PRAKTIKUM 9

ABSTRACT CLASS DAN INTERFACE

A. TUJUAN PEMBELAJARAN

1. Memahami konsep Abstract Class dan Interface dalam OOP serta perbedaannya.
2. Mengimplementasikan Abstract Class dan Interface dalam program Java atau C# untuk menyelesaikan masalah berbasis objek.
3. Menerapkan teknik casting (upcasting dan downcasting) secara aman dan efisien dalam konteks pewarisan dan polimorfisme

B. DASAR TEORI

Abstract Class

- **Definisi**: Abstract class adalah kelas yang tidak dapat diinstansiasi secara langsung dan dapat memiliki metode abstrak (tanpa implementasi) maupun metode konkret.
- **Tujuan**: Menyediakan kerangka dasar bagi subclass untuk mengimplementasikan perilaku spesifik.
- **Ciri-ciri**:
  - Dapat memiliki konstruktor, field, dan metode dengan implementasi.
  - Digunakan saat ada relasi "is-a" dan sebagian perilaku sudah diketahui.
- **Contoh (Java)**:

```
abstract class Animal {
    abstract void makeSound();
    void breathe() {
        System.out.println("Breathing ... ");
    }
}
```

Interface

- **Definisi**: Interface adalah kontrak yang hanya berisi deklarasi metode (tanpa implementasi) dan digunakan untuk mendefinisikan perilaku yang harus diimplementasikan oleh kelas.
- **Tujuan**: Mendukung multiple inheritance dan pemisahan antar modul.
- **Ciri-ciri**:
  - Semua metode bersifat public dan abstract secara default.
  - Tidak memiliki konstruktor atau field non-final.
- **Contoh (Java)**

```java
interface Flyable {
    void fly();
}

class Bird implements Flyable {
    public void fly() {
        System.out.println("Bird is flying");
    }
}
```

Casting (Upcasting dan Downcasting)

- **Upcasting**: Mengubah objek subclass menjadi tipe superclass. Aman dan umum digunakan dalam polimorfisme

```java
Animal a = new Dog(); // Upcasting
```

- **Downcasting**: Mengubah objek superclass menjadi tipe subclass. Perlu pengecekan tipe agar tidak terjadi ClassCastException.

```java
Dog d = (Dog) a; // Downcasting
```

C. PERCOBAAN

1. Abstract Class

```
public abstract class Binatang {
        public void bernafas(){
                System.out.println(“semua binatang bernafas”);
        }
        public void makan(){
                System.out.println(“semua binatang makan”);
        }
        public void berkembangBiak ();
}

public class Burung extends Binatang{

        public void makan(){
```

```
                super.makan();
                System.out.println("burung makan biji-bijian");
        }
        public void berkembangBiak (){
                System.out.println("burung berkembang biak dengan cara bertelur");
        }
}

public class Mamalia extends Binatang{
        public void berkembangBiak (){
            System.out.println("mamalia berkembang biak dengan cara melahirkan");
        }
}
```

Class Test:

```
public class TestAnimal {
      public static void main(String args[]){
          Animal loveBird = new Burung();
          Animal cat = new Mamalia();
          Mamalia dolphin = new Mamalia();
          lovebird.bernafas();
          lovebird.makan();
          lovebird.berkembangBiak();
          cat.bernafas();
          cat.makan();
          cat.berkembangBiak();
          dolphin.berkembangBiak();
      }
}
```

Output:

