Struktur Data dengan Collection Framework

Inheritance

Relasi antar object

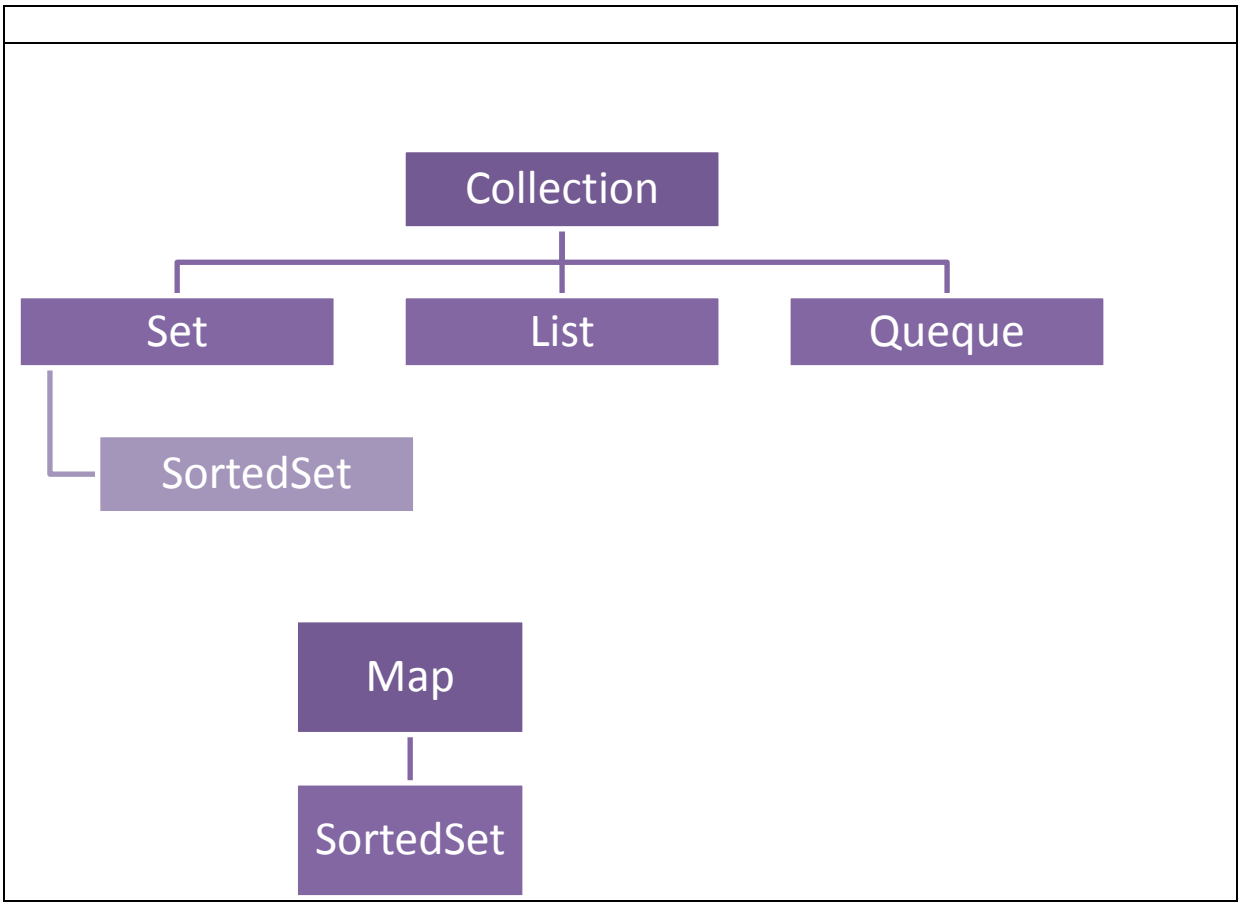
Struktur Data dengan Collection Framework

collection (container) merupakan object yang mengelompokkan sekelompok elemen ke dalam satu kelompok.

Collection digunakan utk menyimpan,mengambil kembali,memanipulasi, dan mengkomunikasikan kumpulan data.

