LAPORAN PRAKTIKUM
TEKNIK PEMROGRAMAN
MINGGU KE-3

DOSEN PEMBIMBING :

Zulkifli Arsyad, S. kom. M.T

IHSAN FAUZI

241524048

1B – D4

LABORATORIUM TEKNIK INFORMATIKA

FAKULTAS TEKNIK

POLITEKNIK NEGERI BANDUNG

2024

DAFTAR ISI

SOAL 1

A. Amati desain setiap class, Apakah desain class tersebut sudah memenuhi konsep OOP yang benar? Jika tidak, coba anda perbaiki dengan mengacu pada Design Hint di Buku Chapter 4.10. Setiap perubahan yang dibuat harus dibubuhi penjelasan serta argumentasiyang jelas.

Class tersebut tidak memenuhi konsep OOP yang benar, karene seharusnya

1. Perubahan pada Struktur Kelas (MenuMakanan)
   4.10 - Keep Classes Small
   Pada awalnya, kelas Restaurant menangani terlalu banyak tanggung jawab, seperti menyimpan data makanan, mengelola stok, dan menampilkan menu. Ini melanggar prinsip Single Responsibility Principle (SRP).

   Untuk memperbaikinya, dibuat kelas MenuMakanan yang menangani informasi makanan. Dengan cara ini, kelas Restaurant hanya bertanggung jawab atas manajemen restoran, bukan detail makanan.
   Kode sebelum perubahan

```
public class Restaurant {
  public String[] nama_makanan;
  public double[] harga_makanan;
  public int[] stok;
  public int id = 0;

  public Restaurant() {
    nama_makanan = new String[10];
    harga_makanan = new double[10];
    stok = new int[10];
  }

  public void tambahMenuMakanan(String nama, double harga, int stok) {
    this.nama_makanan[id] = nama;
    this.harga_makanan[id] = harga;
    this.stok[id] = stok;
    id++;
  }
}
```

Kode Setelah Perubahan:

```
public class MenuMakanan {
  private String nama;
  private double harga;
  private int stok;

  public MenuMakanan(String nama, double harga, int stok) {
    this.nama = nama;
    this.harga = harga;
    this.stok = stok;
  }

  public String getNama() { return nama; }
  public double getHarga() { return harga; }
  public int getStok() { return stok; }

  public boolean stokHabis() {
    return stok <= 0;
  }
}
```

Keuntungan Perubahan:

- **Setiap kelas memiliki satu tanggung jawab.**
- **Kelas lebih kecil dan mudah dikelola.**
- **Memudahkan pemeliharaan dan pengembangan aplikasi.**

2. Penerapan Encapsulation (Aturan 4.10 - Keep Data Private)

   4.10 - Keep Data Private

   Sebelumnya, atribut nama_makanan, harga_makanan, dan stok bersifat public, yang memungkinkan akses langsung dari luar kelas. Ini bertentangan dengan prinsip Encapsulation, di mana data seharusnya hanya bisa diakses melalui metode yang ditentukan.

   Kode Sebelum Perubahan:

```
public class Restaurant {
  public String[] nama_makanan;
  public double[] harga_makanan;
  public int[] stok;
}
```

Kode Setelah Perubahan:

```
public class MenuMakanan {
  private String nama;
  private double harga;
  private int stok;

  public String getNama() { return nama; }
  public double getHarga() { return harga; }
  public int getStok() { return stok; }
}
```

Keuntungan Perubahan:

- **Data tidak bisa diubah langsung dari luar kelas.**
- **Memastikan data hanya dimodifikasi melalui metode yang sesuai.**
- **Mencegah akses yang tidak diinginkan atau perubahan yang tidak terkontrol.**

3. Perubahan pada Manajemen ID Makanan (nextId())
   4.10 - Use Consistent Naming Conventions
   Metode nextId() awalnya digunakan untuk mengelola ID makanan, tetapi kurang jelas fungsinya karena bisa menyebabkan bug jika dipanggil di tempat yang salah. Nama metode juga kurang mendeskripsikan bahwa ID bertambah secara otomatis.

