Kelas Abstrak & *Interface*

Teosofi Hidayah Agung
Hafidz Mulia

Departemen Matematika
Institut Teknologi Sepuluh Nopember

26 Mei 2025

Masalah

Pewarisan sifat atau implementasi dari kelas ke objek lain itu sangat luas terlebih dalam pendefinisian method didalam kelas tersebut. Oleh karena itu, Agar tidak selalu harus mendefinisikan method pada setiap kelas yang dibuat, maka kita dapat menggunakan pendekatan umum yang bersifat **abstrak** dengan tujuan bisa mencakup case yang lebih luas nantinya.

Daftar isi

Abstract

Definisi

Kelas abstrak *adalah kelas yang tidak dapat diinstansiasi secara langsung dan biasanya berfungsi sebagai kerangka dasar (blueprint) bagi class-class turunan. Kelas abstrak bisa memiliki method abstrak (tanpa implementasi), method konkret (dengan implementasi), atribut, dan konstruktor.*

Dikatakan/didefinisikan kelas abstrak ketika kelas tersebut minimal memiliki satu method abstrak.

Abstract

Kode: Syntax Kelas Abstrak

```
1  abstract class NameClass {
2    // Atribut
3    String ID;
4    ...
5
6    // Method konkret
7    void print() {
8        System.out.println(this.ID);
9    }
10   ...
11
12   // Method abstrak
13   abstract void absMethod();
14 }
```

Contoh

Penggunaan kelas abstrak pada kelas `BangunDatar` yang memiliki method abstrak `hitungLuas()` dan `hitungKeliling()`.

```
1   abstract class Bentuk {
2     String nama;
3     Bentuk(String nama) { this.nama = nama; }
4
5     void printNama() {
6         System.out.println("Bentuk: " + this.nama);
7     }
8
9     abstract double luas();
10    abstract double keliling();
11  }
```

Abstract

```java
class Persegi extends Bentuk {
  double sisi;
  Persegi(double s) { this.sisi = s; }

  double luas() {// Jika tidak ada akan error
      return this.sisi * this.sisi;
  }
  double keliling() {// Jika tidak ada akan error
      return 4 * this.sisi;
  }

  void printNama() {// Override opsional
        System.out.println("Persegi: " + this.nama);
    }
}
```

Abstract

Penting!

Kelas abstrak tidak dapat diinstansiasi secara langsung.

```
1   Bentuk b = new Bentuk(); // ERROR!
```

Daftar isi

Interface

Definisi

Interface *adalah template yang hanya mendefinisikan method tanpa implementasi, atau method dengan implementasi default. Semua method dalam interface secara default* `public abstract`.

Simpelnya semua yang didefinisikan dalam interface adalah method abstrak, dan tidak ada atribut yang didefinisikan di dalamnya.

Interface

Kode: Syntax Interface

```
interface NameInterface {
  void method1();
  String method2(char c);
  int method3(int a, int b);
  ...
}
```

Contoh

Penggunaan interface pada interface `Kendaraan`.

Kode: Contoh Interface

```
1 interface Kendaraan {
2   void jalan();
3   void berhenti();
4   void belokKiri();
5   void belokKanan();
6   int getKecepatan();
7 }
```

Kode: Contoh Kelas Implementasi Interface

```
1 class Mobil implements Kendaraan {
2   int kecepatan;
3
4   public void jalan() { System.out.println("Mobil berjalan");}
5   public void berhenti() { System.out.println("Mobil berhenti");}
6   public void belokKiri() { System.out.println("Mobil belok kiri");}
7   public void belokKanan() { System.out.println("Mobil belok kanan");}
8   public int getKecepatan() { return this.kecepatan;}
9 }
```

Penting!

Kita tahu sebuah kelas hanya bisa mengimplementasikan satu kelas (superclass), namun sebuah kelas bisa mengimplementasikan banyak interface.

Kode: Syntax untuk Implementasi Banyak Interface

```
1  class A implements Interface1, Interface2 {
2    ...
3  }
```

Daftar isi

Latihan 1

Buatlah kode program yang mendefinisikan kelas dan interface berikut: