Object Oriented Programming di Java: Class dan Object

Yudi Wibisono (yudi@upi.edu)
Masayu Leylia Khodra (masayu@informatika.org)

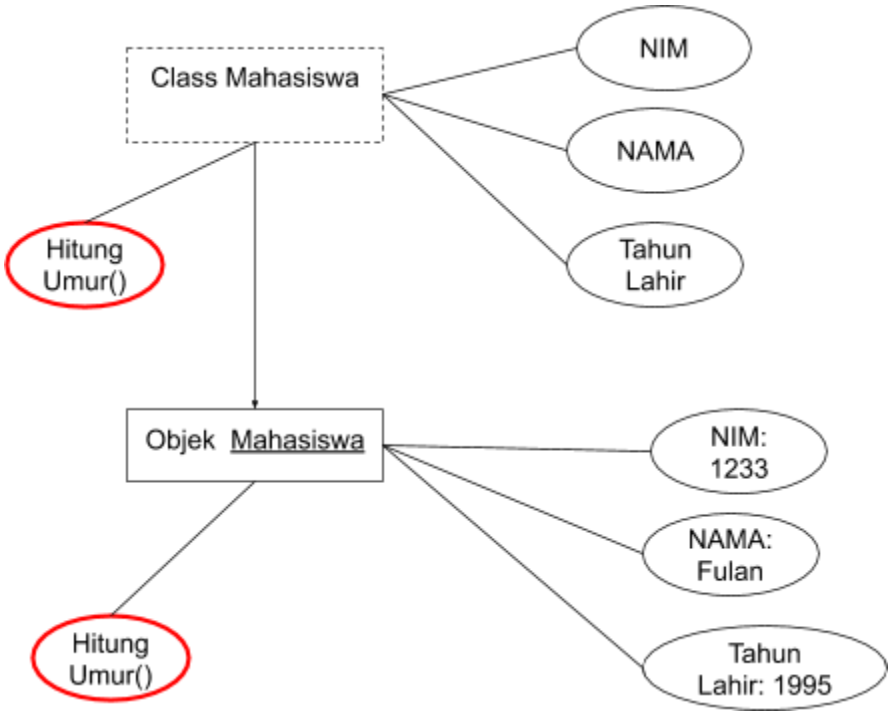
Feb 2017

Slide:
https://docs.google.com/presentation/d/1drLUilla56jGGz8hqlQWlJ_tDLo3nbk7BjkL4dlsv3o/edit?usp=sharing

todo: tambahkan soal latihan praktikum berbentuk kasus. Sesuaikan materi spt: obj1.obj2.obj3; parsing objek, satu objek dua kali new dst)

Object dan Class

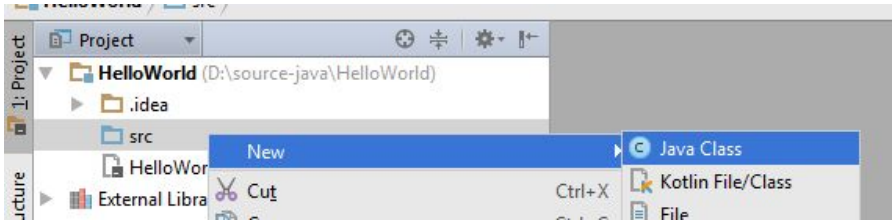
Class adalah tipe yang berisi atribut (variabel) dan fungsi (fungsi pada class seringkali disebut method). Sedangkan objek adalah instansiasi dari class. Sebagai contoh class Mahasiswa. Class ini memiliki atribut nama, nim, tahun lahir dan method menghitung umur. Class ini dapat di-instansiasi atau dicreate menjadi objek, misalnya objek mahasiswa Budi, Ahmad, Elfan dan seterusnya. Jadi class dapat dianggap sebagai pola atau "cetakan" objek. Gambar dibawah memperjelas hubungan antara class dan objek pada contoh ini



Di Java, akses terhadap method, menggunakan simbol titik. Misalnya `budi.nama`, `budi.alamat`, `budi.getUmur()` dan seterusnya.

