UAS
PBO

Car Coin adalah game yang bertujuan untuk mengumpulkan score sebanyak-banyaknya dengan menghindari rintangan

Nama Anggota Kelompok

Muhammad Nabil Mu'afi – 2211102441154

Muhamad adji Mukti – 2111102441065

Muhamad Fuad Ilham – 2111102441037

CLASS & OBJECT

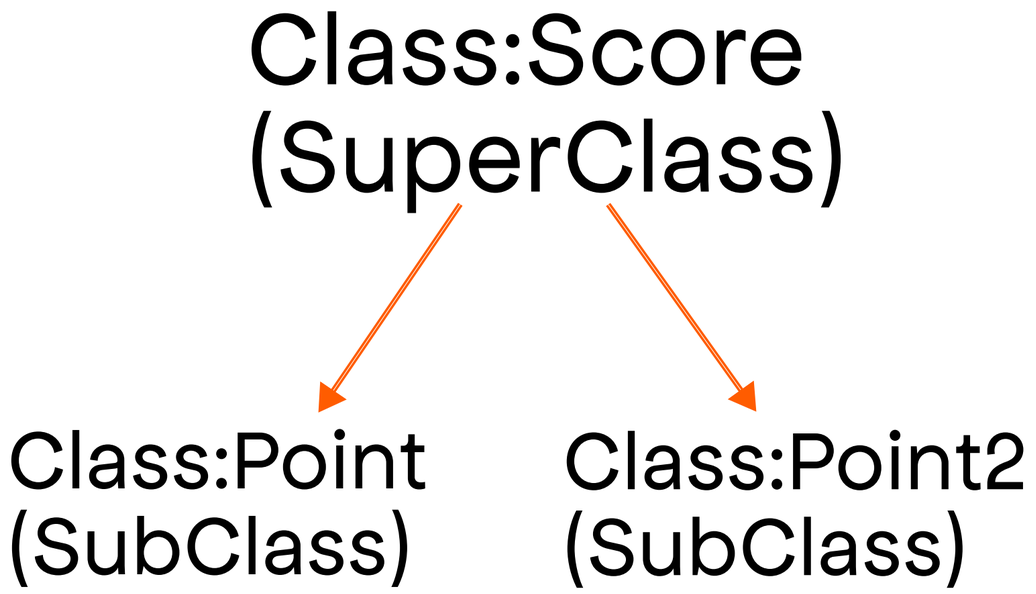
- Car adalah Class
- speed dan score adalah atribut/properti
- method act() adalah behaviour/perilaku objek

```
public class Car extends Actor {
    private int speed = 0;
    private int score = 0;

    public void act() {
        move(5);
        handleControls();
        checkCollision();
        checkEdge();
    }
```

