksiyonlar(Collections)

```
Kisi.java x
package com.comu.gorsel_programlama.ders04;

public class Kisi {
    private String ad;
    private String soyad;
    private int yas;

    public Kisi(){}
    public Kisi(String ad,String soyad,int yas){
        this.ad=ad; this.soyad=soyad;this.yas=yas;
    }
    public String getAd() { return ad; }
    public void setAd(String ad) {this.ad = ad;}

    public String getSoyad() {return soyad;}
    public void setSoyad(String soyad) {this.soyad = soyad; }

    public int getYas() {return yas;}
    public void setYas(int yas) {this.yas = yas;}

    public String toString(){return getAd()+" "+getSoyad()+
        " "+getYas();}
}
```

Diziler ve Koleksiyonlar(Collections)

```
DiziKullan.java X
package com.comu.gorsel_programlama.ders04;

public class DiziKullan {
    public static void main(String[] args) {
        Kisi[] kisiler = new Kisi[2];
        kisiler[0] = new Kisi("Ahmet", "Veli", 30);
        kisiler[1] = new Kisi();
        kisiler[1].setAd("Yücel");
        kisiler[1].setSoyad("Kars");
        kisiler[1].setYas(20);
        System.out.println(kisiler[0]);
        System.out.println(kisiler[1]);
    }
}
```

Diziler ve Koleksiyonlar(Collections)

Dizi oluşturma'nın bir başka yolunda ilk değer atama yöntemidir.

`int[] n={10,20,30,40,50};` Bu şekilde 5 elemanlı bir dizi oluşturulur.

Çok Boyutlu Diziler

```
int[][] b={ {1,2} , {3,4} };
```

Java da çok boyutlu diziler tek boyutluymuş gibi algılanır. b dizisinin kendisi 2 elemanlı bir dizidir ve bu elemanların her biri tek boyutlu int dizisine referans olarak algılanır.

```
int[][] b;  
b=new int[2][];  
b[0]=new int[5];  
b[1]=new int[3];
```


Dizilerin Metotlara Parametre Olarak Gönderilmesi

Bildiğimiz gibi değişkenleri metotlara parametre olarak aktarmanın iki şekli vardır. **Pass-by-value(değer olarak aktarma)**, **pass-by-reference(referans olarak aktarma)**.

Bir değişken değer olarak aktarıldığında bunun bir kopyası oluşturulur ve bu kopya metoda gönderilir. Böyle aktarımlarda, aktarılan parametre üzerindeki değişiklikler normal değişkenin değerini etkilemez.