Satu class umumnya disimpan dalam satu file. **Nama file harus sama dengan nama class** (termasuk huruf besar dan kecilnya!). Satu project di Java umumnya terdiri atas beberapa class. Berikut adalah contoh implementasi sebagian class Mahasiswa. Buatlah terlebih dulu Java project dengan nama "SistemAkademik".

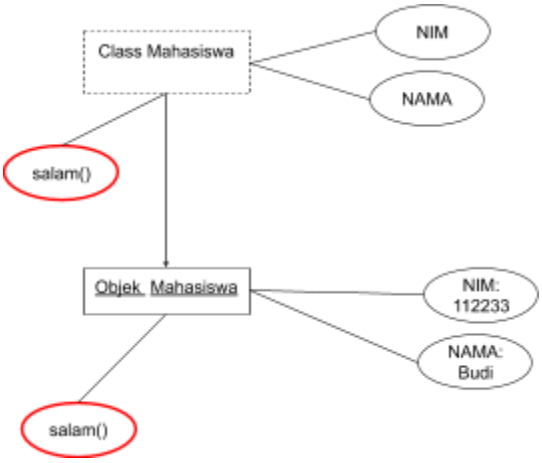
Sekarang kita akan membuat class Mahasiswa. Pilih "src" di project explorerl lalu klik kanan File → New → Java Class (gambar bawah)



Beri nama class Mahasiswa. Standard penamaan class di Java diawali dengan huruf besar, berbeda dengan variabel atau method yang diawali huruf kecil.

Ketik code berikut, class ini memiliki tiga atribut: nim, nama dan satu method sederhana untuk menampilkan nama. Semua atribut ini diset public, artinya boleh diakses secara langsung. Misalnya budi.nim, budi.nama. Konvensi penamaan method mengikuti penamaan variabel, diawali huruf kecil.

Perhatikan bagaimana class Mahasiswa didefinisikan, bagaimana objek dibuat dan cara mengakses atribut dan method.




```
public class Mahasiswa {
    String nim; //atribut
    String nama; //atribut

    public void  salam()  {  //method
      System.out.println("halo, nama saya " + nama);
    }

    public static void main(String[] args) {
      //create objek dari class Mahasiswa
      Mahasiswa mhsObjek = new Mahasiswa();
      mhsObjek.nim = "112233"; //  isi atribut
      mhsObjek.nama = "Budi"; //  isi atribut
```

```
      mhsObjek.salam();        //  panggil method
    }
}
```

Terlihat bahwa penggunaan Object mirip dengan penggunaan variabel. Class sama dengan tipe dan object sama dengan variabel.

Latihan 1

Tambahkan atribut tahunLahir pada class Mahasiswa. Tambahkan method hitungUmur dengan asumsi umur adalah 2017 dikurangi tahun lahir.

```
public class Mahasiswa {
   //atribut tahunLahir

   //method hitungUmur()
   //.....

   public static void main(String[] args) {
      Mahasiswa rudi = new Mahasiswa();
      mhs.nama = "Rudi Martami";
      mhs.thnLahir = 2000;
      int umur = mhs.hitungUmur();
      System.out.println("Umur: "+umur); //hasilnya Umur:17
   }

}
```

Latihan 2

Buatlah sebuah class bernama Kucing, yang memiliki method suara() dan atribut status. Jika statusnya diset "lapar" maka output dari method suara() adalah "meong meong meong!" dan jika diset "kenyang", maka output dari method suara() adalah "zzzz".

```
public class Kucing {
   //atribut
   //....

   //method suara()
   //.....

      public static void main(String[] args) {
            Kucing objKucing = new Kucing();
            objKucing.status = "lapar";
```

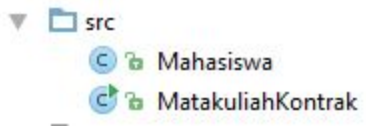
```
            objKucing.suara(); //meong meong meong
            objKucing.status = "kenyang";
            objKucing.suara(); //zzzzz
        }

}
```

Saat menjalankan sebuah class, Java secara otomatis akan mencari method khusus yang bernama `main(String args[])` untuk dijalankan (coba lihat-lihat contoh code sebelumnya, selalu ada method `main`). Method main ini cocok untuk memeriksa apakah class yang dibuat telah sesuai dengan keinginan.