Mewakili item data yang membentuk kelompok secara alami, contoh kartu poker (collection of card), mail folder (a collection of letters), atau telephone directory (a mapping of names to phone numbers) .

collections framework merupakan kesatuan arsitektur utk mewakili dan memanipulasi collection



1. Object Array

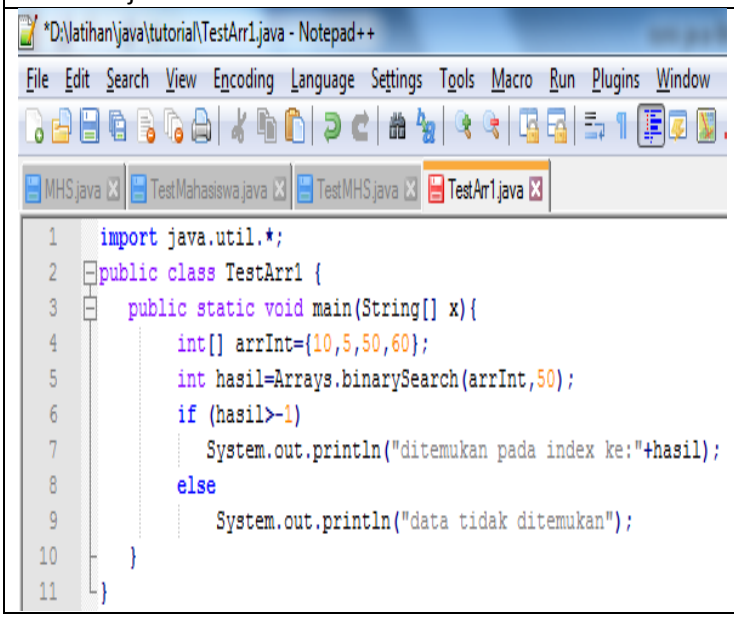
Array standard yang dimiliki oleh Java API (java.util)

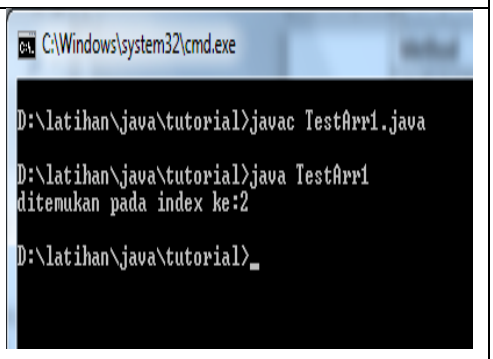
Array memiliki method-method:

Method	Keterangan
static int binarySearch(array, key)	Pencarian nilai dalam array
boolean equals(array1, array2)	Membandingkan apakah dua array memiliki nilai sama. Bekerja pada array satu dimensi
static void sort(array)	Mengurutkan isi array
static String toString(array)	Mengubah nilai array menjadi String

Contoh penggunaan array

TestArr1.java





2. ArrayList

ArrayList mirip dengan array, tapi memiliki kemampuan lebih baik.

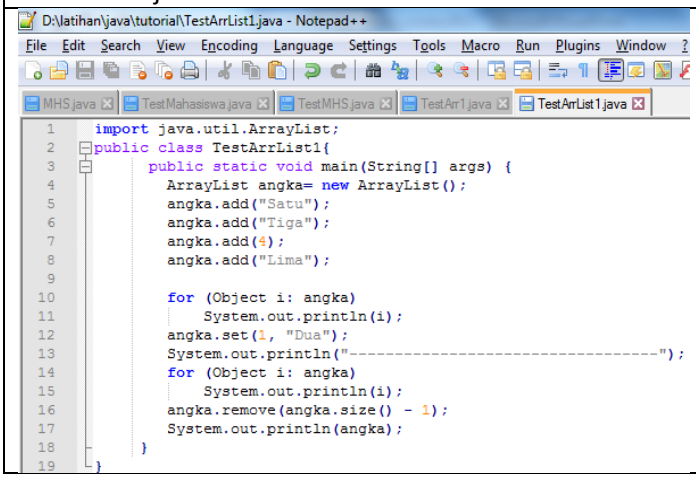
Jumlah elemen dalam ArrayList dapat berubah secara fleksibel tergantung jumlah data yang ada di dalamnya.

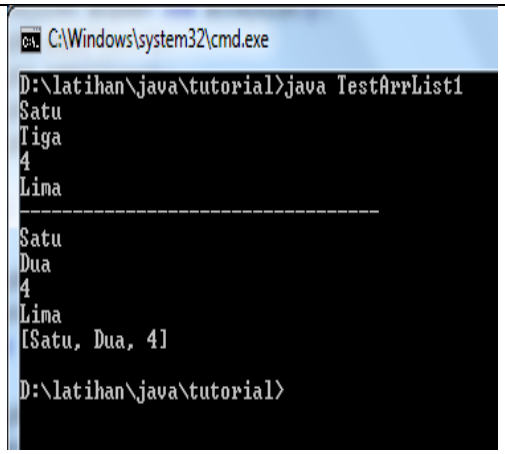
Setelah array terbentuk, data baru dapat dimasukkan di tengah-tengah, tidak harus di akhir elemen array.