   Untuk memperbaikinya, nextId() dihapus, dan id++ dimasukkan langsung ke dalam metode tambahMenuMakanan(), sehingga ID bertambah saat makanan baru ditambahkan.
   Kode Sebelum Perubahan:

```
public int nextId() {
  return id++;
}
```

   Kode Setelah Perubahan:

```
public void tambahMenuMakanan(String nama, double harga, int stok) {
  daftarMakanan[id] = new MenuMakanan(nama, harga, stok);
  id++;
}
```

   Keuntungan Perubahan:

- **Kode lebih mudah dibaca dan dipahami.**
- **Menghindari kesalahan penggunaan nextId().**
- **Nama metode tambahMenuMakanan() lebih mendeskripsikan fungsinya.**

4. Pemindahan isOutOfStock() ke MenuMakanan
   4.10 - Be Explicit About State Dependencies

   Sebelumnya, metode isOutOfStock() berada di dalam Restaurant, yang berarti kelas Restaurant harus tahu tentang stok setiap makanan. Ini tidak efisien karena setiap makanan seharusnya bisa mengetahui sendiri apakah stoknya habis atau tidak.

   Sebagai solusinya, metode ini dipindahkan ke dalam MenuMakanan sebagai stokHabis().

   Kode Sebelum Perubahan:

   ```
   public boolean isOutOfStock(int id) {
     return stok[id] <= 0;
   }
   ```

   Kode Setelah Perubahan:

   ```
   public boolean stokHabis() {
     return stok <= 0;
   }
   ```

   Keuntungan Perubahan:

- **Setiap objek MenuMakanan bertanggung jawab atas stoknya sendiri.**
- **Mengurangi ketergantungan antar kelas (Restaurant tidak perlu mengecek stok makanan).**
- **Kode lebih modular dan lebih mudah diuji.**

B. Ada kebutuhan untuk mengembangkan aplikasi tersebut, dengan menambah fitur pemesanan dan mengurangi setiap stok yang ada. Apakah dengan desain program yang ada dapat dikembangkan? Jika Sulit kemukakan alasannya dan bandingkan dengan desain class hasil modifikasi anda.

Metode kurangiStok() ditambahkan dalam MenuMakanan untuk memastikan stok hanya dapat dikurangi melalui prosedur yang benar.

```
public void kurangiStok(int jumlah) { // Perubahan untuk mengurangi stok
  if (jumlah <= stok) {
    stok -= jumlah;
  } else {
    System.out.println("Stok tidak cukup!");
  }
}
```

Metode PesanMakanan() diimplementasikan dalam Restaurant untuk memungkinkan pelanggan memesan makanan dengan pengecekan stok otomatis.

```
public void PesanMakanan(String nama, int jumlah) { // Implementasi method pemesanan
  for (int i = 0; i < id; i++) {
    if (Menu[i].getNama().equals(nama)) {
      if (Menu[i].getStok() >= jumlah) {
        Menu[i].kurangiStok(jumlah);
        System.out.println("Pesanan " + nama + " sebanyak " + jumlah + " berhasil.");
      } else {
        System.out.println("Stok " + nama + " tidak mencukupi.");
      }
      return;
    }
  }
  System.out.println("Menu tidak ditemukan.");
}
```

Program utama (RestaurantMain) diperbarui untuk menguji fitur pemesanan dengan skenario stok mencukupi dan stok tidak mencukupi.

```
class RestaurantMain {
  public static void main(String[] args) {
    Restaurant menu = new Restaurant();
    menu.tambahMenuMakanan("Bala-Bala", 1000, 20);
    menu.tambahMenuMakanan("Gehu", 1000, 10);
    menu.tambahMenuMakanan("Tahu", 1000, 15);
    menu.tambahMenuMakanan("Molen", 1000, 5);

    menu.tampilMenuMakanan();

    System.out.println("\n=== Pemesanan ===");
    menu.PesanMakanan("Gehu", 5);
    menu.PesanMakanan("Molen", 6); // Tidak cukup stok

    System.out.println("\n=== Menu Setelah Pemesanan ===");
    menu.tampilMenuMakanan();
  }
}
```