Berikutnya kita akan membuat class MatakuliahKontrak. Class ini menyimpan matakuliah yang dikontrak mahasiswa beserta nilainya (kode matakuliah, tahun, semester dan nilai). Class ini memiliki method untuk menghitung mutu (4 untuk nilai A, 3 untuk nilai B dan seterusnya).

Pilih File → New →New Class, lakukan hal yang sama seperti saat membuat file Mahasiswa.

Sehingga dalam project sekarang terdapat dua class:



Buat code berikut:

```
public class MatakuliahKontrak {
   public String kodeMatakuliah;
   public int sem;
   public int tahun;
   public String nilai;
   public int sks;

   public int getMutu() {
        int mutu;
        if (nilai.equals("A")) {
            mutu = 4;
        } else if (nilai.equals("B")) {
            mutu = 3;
        } else if (nilai.equals("C")) {
            mutu = 2;
        } else  {
            mutu = 1;
        }
        return mutu;
   }

   public static void main(String args[]) {
        //testing class
        MatakuliahKontrak mk = new MatakuliahKontrak();
```

```
        mk.kodeMatakuliah="IK111";
        mk.nilai="A";
        mk.sem=1;
        mk.tahun=2008;
        System.out.println(mk.kodeMatakuliah + "  mutunya "+ mk.getMutu());
    }

}
```

Sekarang kita perlu mengubah class Mahasiswa untuk menyimpan matakuliah yang dikontrak mahasiswa tersebut. Satu Mahasiswa dapat memiliki lebih dari satu MatakuliahKontrak. Pada class Mahasiswa tambahkan variabel array untuk menyimpan matakuliah yang dikontrak mahasiswa. Tambahkan juga dua method, pertama method untuk menambahkan matakuliah (`tambahMatakuliah()`) dan yang kedua method untuk mencetak matakuliah yang telah diambil siswa (`printMatakuliah()`).

Tambahkan code di kelas mahasiswa sehingga menjadi sebagai berikut:

```
public class Mahasiswa {
    String nama;
    String nim;

    private MatakuliahKontrak[] daftarMK = new MatakuliahKontrak[10];
    private int pos=0; //posisi terakhir di array

    public void tambahMatakuliah (MatakuliahKontrak mk) {
        //tambah matakuliah ke array
        daftarMK[pos]=mk;
        pos++;
    }

    public void printMatakuliah(){
        //mencetak isi array daftar matakuliah
        System.out.println("KODE-MK;SEM;TAHUN;NILAI;MUTU");

        MatakuliahKontrak mk;
        for (int i=0;i<pos;i++) {
            mk = daftarMK[i]; //bisa juga langsung, misalnya daftarMK[i].sem
            System.out.println( mk.kodeMatakuliah+";"+mk.sem+";"+mk.tahun+";"+
             mk.nilai+ ";"+ mk.getMutu());
        }
    }


    public void  salam()  {  //method
        System.out.println("halo, nama saya " + nama);
    }

    public static void main(String[] args) {
        //create objek dari class Mahasiswa
        Mahasiswa budi  = new Mahasiswa();
        budi.nama = "Budi Martami"; //isi atribut
        budi.nim = "1111";


        MatakuliahKontrak mk1 = new MatakuliahKontrak();
        mk1.kodeMatakuliah="IK111";
        mk1.nilai="A";
```

```
        mk1.sem=1;
        mk1.sks=3;
        mk1.tahun=2008;
        budi.tambahMatakuliah(mk1);

        MatakuliahKontrak mk2 = new MatakuliahKontrak();
        mk2.kodeMatakuliah="MA222";
        mk2.nilai="C";
        mk2.sem=1;
        mk2.sks=3;
        mk2.tahun=2008;
        budi.tambahMatakuliah(mk2);

        budi.printMatakuliah();
    }
}
```

Tidak diperlukan import karena kedua class berada di satu package (mengenai package akan dijelaskan kemudian).

