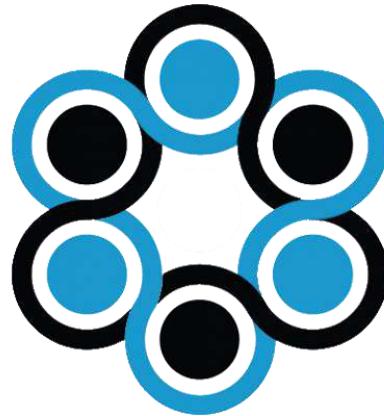


LAPORAN AKHIR
PRAKTIKUM PEMROGRAMAN
BERORIENTASI OBJEK
Periode X



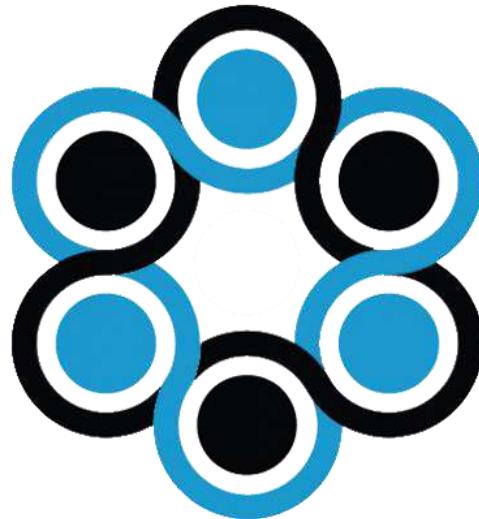
Disusun Oleh :

NAMA : FAUZI SAPUTRA

NPM : 06.2023.1.07748

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA
2024

LAPORAN AKHIR
PRAKTIKUM PEMROGRAMAN
BERORIENTASI OBJEK
Periode X



Disusun Oleh:

FAUZI SAPUTRA [06.2023.1.07748]
FERRY IRAWAN

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA
2024

LEMBAR PENGESAHAN
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

Periode X

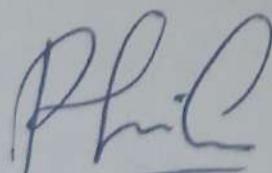
Disusun oleh :

NAMA : Fauzi Saputra

NPM : 06.2023.1.07748

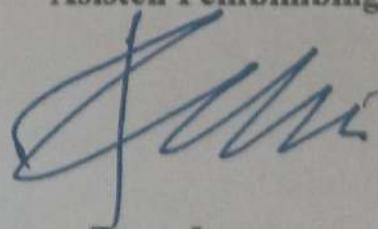
Tanggal 22/01/2025

Dosen Pembimbing



M. Kurniawan, S.Kom., M.Kom.
NIP 153045

Asisten Pembimbing



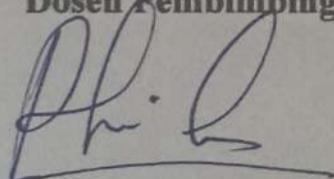
Ferry Irawan

BIMBINGAN DOSEN
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

NAMA : FAUZI SAPUTRA
NPM : 06.2023.1.07748

NO	TANGGAL	POKOK BAHASAN	PARAF DOSEN PEMBIMBING
1	22-10-2024	Class attribute variable method constructor	/
2	05-11-2024	Data collection Input output	/
3	18-11-2024	Inheritance	/
4	02-12-2024	Polymorph	/
5	16-12-2024	Package abstraction interface relation	/
6	31-12-2024	exception handling Pemrograman multimedia	/
7	13-01-2025	Graphical User interface	/

Surabaya,
Dosen Pembimbing



M. KURNIAWAN, S.Kom, M.Kom
410110150087

**BIMBINGAN ASISTEN
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK**

NAMA : Fauzi Saputra
NPM : 06.2023.1.07748

NO	TANGGAL	POKOK BAHASAN	PARAF ASISTEN PEMBIMBING
1	21-10-2024	Class attribute Variable Method constructor	✓
2	09-11-2024	Data Collection Input Process Output	✓
3	15-11-2024	Encapsulation & Inheritance	✓
4	01-12-2024	Relasi Antar Class & Polymorphism	✓
5	15-12-2024	Package abstraction interfaces	✓
6	30-12-2024	Exception handling Pemrograman multimedia	✓
7	12-01-2025	Graphic User Interface (GUI)	✓

Surabaya, 22-01-2025
Asisten Pembimbing

Ferry Irawan



PERTEMUAN 1

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Feni, Sarwita
NPM : 06.2023.1.07746
Pertemuan : 1

Feni.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 1

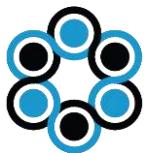
RPL-MF1T3-A

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
PertemuanX_NPM AKHIR
Ganti “X” menjadi Pertemuan yang sedang berlangsung.
 - 2) Pada saat Praktikum, Jawabanlah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar **“Laporan Praktikum”**.
 - 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
 - 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
 - 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
PertemuanX_NPM AKHIR.pdf
 - 6) Upload File **Laporan Praktikum [PDF]** pada form yang sudah disediakan.
-

TUGAS PRAKTIKUM

1. Jelaskan dengan rinci apa yang dimaksud dengan “Class”, “Object”, dan “Method” dalam pemrogramman Java! [wajib]
2. Buatlah Class “Character” yang memiliki atribut “Name”, “Level”, “Agility”. Dan berikan constructor! [wajib]
3. Dari soal nomor 2 berikan method “Walking” untuk mengoutput “[Character.Name] is Walking with [Character.Agility] speed...” [wajib]
4. Jelaskan apa itu static atribut/ method pada paradigma OOP!
5. Buatlah object dari class *Scanner* dengan nama “input”.
6. Dari soal nomor 2 buatlah method untuk membuat Object dari class Character Bernama “createCharacter” dan gunakan Scanner untuk menginput atribut atributnya!



