Class & Object

Teosofi Hidayah Agung
Hafidz Mulia

Departemen Matematika
Institut Teknologi Sepuluh Nopember

28 April 2025

Masalah

Dengan semua ilmu yang telah kalian pelajari dari Alpro 1 hingga sekarang, coba buatlah gambaran kasar gimana kalian membuat sebuah game atau mungkin aplikasi sederhana (kalkulator, database, dll). Kemudian coba lihat berapa banyak baris yang kalian butuhkan untuk membuat kode tersebut?

Jika baris kodenya cuma sedikit maka kalian layak dapat *title* **GOAT**, namun jika tidak itu wajar saja.

Contoh

Misal aku ingin membuat sebuah program yang menerima nama mahasiswa, umur, jurusan, dan ipk-nya.

```
1  import java.util.Scanner;
2  public class DatabaseMahasiswa {
3    public static void main(String[] args) {
4        Scanner scanner = new Scanner(System.in);
5
6        String[] nama = new String[100];
7        int[] umur = new int[100];
8        String[] jurusan = new String[100];
9        double[] ipk = new double[100];
10
11       System.out.println("Masukkan jumlah mahasiswa (maks 100): ");
12       int jumlah = scanner.nextInt();
13       scanner.nextLine();
```

```
        for (int i = 0; i < jumlah; i++) {
            System.out.println("Data Mahasiswa ke-" + (i + 1));
            System.out.print("Nama: ");
            nama[i] = scanner.nextLine();
            System.out.print("Umur: ");
            umur[i] = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Jurusan: ");
            jurusan[i] = scanner.nextLine();
            System.out.print("IPK: ");
            ipk[i] = scanner.nextDouble();
            scanner.nextLine();
        }
        System.out.println("\nDaftar Mahasiswa:");
        for (int i = 0; i < jumlah; i++) {
            System.out.println("Nama: " + nama[i] + ", Umur: " + umur[i] +
                ", Jurusan: " + jurusan[i] + ", IPK: " + ipk[i]);
    }
  }
}
```

Kesimpulan

Dapat dilihat bahwa kode sebelumnya sangatlah tidak efisien dan datanya pun tidak bisa fleksibel (maksimal 100 mahasiswa). Sehingga diperlukan sebuah paradigma baru yang lebih efisien dan fleksibel dalam penyelesaian masalah ini.

Daftar isi

Definisi

Pemrograman Prosedural *dapat didefinisikan sebagai model pemrograman yang berasal dari pemrograman terstruktur, berdasarkan konsep pemanggilan prosedur. Prosedur (yang juga bisa disebut sebagai blok perintah atau fungsi) hanya terdiri dari serangkaian langkah komputasi yang harus dilakukan.*

Ciri-ciri Pemrograman Prosedural

- Menggunakan fungsi untuk menyelesaikan masalah.
- Data (biasanya disimpan dalam variabel global atau array.) dan fungsi terpisah.
- Kalau ada perubahan struktur data, harus ubah banyak bagian kode.
- Sangat bergantung pada kelas `Main`.

Prosedural vs. Berorientasi Objek

Prosedural

Kelebihan

- Mudah dipahami dan diimplementasikan.
- Lebih mudah untuk debugging.
- Sederhana untuk program kecil.