Latihan 3a
Tambahkan method print() pada class MatakuliahKontrak yang akan mencetak isi MataKuliahKontrak yang mencakup kode matakuliah, nama matakuliah dst.

Latihan 3b
Tambahkan method pencarian matakuliah yang pernah dikontrak pada class Mahasiswa. Parameter method ini adalah kode matakuliah, semester dan tahun. Return dari method ini adalah class matakuliahkontrak. Rangka method-nya sebagai berikut:

```
public MataKuliahKontrak getMkKontrak(String kodeMatakuliah, int
tahun, int sem) {

}
```

contoh kalau dijalankan:

```
public static void main(String[] args) {
        //create objek dari class Mahasiswa
        Mahasiswa budi  = new Mahasiswa();
        //isi data…
        //...
        MatakuliahKontrak mk = budi.getMkKontrak("IK111",2001, 2)
        mk.print();
    }
}
```

Latihan 3c
Tambahkan method penghitungan IPK mahasiswa berdasarkan nilai mutu (A=4 dst)

menggunakan getMutu.

```
public double hitungIPK() {

}
```

Constructor

Constructor adalah method khusus yang dipanggil saat object di-create. Contoh:

```
Mahasiswa mhs = new Mahasiswa();
```

Pada code di atas, maka akan dipanggil constructor Mahasiswa() yang merupakan constructor default.

Karakteristik constructor:

- Nama method constructor harus sama dengan nama class.
- Satu class dapat memiliki lebih dari satu constructor (dengan parameter yang berbeda-beda).
- Constructor tidak memiliki return
- Constructor dipanggil dengan new.

Apa kegunaan constructor? Constructor dapat digunakan untuk menginisialisasi object karena pasti dipanggil saat objek di-create. Sebagai contoh code berikut menambahkan constructor (bagian yang di-bold) sehingga nim, nama langsung diisi pada saat object dicreate.

```
public class Mahasiswa {
     public String nim;
     public String nama;
     private MatakuliahKontrak[] daftarMK = new MatakuliahKontrak[10];
     private int pos=0;

     public Mahasiswa(String vNim, String vNama) {
          nim = vNim;
          nama = vNama;
     }

     // … yang lain sama
}
```

Perhatikan penggunaan variabel vNim, Vnama. Penamaan ini untuk membedakan nama variabel dengan atribut method. Alternatif lain adalah dengan menggunakan keyword `this`. Contoh:

```
public class Mahasiswa {
     public String nim;
```

```
    public String nama;
    private MatakuliahKontrak[] daftarMK = new MatakuliahKontrak[10];
    private int pos=0;

    public Mahasiswa(String nim, String nama) {
        this.nim = nim;
        this.nama = nama;
    }
    // … yang lain sama
}
```

Keyword this menyatakan objek yang sedang dieksekusi. Sedangkan contoh penggunaan constructor adalah sebagai berikut :

```
Mahasiswa budi  = new Mahasiswa("1111","Budi Martami");
```

Catatan tentang urutan inisiasi. Code yang berada pada badan class akan dieksekusi lebih dulu dibandingkan constructor. Pada kode berikut `int nama= "-"` akan dieksekusi **lebih dulu** dibandingkan constructor.

```
public class Mahasiswa {
    private int nama="-"; //dieksesusi pertama, nama terisi "-"

    public Mahasiswa(String nama) {
        this.nama = nama;  //dieksekusi berikutnya saat di create
    }

}
```

Latihan 4

Tambahkan constructor untuk class MatakuliahKontrak sehingga atribut dapat diisi saat object di-create. Tambahkan pengecekan sbb: Semester harus berisi 1,2 dan 0 (1; ganjil, 2 genap, 0 semester pendek). Jika diluar itu, maka nilai semester yang salah diganti dengan -1. Nilai harus "A" sd "E", diluar itu nilai yang salah diganti dengan "-"

Contoh penggunaanya:

```
public static void main(String args[]) {
       //testing class
       MatakuliahKontrak mk = new MatakuliahKontrak("IK111", "F", 1, 2008);
       mk.print(); // nilai F berganti -
}
```

Static Method

Static method adalah method dapat dipanggil langsung melalui class-nya dan tidak memerlukan object. Contoh method Math.pow() untuk menghitung pangkat tidak memerlukan pembuatan object bertipe Math terlebih dulu.

