Modul 6
Inheritance

A) Pokok Bahasan

1. Konsep Inheritance
2. Access Modifier protected
3. Overriding
4. Keyword `super`

B) Tujuan

1. Mahasiswa Mampu Memahami Konsep Inheritance Pada PBO

C) Dasar Teori

1. Inheritance

   Inheritance adalah suatu konsep mewariskan attribute dan method yang dimiliki oleh suatu class kepada class lainnya

   Agar suatu class dapat mewarisi attribute dan method dari class lain, class tersebut harus menggunakan keyword extends

   Class yang ingin mewarisi biasanya disebut dengan subclass atau child class, sedangkan class yang mewarisi attribute dan method nya kepada subclass dinamakan superclass atau parent class

   Suatu class hanya boleh memiliki sebuah superclass

   Subclass tidak akan mewarisi attribute dan method yang memiliki access modifier private

   Subclass hanya mewarisi attribute dan method yang memiliki access modifier public

```
Syntax :
class [nama_class] extends [super_class] {}

Contoh :
class Binatang {
   private int jumlahKaki;

   public void berjalan() {
      System.out.println("Binatang Berjalan");
   }
}
```

```
class Ayam extends Binatang {
   /* Attribute jumlah kaki tidak diwariskan */
   /* Secara otomatis method "berjalan" milik Binatan Juga akan
   dimiliki oleh class Ayam */
   public void berkokok() {
      //System.out.println( this.jumlahKaki ); //Error
      System.out.println("Kukuruyuk");
   }
}
Ayam a1 = new Ayam();
a1.berjalan();  //Binatang Berjalan
a1.berkokok();  //Kukuruyuk
```

2. Access modifier protected

   Class, attribute, method yang menggunakan access modifier protected hanya dapat di akses oleh subclass, dan class dengan package yang sama

```
Contoh :
class Kendaraan {
   protected double kecepatan = 10;

   public void jalan() {
      System.out.println("Kendaraan Berjalan");
   }
}
class Mobil extends Kendaraan {
   // Attribute kecepatan diwariskan
   public double ukurKecepatan() {
      this.jalan();
      System.out.println( this.kecepatan );
   }
}
Mobil m1 = new Mobil();
m1.ukurKecepatan();
```

3. Overriding

   Overriding adalah konsep menulis ulang sebuah method yang diwariskan oleh superclass dengan nama dan parameter yang sama namun dengan implementasi yang berbeda

```
Contoh :
class Binatang {
   public class makan() {
      System.out.println("Binatang Makan dengan Lahapnya");
   }
   public void berjalan() {
      System.out.println("Binatang Berjalan dengan Kaki");
   }
}
class Ular extends {
```

```
    // method berjalan milik Binatang di override
    public void berjalan() {
        System.out.println("Ular Berjalan dengan Badannya");
    }
}
```

4. Keyword super

   Pada materi sebelumnya, kita sudah mempelajari keyword `this` yang merujuk kepada objek saat ini

   Keyword `super` merujuk kepada attribute dan method yang telah diwariskan oleh superclass

```
Syntax :
/* merujuk ke attribute yang diwariskan superclass */
super.[attribute]
/* merujuk ke method yang diwariskan superclass */
super.[method]()
/* merujuk ke constructor milik superclass */
super()

Contoh :
class Kendaraan {
    public void jalan() {
        System.out.print("Kendaraan Berjalan");
    }
}
class Goku extends Kendaraan {
    public double kecepatan = 9000;

    public void jalan() {
        // Memanggil method jalan yang diwariskan
        super.jalan();
        System.out.println(" Dengan Kecepatan diatas "+this.kecepatan);
    }
}
```

5. Overriding Constructor
   Dalam bahasa Java, constructor milik superclass tidak diwariskan kepada subclass. Jadi, apabila pada superclass tidak memiliki constructor default, maka constructor milik superclass harus dipanggil pada constructor subclass

   Pemanggilan constructor milik superclass juga harus dilakukan terlebih dahulu sebelum statement lainnya

```
Contoh :
class PersegiPanjang {
    protected double panjang;
    protected double lebar;
```

```
        public PersegiPanjang(double panjang, double lebar) {
            this.panjang = panjang;
            this.lebar = lebar;
        }
    }
    class Balok extends PersegiPanjang {
        protected double tinggi;

        public Balok(double panjang, double lebar, double tinggi) {
            // memanggil constructor milik class PersegiPanjang
            super(panjang, lebar);
            this.tinggi = tinggi;
        }
    }
```

D) Praktik

Buatlah direktori dengan susunan sebagai berikut

```
/main
    Main.java
    KipasAngin.java
/parent
    AlatElektronik.java
```

file : parent/AlatElektronik.java

```
package parent;

public class AlatElektronik {
    protected String merk;
    private boolean sedangMenyala;

    public AlatElektronik(String merk) {
        this.merk = merk;
        this.sedangMenyala = false;
    }

    public void nyalakan() {
        System.out.println(this.merk+ " Dinyalakan");
        this.sedangMenyala = true;
    }

    public void matikan() {
        System.out.println(this.merk+ " Dimatikan");
        this.sedangMenyala = false;
    }

    public boolean isMenyala() {
        return sedangMenyala;
    }
}
```

file : main/KipasAngin.java

```
package main;
import parent.AlatElektronik;

public class KipasAngin extends AlatElektronik {

    // Overriding Constructor milik class AlatElektronik
    public KipasAngin(String merk) {
        super(merk);
    }

    // Overriding method matikan
    @Override
    public void matikan() {
        // System.out.println(this.sedangMenyala);    // Error : Private
        if(!this.isMenyala()) {
            // Merk dapat diakses di subclass
            System.out.println(this.merk+" Sudah Mati");
        } else {
            super.matikan();
        }
    }

    public void cuci() {
        System.out.println(super.merk+" Sudah dicuci dan sudah bersih");
    }
}
```

file : main/Main.java

```
package main;
import parent.AlatElektronik;

public class Main {
    public static void main(String []args) {
        AlatElektronik elek = new AlatElektronik("Mitsubishi");

        // System.out.println( elek.merk );       // Error : Protected
        elek.nyalakan();
        elek.matikan();
        System.out.println(elek.isMenyala());

        KipasAngin kipas = new KipasAngin("Miyako");

        // System.out.println( kipas.merk );  // Error : Protected
        kipas.nyalakan();
        kipas.matikan();
        System.out.println(elek.isMenyala());
        kipas.cuci();
    }
}
```