semua binatang bernafas

semua binatang makan

burung makan biji-bijian

burung berkembang biak dengan cara bertelur

semua binatang bernafas

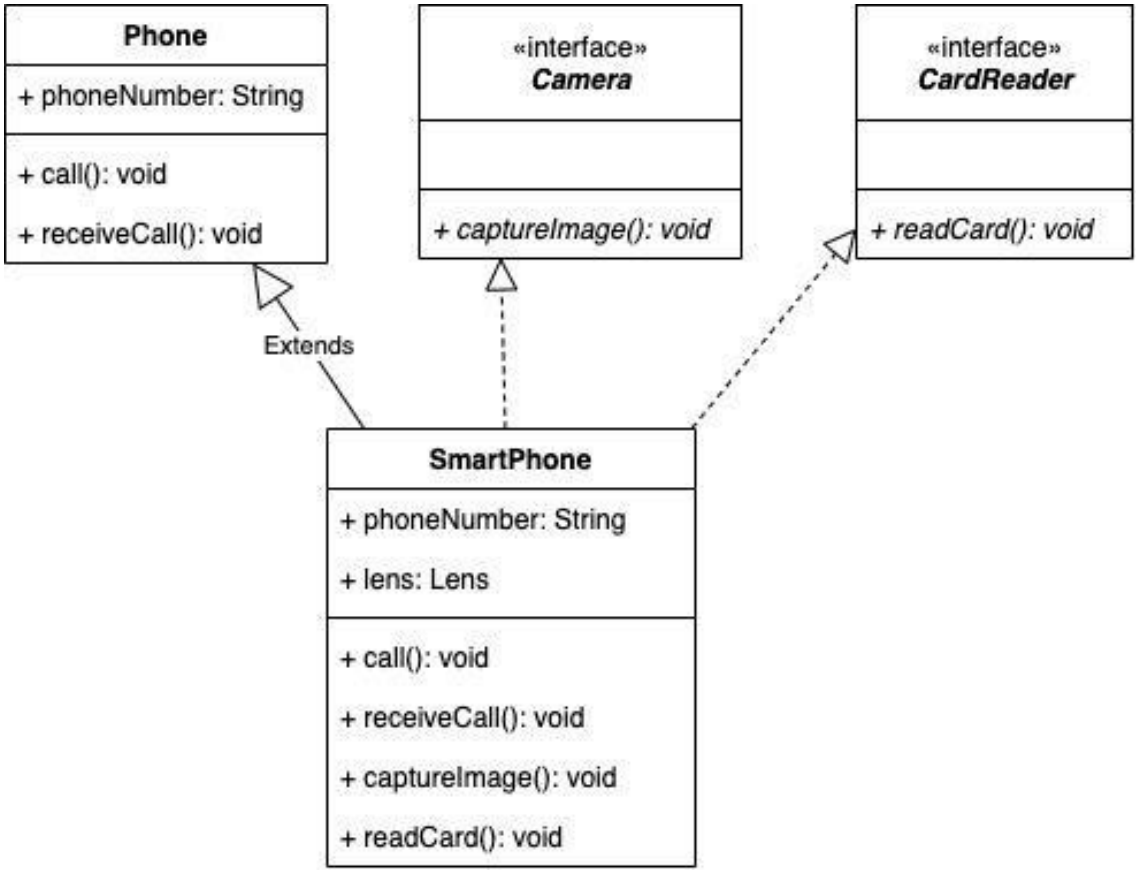
semua binatang makan

mamalia berkembang biak dengan cara melahirkan

mamalia berkembang biak dengan cara melahirkan

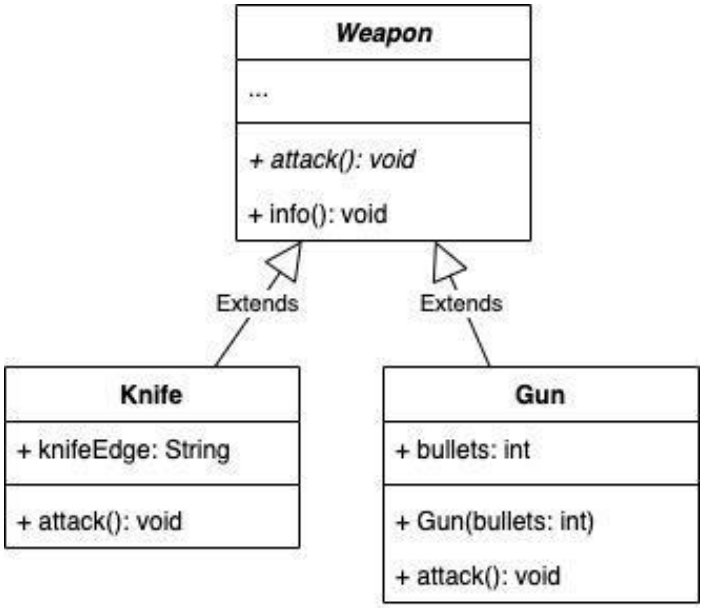
2. Interface Class

Implementasikan class diagram berikut dalam kode program.



D. LATIHAN

Latihan 1. Implementasikan dalam kode program. Perbaikilah kesalahan pada class TestAbstract dan jelaskan kesalahan yang terjadi tersebut!