Isi dalam array bisa dihapus, dan index dalam array sesudahnya akan maju satu langkah untuk mengisi slot kosong tersebut.

Contoh: penggunaan arrayList

TestArrList.java





Contoh Penggunaan ArrayList dengan elemen berupa object

MHS.java

```
1  public class MHS {
2     private String nim;
3     private String nama;
4     private float ipk;
5       MHS(){
6         this.nim="";
7         this.nama="";
8         this.ipk=0f;
9     }
10    MHS(String nim,String nama,float ipk){
11        this.nim=nim;
12        this.nama=nama;
13        this.ipk=ipk;
14    }
15    public void setNim(String nim){
16        this.nim=nim;
17    }
18    void setNama(String nama){
19        this.nama=nama;
20    }
21    void setIPK(float ipk){
22        this.ipk=ipk;
23    }
24    static String ketLulus(MHS m){
25          String ket;
26          float nilai;
27          nilai=m.getIPK();
28          if (nilai>3.7)
29              ket="Cumlaude";
30          else if (nilai>3.5)
31              ket="Sangat Memuaskan";
32          else if (nilai>3.0)
33              ket="Memuaskan";
34          else
35              ket="Cukup";
36          return ket;
37      }
```

MHS.java (lanjutan)

```
38    float getIPK(){
39        return ipk;
40    }
41    String  getNama(){
42        return nama;
43    }
44    String  getNim(){
45        return nim;
46    }
47    public static MHS bandingkanIPK(MHS m1,MHS m2){
48        if (m1.getIPK()>=m2.getIPK())
49            return m1;
50        else
51            return m2;
52    }
53
54    public static int totalArr(int... x){
55        int sum=0;
56        for (int i=0;i<x.length;i++){
57            sum+=x[i];
58        }
59        return sum;
60    }
61
62    static void printMHS(MHS...x){
63        int i;
64        System.out.println("==============");
65        for (i=0;i<x.length;i++){
66            System.out.println(x[i].nim + "," + x[i].nama+ "," + x[i].ipk);
67        }
68    }
69 }
```

Test2MHS.java

```
1  import java.util.ArrayList;
2  import java.util.Scanner;
3  public class Test2MHS{
4      ArrayList<MHS> mhs;
5      Test2MHS(){
6          mhs = new ArrayList<MHS>();
7      }
8      void isiData(){
9          Scanner in = new Scanner(System.in);
10         System.out.println("");
11         System.out.print("NIM:");
12         String nim = in.next();
13         in.nextLine();
14         System.out.print("Nama:");
15         String nama=in.nextLine();
16         System.out.println("IPK:");
17         float ipk=in.nextFloat();
18         mhs.add(new MHS(nim,nama,ipk));
19     }
20     void printMHS(){
21         for (MHS x:mhs)
22             System.out.println(x.getNim()+" "+x.getNama()+","+x.getIPK());
23     }
24     public static void main(String[] x){
25         Scanner in = new Scanner(System.in);
26         Test2MHS f = new Test2MHS();
27         String lagi;
28         do{
29             f.isiData();
30             System.out.print("Isi lagi[y/t]:");
31             lagi=in.next();
32         }
33         while(lagi.equals("y")||lagi.equals("Y"));
34         f.printMHS();
35     }
36 }
```

```
D:\latihan\java\tutorial>javac Test2MHS.java

D:\latihan\java\tutorial>java Test2MHS

NIM:A11.2016.00101
Nama:Mahendra Sutiawan
IPK:
3,2
Isi lagi[y/t]:y

NIM:A11.2016.01101
Nama:Susi Damayanti
IPK:
3,6
Isi lagi[y/t]:t
A11.2016.00101 Mahendra Sutiawan,3.2
A11.2016.01101 Susi Damayanti,3.6

D:\latihan\java\tutorial>
```