Contoh berikut akan menghasilkan error, karena salam() bukan static method yang dapat dijalankan pada class Mahasiswa langsung.

```
Mahasiswa.salam();//<--error: method salam() tdk dpt dijalankan pd class!
```

perbaikannya:

```
Mahasiswa budi = new Mahasiswa();
budi.salam();  // ← boleh , salam() dijalankan pada objek.
```

Contoh static method yang selama ini kita terus gunakan adalah method main(String[] args). Perhatikan deklarasinya:

```
public static void main(String[] args)
```

Mengapa method main static? karena pada saat Java menjalankan class tersebut, belum ada object yang dicreate.

Static method tidak diperbolehkan mengakses atribut atau method non static pada class tersebut (mengapa?). Perhatikan walaupun method main berada di dalam class mahasiswa, tapi di dalam method ini kita tidak dapat mengakses atribut.

```
Class Mahasiswa {
   string nama;
   public static void main(String[] args) {
         nama = "Budi"; //ERROR!
  }
}
```

Latihan 5

Tambahkan method static untuk class MatakuliahKontrak, yaitu
`int getMutuStatic(String nilai)`, yang menerima input nilai A sd E dan mengeluarkan output nilai mutu yaitu 4 sd 0.

Apa bedanya dengan getMutuStatic dengan getMutu?

Atribut Static

Pada contoh class Mahasiswa, terdapat atribut seperti Nama dan NIM yang memiliki nilai berbeda di setiap objek.

Contoh:

```
    public static void main(String[] args) {
      Mahasiswa mhs1 = new Mahasiswa();
      Mahasiswa mhs2 = new Mahasiswa();
      mhs1.nim="1234";
      mhs2.nim="5678";
    }
```

Terkadang kita membutuhkan atribut yang berlaku untuk semua objek. Atribut ini disebut dengan static atribut.

Cobalah code berikut

```
public class Siswa {
    double nilaiMentah;
    static double bobot = 0.0; //perhatikan penggunaan keyword static

    public double getNilai() {
      return nilaiMentah * bobot;
    }

    public static void main(String[] args) {
      Siswa s1 = new Siswa();
      Siswa s2 = new Siswa();
      s1.nilaiMentah = 50;
      s2.nilaiMentah = 100;
      Siswa.bobot = 0.1;// class Siswa,bukan objek s1 atau s2

      //atribut static bobot berlaku untuk semua objek
      System.out.println(s1.getNilai());
      System.out.println(s2.getNilai());
    }

}
```

Static Final

Umumnya atribut static digunakan untuk menyimpan konstanta. Untuk menjamin agar nilainya tidak berubah, maka dapat ditambahkan keyword final (gambar bawah)

```
public static final double batasSem = 8;
```

Method Overloading

Method overloading adalah method yang memiliki nama sama tetapi berbeda parameter. Contoh: `String.subString` yang memiliki dua bentuk.

```
public String subString(int beginIndex)
public String subString(int beginIndex, int endIndex)
```

Method yang akan dipanggil akan disesuaikan dengan urutan dan tipe parameternya. Jadi "Budi".substring(1) dan "Budi".substring(2,4) akan memanggil dua method yang berbeda.

Sebagai contoh, pada class Mahasiswa telah disediakan method
`public void tambahMatakuliah (MatakuliahKontrak mk)`

Parameter pada method diatas adalah MatakuliahKontrak, kita akan membuat method tambahMatakuliah yang menerima parameter lain:

```
public void tambahMatakuliah(String kodeMatakuliah, int sem,int
tahun, String nilai) {
   MatakuliahKontrak mk = new MatakuliahKontrak();
   mk.kodeMatakuliah = kodeMatakuliah;
   mk.sem = sem;
   mk.tahun = tahun;
   mk.nilai = nilai;
   tambahMatakuliah(mk); //method tambahMatakuliah yg sebelumnya
}
```

Latihan 6

Buat method tambahan `printMatakuliah(String kodeMk)` di class Mahasiswa dengan parameter string kode matakuliah yang akan diprint.

