Modul 7
Polymorphism

A) Pokok Bahasan

1. Konsep Polymorphism
2. Virtual Method
3. Operator instanceof
4. Casting Object

B) Tujuan

1. Mahasiswa Mampu Memahami Konsep Polymorphism Pada PBO

C) Dasar Teori

1. Polymorphism

   Polimorphism adalah suatu konsep dimana suatu objek memiliki kemampuan untuk memiliki banyak bentuk

   Dibawah ini adalah contoh Polymorphism. Walaupun objek b1 bertipe Binatang, objek b1 mampu diisi dengan objek a1 yang bertipe Ayam, juga dapat diisi dengan objek u1 yang bertipe Ular

```
Contoh :
class Binatang {}
class Ayam extends Binatang {}
class Ular extends Binatang {}

Ayam a1 = new Ayam();
Ular u1 = new Ular();
Binatang b1;
b1 = a1;
b1 = u1;
```

2. Virtual Method

   Virtual Method adalah method yang telah di-override oleh subclass. Ketika sebuah method dari objek yang berisi objek dengan tipe subclassnya, maka yang terpanggil adalah Virtual Method. Kondisi ini dinamakan dengan Virtual Method Invocation

```
Contoh :
class Binatang {
   public void jalan() {
      System.out.println("Binatang Berjalan");
   }
```

```
    public void teriak() {
        System.out.println("Binatang Berteriak");
    }
}
class Ayam {
    // Virtual Method
    public void teriak() {
        System.out.println("KUKURUYUUK");
    }
}
Binatang b1 = new Binatang();
b1.jalan();     // Binatang Berjalan
b1.teriak();    // Binatang Berteriak
Ayam a1 = new Ayam();
b1 = a1;
b1.jalan();     // Binatang Berjalan
/* Pemanggilan Virtual Method (Virtual Method Invocation) */
b1.teriak();    // KUKURUYUUK
```

3. Operator `instanceof`

   Operator instancof digunakan untuk mengecek apakah objek tersebut merupakan instance dari class tertentu atau tidak

```
Syntax :
[objek] instanceof [class]

Contoh :
class Binatang {}
class Ayam extends Binatang{}
class Ular extends Binatang{}

Binatang b1 = new Ayam();

System.out.println( b1 instanceof Binatang );      //true
System.out.println( b1 instanceof Ular );          //false
if (b1 instanceof Ayam) { //true
    System.out.println("Variabel ini bertipe Ayam");
}
```

4. Casting Object

   Suatu objek dapat diisi dengan objek bertipe subclassnya. Namun suatu objek tidak bisa diisi dengan objek bertipe superclassnya. Objek tersebut harus dikonversi terlebih dahulu tipenya menjadi subclassnya. Proses konversi ini disebut dengan Casting Object

```
Syntax :
([subclass]) [objek]

Contoh :
class Binatang {}
class Ayam extends Binatang{}
```

```
        Binatang b1 = new Binatang();
        // Proses Casting Object
        Ayam a1 = (Ayam) b1;
```

D) Praktik

file : Contoh1.java

```
public class Contoh1 {

    // Method dengan Number sebagai Parameternya
    public static void tulisAngka(Number n) {
        tulis(n);
    }

    // Method dengan Object sebagai Parameternya
    public static void tulis(Object text) {
        // Virtual Method Invocation
        System.out.println(text.toString());
    }

    public static void main(String []args) {
        int a = 1;
        double b = 2;
        long c =3;
        String d = "Hai";
        boolean e = true;

        tulisAngka(a);
        tulisAngka(b);
        tulisAngka(c);

        tulis(a);
        tulis(b);
        tulis(c);
        tulis(d);
        tulis(e);
    }
}
```

file : Contoh2.java

```
public class Contoh2 {

    private static class Burung {
        protected double ketinggian = 0;

        public void terbang() {
            System.out.println("Burung terbang");
            this.ketinggian += 1;
        }
```

```
        public void cekKetinggian() {
            System.out.println("Burung ini terbang "+this.ketinggian+" meter
diatas tanah");
        }
    }

    private static class Ayam extends Burung {
        // Virtual Method
        @Override
        public void terbang() {
            System.out.println("Ayam terbang sebentar lalu jatuh lagi");
            this.ketinggian = 0;
        }

        public void berkokok() {
            System.out.println("Kukuruyuuk");
        }
    }

    private static class Merpati extends Burung {
        // Virtual Method
        @Override
        public void terbang() {
            System.out.println("Merpati terbang dengan anggun");
            this.ketinggian += 0.5;
        }
    }

    // method dengan Polymorpism Parameter
    private static void berkokokKalauBisa(Burung burung) {
        // instanceof untuk mengecek bahwa Burung tersebut merupakan Ayam
        if (burung instanceof Ayam) {
            // burung.berkokok();    // Belum bisa Berkokok karna masih bertipe
burung
            // Harus di Casting terlebih dahulu menjadi Ayam
            Ayam ayam = (Ayam)burung;
            ayam.berkokok();
        } else {
            System.out.println("Jenis burung ini tidak bisa berkokok");
        }
    }

    public static void main(String []args) {
        Burung b1;

        b1 = new Burung();
        // Virtual Method Invocation
        b1.terbang();
        berkokokKalauBisa(b1);

        b1 = new Ayam();
        b1.terbang();
```

```
        berkokokKalauBisa(b1);

        b1 = new Merpati();
        b1.terbang();
        berkokokKalauBisa(b1);
    }
}
```