Kekurangan

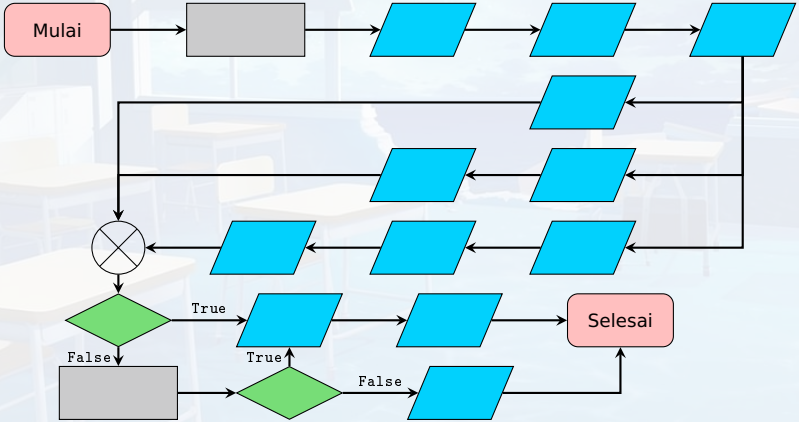
- Sulit dikembangkan kalau program makin besar.
- Mudah error sebab data berserakan.
- Semua data bisa diakses dimana saja.
- Maintenance sulit.

Contoh

Flowchart adalah gambaran dari pemrograman prosedural.

Prosedural vs. Berorientasi Objek

Prosedural



Gambar: Contoh Flowchart sebagai gambaran dari pemrograman prosedural

Definisi

Pemrograman Berorientasi Objek (PBO) *adalah paradigma pemrograman yang menggunakan objek-objek untuk menyelesaikan masalah. Objek adalah entitas yang memiliki data dan perilaku. Pemrograman berorientasi objek mengorganisir kode ke dalam kelas-kelas yang dapat digunakan kembali.*

Ciri-ciri PBO

- Ada class (template/blueprint) dan object (instance dari class).
- Data dan fungsi terkait dibungkus dalam satu class.
- Enkapsulasi, inheritance (pewarisan), dan polymorphism adalah prinsip utama.

Kelebihan

- Lebih mudah untuk dikembangkan dan diperluas (scalable).
- Lebih rapi, data dan perilaku satu objek terbungkus rapi.
- Mudah untuk maintenance.
- Lebih aman, karena ada kontrol akses (private, public, protected).

Kekurangan

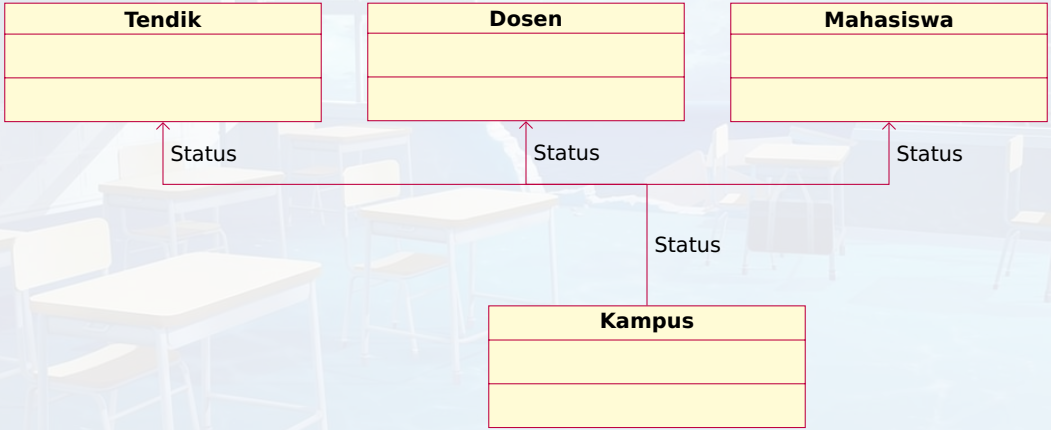
- Lebih kompleks dibandingkan prosedural.
- Memerlukan waktu lebih lama untuk belajar.
- Memerlukan lebih banyak memori.

Contoh

Diagram UML adalah gambaran dari pemrograman berorientasi objek.

Prosedural vs. Berorientasi Objek

Berorientasi Objek



Gambar: Contoh Diagram UML sebagai gambaran dari pemrograman berorientasi objek

Prosedural vs. Berorientasi Objek

“Sekarang sudah waktunya untuk beralih dari Pemrograman Prosedural ke Pemrograman Berorientasi Objek.”