C. Source code

```
class RestaurantMain {
  public static void main(String[] args) {
    Restaurant menu = new Restaurant();
    menu.tambahMenuMakanan("Bala-Bala", 1000, 20);
    menu.tambahMenuMakanan("Gehu", 1000, 10);
    menu.tambahMenuMakanan("Tahu", 1000, 15);
    menu.tambahMenuMakanan("Molen", 1000, 5);

    menu.tampilMenuMakanan();

    System.out.println("\n=== Pemesanan ===");
    menu.PesanMakanan("Gehu", 5);
```

```
        menu.PesanMakanan("Molen", 6); // Tidak cukup stok

        System.out.println("\n=== Menu Setelah Pemesanan ===");
        menu.tampilMenuMakanan();
    }
}




class MenuMakanan {
    private String nama_makanan;
    private double harga_makanan;
    private int stok;

    public MenuMakanan(String nama_makanan, double harga_makanan, int stok) {
        this.nama_makanan = nama_makanan;
        this.harga_makanan = harga_makanan;
        this.stok = stok;
    }

    public String getNama() {
        return nama_makanan;
    }

    public double getHarga() {
        return harga_makanan;
    }

    public int getStok() {
        return stok;
    }

    public boolean stokhabis() {
        return stok == 0;
    }

    public void kurangiStok(int jumlah) { // Perubahan untuk mengurangi stok
```

```
        if (jumlah <= stok) {
            stok -= jumlah;
        } else {
            System.out.println("Stok tidak cukup!");
        }
    }
}



class Restaurant {
    private MenuMakanan[] Menu;
    private static byte id = 0;

    public Restaurant() {
        Menu = new MenuMakanan[10];
    }

    public void tambahMenuMakanan(String nama, double harga, int stok) {
        Menu[id] = new MenuMakanan(nama, harga, stok);
        id++;
    }

    public void tampilMenuMakanan() {
        for (int i = 0; i < id; i++) {
            if (!isOutOfStock(i)) {
                System.out.println(Menu[i].getNama() + " (" + Menu[i].getStok() + ")\tRp. " +
Menu[i].getHarga());
            }
        }
    }

    public void PesanMakanan(String nama, int jumlah) { // Implementasi method
pemesanan
        for (int i = 0; i < id; i++) {
            if (Menu[i].getNama().equals(nama)) {
                if (Menu[i].getStok() >= jumlah) {
```

```
            Menu[i].kurangiStok(jumlah);
            System.out.println("Pesanan " + nama + " sebanyak " + jumlah + " berhasil.");
          } else {
            System.out.println("Stok " + nama + " tidak mencukupi.");
          }
          return;
        }
      }
      System.out.println("Menu tidak ditemukan.");
    }

    public boolean isOutOfStock(int index) {
      return Menu[index].stokhabis();
    }
}
```

Output

```
"C:\Program Files (x86)\graalvm-jdk-21.0.
Bala-Bala (20)  Rp. 1000.0
Gehu (10)   Rp. 1000.0
Tahu (15)   Rp. 1000.0
Molen (5)   Rp. 1000.0

=== Pemesanan ===
Pesanan Gehu sebanyak 5 berhasil.
Stok Molen tidak mencukupi.

=== Menu Setelah Pemesanan ===
Bala-Bala (20)  Rp. 1000.0
Gehu (5)    Rp. 1000.0
Tahu (15)   Rp. 1000.0
Molen (5)   Rp. 1000.0

Process finished with exit code 0
```

SOAL 2

Kelas Film

Menyimpan informasi tentang film yang tersedia.

