Polymorphism

Pokok Pembahasan

- Polimorfisme

Tujuan Pembelajaran

Dengan materi pembelajaran ini teman-teman diharapkan dapat:

- Mengtahui apa itu Polimorfisme
- Dapat mengimplementasi kan Polimorfisme di dalam OOP

Polymorphism (polimorfisme –perbedaan bentuk) Polymorphisme artinya mempunyai banyak bentuk. Dua objek atau lebih dikatakan sebagai polymorphic, bila objek-objek itu mempunyai antar muka yang identik namun mempunyai perilaku perilaku yang berbeda.

Polimorfisme

Kata polimorfisme yang berarti satu objek dengan banyak bentuk yang berbeda, adalah konsep sederhana dalam bahasa pemrograman berorientasi objek yang berarti kemampuan dari suatu variabel referensi objek untuk memiliki aksi berbeda bila method yang sama dipanggil, dimana aksi method tergantung dari tipe objeknya.

Kondisi yang harus dipenuhi supaya polimorfisme dapat diimplementasikan adalah :

- Method yang dipanggil harus melalui variabel dari basis class atau superclass.
- Method yang dipanggil harus juga menjadi method dari basis class.
- Signature method harus sama baik pada superclass maupun subclass.
- Method access attribute pada subclass tidak boleh lebih terbatas dari basis class.

Polimorfisme pada Java memiliki 2 macam yaitu diantaranya:

- Static Polymorphism (Polimorfisme statis).
- Dynamic Polymorphism (Polimorfisme dinamis).

Perbedaan sederhana, jika Polymorphism yang statis dapat di implementasikan dengan overloading atau secara langsung. Sementara Polymorphism dinamis di implementasikan menggunakan interface atau abstract.

Percobaan Polymorphism dinamis menggunakan Abstract Class

- Polimorfisme.java

```
package main;

abstract class Bentuk {

   protected int panjang;
   protected int lebar;

   //   method super-class yang akan di overloading pada sub-class
   public String getBentuk() {
            return "Bentuk Dasar";
   }

   //   method super-class yang akan di overloading pada sub-class
   public abstract int hitungLuas();
}

class BujurSangkar extends Bentuk {


   public BujurSangkar(int panjang1, int lebar1) {
            this.panjang = panjang1;
            this.lebar = lebar1;
   }

   // implementasi baru getBentuk pada sub-class BujurSangkar
   public String getBentuk() {
            return "Bentuk Bujur Sangkar";
   }

   // implementasi baru hitungLuas pada sub-class BujurSangkar
   public int hitungLuas() {
            return panjang * lebar;
   }

}

class SegiTiga extends Bentuk {

   public SegiTiga(int panjang2, int lebar2) {
            this.panjang = panjang2;
            this.lebar = lebar2;
   }
```

```
    // implementasi baru getBentuk pada sub-class SegiTiga
    public String getBentuk() {
                return "Bentuk Segi Tiga";
    }

    // implementasi baru hitungLuas pada sub-class SegiTiga
    public int hitungLuas() {
                return this.panjang * this.lebar / 2;
    }

}

class Polimorfisme {

    /*
    definisikan method cetakLuasBentuk() dengan parameter super-class Bentuk
    untuk mengakses member method dari sub-class
    */
    public static void cetakLuasBentuk(Bentuk btk) {
        System.out.println(btk.getBentuk() + "dengan luas"
                + btk.hitungLuas());
    }

    public static void main(String[] args) {

        BujurSangkar bs = new BujurSangkar(10, 20);
        BujurSangkar bs1 = new BujurSangkar(10, 20);

        SegiTiga st = new SegiTiga(5, 10);
        SegiTiga st1 = new SegiTiga(50, 100);

//      cetak member method dari sub-class BujurSangkar
        cetakLuasBentuk(bs);
        cetakLuasBentuk(bs1);

//      cetak member method dari sub-class SegiTiga
        cetakLuasBentuk(st);
        cetakLuasBentuk(st1);
    }

}
```

Percobaan Polymorphism statis

- Polimorfisme_2.java

```
class Polimorfisme_2 {

 static double maxNumber(double a, double b) {
                if (a < b) {
                        return a;
        } else {
                         return b;
                }
   }

   // Method sama, namun parameter berbeda
   // Tipe data int
   static int maxNumber(int a, int b) {
                if (a < b) {
                  return a;
             } else {
      return ;
   }   }

   public static void main(String[] args) {
             System.out.println(maxNumber(5.5, 7.5));
                  System.out.println(maxNumber(10, 20));
   }
}
```