PRAKTIKUM 2

PEMBUATAN CLASS DAN OBJEK

A. TUJUAN PEMBELAJARAN

1. Memahami mengenai konsep Class dan Objek
2. Mampu mengubah konsep Objek di dunia nyata menjadi objek dalam pemrograman.
3. Memahami cara membuat class.
4. Memahami cara mengakses data member dan method dari class.

B. DASAR TEORI

MENGENAL OBJEK & CLASS

1. Paradigma Objek

- Paradigma adalah suatu cara pandang atau cara berpikir
- Paradigma objek adalah cara pandang yang memandang SEGALA SESUATU sebagai OBJEK
- Semua aspek dalam Java programming dapat dianggap sebagai objek, -kecuali TIPE DATA PRIMITIF-, karena semua library dan objek dalam Java memiliki akar awal class java.lang.Object
- Berbagai benda di sekitar kita adalah objek nyata yang dapat dilihat, seperti : kucing, meja, rumah, orang , dll

2. Konsep Objek & Class

- Java adalah merupakan OOP sehingga konsep objek dan class menjadi penting untuk dipahami
- OOP memiliki banyak kelebihan dibandingkan dengan bahasa prosedural, di antaranya :
    - Reusabilitas
    - Pembangunan program lebih cepat
    - Fleksibilitas lebih tinggi

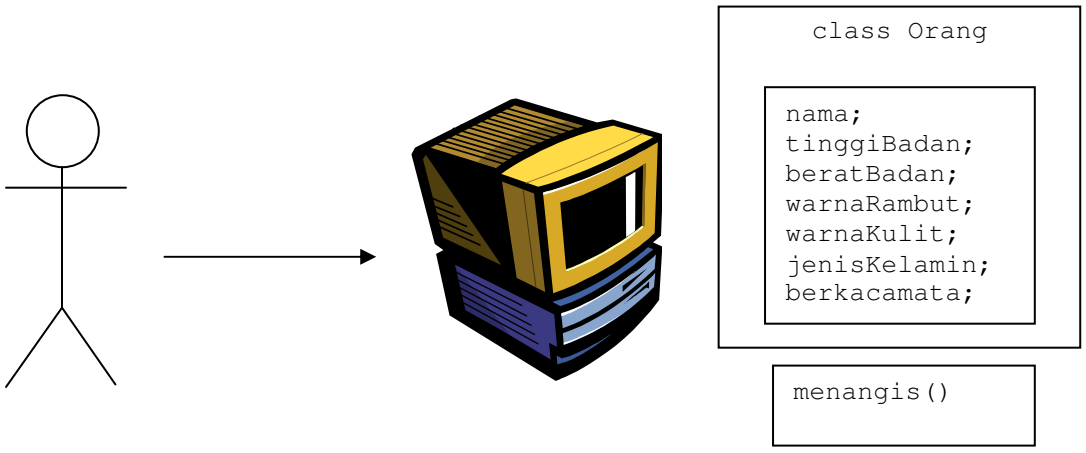
- Ekstensibilitas
- Less maintenance

- Persoalannya, bagaimana memindahkan pemikiran objek di dunia nyata menjadi objek di dunia software atau pemrograman, khususnya Java
- Ambil contoh objek nyata yang akan dipindahkan adalah objek orang

3. Data Member

- Setiap objek yang dinamakan 'orang' pasti memiliki : nama, tinggi badan, berat badan, warna rambut, warna kulit, jenis kelamin, menggunakan kacamata, dll
- Ciri-ciri tersebut dapat dipindahkan menjadi variabel-variabel dari class yang sering disebut sebagai : data member
- Contoh pemisalan objek orang nyata menjadi kode program dalam class Orang :

```
class Orang {
   String nama;          //nama orang
   int tinggiBadan;      //dalam cm
   int beratBadan;       //dlm kg
   String warnaRambut;   //hitam, pirang, coklat
   String warnaKulit;    //sawoMatang, hitam, putih
   String jenisKelamin;  //pria atau wanita
   boolean berkacamata;  //bila berkacamata berarti true
}
```



Gambar 2.1 Memindahkan orang dari dunia nyata menjadi class Orang

Class dapat diumpamakan seperti spesifikasi atau blueprint. Dalam hal ini, Tuhan menciptakan manusia dengan spesifikasi tertentu, sehingga kita mengenal istilah SPESIES manusia. Jadi dapat diumpamakan bahwa Tuhan memiliki class Orang yang kemudian membuat banyak objek dari class Orang tsb, dan contoh objek tersebut adalah Anda sendiri.

Objek dalam pemrograman adalah objek yang dibuat dari class tertentu. Dari definisi class Orang di atas, kita bisa membuat objek-objek berdasar class tersebut. Objek-objek yang dibuat perlu disimpan dalam variabel yang akan menyimpan referensi/address dari objek yang dibuat. Proses pembuatan objek sering disebut sebagai instansiasi class, sedangkan objeknya disebut sebagai instance dari class.

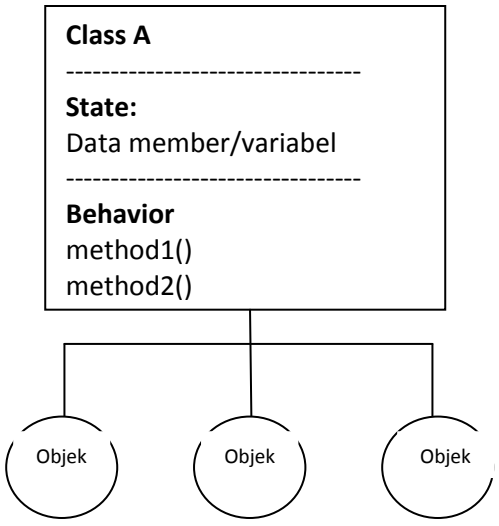
4. Method

- Selain memiliki atribut atauSTATE yang diimplementasikan sebagai data member di atas, manusia juga dapat melakukan suatu aksi atau pekerjaan tertentu (BEHAVIOR)
- Contoh aksi/behavior yang umum adalah menangis dan tertawa
- Kedua behavior tsb bisa dipindahkan ke dalam bahasa pemrograman menjadi method sbb :

```
void menangsi() {
   System.out.println("hik..hikk..hik…");
}

void tertawa() {
   System.out.println("ha..ha..ha..ha..");
}
```

- Method merupakan perwujudan aksi atau tindakan dari dunia nyata di dalam pemrograman komputer.
- Method dalam dunia pemrograman juga "pasti melakukan sesuatu aksi", misalnya menampilkan String di konsol



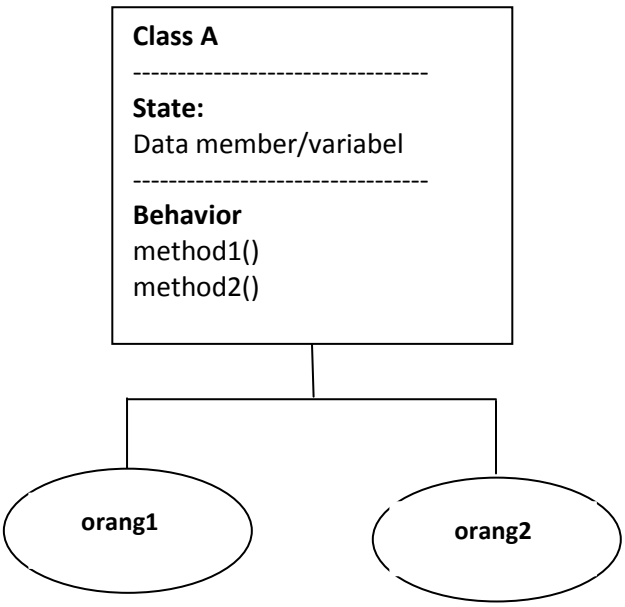
Gambar 2.2 Ilustrasi perbedaan antara class dan objek

- Dari gambar di atas dapat dipahami bahwa suatu class dapat memiliki banyak objek, dan setiap objek akan mewarisi data member dan method yang sama dari class
- Untuk membuat objek Orang dari class Orang, gunakan keyword **new** sbb :

```
Orang orang1 = new Orang("Izzuddin AAfif");

Orang orang2 = new Orang("Muhammad Fairuz");
```

- Kesimpulannya : objek bertipe Orang dapat dibuat dari class Orang dan tiap objek perlu disimpan dalam variabel untuk menyimpan referensi/address dari lokasi di mana sebenarnya objek disimpan
- Semua objek memiliki data member yang sama tetapi setiap objek dapat memiliki state atau nilai data member yang berbeda (hanya nama dan tipe variabel yang sama)



Gambar 2.3 Ilustrasi pembuatan objek dari class

ATURAN PEMBUATAN CLASS

Deklarasi class dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier> class <nama_class> {
      [deklarasi_atribut]
      [deklarasi_konstruktor]
      [deklarasi_metode]
}
```

Contoh:

```
public class Siswa {
      …
}
```

Deklarasi atribut dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier><tipe><nama_atribut> ;
```

Contoh:

```
public class Siswa {
      public int nrp;
      public String nama;
}
```

Deklarasi metode dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier><return_type><nama_metode> ([daftar_argumen]) {
      [<statement>]
}
```

Contoh:

```
public class Siswa {
      public int nrp;
      public String nama;
      public void info() {
            System.out.println("Ini siswa PENS");
      }
}
```

Untuk dapat mengakses anggota-anggota dari suatu obyek, maka harus dibuat instance dari class tersebut terlebih dahulu. Berikut ini adalah contoh pengaksesan anggota-anggota dari class Siswa:

```
public class TesSiswa {
      public static void main(String args[]) {
            Siswa it=new Siswa();
            it.nrp=5;
            it.nama="Andi";
            it.info();
      }
}
```

C. TUGAS PENDAHULUAN

Membuat review mengenai Class dan Objek dan beri contoh 1 membuat class dan objek.

D. PERCOBAAN

Percobaan 1 : Mengakses data member suatu class.

Amati program dibawah ini:

```
public class Siswa {
    int nrp;
    public void setNrp(int i) {
        nrp=i;
    }
```

```
}

public class Test {
    public static void main(String args[]) {
        Siswa anak=new Siswa();
        anak.nrp = 5 ;
        System.out.println(anak.nrp);
    }
}
```

Percobaan 2 : Mengakses method suatu class.
Amati program dibawah ini:

```
public class Siswa {
    int nrp;
    public void setNrp(int i) {
        nrp=i;
    }
}

public class Test {
    public static void main(String args[]) {
        Siswa anak=new Siswa();
        anak.setNrp(5);
        System.out.println(anak.nrp);
    }
}
```

Percobaan 3 : Mengakses method suatu class.
Amati program dibawah ini:

```
public class Siswa {
    int nrp;
    String nama;

    public void setNrp(int i) {
        nrp=i;
    }

    public void setNama(String i) {
        nama=i;
    }
}

public class Test {
    public static void main(String args[]) {
        Siswa anak=new Siswa();
        anak.setNrp(5);
      anak.setNama("Budi");
      System.out.println(anak.nrp);
       System.out.println(anak.nama);
    }
}
```

Percobaan 4 : Membuat Class dan Konstruktor dengan parameter.

```
class Puppy{

   int puppyAge;

   public Puppy(String name){
      // This constructor has one parameter, name.
      System.out.println("Passed Name is :" + name );
   }
   public setAge( int age ){
       puppyAge = age;
   }

   public getAge( ){
       System.out.println("Puppy's age is :" + puppyAge );
       return puppyAge;
   }
   public static void main(String []args){
      Puppy myPuppy = new Puppy( "tommy" );

      myPuppy.setAge( 2 );

      myPuppy.getAge( );

      System.out.println("Variable Value :" + myPuppy.puppyAge );
   }
}
```

Percobaan 5 :Membuat class Point dan menghitung jarak antara dua Point. Buatlah fungsi utama untuk membuat dua Point dan menghitung jarak antara dua Point tersebut !

```
class Point {
   public double x, y;

Point(double x_value, double y_value) {
      x = x_value;
      y = y_value;
   }
   public void clear() {
      this.x = 0;
      this.y = 0;
   }
   public double distance(Point that) {
      double xDiff = x - that.x;
      double yDiff = y - that.y;
      return Math.sqrt(xDiff * xDiff + yDiff * yDiff);
   }
}
```

Percobaan 6 :Membuat array objek

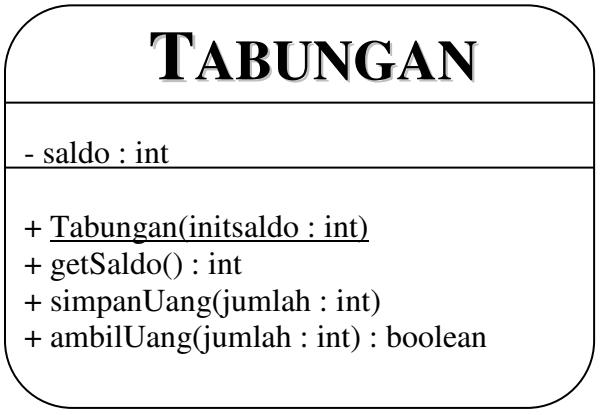
```
class Account{
```

```
  int a;
  int b;
 public void setData(int c,int d){
   a=c;
   b=d;
 }
 public void showData(){
   System.out.println("Value of a ="+a);
   System.out.println("Value of b ="+b);
 }
}

class ObjectArray{
   public static void main(String args[]){
    Account obj[] = new Account[2] ;
obj[0] = new Account();
    obj[1] = new Account();
    obj[0].setData(1,2);
    obj[1].setData(3,4);
    System.out.println("For Array Element 0");
    obj[0].showData();
    System.out.println("For Array Element 1");
     obj[1].showData();
  }
}
```

E. LATIHAN

Latihan1 : Mengimplementasikan UML class diagram dalam program untuk class Tabungan

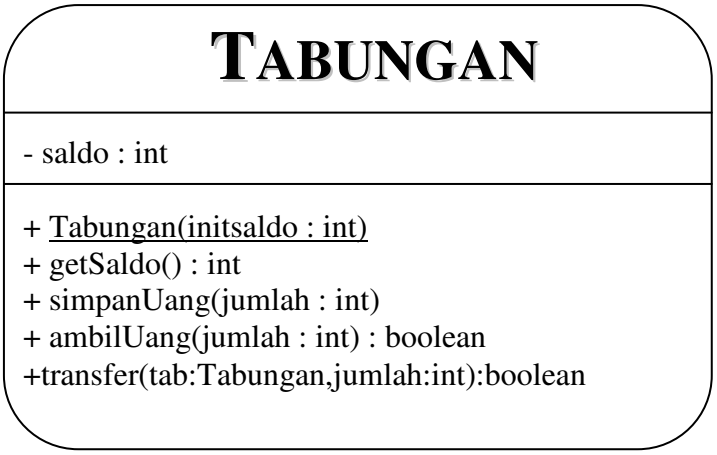


Gambar 2.4Class Tabungan

Transformasikan class diagram diatas ke dalam bentuk program? Jalankan file TesTugas1.class. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti diatas.

```
Saldo awal : 5000
Jumlah uang yang disimpan : 3000
Jumlah uang yang diambil : 6000   ok
Jumlah uang yang disimpan : 3500
Jumlah uang yang diambil : 4000   ok
Jumlah uang yang diambil : 1600   gagal
Jumlah uang yang disimpan : 2000
Saldo sekarang = 3500
```

Latihan 2 : Menambahkan pada class Tabungan method transfer().



Gambar 2.4Class Tabungan dengan tambahan method transfer()

F. LAPORAN RESMI

Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.