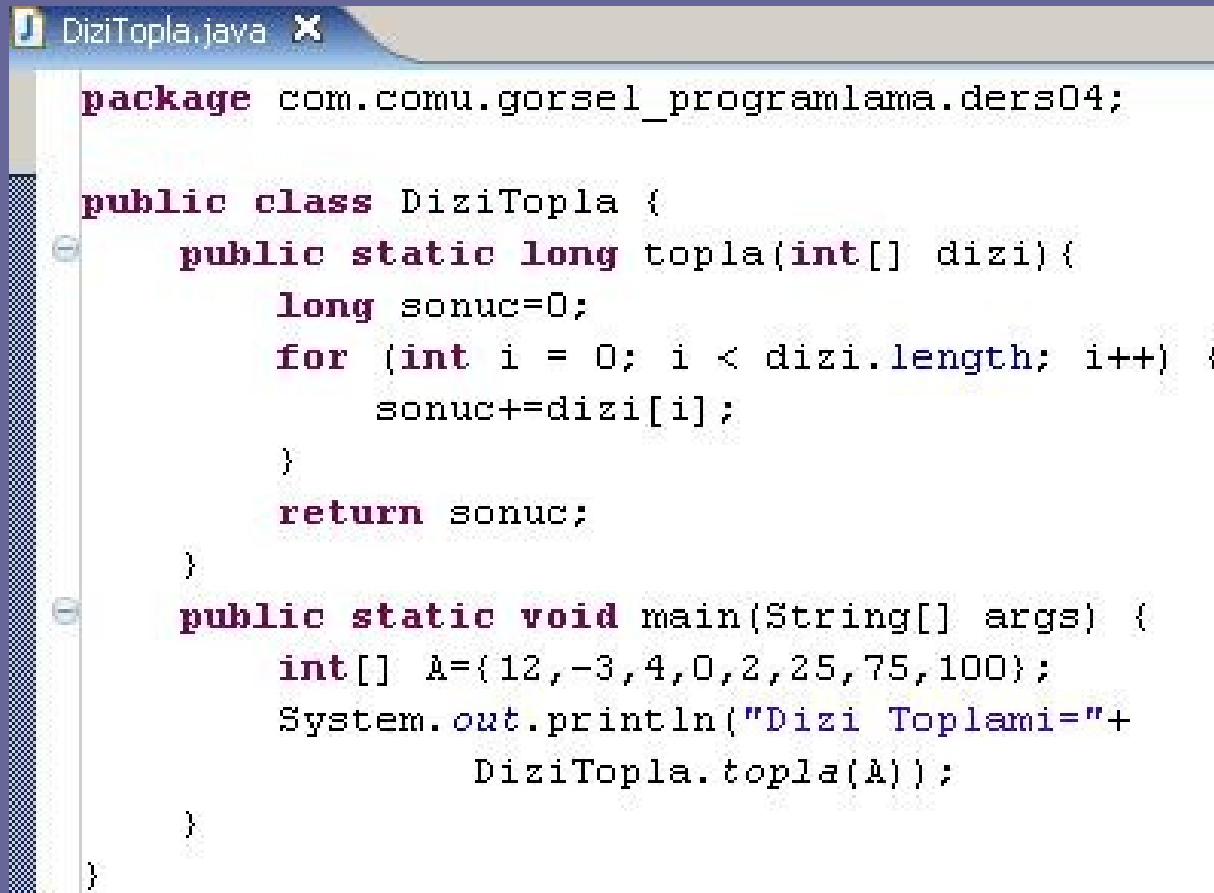
Bir değişken referans olarak gönderildiğinde, metodu çağıran kişi metoda değişkenin verisine erişim ve değiştirme yetkisi vermiştir.

Dizilerin Metotlara Parametre Olarak Gönderilmesi

İlkel tipler her zaman değer olarak gönderilirler. Nesneler metotlara parametre olarak gönderilmez, ama bu nesnelerin referansları parametre olarak gönderilebilir.

Referansların kendileri değer olarak gönderilir.

Dizilerin Metotlara Parametre Olarak Gönderilmesi

A screenshot of a Java IDE window titled 'DiziTopla.java'. The code defines a package 'com.comu.gorsel_programlama.ders04', a class 'DiziTopla', a static method 'topla' that sums an integer array, and a 'main' method that calls 'topla' on a specific array.

```
package com.comu.gorsel_programlama.ders04;

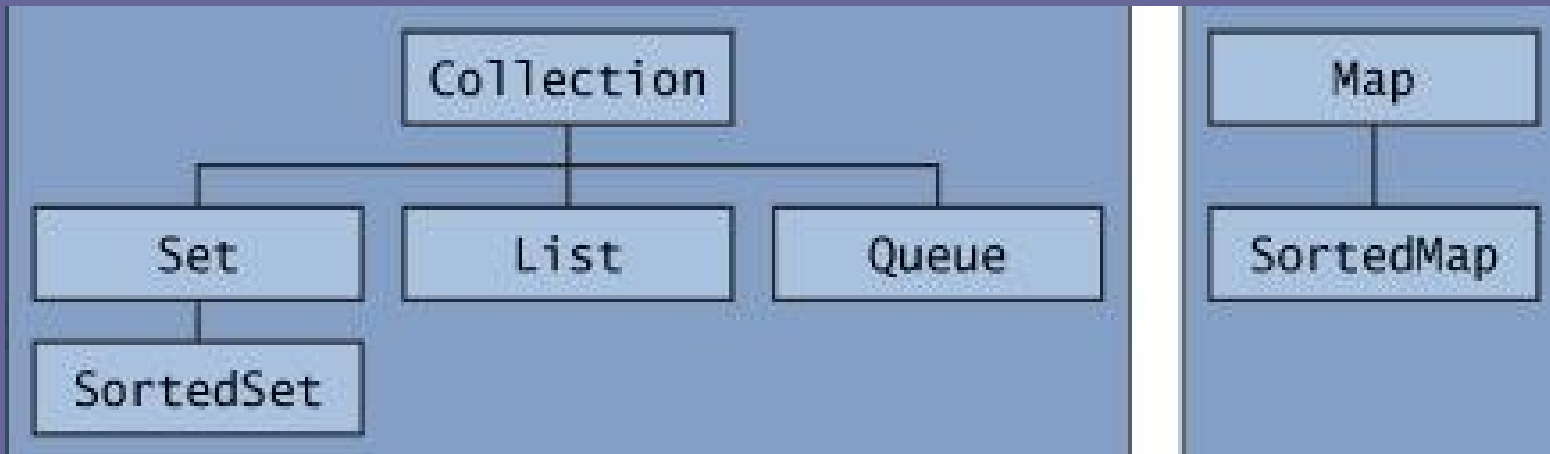
public class DiziTopla {
    public static long topla(int[] dizi){
        long sonuc=0;
        for (int i = 0; i < dizi.length; i++) {
            sonuc+=dizi[i];
        }
        return sonuc;
    }
    public static void main(String[] args) {
        int[] A={12,-3,4,0,2,25,75,100};
        System.out.println("Dizi Toplami="+
            DiziTopla.topla(A));
    }
}
```

Koleksiyonlar (Collections)

Bir dizide aynı tipli ve belirli bir sayıdaki nesneyi tutabiliriz. Bazı durumlarda dizi içinde tutacağımız elemanın sayısı ve tipi belirli olmayabilir ve çalışma zamanında belirli hale gelebilir. Java da bu tip durumlarda kullanılmak için çeşitli sınıflar bulunmaktadır. Bu sınıflara koleksiyon sınıfları denilir ve “**java.util.***” paketinde bulunurlar.

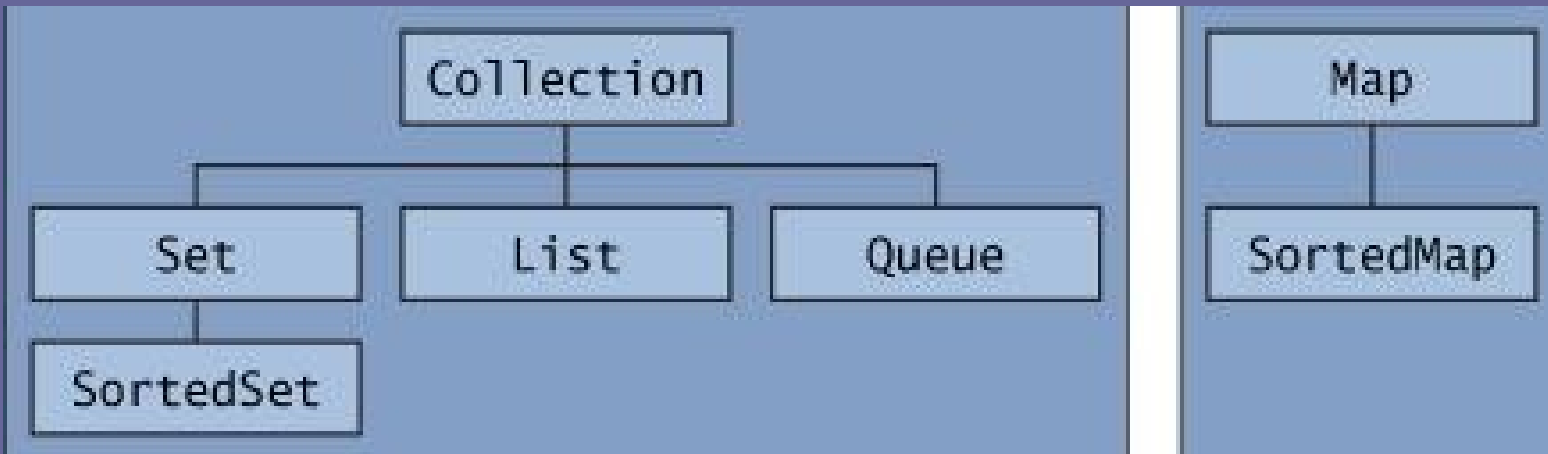
Bu sınıfların hepsinde tutulabilen eleman nesne olmalıdır, ilkel veri tipleri kabul edilmez. İlkel veri tipleri nesne karşılıklarına çevrilerek kullanılırlar.

Koleksiyonlar (Collections)



Temel koleksiyon arayüzleri yukarıdadır. Bu arayüzlerin gerçekleştirimi ile farklı amaçlar için kullanılabilecek sınıflar bulunmaktadır. Bu sınıfların bazıları aynı elemanın iki kez eklenmesine izin verirken bazıları vermez, bazıları elemanları sıralı tutarken bazıları tutmaz.

Set



SET: Bu kolleksiyon aynı elemanların eklenmesine izin vermez. Matematikteki SET kavramının soyutlanmış şeklidir.

Set

Java platformunda üç farklı tipli Set bulunmaktadır.

HashSet: elemanlarını bir Hash tablosunda saklar. Elemanlar sıralı tutulmazlar.

TreeSet: Elemanları red-black ağacı yapısında tutar. Performansı HashSet e göre daha azdır. Elemanları değerlerine göre sıralı tutar.

LinkedHashSet: Hash tablosunu bağlı liste şeklinde gerçekleştirir. Elemanlar eklenme sıralarına göre sıralıdırlar.

Set

```
SetOrnek.java x
package com.comu.gorsel_programlama.ders04;

import java.util.HashSet;
import java.util.Set;

public class SetOrnek {
    public static void main(String[] args) {
        String[] elemanlar = {"dene", "ali", "dene", "deneme", "kume"};
        Set kume = new HashSet();
        for (int i = 0; i < elemanlar.length; i++) {
            if (!kume.add(elemanlar[i])) {
                System.out.println("eleman iki kez giriliyor:" + elemanlar[i]);
            }
        }
        System.out.println(kume.size() + " farklı eleman eklendi");
    }
}
```