Inheritance

Person.java

```
1  public class Person {
2      private String nik;
3      private String nama;
4      public Person(){
5          nik="";
6          nama="";
7      }
8      public Person(String nik,String nama){
9          setNik(nik);
10         setNama(nama);
11     }
12     public void setNik(String nik){
13         this.nik=nik;
14     }
15     void setNama(String nama){
16         this.nama=nama;
17     }
18     String getNik(){
19         return nik;
20     }
21     String getNama(){
22         return nama;
23     }
24     void display(){
25         System.out.println("Informasi Person");
26         System.out.println("------------------");
27         System.out.println("NIK:"+nik);
28         System.out.println("Nama:"+nama);
29         System.out.println("------------------");
30     }
31 }
```

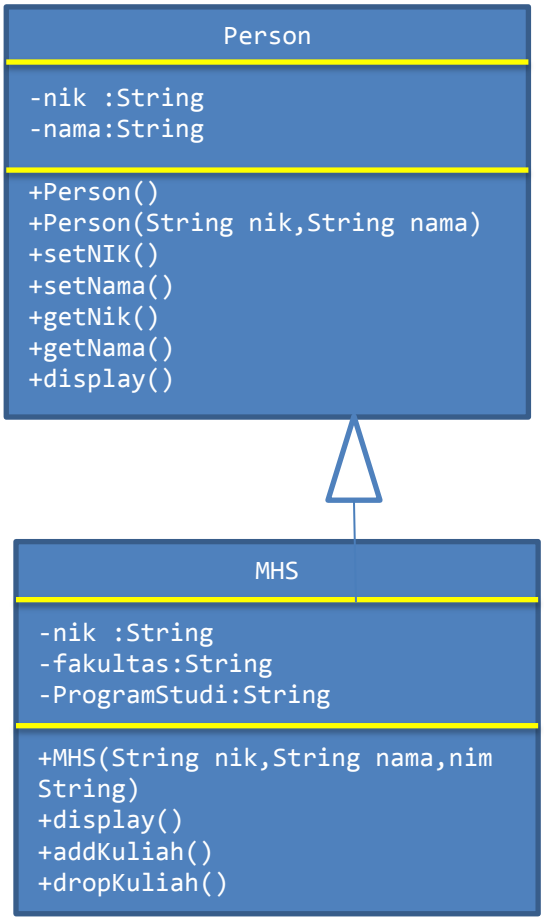
MHS.java

```
1  public class MHS extends Person {
2      private String nim;
3      MHS(String nik,String nama,String nim){
4        super(nik,nama);
5        setNim(nim);
6      }
7      void setNim(String nim){
8          this.nim=nim;
9      }
10     String getNim(){
11         return nim;
12     }
13     void display(){
14         super.display();
15         System.out.println("NIM:"+nim);
16         System.out.println("------------------");
17     }
18 }
```

TestInheritance.java

```
1  public class TestInheritance {
2      public static void main(String[] args){
3          Person obj1 = new Person("12345","Heru");
4          obj1.display();
5
6          MHS obj2 = new MHS("12344","Fatima","A11.2016.00007");
7          obj2.display();
8      }
9  }
```

Relasi antar Object

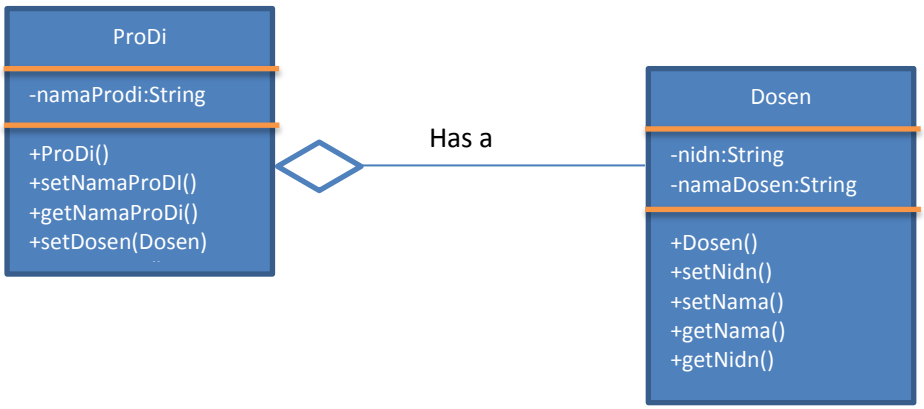


Generalisasi (Inheritance)
Merupakan bentuk relasi **Is-a**
Relasi antara class yang lebih umum(general)/(superclass) ke class yang lebih kusus (specialized)/ (subclass)

Aggregation

Has-a relationship

Objects pada suatu class berisi referensi ke object dari class lain.