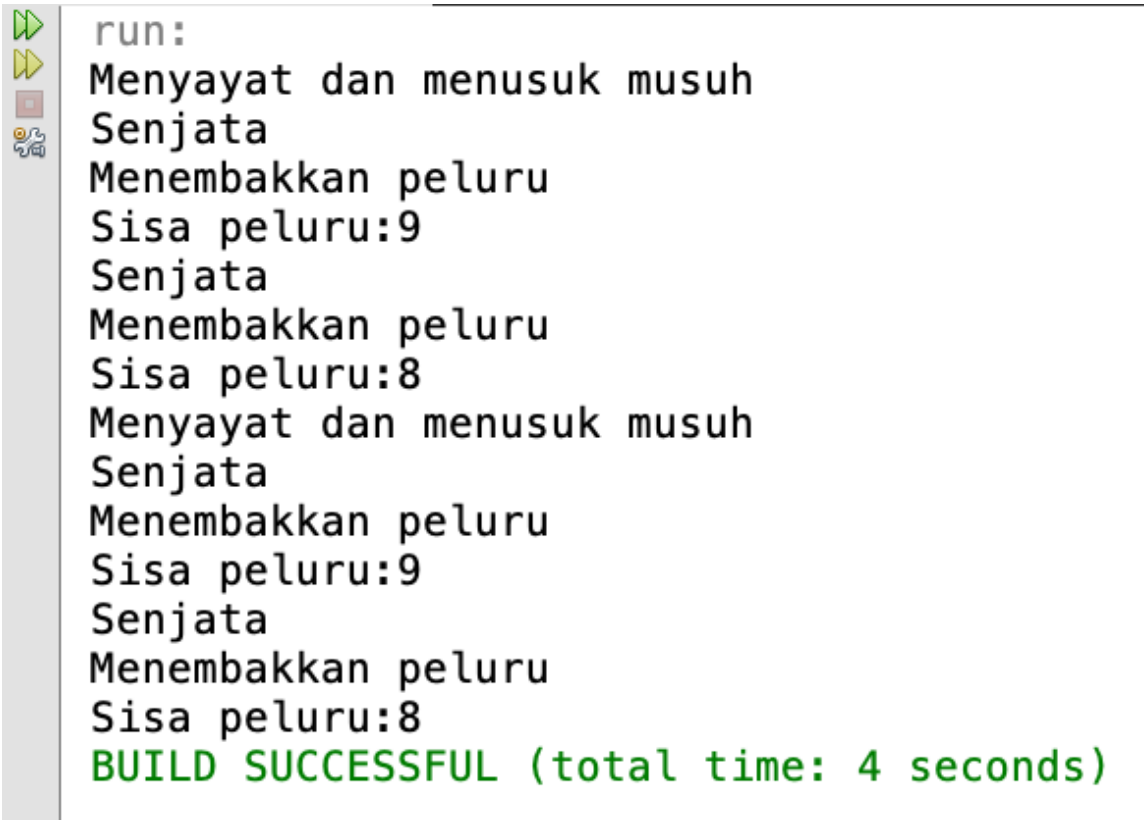


Test Class:

```
public class TestAbstract {
        public static void main(String args[]){
            Weapon weapon = new Knife();

            Weapon knife = new Knife();
            Weapon gun = new Gun(10);

            knife.attack();
            gun.attack();
            gun.attack();

            Knife knife2 = new Knife();
            Weapon gun2 = new Gun(10);

            knife2.attack();
```

```
            gun2.attack();
            gun2.attack();


        }
}
```

Output:



Perbaiki kesalahan yang terjadi pada test class dan jelaskan kesalahan tersebut.

Latihan 2. Perbaikilah kesalahan pada program berikut dan jelaskan kesalahan yang terjadi tersebut!

```
public interface Scanner{
    public void scaneImage(){
        System.out.println(“Scanning image...”);
    }
}
```

```
public interface Copier{
        public void copyImage();
}

public class Printer implements Copier, Scanner{
        public void scaneImage(){
                System.out.println("Scanning image...");
        }
        public void copyImage(){
                System.out.println("Copy image...");
        }
        public void printImage(){
                System.out.println("Print image...");
        }
}
```

E. TUGAS

1. Dalam sebuah program terdapat tiga buah object yaitu : ballpoint, gun, usbFlash. usbFlash memiliki fungsi storageMedia. Gun memiliki fungsi laserPointer. Ballpoint memiliki fungsi draw, laserPointer, dan storageMedia. Buatlah class diagram untuk object-object tersebut kemudian implementasikan menjadi kode program.
2. Buat program abstract class "BangunDatar" dengan subclass "Persegi" dan "Lingkaran".
3. Buat interface "Playable" dan implementasikan pada dua kelas "Musik" dan "Video".
4. Tunjukkan contoh upcasting dan downcasting pada program Anda.

F. LAPORAN RESMI

Kumpulkan hasil latihan dan tugas di atas. Tambahkan analisa dalam laporan resmi.