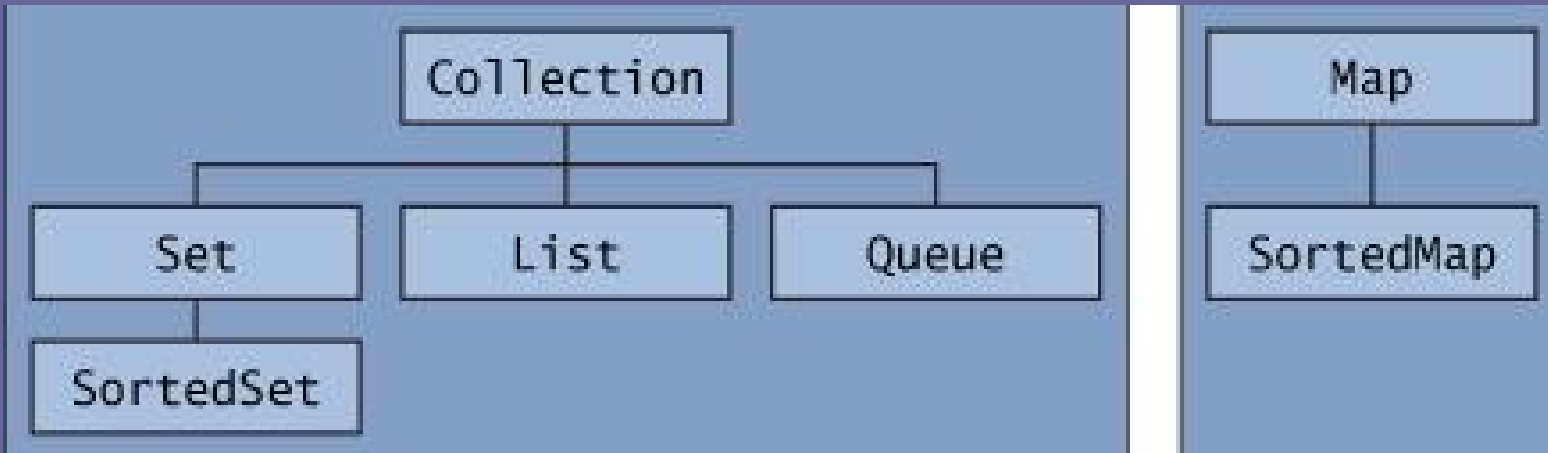

Set

```
SetOrnek2.java X
package com.comu.gorsel_programlama.ders04;

import java.util.HashSet;
import java.util.Set;

public class SetOrnek2 {
    public static void main(String[] args) {
        String[] elemanlar = { "dene", "ali", "dene", "deneme", "kume" };
        Set<String> kume = new HashSet<String>();
        for (String el : elemanlar){
            if (!kume.add(el)) {
                System.out.println("eleman iki kez giriliyor:" + el);
            }
        }
        System.out.println(kume.size()+" farkli eleman eklendi");
    }
}
```

List



List: Bu kolleksiyon aynı elemanların eklenmesine izin verilir.

LIST

Java platformunda iki farklı tipli List bulunmaktadır.

ArrayList: Elemanları dizi yapısında tutar. Dizinin kapasitesi eleman ekledikçe otomatik olarak arttırılır.

LinkedList: Elemanları bağlı liste şeklinde tutar. Elemanları listenin sonuna yada başına ekleme yada silme için metotlar sunar. ArrayList e göre performansı daha azdır.

LIST

```
ListOrnek.java x
package com.comu.gorsel_programlama.ders04;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Random;

public class ListOrnek {
    public static void main(String[] args) {
        String[] lis = {"B", "C", "D", "F", "H", "K", "L"};
        List<String> liste = new ArrayList<String>();
        for (String s : lis) {
            liste.add(s);
        }
        System.out.println("liste elemanları sıralı");
        System.out.println(liste);
        System.out.println("rasgele karıştırılmış liste");
        Collections.shuffle(liste, new Random());
        System.out.println(liste);
    }
}
```

