LAPORAN PRATIKUM PEMROGRAMAN BERORIENTASI OBJEK

MODUL 5

Nama: Mahran Radifan Zhafir

NIM: 105219001

LABORATORIUM SISTEM OPERASI PROGRAM STUDI ILMU KOMPUTER
FAKULTAS SAINS DAN ILMU KOMPUTER

UNIVERSITAS PERTAMINA 2021

A. Pendahuluan

Abstraction adalah salah satu metode inheritance dimana superclass yang abstrak tidak dapat di inisiasi sebagai objek walaupun sudah mempunyai beberapa method. Gunanya adalah apabila kita memerlukan sebuah objek, tapi superclass belum bisa dikatakan sebagai objek karena belum memenuhi syarat sebagai sebuah objek. Abstraction akan berguna pada saat penggunaan basis data kedepannya

Interface adalah salah satu metode inheritance dimana classs yang menjadi sebuah child dari superclass harus menyediakan method yang sudah didefinisikan di superclass. Superclass sebuah interface tidak mempunyai method yang dapat dijalankan, tapi terdapat nama method yang harus dipunyai oleh child class dari superclass yang di implementasikan

Rumusan masalah pada praktikum kali ini bagaimana abstract menurunkan method nya tanpa dapat menginisiasi dirinya sendiri dan bagaimana cara penerapan interface pada pemrograman berorientasi objek

B. Pembahasan

```
1   public abstract class shape {
2       String warna;
3
4       public shape(String warna) {
5           this.warna = warna;
6       }
7
8       String getWarna() {
9           return this.warna;
10      }
11  }
12
```

Gambar 1,1 Abstract dari sebuah bangun datar

Sebuah bangun datar pasti memiliki warna. Tapi apabila warna saja, itu belum dapat dikatakan sebagai bangun datar. Maka jika kita menginisiasi sebuah objek bernama bangun datar tapi hanya memiliki warna saja, itu salah. Maka kita memerlukan sebuah child untuk melengkapi abstrak tersebut.

```java
public class persegi extends shape {
    private float sisi;

    public persegi(String warna,float sisi){
        super(warna);
        this.sisi = sisi;
    }

    float getLuas(){
        return sisi * sisi;
    }

    @Override
    public String toString(){
        return "warna persegi adalah " + super.getWarna() + " luasnya adalah " + getLuas();
    }
}
```

Gambar 1.2 Child dari superclass bangun datar

Dengan menambahkan method baru, sebuah bangun datar dapat dikatakan sebuah bangun datar karena sudah memiliki sisi dan luasnya. Dengan ini kita dapat membangun sebuah objek bangun datar karena sudah melengkapi kriteria tertentu pada sebuah objek bangun datar.

```java
interface kendaraan {
    void gantiGigi(int gigi);
    void maju(int kecepatan);
    void berhenti(int rem);
}
```

Gambar 1.3 Method pada kendaraan yang selalu ada pada kendaraan

Pada kendaraan, method diatas pasti dimiliki sehingga untuk menciptakan kendaraan kedepannya, kita selalu memerlukan method tersebut. Interface hadir untuk memastikan child dari superclass yang berjenis interface wajib memiliki method diatas

```java
public class sepeda implements kendaraan {
    int kecepatan;
    int gigi;
    int rem;

    public void gantiGigi(int gigi){
        this.gigi = gigi;
    }
    public void maju(int kecepatan){
        this.kecepatan = this.kecepatan + kecepatan;
    }

    public void berhenti(int rem){
        this.rem = rem;
        this.kecepatan = this.kecepatan - rem;
    }

    public void keadaanSekarang(){
        System.out.println("sekarang gigi " + gigi + " dengan kecepatan " + this.kecepatan + "
    }
}
```

Gambar 1.4 Child dari interface yang menggunakan method interface

Pada child, kita harus menggunakan semua method yang sudah disediakan sehingga tidak ada kendaraan yang melupakan fungsi yang selalu ada pada kendaraan.

C. Kesimpulan

Abstraction digunakan sebagai inisiasi objek Ketika kita belum mengetahui keseluruhan data yang akan dimasukkan. Abstraction sangat berguna apabila method di dalamnya yang selalu ada tapi bila hanya ada dia, itu belum dapat dikatakan sebagai objek,

Interface berguna sama seperti abstraction, hanya saja interface hanya menyediakan nama method yang harus ada pada setiap childnya sehingga cara kerja abstraction dan interface agak sedikit berbeda walaupun tujuannya sama,