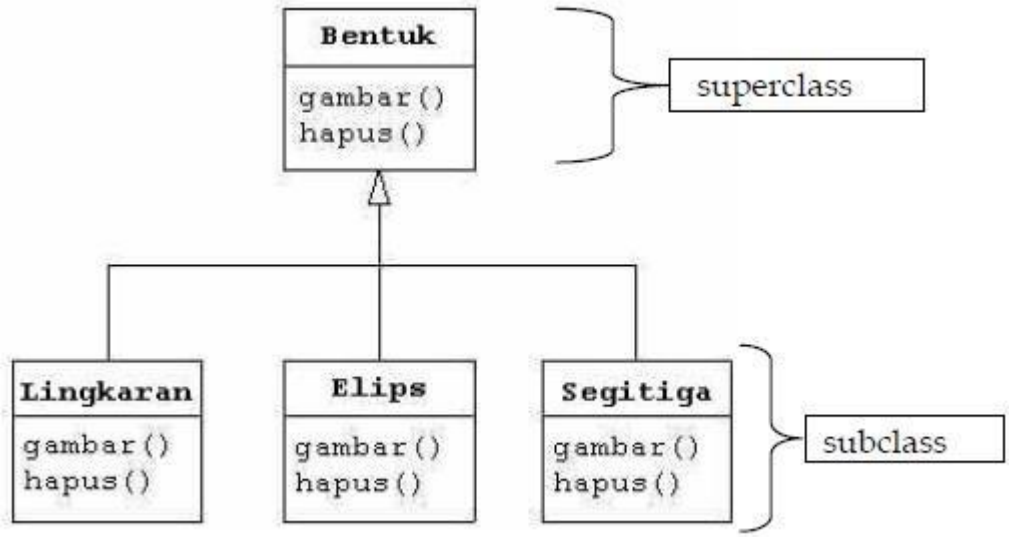
MODUL 7
POLIMORFISME

7.1 Polimorfisme

Polimorfisme mempunyai makna sesuatu yang memiliki banyak bentuk, yaitu memiliki nama sama, tetapi memiliki kelakuan (*behaviour*) yang berbeda.



Perhatikan gambar diagram di atas !

Class Bentuk yang merupakan class induk (*superclass*) dari class Lingkaran, Elips dan Segitiga mempunyai method **gambar()** dan **hapus()**. Class-class anak (*subclass*) juga mempunyai method **gambar()** dan **hapus()**. Meskipun keempat class tersebut mempunyai nama method yang sama, tetapi isi (source code/yang dilakukan/output) dari masing-masing method tersebut berbeda.

Jika kita menginginkan sebuah objek yang dapat memanggil setiap method (yaitu method gambar & hapus) yang ada pada setiap class (pada superclass maupun subclass), maka gunakanlah teknik **Polimorfisme**. Polimorfisme hanya berlaku pada method dan tidak berlaku untuk atribut.

Untuk mendapatkan operasi Polimorfisme dari suatu method, maka method tersebut haruslah merupakan method yang ada di class induk (lihat diagram diatas bahwa method gambar() dan hapus(), selain terdapat di class-class turunan class Bentuk, juga terdapat di class Bentuk).

Contoh source code implementasi Polimorfisme :

Bentuk.java

```
1  public class Bentuk {
2      protected void gambar() {
3        System.out.println("superclass -> Menggambar ");
4       }
5
6       protected void hapus() {
7        System.out.println("superclass -> Menghapus Gambar");
8       }
9  }
```

Lingkaran.java

```
1  public class Lingkaran extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Lingkaran");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Lingkaran");
8      }
9  }
```

Segitiga.java

```
1  public class Segitiga extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Segitiga");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Segitiga");
8      }
9  }
```

Elips.java

```
1  public class Elips extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Elips");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Elips");
8      }
9
10 }
```

Cetakgambar.java

```
1  public class Cetakgambar extends Bentuk {
2
3      private void tampil(Bentuk[] obj) {
4          // Polimorfisme
5          // Memanggil method yang sama yaitu method gambar() dan hapus()
6          // pada masing-masing class
7          for (int i=0;i<obj.length;i++)
8           {  obj[i].gambar();
9              obj[i].hapus();
10             System.out.println("==================");
11          }
12     }
13
14     public static void main (String []args) {
15          Bentuk[] obj = {  new Lingkaran(),
16                            new Elips(),
17                            new Segitiga()
18                         };
19          Cetakgambar cetak = new Cetakgambar();
20
21          // Menampilkan method gambar() & hapus() pada class Bentuk (superclass)
22          cetak.gambar();
23          cetak.hapus();
24          System.out.println("==================");
25
26          // Overriding
27          // Menumpuk method gambar() & hapus() pada class Bentuk (superclass)
28          // dengan method gambar() & hapus() pada subclass-nya
29          // yaitu class Lingkaran, Elips dan Segitiga
30          cetak.tampil(obj);
31     }
32 }
```

Pada class Cetakgambar terdapat variabel/objek *obj* yang bertipe *class Bentuk*. Maka dapat dikatakan bahwa variabel *obj* dapat berperan sebagai *Lingkaran*, *Elips*, atau *Segitiga*. Hal ini didasarkan bahwa pada kenyataannya setiap objek dari class Induk (*superclass*) dapat berperan sebagai class-class turunannya sebagaimana sepeda motor adalah kendaraan, pelajar dan mahasiswa adalah orang/manusia.

Perhatikan bahwa Polimorfisme tidak sama dengan overloading !!!

7.2. Method Overriding

Overriding method adalah kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya. Contoh overriding method dapat dilihat pada class-class turunan dari class Bentuk yang mendefinisikan kembali method gambar() dan method hapus() dari class induknya.

LATIHAN 7

1. Buatlah aplikasi penjumlahan, pengurangan, perkalian dan pembagian menggunakan konsep polimorfisme dengan memasukkan parameter bilangan A = 10.5 dan bilangan B = 0.5 !