Daftar isi

Class

Definisi

Class *adalah template atau blueprint untuk membuat objek. Class mendefinisikan atribut (data) dan metode (fungsi) yang dimiliki oleh objek. Class adalah struktur data yang dapat digunakan untuk mengelompokkan data dan perilaku yang terkait.*

Kode: Struktur Dasar Class

```
1 class NamaClass {
2   // Atribut (variabel)
3   // Method (fungsi)
4 }
```

Class

Atribut

Variabel yang dimiliki oleh class. Atribut dapat berupa variabel primitif (int, double, char, dll) atau objek dari class lain.

Contoh

Mobil memiliki atribut seperti warna, merk, dan tahun keluaran.

Kode: Contoh Penambahan Atribut pada

```
1  class Mobil {
2    String merk;
3    String warna;
4    int tahun;
5  }
```

Class

Method

Fungsi yang dimiliki oleh class. Method dapat berupa fungsi yang mengubah nilai atribut atau fungsi yang melakukan operasi tertentu.

Contoh

Mobil memiliki method seperti mesin nyala dan mesin mati.

Kode: Contoh Penambahan Method pada Class Mobil

```
1  class Mobil {
2    //...
3    void MesinModeAktifOn() {System.out.println("Mesin " + merk + "
         dinyalakan!");}
4    void MesinModeNonaktifOff() {System.out.println("Mesin " + merk + "
         dimatikan!");}
5  }
```

Definisi

Constructor *adalah method khusus dalam class yang digunakan untuk menginisialisasi objek. Constructor memiliki nama yang sama dengan class dan tidak memiliki tipe pengembalian. Constructor dapat memiliki parameter untuk menginisialisasi atribut objek.*

Kode: Struktur Constructor pada Class

```
1  class NamaClass {
2    Type atribut1, atribut2,...;
3    NamaClass(Type param1, Type param2,...) { // Constructor
4      this.atribut1 = param1;
5      this.atribut2 = param2;
6      //...
7    }
8  }
```

Kode: Constructor pada Class Mobil

```
1  class Mobil {
2    String merk;
3    String warna;
4    int tahun;
5
6    // Constructor
7    Mobil(String merkInput, String warnaInput, int tahunInput) {
8        this.merk = merkInput;
9        this.warna = warnaInput;
10       this.tahun = tahunInput;
11   }
12 }
```

Daftar isi

Object

Definisi

Object *adalah instansiasi dari class. Object memiliki atribut dan metode yang didefinisikan dalam class. Setiap object memiliki nilai yang berbeda untuk atribut yang sama.*

Kode: Contoh Instansiasi Object

```
1  public static void main(String[] args) {
2    NamaClass namaObject1 = new NamaClass();
3    NamaClass namaObject2 = new NamaClass(parameter1, parameter2, ...);
4  }
```

Contoh

Contoh pembuatan objek mobil dan menggunakan perilakunya.

Kode: Contoh Penggunaan Object Mobil

```
1   public static void main(String[] args) {
2       Mobil mobilSaya = new Mobil(); // Membuat object
3       Mobil mobilTeman = new Mobil("Esemka", "RGB", 2030); // Constructor
4
5       mobilSaya.merk = "Toyota"; // Mengisi atribut
6       mobilSaya.warna = "Putih";
7       mobilSaya.tahun = 2020;
8
9       mobilSaya.nyalakanMesin(); // Menggunakan method
10      mobilSaya.matikanMesin();
11  }
```

Daftar isi

Latihan 1

Buatlah class `Mahasiswa` yang memiliki atribut sebagai berikut:

- Nama
- NRP
- Jurusan
- TTL
- Tahun Masuk
- IPS
- IPK
- Alamat
- No HP
- Email
- Semester
- Status

Kemudian buatlah beberapa method berikut:

- Menampilkam biodata diri