TUGAS PRAKTIKUM

Soal Praktikum

1. Jelaskan dengan rinci apa yang dimaksud dengan “Class”, “Object”, dan “Method” dalam pemrograman Java!

Jawaban

- Class adalah prototype, atau blueprint yang mendefinisikan atribut dan method pada sebuah object tertentu.
- Object adalah bentuk nyata dari sebuah class yang dibuat.
- Method adalah sejenis function (fungsi) yang didefinisikan dalam Class untuk menggambarkan Tindakan atau perilaku objek

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Buatlah Class “Character” yang memiliki atribut “Name”, “Level”, “Agility”, Dan berikan constructor!

Jawaban

Ketik jawaban disini ...

Source Code

```
Import.java.util.Scanner;
public class Character {
    // Atribut
    String name;
    int level;
    String agility;
}
```

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

3. Dari soal nomor 2 berikan method “Walking” untuk mengoutput “[Character.Name] is Walking with [Character.Agility] speed...”

Jawaban

Ketik jawaban disini ...

Source Code

```
public class Character {  
    // Atribut  
    String name;  
    int level;  
    String agility;  
  
    //Constructor  
    public Character(String name, int level, String agility){  
        this.name = name;  
        this.level = level;  
        this.agility = agility;  
    }  
    public void Walking (){  
        System.out.println(name + " is walking with " + agility +  
" speed...");  
    }  
}
```

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

4. Jelaskan apa itu static atribut/method pada paradigma OOP!

Jawaban

- Static Atribut adalah variabel yang dideklarasikan dengan kata kunci static dalam sebuah kelas.
- Static Method adalah metode yang dideklarasikan dengan kata kunci static. Metode ini dapat dipanggil tanpa perlu membuat objek dari kelas tersebut.

Source Code

Tulis kode program dikotak ini...

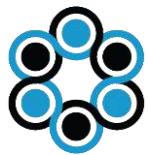
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

5. Buatlah object dari class *Scanner* dengan nama “input”.

Jawaban

Ketik jawaban disini ...

Source Code

```
Scanner input = new Scanner (System.in);
```

Penjelasan

Membuat object Scanner bernama input

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

6. Dari soal nomor 2 buatlah method untuk membuat Object dari class Character Bernama “createCharacter” dan gunakan Scanner untuk menginput atribut-atributnya!

Jawaban

Ketik jawaban disini ...

Source Code

```
public static Character createCharacter(Scanner input) {  
    System.out.print("Nama Karakter: ");  
    String name = input.nextLine();  
  
    System.out.print("Level Karakter: ");  
    int level = input.nextInt();  
  
    System.out.print("Agility Karakter: ");  
    int agility = input.nextInt();  
  
    // Membersihkan input buffer setelah nextInt  
    input.nextLine();  
  
    // Membuat object karakter baru dengan input dari user  
    return new Character(name, level, agility);  
}
```

Penjelasan

nextLine() untuk membaca inputan berupa kata dan kalimat(String)

nextInt() untuk membaca inputan berupa int(integer)

Oh iya jangan lupa mengakhiri inputan pertama dengan menambahkan method nextLine(). Hal ini bertujuan supaya inputan selanjutnya yang merupakan tipe data String dapat terbaca atau tidak terlewati.



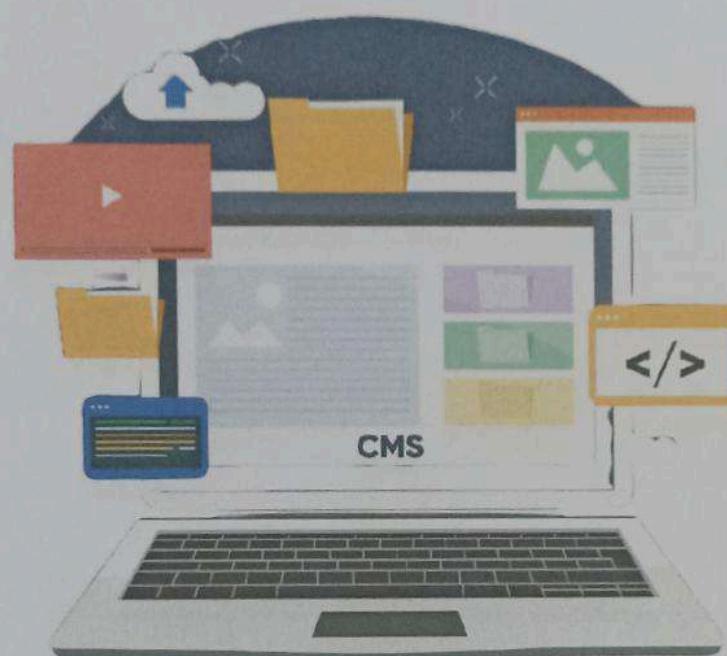
TUGAS PRAKTIKUM

Output

```
File Edit Selection View Go Run ... ← → 🔍 Praktikum p-1
EXPLORER ...
PRAKTIKUM P-1
    ~$rtemuan1_06.2023....
    J Character.class
    J Character.java
    Pertemuan1_06.2023...
    Pertemuan1_06.2023...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS G:\kuliah\praktikum\Semester 3\Praktikum p-1> cd "g:\kuliah\praktikum\Semester 3\Praktikum p-1"
Nama Karakter : Fauzi
Level Karakter : 95
Agility Karakter : 1000
Fauzi is Walking with 1000 speed...
PS G:\kuliah\praktikum\Semester 3\Praktikum p-1>
```

TUGAS DAN EVALUASI



**PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X**

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Modul : 1

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Sebutkan dan jelaskan secara rinci ke empat struktur utama paradigma pemrograman OOP!

Jawaban

- a) Class, adalah prototype, atau blueprint yang mendefinisikan atribut dan method pada sebuah objek tertentu
- b) Object, adalah bentuk nyata dari sebuah class yang dibuat. Objek bisa mewakili benda nyata atau konsep abstrak dari suatu hal
- c) Method, sejenis function(fungsi) yang didefinisikan dalam Class untuk menggambarkan tindakan atau perilaku objek
- d) Attribute, adalah variabel yang menyimpan data atau informasi tentang objek yang dibuat dari class tertentu.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Berikan contoh perumpamaan antara Class dan Object di kehidupan nyata!

Jawaban

Perumpamaan

Class: Mobil

Deskripsi: Mobil adalah sebuah konsep atau blueprint yang mendefinisikan berbagai atribut dan perilaku yang dimiliki oleh mobil seperti warna, merek, model, dan kemampuan berkendara.

Object: Mobil Toyota Avanza Hitam

Object adalah intansi dari class. Dalam contoh ini, "Mobil Toyota Avanza Hitam" adalah sebuah objek yang spesifik. Ia memiliki atribut warna hitam, merk Toyota, dan tipe Avanza, serta dapat melakukan metode seperti ini mengemudi dan berhenti.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Implementasikan soal nomor 2 ke dalam coding !

Jawaban

Ketik jawaban disini ...

Source Code

```
// Mendefinisikan class mobil
class Mobil {
    String merk;
    String warna;
    String tipe;

    // Constructor
    public Mobil(String merk, String warna, String tipe) {
        this.merk= merk;
        this.warna= warna;
        this.tipe= tipe;
    }

    // Metode untuk mengemudi
    public void mengemudi() {
        System.out.print("Mobil " + " " + merk + " " + tipe
+ " " + "berwarna" + warna + " " + "sedang mengemudi");
    }
}

// class utama untuk menjalankan program
public class Main {
    public static void main(String[] args) {
        Mobil mobil1 = new Mobil(merk:"Toyota", warna:"Hitam",
tipe:"Avanza");
        Mobil mobil2 = new Mobil("Lamborghini", "Orange",
"Aventador");
```



TUGAS & EVALUASI

```
mobil1.mengemudi();
mobil2.mengemudi();
}
}
```

Penjelasan

Di dalam codingan saya menggunakan tipe data String[] args berfungsi untuk menyimpan sejumlah nilai string dalam satu variabel dan untuk menerima argumen dari baris perintah saat program dijalankan. Oh iya ada tambahn sedikit penjelasan yaitu jangan awali void sebelum nama constructor kalau tidak ia akan terjadi error.

Output

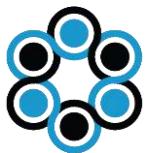
The screenshot shows the Eclipse IDE interface. In the center, there is a code editor with the following Java code:

```
PROJECT BAB I
Main.java X J Mainclass J Mobilclass
Main.java
Mobil
mengemudi()

Main.java > Mobil > mengemudi()
1 // Mendeфиниsikan class mobil
2 class Mobil {
3     private String merk;
4     private String warna;
5     private String tipe;
6
7     // Constructor
8     public Mobil(String merk, String warna, String tipe) {
9         this.merk= merk;
10        this.warna= warna;
11        this.tipe= tipe;
12    }
13
14    // Metode untuk mengemudi
15    public void mengemudi() {
16        System.out.println("Mobil " + " " + merk + " " + tipe + " " + "berwarna" + " " + warna + " " + "sedang mer");
17    }
18
19
20 // class utama untuk menjalankan program
```

Below the code editor is a terminal window showing the command-line output:

```
PS G:\kuliah\praktikum\Semester 3\praktikum p-1\Project Bab 1> cd "G:\kuliah\praktikum\Semester 3\praktikum p-1\Project Bab 1"> if ($?) { javac Main.java } ; if ($?) { java Main }
Mobil Toyota Avanza berwarna Hitam sedang mengemudi
Mobil Lamborghini Aventador berwarna Orange sedang mengemudi
PS G:\kuliah\praktikum\Semester 3\praktikum p-1\Project Bab 1>
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Apa itu Polymorphism dan Inheritance?

Jawaban

- Polymorphism, prinsip terakhir dalam OOP adalah polymorphism. Pada dasarnya polymorphism adalah kemampuan suatu pesan atau data untuk diproses lebih dari satu bentuk.

Inheritance, dalam konsep OOP adalah kemampuan untuk membuat class baru yang memiliki fungsi turunan atau mirip dengan fungsi yang ada sebelumnya.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Modul : 2

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Jelaskan apa yang dimaksud dengan Class, Attribute, dan Method berdasarkan pemahaman-mu!

Jawaban

- a) Class, adalah *prototype*, atau *blueprint* yang mendefinisikan atribut dan method pada sebuah objek tertentu. Class berfungsi untuk menampung isi dari kode program yang akan dijalankan, di dalam class berisi atribut dengan tipe data tertentu dan sebuah method.
- b) Attribute, atribut adalah variabel yang ada didalam class. Atribut ini mendefinisikan data atau informasi yang terkait dengan class tersebut.
- c) Method, adalah fungsi yang berkaitan dengan objek yang dibuat dari class tersebut. Method memberikan kemampuan kepada objek untuk melakukan tindakan atau operasi tertentu, dan mereka bekerja dengan data yang disimpan dalam atribut-atribut objek.

Source Code

Tulis kode program dikotak ini...

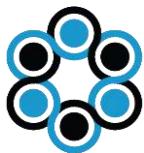
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Jelaskan apa yang dimaksud dengan Constructor dan Object!

Jawaban

- a) Constructor adalah sebuah method khusus yang digunakan untuk menginisialisasi (*instance*) objek dari sebuah Class. a berfungsi untuk memberikan nilai awal kepada atribut dalam objek dan melakukan tugas-tugas lain yang diperlukan saat pembuatan objek.
- b) Object adalah *instance* dari sebuah class. Dalam arti yang lebih sederhana, Objek adalah wujud konkret dari suatu entitas yang terdefinisi oleh class. Objek memiliki atribut yang merepresentasikan sifat dan method yang merepresentasikan tindakan yang bisa dilakukan oleh objek tersebut.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah class Buku yang memiliki atribut judul, penulis, dan tahunTerbit.
Kemudian tampilkan data dengan static method tampilkanData()!

Jawaban

Ketik jawaban disini ...

Source Code

```
// Mendefinisikan class Buku
class Buku {
    String judul;
    String pengarang;
    int tahunTerbit;

    // Constructor
    public Buku(String judul, String pengarang, int tahunTerbit)
    {
        this.judul= judul;
        this.pengarang= pengarang;
        this.tahunTerbit= tahunTerbit;
    }

    // Method tampilkanData
    public void tampilkanData() {
        System.out.println("Judul Buku : " + judul);
        System.out.println("Pengarang : " + pengarang);
        System.out.println("Tahun Terbit : " +
tahunTerbit);
    }
}

// Static Method
public class Main {
    public static void main(String[] args) {
        Buku buku1 = new Buku("Paradigma", "Syahid Muhammad",
2018);
```



TUGAS & EVALUASI

```
buku1.tampilkanData();  
}  
}
```

Penjelasan

Pada Constructor Buku saya menggunakan `this.judul`: `this` digunakan untuk membedakan antara parameter local dan atribut kelas. Artinya, `this.judul` mengacu pada atribut mengacu pada atribut judul dari objek tersebut, sedangkan judul di sebelah kanan adalah parameter yang diterima oleh constructor.

Output

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows multiple Java projects under "SEMESTER 3". One project, "Main.java", contains a "Buku" class definition.
- Buku Class Definition:**

```
1 // Mendefinisikan class buku  
2 class Buku {  
3     private String judul;  
4     private String pengarang;  
5     private int tahunTerbit;  
6  
7     // Constructor  
8     public Buku(String judul, String pengarang, int tahunTerbit) {  
9         this.judul = judul;  
10        this.pengarang = pengarang;  
11        this.tahunTerbit = tahunTerbit;  
12    }  
13  
14    // Method tampilkanData  
15    public void tampilkanData() {  
16        System.out.println("Judul: " + judul);  
17        System.out.println("Pengarang: " + pengarang);  
18        System.out.println("Tahun Terbit: " + tahunTerbit);  
19    }  
20}
```
- Main.java Driver Class:**

```
1 error:  
PS G:\Voliah\praktikum\Semester 3\Project Bab 2> cd "G:\Voliah\praktikum\Semester 3\Project Bab 2"&; if ($?) { javac Main.java } ; if ($?) {  
java Main }  
Judul: Paradigmas  
Pengarang: Syarid Muhammad  
Tahun Terbit: 2018  
Judul: Teks Pengantar  
Pengarang: Faizi Saputra  
Tahun Terbit: 2022
```
- Output Console:** Displays the output of running the Main.java program, showing two book entries with their titles, authors, and publication years.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buat class Laptop dengan atribut merek, prosesor, dan ram

Jawaban

Ketik jawaban disini ...

Source Code

```
public class Laptop {  
    // Atribut  
    String merek;  
    String prosesor;  
    int ram;
```

Penjelasan

Pendeklarasian class laptop yang berisi string merek, string prosesor, dan int ram

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

5. Gunakan Constructor untuk inisialisasi atribut-atribut tersebut. Tambahkan metode upgradeRam() yang menerima parameter untuk menambah kapasitas RAM. Cetak informasi setiap objek dengan method tampilkanInfo()!

Jawaban

Ketik jawaban disini ...

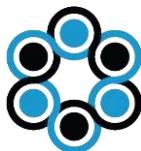
Source Code

```
// Constructor untuk inisialisasi atribut
public Laptop(String merek, String prosesor, int ram) {
    this.merek = merek;
    this.prosesor = prosesor;
    this.ram = ram;
}

// Metode untuk upgrade RAM
public void upgradeRam(int tambahanRam) {
    this.ram += tambahanRam;
    System.out.println("RAM berhasil di-upgrade. RAM saat ini: " + this.ram + " GB");
}

// Metode untuk menampilkan informasi laptop
public void tampilkanInfo() {
    System.out.println("Merek Laptop: " + merek);
    System.out.println("Prosesor: " + prosesor);
    System.out.println("RAM: " + ram + " GB");
}

// Main method untuk menjalankan program
public static void main(String[] args) {
    // Membuat objek Laptop
```



TUGAS & EVALUASI

```
Laptop laptop1 = new Laptop("Lenovo", "Intel Core i7",
4);
laptop1.tampilkanInfo();

System.out.println("\nUpgrade RAM...");
laptop1.upgradeRam(4); // Upgrade RAM

System.out.println("\nInfo setelah upgrade:");
laptop1.tampilkanInfo(); // Menampilkan info setelah
upgrade
}
}
```

Penjelasan

Terlihat dibawah public void upgradeRam ada code yang seperti `(this.ram +=)` kenapa ada `(+=)?` itu berfungsi untuk menambahkan object laptop ram yang diperintahkan, lalu ditambahkan pada saat object menampilkan data yaitu di `laptop1.upgradeRam(4)`, didalam parameter `upgradeRam` berisi 4 itu berfungsi angka yang akan ditambahkan.

Output

The screenshot shows a Java IDE interface with the following details:

- File Explorer:** Shows the project structure with files like Main.java, Laptop.class, and Laptop.java.
- Code Editor:** Displays the `Laptop.java` file containing the following code:

```
public class Laptop {
    // Metode untuk upgrade RAM
    public void upgradeRam(int tambahanRam) {
        this.ram += tambahanRam;
        System.out.println("RAM berhasil di-upgrade. RAM saat ini: " + this.ram + " GB");
    }

    // Metode untuk menampilkan informasi laptop
    public void tampilInfo() {
        System.out.println("Merek Laptop: " + this.merek);
        System.out.println("Prosesor: " + this.prosesor);
        System.out.println("RAM: " + this.ram + " GB");
    }

    // Main method untuk menjalankan program
    public static void main(String[] args) {
        // Instansiasi objek Laptop
        Laptop laptop1 = new Laptop(merek:"Lenovo", prosesor:"Intel Core i7", ram:4);
    }
}
```
- Terminal:** Shows the output of running the program:

```
Merek Laptop: Lenovo
Prosesor: Intel Core i7
RAM: 4 GB

Upgrade RAM...
RAM berhasil di-upgrade. RAM saat ini: 8 GB
```
- Status Bar:** Shows Java Ready, 21:00, 0 Java Ready, and a search bar at the bottom.
- Bottom Right:** Activation message for Windows.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

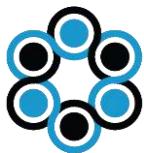
6. Buatlah class Mahasiswa dengan atribut namaLengkap, npm, dan jenisKelamin. Buat tiga objek: satu menggunakan constructor tanpa parameter, dua lainnya dengan Constructor dengan Parameter. Cetak informasi setiap objek dengan method tampilkanInfo()!

Jawaban

Ketik jawaban disini ...

Source Code

```
public class Mahasiswa {  
    // Attribute  
    String namaLengkap;  
    String npm;  
    String jenisKelamin;  
  
    //Constructor Tanpa Parameter  
    public Mahasiswa(){  
        this.namaLengkap = "Zayden Putra";  
        this.npm = "06.2023.1.07746";  
        this.jenisKelamin = "Laki-Laki";  
    }  
  
    //Constructor Menggunakan Parameter  
    public Mahasiswa(String namaLengkap, String npm, String  
jenisKelamin){  
        this.namaLengkap = namaLengkap;  
        this.npm = npm;  
        this.jenisKelamin = jenisKelamin;  
    }  
    public void tampilkanInfo(){  
        System.out.println("Nama: " + namaLengkap);  
        System.out.println("Npm: " + npm);  
        System.out.println("Jenis Kelamin: " + jenisKelamin);  
    }  
}
```



TUGAS & EVALUASI

```
}

public static void main (String[] args){
    //Membuat Object Mahasiswa Menggunakan Constructor Tanpa Parameter
    Mahasiswa mahasiswa1 = new Mahasiswa();

    //Membuat Object Mahasiswa Menggunakan Constructor Tanpa Parameter
    Mahasiswa mahasiswa2 = new Mahasiswa("Erlinda",
    "06.2023.1.07747", "Perempuan");
    Mahasiswa mahasiswa3 = new Mahasiswa("Fauzi Saputra",
    "06.2023.1.07748", "Laki-Laki");

    //Menampilkan Information masing-masing object
    System.out.println();
    mahasiswa1.tampilkanInfo();

    System.out.println();
    mahasiswa2.tampilkanInfo();

    System.out.println();
    mahasiswa3.tampilkanInfo();
}

}
```

Penjelasan

disitu diperintahkan membuat 2 tipe constructor yaitu constructor berparameter dan constructor tanpa parameter. Contoh syntax constructor tanpa parameter: `public Mahasiswa();` dan ini Contoh syntax constructor menggunakan parameter: `public Mahasiswa(String nama, String npm, String jenisKelamin)` bedanya constructor menggunakan parameter dan constructor tanpa parameter ialah cara untuk menampilkan data. Constructor menggunakan parameter ia menampilkan datanya di method dan di object. Berbeda dengan constructor tanpa parameter ia menampilkan datanya di constructornya langsung.



TUGAS & EVALUASI

Output

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Structure (Left):** Shows projects like SEMESTER 3, praktikum p-1, Project Bab 1, Tugas&Evaluasi_06..., Project Bab 2, and Mahasiswa.
- Middle Panel:** The code editor displays `Mahasiswa.java` with two constructors and a `tampilkanInfo()` method.

```
public class Mahasiswa {
    // Attribute
    String namalengkap;
    String npm;
    String jeniskelamin;

    //Constructor Tanpa Parameter
    public Mahasiswa() {
        this.namalengkap = "Zayden Putra";
        this.npm = "06.2023.1.07746";
        this.jeniskelamin = "Laki-Laki";
    }

    //Constructor Menggunakan Parameter
    public Mahasiswa(String namalengkap, String npm, String jeniskelamin) {
        this.namalengkap = namalengkap;
        this.npm = npm;
        this.jeniskelamin = jeniskelamin;
    }

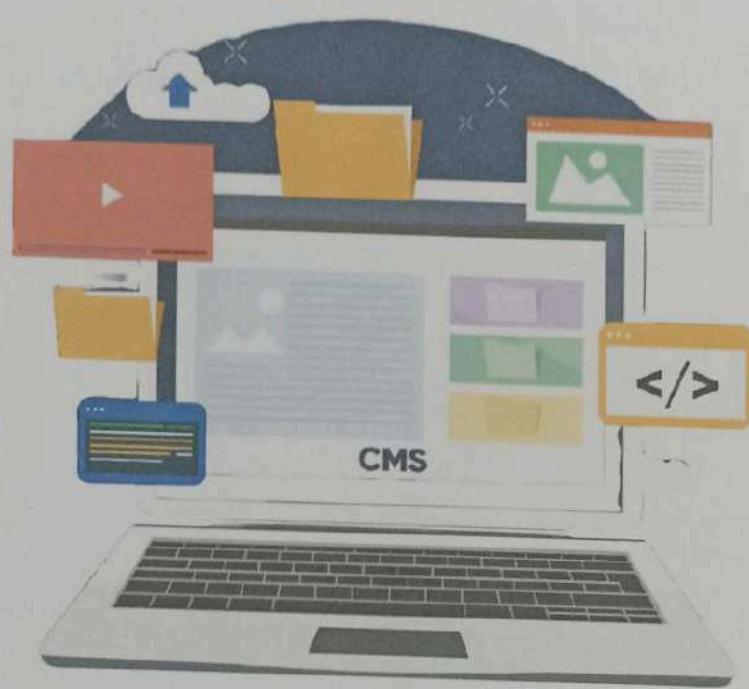
    public void tampilkanInfo() {
    }
}
```
- Output Console (Bottom):** Displays the output of the `tampilkanInfo()` method for two students:

```
Npm: 06.2023.1.07746
Jenis Kelamin: Laki-Laki

Nama: Erlinda
Npm: 06.2023.1.07747
Jenis Kelamin: Perempuan

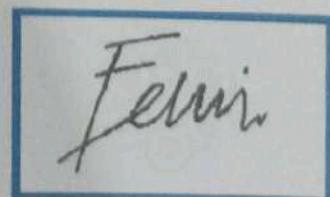
Nama: Faizi Saputra
Npm: 06.2023.1.07748
Jenis Kelamin: Laki-Laki
```
- Right Side:** Shows system status (Activate Windows), terminal tabs (PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PAGES), and system icons (Windows Start, Task View, File Explorer, etc.).

TUGAS DAN EVALUASI



**PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X**

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Modul : 3



Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Buatlah Class Transaksi dengan atribut tanggal(string) dan nominal(float) beserta constructornya!

Jawaban

Ketik jawaban disini ...

Source Code

```
public class Transaksi {  
    String tanggal;  
    float nominal;  
  
    public Transaksi(String tanggal, float nominal){  
        this.tanggal = tanggal;  
        this.nominal = nominal;  
    }  
}
```

Penjelasan

Class Transaksi dengan atribut string tanggal dan float nominal, menggunakan constructor berparameter

Output

The screenshot shows a Java development environment with two files open:

- Main.java**: Contains the main method and imports the Scanner class.
- Transaksi.java**: Contains the definition of the Transaksi class with its constructor.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Buatlah class scanner, lalu buatlah 3 objek dari class Transaksi(gunakan loop) dan isi value atribut dari objek tersebut menggunakan scanner!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.Scanner; // Impor class Scanner
public class Transaksi {
    // Atribut
    String tanggal;
    float nominal;

    // Constructor
    public Transaksi(String tanggal, float nominal) {
        this.tanggal = tanggal;
        this.nominal = nominal;
    }

    // Method untuk menampilkan informasi transaksi
    public void tampilkanTransaksi() {
        System.out.println("Tanggal: " + tanggal + ", Nominal: "
+ nominal);
    }
}

public class Main {
    public static void main(String[] args) {
        // Membuat scanner untuk input data
        Scanner scanner = new Scanner(System.in);

        // Membuat array untuk menyimpan objek Transaksi
        Transaksi[] transaksiArray = new Transaksi[3];
```



TUGAS & EVALUASI

```
// Loop untuk mengisi 3 objek transaksi
for (int i = 0; i < 3; i++) {
    System.out.println("Masukkan data transaksi ke-" +
(i + 1) + ":");

    System.out.print("Tanggal (format dd/mm/yyyy): ");
    String tanggal = scanner.nextLine();

    System.out.print("Nominal: ");
    float nominal = scanner.nextFloat();

    // Konsumsi karakter newline setelah input float
    scanner.nextLine();

    // Membuat objek transaksi dan memasukkannya ke
array
    transaksiArray[i] = new Transaksi(tanggal, nominal);
}

// Menampilkan informasi semua transaksi
System.out.println("\nInformasi Transaksi:");
for (Transaksi transaksi : transaksiArray) {
    transaksi.tampilkanTransaksi();
}

// Menutup scanner
scanner.close();
}
```

Penjelasan

Pada `Transaksi[] transaksiArray = new Transaksi[3];` : Membuat array untuk menampung 3 objek Transaksi. Dan pada `scanner.nextLine();` Digunakan setelah `scanner.nextFloat();` untuk menangkap sisa karakter newline (\n) setelah pengguna memasukkan nominal.



TUGAS & EVALUASI

Output

The screenshot shows a terminal window titled "Project Bab 3" with the following content:

```
File Edit Selection View Go Run ... ← → Project Bab 3
EXPLORER PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PROJECT BAB 3
Main.class
Transaksi.class
Transaksi.java

Masukkan data transaksi ke-1:
Tanggal (format dd/mm/yyyy): 20
Nominal: 2000
Masukkan data transaksi ke-2:
Masukkan data transaksi ke-3:
Tanggal (format dd/mm/yyyy): 22
Nominal: 4000

Informasi Transaksi:
Tanggal: 20, Nominal: 2000.0
Tanggal: 21, Nominal: 3000.0
Tanggal: 22, Nominal: 4000.0
PS G:\kuliah\praktikum\Semester 3\Project Bab 3> cd "g:\kuliah\praktikum\Semester 3\Project Bab 3\"
If ($?) { javac Transaksi.java } ; If ($?) { java Transak
$1 }
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Beri pengujian pada saat proses pemasukan data apabila user menginput nominal bukan kelipatan 50.000, beri output “nominal harus kelipatan 50.000”

Jawaban

Ketik jawaban disini ...

Source Code

```
while (!validNominal) {  
    System.out.print("Nominal: ");  
    nominal = scanner.nextFloat();  
  
    // Cek apakah nominal kelipatan 50.000  
    if (nominal % 50000 == 0) {  
        validNominal = true; // validasi berhasil  
    } else {  
        System.out.println("Nominal harus kelipatan  
50.000, coba lagi.");  
    }  
}
```

Penjelasan

Pada saat input nominal, ada validasi di dalam loop while yang akan meminta pengguna untuk memasukkan nominal hingga valid (yaitu kelipatan 50.000).

- Jika nominal yang diinput bukan kelipatan 50.000, akan muncul pesan: "Nominal harus kelipatan 50.000, coba lagi."



TUGAS & EVALUASI

Output

The screenshot shows a Java project named "PROJECT BAB 3" with files Main.java, Main.class, Main.java, and Transaksi.class. The Main.java file contains the following code:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Nominal harus kelipatan 50.000, coba lagi.");
    }
}
```

The terminal window shows the following output:

```
tempCodeRunnerFile.java:4: error: class, interface, enum, or record expected
}
^
3 errors
PS G:\kuliah\praktikum\Semester 3\Project Bab 3> cd "g:\kuliah\praktikum\Semester 3\Project Bab 3\" ; if ($?) { java
Masukkan data transaksi ke-1:
Tanggal (format dd/mm/yyyy): 10
Nominal: 20000
Nominal harus kelipatan 50.000, coba lagi.
Nominal: []
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Output semua object dari Class Transaksi yang telah dibuat!

Jawaban

Ketik jawaban disini ...

Source Code

```
System.out.println("\nInformasi Transaksi:");
    for (Transaksi transaksi : transaksiArray) {
        transaksi.tampilkanTransaksi();
    }
```

Penjelasan

Setelah ketiga objek Transaksi diinput, program akan menampilkan semua transaksi yang telah dibuat.

Output

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Bar:** Project Bab 3
- File Structure (EXPLORER):** PROJECT BAB 3, Main.java, Main.java, Transaksi.java
- Main.java:** Contains the code for printing three transactions.
- Transaksi.java:** Contains the class definition for Transaksi with attributes tanggal and nominal, and a method tampilkanTransaksi().
- Terminal (OUTPUT):** Shows the execution of the program, prompting for three transactions and then printing them out.
- Bottom Status Bar:** Activate Windows, Go to Settings to activate Windows.