- **Atribut:** judul, genre, durasi, hargaTiket
- **Method:**
    - Film(String judul, String genre, int durasi, double hargaTiket) → Konstruktor untuk inisialisasi objek Film.
    - getJudul(), setJudul(String judul) → Mengambil dan mengubah judul film.
    - getGenre(), setGenre(String genre) → Mengambil dan mengubah genre film.
    - getDurasi(), setDurasi(int durasi) → Mengambil dan mengubah durasi film.
    - getHargaTiket(), setHargaTiket(double hargaTiket) → Mengambil dan mengubah harga tiket film.

Kelas Tiket

Merepresentasikan tiket yang dipesan oleh pelanggan.

- **Atribut:** film, jadwalTayang, nomorKursi (menggunakan List<String> agar bisa menyimpan lebih dari satu kursi)
- **Method:**
    - Tiket(Film film, String jadwalTayang, List<String> nomorKursi) → Konstruktor untuk membuat objek tiket berdasarkan film, jadwal, dan nomor kursi.
    - getFilm() → Mengambil informasi film yang terkait dengan tiket.
    - getJadwalTayang() → Mengambil jadwal tayang tiket.
    - getNomorKursi() → Mengambil daftar nomor kursi yang dipesan.

Kelas Pelanggan

Menyimpan informasi pelanggan yang memesan tiket.

- **Atribut:** nama, email, notelp
- **Method:**

- Pelanggan(String nama, String email, String notelp) → Konstruktor untuk inisialisasi data pelanggan.
- getNama() → Mengambil nama pelanggan.
- getEmail() → Mengambil email pelanggan.
- getNotelp() → Mengambil nomor telepon pelanggan.

Kelas Pemesanan

Menyimpan transaksi pemesanan tiket.

- **Atribut:** pelanggan, tiket, jumlahTiket, totalHarga
- **Method:**
  - Pemesanan(Pelanggan pelanggan, Tiket tiket, int jumlahTiket) → Konstruktor untuk membuat objek pemesanan berdasarkan pelanggan, tiket, dan jumlah tiket yang dipesan.
  - hitungTotalHarga() → Menghitung total harga berdasarkan jumlah tiket yang dipesan.
  - tampilkanDetailPemesanan() → Menampilkan detail pemesanan seperti nama pelanggan, film yang dipesan, jadwal tayang, nomor kursi, dan total harga.

Kelas Main

Menyimulasikan sistem pemesanan tiket bioskop.

- **Method:**
  - main(String[] args) → Fungsi utama yang menampilkan daftar film, membuat pelanggan, melakukan pemesanan tiket, dan menampilkan detail pemesanan.

1. Film.java

```
// Kelas Film
public class Film {
    private String judul;
    private String genre;
    private int durasi;
    private double hargaTiket;

    public Film(String judul, String genre, int durasi, double hargaTiket){
        this.judul = judul;
        this.genre = genre;
        this.durasi = durasi;
        this.hargaTiket = hargaTiket;
    }

    public String getJudul(){
        return judul;
    }

    public String getGenre(){
        return genre;
    }

    public int getDurasi(){ // Perbaikan nama method
        return durasi;
    }

    public double getHargaTiket() { // Getter harga tiket
        return hargaTiket;
    }

    public void setHargaTiket(double hargaTiket){ // Perbaikan setter
        this.hargaTiket = hargaTiket;
    }
}
```

2. Tiket.java

```
// Kelas Tiket
public class Tiket {
    private Film film;
    private String jadwalTayang;
    private String nomorKursi;

    public Tiket(Film film, String jadwalTayang,
```

```
    String nomorKursi){
            this.film = film;
            this.jadwalTayang = jadwalTayang;
            this.nomorKursi = nomorKursi;
        }

        public Film getFilm(){
            return film;
        }

        public String getJadwalTayang(){
            return jadwalTayang;
        }

        public String getNomorKursi(){
            return nomorKursi;
        }
    }
```

3. Pelanggan.java

```
// Kelas Pelanggan
public class Pelanggan {
  private String nama;
  private String email;  // Perbaikan nama variabel
  private String notelp;

  public Pelanggan(String nama, String email, String notelp){
    this.nama = nama;
    this.email = email;
    this.notelp = notelp;
  }

  public String getNama(){
    return nama;
  }

  public String getEmail(){ // Perbaikan method getter
    return email;
  }

  public String getNotelp(){
    return notelp;
  }
}
```

4.Pemesanan.java

```
// Kelas Pemesanan
public class Pemesanan {
  private Pelanggan pelanggan; // Perbaikan nama variabel
  private Tiket tiket;
  private int jumlahTiket;
  private double totalHarga;

  public Pemesanan(Pelanggan pelanggan, Tiket tiket, int jumlahTiket) {
    this.pelanggan = pelanggan;
    this.tiket = tiket;
    this.jumlahTiket = jumlahTiket;
    this.totalHarga = hitungTotalHarga();
  }

  private double hitungTotalHarga() {
    return jumlahTiket * tiket.getFilm().getHargaTiket();
  }

  public void tampilkanDetailPemesanan() {
    System.out.println("=== Detail Pemesanan Tiket ===");
    System.out.println("Nama Pelanggan  : " + pelanggan.getNama());
    System.out.println("Email        : " + pelanggan.getEmail());
    System.out.println("No. Telepon    : " + pelanggan.getNotelp()); //
Tambahan informasi
    System.out.println("Film         : " + tiket.getFilm().getJudul());
    System.out.println("Genre        : " + tiket.getFilm().getGenre());
    System.out.println("Durasi       : " + tiket.getFilm().getDurasi() + "
menit");
    System.out.println("Jadwal Tayang  : " + tiket.getJadwalTayang());
    System.out.println("Nomor Kursi    : " + tiket.getNomorKursi());
    System.out.println("Jumlah Tiket   : " + jumlahTiket);
    System.out.println("Total Harga    : Rp. " + totalHarga);
  }
}
```

5. Main.java

```
// Kelas Main
public class Main {
    public static void main(String[] args) {
        Film film1 = new Film("Hachiko", "Drama", 93, 35000);
        Film film2 = new Film("Spider-man no way home",
"Fiction", 148, 40000);

        // Menampilkan informasi film
```

```
        System.out.println("=== Daftar Film ===");
        System.out.println("1. " + film1.getJudul() + " (" +
film1.getGenre() + ") - " + film1.getDurasi() + " menit - Rp. "
+ film1.getHargaTiket());
        System.out.println("2. " + film2.getJudul() + " (" +
film2.getGenre() + ") - " + film2.getDurasi() + " menit - Rp. "
+ film2.getHargaTiket());

        // Membuat pelanggan
        Pelanggan pelanggan1 = new Pelanggan("Ihsan Fauzi",
"ihsan@email.com", "081234567890"); // Tambah nomor telepon

        // Memesan tiket
        Tiket tiket1 = new Tiket(film1, "27 Februari 2025 -
19:00 WIB", "A5");
        Pemesanan pemesanan1 = new Pemesanan(pelanggan1, tiket1,
1);

        // Menampilkan detail pemesanan
        pemesanan1.tampilkanDetailPemesanan();
    }
}
```

Output

```
"C:\Program Files (x86)\graalvm-jdk-21.0.6+8.1\bin\java.exe" "-ja
=== Daftar Film ===
1. Hachiko (Drama) - 93 menit - Rp. 35000.0
2. Spider-man no way home (Fiction) - 148 menit - Rp. 40000.0
=== Detail Pemesanan Tiket ===
Nama Pelanggan  : Ihsan Fauzi
Email           : ihsan@email.com
No. Telepon     : 081234567890
Film            : Hachiko
Genre           : Drama
Durasi          : 93 menit
Jadwal Tayang   : 27 Februari 2025 - 19:00 WIB
Nomor Kursi     : A5
Jumlah Tiket    : 1
Total Harga     : Rp. 35000.0

Process finished with exit code 0
```

Link Github

https://github.com/isanzzi/Tekpro-P3