Dosen.java

```
1  public class Dosen {
2      private String nidn;
3      private String nama;
4      Dosen(String nidn,String nama){
5          this.nidn=nidn;
6          this.nama=nama;
7      }
8      void setNidn(String nidn){
9          this.nidn=nidn;
10     }
11     String getNidn(){
12         return nidn;
13     }
14     String getNama(){
15         return nama;
16     }
17 }
18
```

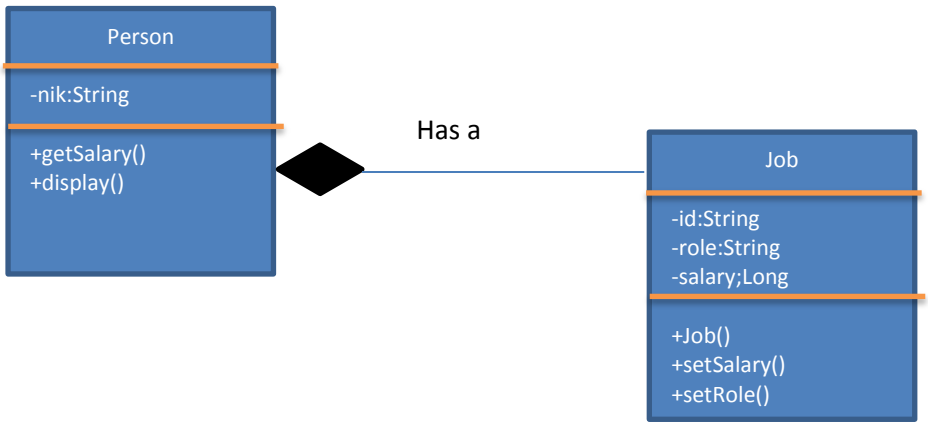
ProDi.java

```
1  public class ProDi {
2      private String namaProDi;
3      public Dosen dsn;
4      ProDi(String namaProDi){
5          this.namaProDi=namaProDi;
6      }
7      void setNamaProDi(String namaProDi){
8          this.namaProDi=namaProDi;
9      }
10     String getNamaProdi(){
11         return namaProDi;
12     }
13     void setDosen(Dosen d){
14        dsn=d;
15     }
16     void printDosen(){
17         System.out.println("nidn:"+
18          dsn.getNidn()+",nama:"+dsn.getNama());
19     }
20 }
```

TestAgregasi.java

```
1  import java.util.*;
2  public class TestAgregasi {
3      public static void main(String[] args){
4          ProDi tis1 = new ProDi("TI-S1");
5          Dosen d1 = new Dosen("123","Dr. Selamet");
6          Dosen d2 = new Dosen("124","Bejo,M.Kom");
7          tis1.setDosen(d1);
8          tis1.printDosen();
9      }
10 }
```

Komposisi



Person2.java

*D:\latihan\java\latihan4\Person2.java - Notepad++

```
1  public class Person2 {
2      //composition has-a relationship
3      private Job job;
4      private String name;
5      public Person2(String name){
6          this.job=new Job();
7          job.setSalary(1000L);
8          job.setRole("Manager");
9          job.setId(1);
10         this.name=name;
11     }
12     public long getSalary() {
13         return job.getSalary();
14     }
15     public void display(){
16         System.out.println(name);
17         System.out.println(job.getId());
18         System.out.println(job.getRole());
19         System.out.println(job.getSalary());
20
21     }
22 }
```

Job.java

*D:\latihan\java\latihan4\Job.java - Notepad++

```
1  public class Job {
2      private String role;
3      private long salary;
4      private int id;
5
6      public String getRole() {
7          return role;
8      }
9      public void setRole(String role) {
10         this.role = role;
11     }
12     public long getSalary() {
13         return salary;
14     }
15     public void setSalary(long salary) {
16         this.salary = salary;
17     }
18     public int getId() {
19         return id;
20     }
21     public void setId(int id) {
22         this.id = id;
23     }
24 }
25
```

TestPerson2.java

*D:\latihan\java\latihan4\TestPerson2.java - Notepad++

```
1  public class TestPerson2 {
2      public static void main(String[] args) {
3          Person2 person = new Person2("Alex");
4          long salary = person.getSalary();
5          person.display();
6      }
7  }
```