Multiple Constructor

Sudah dipelajari sebelumnya bahwa constructor adalah method khusus yang dipanggil saat objek dicreate. Menggunakan teknik yang sama dengan overload, maka dapat terdapat lebih dari satu constructor di dalam sebuah class.

Pada contoh sebelumnya, kita telah memiliki constructor pada class mahasiswa:

```
public Mahasiswa(String nim, String nama)
```

Sekarang kita akan membuat constructor lain yang menerima input nama saja, nim dibuat secara manual (misal mahasiswa khusus). Perhatikan penggunaan this()

```
// … sama
public Mahasiswa(String nama) {
   this("0000",nama); // pemanggilan constructor sebelumnya
}

public static void main(String[] args {
   Mahasiswa fulan = new Mahasiswa("111", "Fulan");
   Mahasiswa budi  = new Mahasiswa("Budi Martami");
}
```

Latihan 7:

Buatlah constructor mahasiswa untuk meng-clone objek mahasiswa lain.

Misalnya kita telah mempunyai objek budi, kita ingin membuat objek ke-2 yang merupakan clone dari objek budi (isinya persis sama), maka kita akan melakukan:

```
Mahasiswa budi2 = new Mahasiswa(budi); //budi2 clone budi
```

Perhatikan bahwa parameter dari construtor diatas adalah class Mahasiswa, sehingga templatenya constructornya:

```
public Mahasiswa(Mahasiswa mhs) {
      //...cloning
}
```

Kontrol Akses terhadap Atribut dan Method: Public dan Private

Dalam code sebelumnya, kita sering melihat penggunaan keyword “public” di depan deklarasi method dan atribut suatu kelas.

Public artinya atribut dan method dapat diakses kelas lain dengan bebas, sedangkan private artinya hanya dapat diakses oleh kelas tempat method/atribut dideklarasikan. Sebagai contoh:

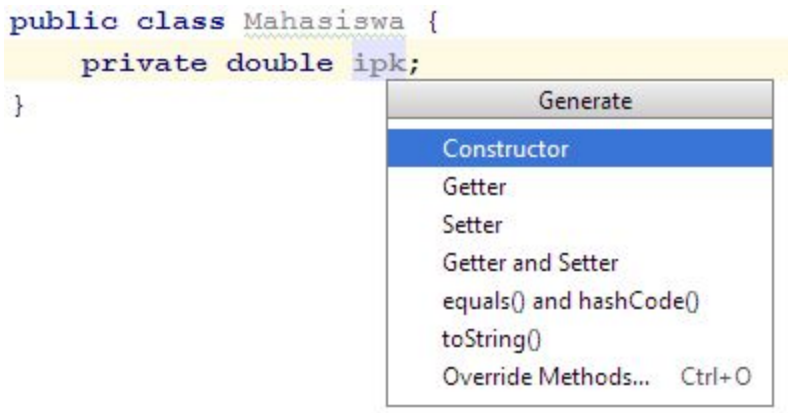
```
public class Mahasiswa {
     public String nim;
     public String nama;
     private int pos=0;
}
```

Nim dan Nama dapat diakses dari luar sedangkan atribut pos hanya dapat diakses oleh kelas Mahasiswa saja.

Sebaiknya semua method dan atribut diset sebagai private kecuali ada alasan kuat. Semakin sedikit method public, semakin stabil kelas (dapat diubah tanpa menggangu kelas yang menggunakannya). Atribut ini dapat diisi dengan “setter” dan “getter”. Sebagai contoh, ketik code berikut:

```
public class Mahasiswa {
   private double ipk;
}
```

Lalu tekan alt-ins dan pillih Getter dan Setter (gambar bawah)



maka secara otomatis akan dibuatkan setter dan getternya

```
public class Mahasiswa {
   private double ipk;

   public double getIpk() {
       return ipk;
   }

   public void setIpk(double ipk) {
       this.ipk = ipk;
   }
}
```

Cakupan lain: protected dan tanpa modifier akan dijelaskan kemudian dalam materi inheritance.