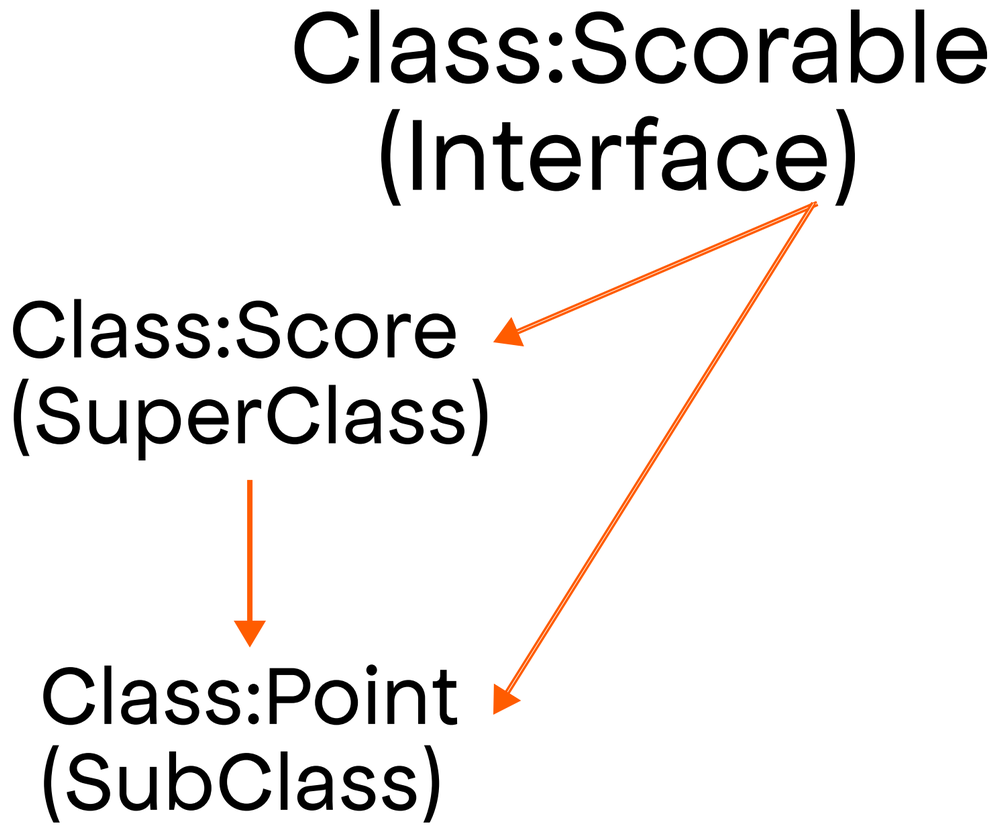
INHERITANCE

- Kelas Score menjadi superclass atau kelas induk.
- Menyediakan fungsionalitas umum yang dapat diwarisi oleh kelas anak.
- Kelas Point dan Point2 menjadi subclass atau kelas anak dari Score.
- Secara otomatis mewarisi sifat dan perilaku dari Score.
- Dapat menambahkan perilaku atau mengubah perilaku yang diwarisi.



POLIMORFISME

konsep polimorfisme diimplementasikan melalui penggunaan antarmuka (Scorable).



Class Scorable (Interface)

Antarmuka Scorable mendefinisikan satu metode yaitu giveScore. Setiap kelas yang mengimplementasikan antarmuka ini akan memberikan implementasi yang spesifik untuk metode giveScore.

```
public interface Scorable {
    void giveScore();
}
```

Class Score

Kelas Score mengimplementasikan metode giveScore sesuai dengan kontrak antarmuka. Dalam implementasi ini, skor bertambah, teks skor ditampilkan, dan objek dihapus.

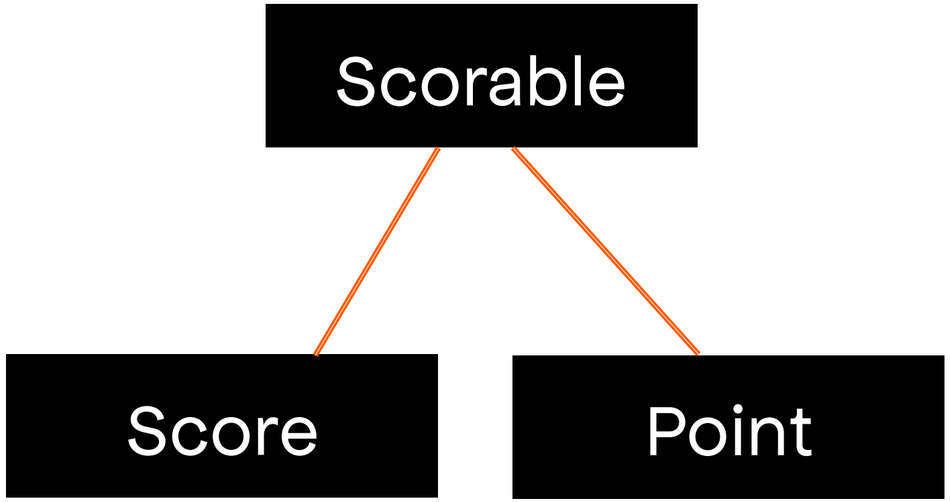
```
public void giveScore() {
    Greenfoot.playSound("sound.wav");
    increaseScoreFromScorable(5);
    getWorld().showText("Score: " + getScore(), 625, 20);
    getWorld().removeObject(this);
    System.out.println("Default Score behavior");
}
```

Class Point

```
    public void giveScore() {
    Greenfoot.playSound("sound.wav");
    Score.increaseScoreFromScorable(scoreValue);
    getWorld().showText("Score: " + Score.getScore(), 625, 20);
    getWorld().removeObject(this);
    System.out.println("Point-specific behavior");
}
```

Kelas Point, sebagai subclass dari Score, juga mengimplementasikan metode giveScore sesuai kontrak antarmuka. Namun, perilaku spesifik dari Point berbeda dari perilaku default Score. Dalam contoh ini, skor ditambahkan dengan nilai yang berbeda dan objek dihapus berdasarkan kondisi yang berbeda.

Class(Interface) Scorable memungkinkan berbagai kelas untuk memberikan perilaku yang sesuai dengan kebutuhan mereka sendiri, menunjukkan konsep polimorfisme di mana objek dari tipe antarmuka dapat memiliki perilaku yang bervariasi sesuai dengan implementasinya.