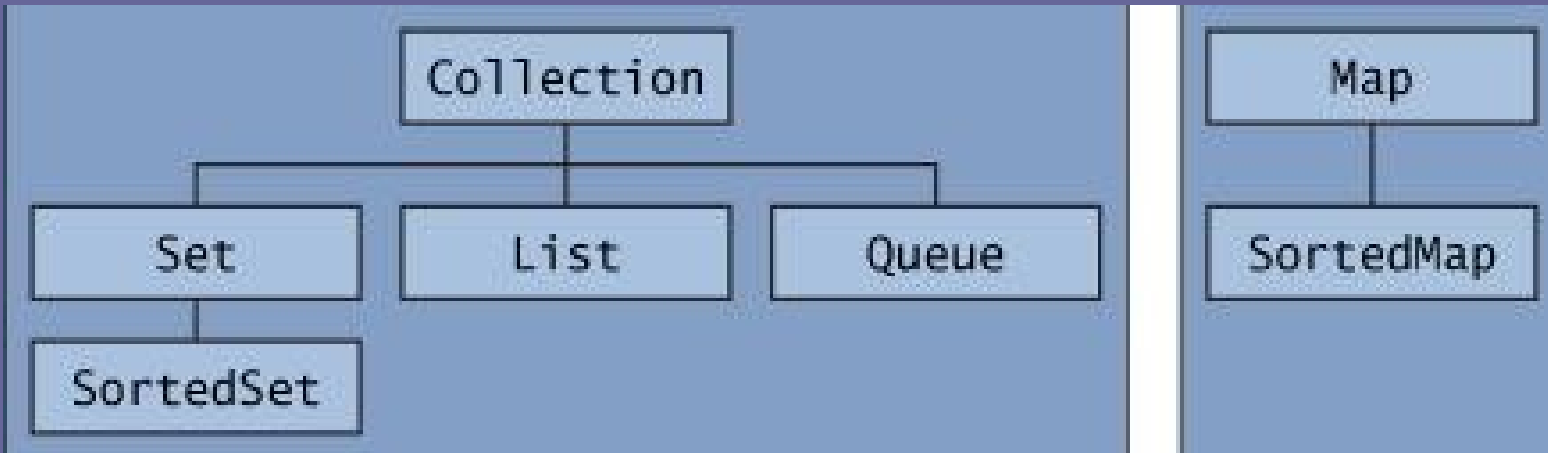
LIST

```
ArrayListKullanimi.java X
package com.comu.gorsel_programlama.ders04;

import java.util.ArrayList;

public class ArrayListKullanimi {
    public static void main(String[] args) {
        ArrayList arr = new ArrayList();
        Integer sayi= new Integer(10);
        Byte b = new Byte((byte)5);
        arr.add(sayi);
        arr.add(b);
        arr.add(new Long((long)1000));
        Object[] sayilar= arr.toArray();
        for (int i = 0; i < sayilar.length; i++) {
            Number n = (Number)sayilar[i];
            System.out.println(n.longValue());
        }
        System.out.println("Toplami="+toplami(arr));
    }
    private static long toplami(ArrayList arr) {
        long sonuc = 0;
        for (int i = 0; i < arr.size(); i++) {
            Number sayi = (Number)arr.get(i);
            sonuc+=sayi.longValue();
        }
        return sonuc;
    }
}
```

Queue



Queue: FIFO yapısında elemanları eklemek için kullanılır. Tüm yeni elemanlar sona eklenir. Elemanlar baştan çıkartılır.

Queue

```
QueueOrnek.java X
package com.comu.gorsel_programlama.ders04;

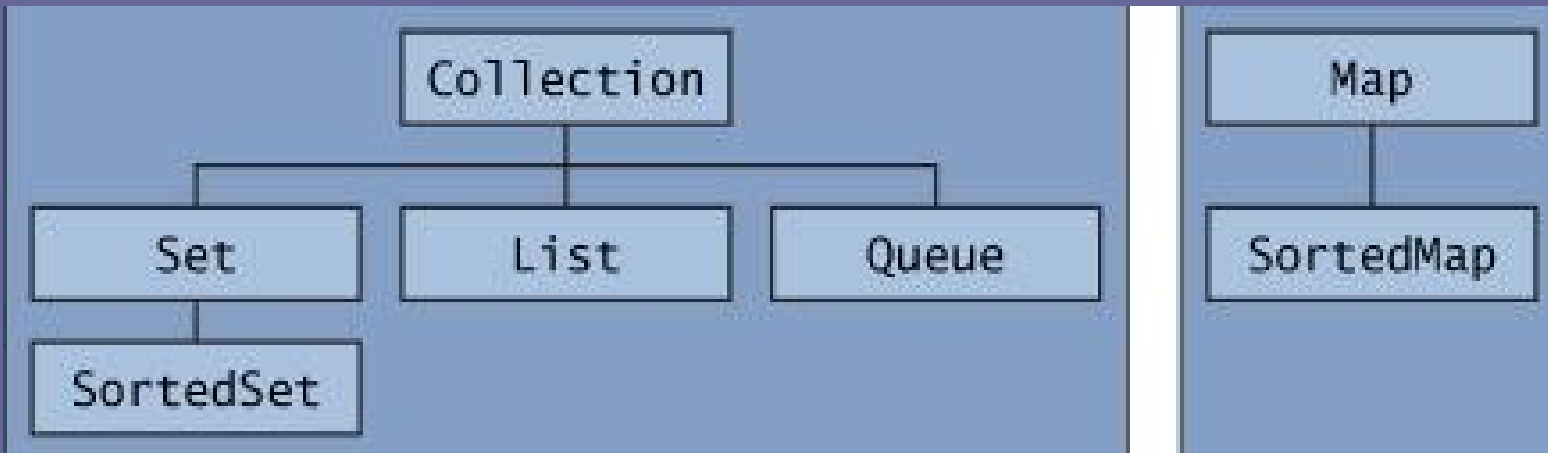
import java.util.PriorityQueue;
import java.util.Queue;

public class QueueOrnek {
    public static void main(String[] args) {
        Queue kuyruk = new PriorityQueue();

        kuyruk.add(new Integer(1));
        kuyruk.add(new Integer(2));
        kuyruk.add(new Integer(3));
        kuyruk.add(new Integer(4));

        System.out.println(kuyruk.poll());
        System.out.println(kuyruk.poll());
        System.out.println(kuyruk.peek());
        System.out.println(kuyruk.poll());
        System.out.println("boyutu="+kuyruk.size());
    }
}
```