Collection

Collection adalah framework yang disediakan Java untuk menyimpan kumpulan objek. Sebenarnya array juga dapat digunakan untuk menyimpan kumpulan objek, tetapi array memiliki kelemahan ukuranya tidak dapat diubah. Kelebihan lain collection juga menyediakan berbagai method standard untuk membantu programmer seperti pencarian.

Contoh collection yang paling banyak digunakan adalah ArrayList. Penggunaannya mirip dengan array.

Pada program berikut kita akan membuat ArrayList yang elemenya adalah objek bertipe Integer. Perlu dibedakan antara Integer dan int. Integer adalah class sedang int adalah tipe data primitif.

```
public static void main(String[] args) {
   ArrayList<Integer> myArray = new ArrayList<>();
   //tambah data
   myArray.add(3);
   myArray.add(5);
   //akses data
   int data = myArray.get(0); //data=3
   System.out.println("Data:"+data);
}
```

Perhatikan penggunaan tanda < > untuk menyatakan jenis elemen, ini disebut generic.

Walaupun Integer dan int berbeda (satu class dan satunya tipe variabel), hal seperti ini bisa dilakukan: `int data = myArray.get(1);` Ini disebabkan Java menerapkan autoboxing yang mengkonversi dari int ke Integer dan sebaliknya secara otomatis.

Dengan mendefinisikan tipe elemen koleksi, maka dapat dijamin tipe elemen akan selalu konsisten. Misalnya kode berikut akan memberikan error pada saat compile:

```
myArray.add("3"); //ERROR, elemen myArray harus bertipe integer
```

Contoh ArrayList untuk tipe class yang lain:

```
ArrayList<Double> myArray = new ArrayList<>();
ArrayList<String> myArray = new ArrayList<>();
ArrayList<Mahasiswa> myArray = new ArrayList<>();
```

Loop pada collection

Untuk loop, dapat dilakukan seperti array biasa, dengan cara yang paling mudah dibaca adalah menggunakan for-each

```
for (int data:myArray) {
   System.out.println("data:"+data);
}
```

Latihan 8:
Ganti implementasi array di class Mahasiswa dan MatakuliahKontrak di halaman 5 (sebelum latihan 3) dengan ArrayList.

Method-method penting di Collection.

Collection menyediakan berbagai method untuk membantu programmer, method yang paling umum digunakan adalah pencarian elemen. Menggunakan contoh sebelumnya:

```
if (myArray.contains(5)) { //cari elemen
   System.out.println("Ada angka 5");
}
```

Method yang lain adalah sorting yang disediakan class Collection, menggunakan contoh sebelumnya:

```
ArrayList<Integer> myArray = new ArrayList<>();
//tambah data
myArray.add(3);
myArray.add(5);
myArray.add(1);
myArray.add(2);

Collections.sort(myArray);
```

```
for (int data:myArray) {
   System.out.println("data:"+data);
}

@todo:custom sorting
```

Latihan 9:
Tambahkan implementasi di class Mahasiswa dan MatakuliahKontrak, untuk pencarian matakuliah yang di kontrak dengan method contains. Jawabannya hanya "ada" atau "tidak"

HashSet

Pada contoh sebelumnya kita menggunakan ArrayList, java menyedikan beberapa class collection yang dapat digunakan sesuai dengan karakteristik akses dan data di dalam koleksi. Jika kita memerlukan pencarian data yang cepat (misalnya untuk mencari ID), kelas koleksi yang cocok digunakan adalah Hashset.

Untuk pencarian, kompleksitas HashSet adalah O(1) sedangkan ArrayList O(n), tabel berikut memperlihatkan perbedaan kecepatan antara HashSet dan ArrayList:

Jumlah Elemen	Peningkatan dibandingkan ArrayList
3	2x
10	3x
50	6x
200	12
10000	532x

Dapat dilihat dengan jumlah data 100000, HashSet sekitar 500 kali lebih cepat dibandingkan ArrayList.

Perbedaan utama antara HashSet dengan ArrayList adalah tidak boleh ada duplikasi elemen di dalam HashSet dan tidak ada jaminan urutan (data tidak dapat diakses berdasarkan indeks).