ENKAPSULASI

- Variabel score diakses secara privat. Hanya metode di dalam kelas Score yang dapat mengakses dan memodifikasinya.
- Metode-metode yang terkait dengan manipulasi skor (seperti increaseScore, getScore, dan increaseScoreFromScorable) diakses melalui antarmuka dan tidak secara langsung dari luar kelas.

```
public class Score extends Actor implements Scorable {
    private static int score = 0;

    public Score() {
        // Konstruktor
    }

    public void act() {
        // Implementasi act (jika diperlukan)
    }

    private static void increaseScore(int value) {
        score += value;
    }

    public static int getScore() {
        return score;
    }

    public static void increaseScoreFromScorable(int value) {
        increaseScore(value);
    }
```

Overriding

Metode giveScore() pada kelas Point meng-override metode yang sama pada kelas Score. Ini terjadi karena kelas Point adalah subclass dari Score, dan kita ingin memberikan perilaku yang berbeda saat objek Point memberikan skor.

Class: Score(SuperClass)

```
public void giveScore() {
    Greenfoot.playSound("sound.wav");
    increaseScoreFromScorable(5);
    getWorld().showText("Score: " + getScore(), 625, 20);
    getWorld().removeObject(this);
    System.out.println("Default Score behavior");
}
```

Class: Point(SubClass)

```
public void giveScore() {
Greenfoot.playSound("sound.wav");
Score.increaseScoreFromScorable(scoreValue);
getWorld().showText("Score: " + Score.getScore(), 625, 20);
getWorld().removeObject(this);
System.out.println("Point-specific behavior");
```

Overloading

metode increaseScoreFromScorable yang di-overload. Satu tanpa parameter dan satu dengan parameter value. Metode increaseScoreFromScorableWithSound dengan nama yang berbeda dan memiliki parameter.

Class: Score(SuperClass)

```
}
public static void increaseScoreFromScorable() {
    increaseScore(5);
}
public static void increaseScoreFromScorable(int value) {
    increaseScore(value);
}
public static void increaseScoreFromScorableWithSound(int value) {
    Greenfoot.playSound("sound.wav");
    increaseScore(value);
}
```

Interaksi Antar Objek

Kelas Car berinteraksi dengan kelas Point. Ketika Object Car bersentuhan dengan Object Point. Maka akan terjadi penambahan score dan penghilangan Object Point.

Class Car

```
private void checkCollision() {
if (isTouching(Point.class)) {
    // Hapus objek Point
    removeTouching(Point.class);

    // Lakukan tindakan tambahan jika diperlukan
    Greenfoot.playSound("sound.wav");
    score += 10; // Misalnya, tambahkan skor setiap kali menabrak Point
    getWorld().showText("Score: " + score, 625, 20);
}

else if (isTouching(Point2.class)) {
    // Hapus objek Point2
    removeTouching(Point2.class);

    // Lakukan tindakan tambahan jika diperlukan
    Greenfoot.playSound("sound.wav");
    score += 30; // Misalnya, tambahkan skor setiap kali menabrak Point2
    getWorld().showText("Score: " + score, 625, 20);
}
```

Interaksi Antar Objek

kelas NPC berinteraksi dengan subclass(amb, green, blue). Jika Object NPC saling bertemu/bertakbrakan, maka salah satu dari NPC akan hilang. Dan jika kelas NPC menyentuh kelas Score. Makan object Point akan menghilang. Dan jika kelas Car menabrak kelas NPC, maka akan Game Over.

Class Car

```
public void crash() {
    if (isTouching(Car.class)) {
        handleCarCrash();
    } else if (isTouching(NPC.class)) {
        handleNPCCrash();
    }
    else if (isTouching(Score.class)) {
        handleScoreCrash();
    }

    else if (isAtEdge()) {
        getWorld().removeObject(this);
    }
}
```

Interaksi Antar Objek

Metode dalam kelas Car

```
private void handleCarCrash() {
    getWorld().showText("You Crashed!!", 550, 300);
    getWorld().showText("Press Enter to try again", 550, 350);
    Greenfoot.playSound("crash.wav");
    removeTouching(Car.class);
}

private void handleNPCCrash() {
    // Logika penghapusan objek NPC saat menabrak objek NPC lainnya
    getWorld().removeObject(this);
}

private void handleScoreCrash() {
    // Logika penghapusan objek Score saat menabrak objek NPC
    getWorld().removeObject(getOneIntersectingObject(Score.class));
}
```