Map



Map: Anahtarlar ile Değerleri eşleştirerek saklayan bir nesnedir. Bir Map aynı anahtar değerlerini tutamaz.

Map

Java platformunda üç farklı tip Map bulunmaktadır.

HashMap: Map elemanlarını hashtable yapısında saklar.

TreeMap: Anahtar-Değer ikililerini red-black ağaç veri yapısı şeklinde tutar.

LinkedHashMap: Anahtar-Değer ikililerini bağlı liste veri yapısında tutar.

Map

```
MapOrnek.java x
package com.comu.gorsel_programlama.ders04;

import java.util.HashMap;
import java.util.Map;

public class MapOrnek {
    public static void main(String[] args) {
        String[] kelimeler = {"Ali", "Ali", "is",
                               "is", "is", "Java", "kod", "C", "C"};
        Map<String, Integer> m = new HashMap<String, Integer>();
        for (String kelime : kelimeler) {
            Integer freq = m.get(kelime);
            m.put(kelime, (freq==null)?1:freq+1);
        }
        System.out.println(m.size()+" farkli kelime");
        System.out.println(m);
    }
}
```

Map

```
Kitap.java x
package com.comu.gorsel_programlama.ders04;

public class Kitap {
    private String ISBN;
    private String baslik;
    public Kitap(String isbn, String baslik) {
        this.ISBN = isbn;
        this.baslik = baslik;
    }
    public String getBaslik() {
        return baslik;
    }
    public void setBaslik(String baslik) {
        this.baslik = baslik;
    }
    public String getISBN() {
        return ISBN;
    }
    public void setISBN(String isbn) {
        ISBN = isbn;
    }
}
```

Map

```
HashMapKullanim.java X
package com.comu.gorsel_programlama.ders04;

import java.util.HashMap;

public class HashMapKullanim {
    public static void main(String[] args) {
        HashMap<String, Kitap> hm = new HashMap<String, Kitap>();
        Kitap k= new Kitap("123H5", "Java ya giriş");
        hm.put(k.getISBN(), k);
        Kitap k2 = new Kitap("35HM22", "Java 5.0");
        hm.put(k2.getISBN(), k2);
        Kitap gelen = hm.get("123H5");
        System.out.println(gelen.getBaslik());
    }
}
```

Map elemanlarının Iterator ile alınması

```
IteratorOrnek.java x
package com.comu.gorsel_programlama.ders04;

import java.util.HashMap;
import java.util.Iterator;

public class IteratorOrnek {
    public static void main(String[] args) {
        HashMap<String, String> a = new HashMap<String, String>();
        a.put("123", "Mustafa");
        a.put("456", "Java");
        Iterator<String> it = a.keySet().iterator();
        while (it.hasNext()) {
            String element = a.get(it.next());
            System.out.println(element);
        }
    }
}
```

Görsel Programlama

DERS 04