Code berikut memperlihatkan contoh penggunaan HashSet

```
public static void main(String[] args) {
   HashSet<Integer> myArray = new HashSet<>();
   myArray.add(1);
   myArray.add(3);
   myArray.add(5);
   myArray.add(2);
   for (int i:myArray) {
       //tidak ada jaminan urutan sama dengan urutan saat add
       System.out.println(i);
   }
   if (myArray.contains(3)) {
       System.out.println("Angka 3 ditemukan!");
   }
}
```

Karena elemen HashSet harus unik (tidak ada duplikasi), maka saat proses add elemen sudah ada sebelumnya, maka elemen tersebut tidak akan masuk ke koleksi, sebagai contoh code berikut menambahkan data yang sudah ada sebelumnya:

```
myArray.add(1);
myArray.add(3);
myArray.add(5);
myArray.add(2);
myArray.add(5); //duplikasi
myArray.add(2); //duplikasi
myArray.add(2); //duplikasi
for (int i:myArray) {
   //tidak ada jaminan urutan sama dengan add
   System.out.println(i);
}
```

hasilnya adalah sbb:

```
1
2
3
5
```

HashMap

Seringkali kita memerlukan penyimpanan untuk elemen berbentuk pasangan <key, value>. Misalnya untuk kelas Mahasiswa, pasangannya seperti ini <String: nim , Mahasiswa mhs> maka kelas koleksi yang paling cocok digunakan adalah HashMap. Berikut contoh penggunaannya:

```
public static void main(String[] args) {
   HashMap<String,Integer> hm = new HashMap<>();
   hm.put("1234",5); //key, value
```

```
hm.put("1235",10);

//search berdasaran key 1235, ambil valuenya
int val = hm.get("1235");

//search berdasarkan key
if (hm.containsKey("1234")) {
    System.out.println("Mengandung key 1234");
}

//loop berdasarkan key
System.out.println("Semua key");
for (String key : hm.keySet()) {
    System.out.println("Key:"+key);
}

//loop berdsarkan value
System.out.println("Semua value");
for (int value : hm.values()) {
    System.out.println("Value:"+value);
}

//loop berdsarkan key-value
System.out.println("Semua key value");
for (Map.Entry<String, Integer> entry : hm.entrySet())
{
    String key = entry.getKey(); //ambil key
    Integer value = entry.getValue();//ambil value
    System.out.println("Key:"+key);
    System.out.println("Value:"+value);
}
```

Latihan 10:
Ganti implementasi MatakuliahKontrak di halaman 5 (sebelum latihan 3) dengan HashMap <String kode_matakuliah, MatakuliahKontrak mk>. Implementasikan pencarian matakuliah yang di kontrak mahasiswa (latihan 3b).

Latihan proyek:

Buatlah software untuk menghitung gaji yang perlu dibayarkan perusahaan. Buatlah class berikut:

1. Class Pegawai yang berisi informasi id, nama, level pegawai, jumlah absen, jumlah lembur.
2. Class InputPegawai yang menangani input.
3. Class KumpulanPegawai untuk menyimpan data semua pegawai dan mencetaknya.
4. Class ProsesGaji yang menghitung gaji pegawai. Buat sendiri asumsi untuk menghitung gaji berdasarkan data pegawai. Simpan pegawai dalam collection.

Tambahkan class, atribut method lain sesuai keperluan.

Contoh tampilan:

```
Menu
   1. Tambah pegawai.
   2. Cetak semua pegawai.
   3. Tampilkan gaji pegawai
```

Catatan: untuk menu 1, user memasukan data id, nama dsb. Untuk menu 2, contoh outputnya adalah sebagai berikut (hint: gunakan String.format):

```
id    nama       level      jumlah absen    jumlah lembur
=========================================================
1     Budi       9          2               10
2     Ahmad      7          1               5
dst..
```

Untuk menu 3, contoh outputnya adalah sebagai berikut :

```
id    nama       Gaji (dlm ribu)
==============================
1     Budi                  5000
2     Ahmad                 6500
dst
```