```
System.out.println("\nInformasi Transaksi:");
    for (Transaksi transaksi : transaksiArray) {
        transaksi.tampilkanTransaksi();
    }
```

```
1. Masukkan data transaksi ke-1:
Tanggal (Format dd/mm/yyyy): 10
Nominal: 20000
Nominal harus kelipatan 50.000, coba lagi.
Nominal: 50000
Masukkan data transaksi ke-2:
Tanggal (Format dd/mm/yyyy): 20
Nominal: 100000
Masukkan data transaksi ke-3:
Tanggal (Format dd/mm/yyyy): 30
Nominal: 150000

Informasi Transaksi:
Tanggal: 10, Nominal: 50000.0
Tanggal: 20, Nominal: 100000.0
Tanggal: 30, Nominal: 150000.0
```



PERTEMUAN 2

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Salufra
NPM : 06.2023.1.07748
Pertemuan : 2

Fauzi.

A handwritten signature in black ink, reading "Fauzi.", enclosed within a thin blue rectangular border.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 2

RPL-MF4T5-B

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
“PertemuanX_NPM AKHIR”
Ganti “X” menjadi Pertemuan yang sedang berlangsung.
- 2) Pada saat Praktikum, Jawabanlah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar **“Laporan Praktikum”**.
- 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
- 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
- 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
“PertemuanX_NPM AKHIR.pdf”
- 6) Upload File **Laporan Praktikum [PDF]** dan pada form yang sudah disediakan.

TUGAS PRAKTIKUM

1. Apa yang dimaksud dengan **Data Collection** dan **Encapsulation** (Enkapsulasi) pada java? Jelaskan! Serta sebutkan macam-macam **Data Collection** yang kamu ketahui! [Wajib]
2. Apa perbedaan antara method **Getter** dan **Setter**? serta sebutkan dan jelaskan **Accesss Modifier** apa saja yang terdapat dalam konsep Enkapsulasi! [Wajib]
3. Buatlah sebuah class **Pegawai** yang memiliki atribut nama (gunakan method **getter & setter**). Kemudian, buatlah objek-objek berikut dari class tersebut. Pada kolom “....” isi dengan **nama kalian** sendiri!

NAMA
Rimuru
Ainz
Saitama
Gon
Asta
“....”



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

Setelahnya, buatlah sebuah **ArrayList** untuk menampung objek-objek pegawai tersebut, kemudian cetak hasilnya dengan menggunakan for loop/foreach loop [Wajib]!

4. Merujuk pada soal sebelumnya, buatlah sebuah program Java untuk menambahkan **Objek Pegawai** dari class **Pegawai** yang telah dibuat tadi ke dalam class **Departemen**!

Note : Buatlah Class Departemen yang memiliki method untuk menambahkan daftar pegawai.

5. Buatlah sebuah program Java yang mengimplementasikan relasi **Aggregation** (agregasi) antar kelas. Pilihlah studi kasus yang relevan dimana satu kelas "memiliki" kelas lain, tetapi kelas yang dimiliki dapat berdiri secara independen tanpa kelas yang memiliki. (Berikan studi kasus selain yang terdapat di modul) !



TUGAS PRAKTIKUM

Soal Praktikum

1. Apa yang dimaksud dengan **Data Collection** dan **Encapsulation** (Enkapsulasi) pada java? Jelaskan! Serta sebutkan macam-macam Data Collection yang kamu ketahui! [Wajib]

Jawaban

1. – Data Collection adalah istilah yang merujuk kepada struktur data yang digunakan untuk mengelompokkan dan menyimpan sejumlah elemen atau objek dalam satu kesatuan. Data Collection memiliki beraneka macam jenis, yaitu List, Set, Queue, dan Map.
- Encapsulation adalah suatu proses pembungkusan data (atribut) dan aksinya (method) menjadi satu. Dengan demikian, data yang ada tidak akan bisa diakses oleh class lain dan hanya dapat diakses melalui method pada class yang diijinkan. Encapsulation memiliki beraneka macam jenis yaitu, Access Modifier, Getter, dan Setter

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Apa perbedaan antara method **Getter** dan **Setter**? serta sebutkan dan jelaskan **Accesss Modifier** apa saja yang terdapat dalam konsep Enkapsulasi! [Wajib]

Jawaban

- **Metode Getter** berkaitan dengan mengambil nilai; suatu variabel, sementara **Metode Setter** digunakan untuk mengatur atau memperbarui nilai dari sebuah variabel yang ada.
- **Access Modifier** di dalam OOP (Object Oriented Programming) akan menentukan apakah Class lain dapat menggunakan field atau meminta izin untuk mengakses data dari suatu Class. Ada beberapa macam **Access Modifier** yang dapat digunakan, yaitu diantaranya **public**, **protected**, **private**, dan **default**.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

3. Buatlah sebuah class **Pegawai** yang memiliki atribut nama (gunakan method **getter & setter**). Kemudian, buatlah objek-objek berikut dari class tersebut. Pada kolom “....” isi dengan **nama kalian** sendiri!

NAMA
Rimuru
Ainz
Saitama
Gon
Asta
“....”

Setelahnya, buatlah sebuah **ArrayList** untuk menampung objek-objek pegawai tersebut, kemudian cetak hasilnya dengan menggunakan for loop/foreach loop [Wajib]!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;

class Pegawai {
    // Atribut private
    private String nama;

    // Constructor
    public Pegawai(String nama) {
        this.nama = nama;
    }
}
```



TUGAS PRAKTIKUM

```
// Getter untuk nama
public String getNama() {
    return nama;
}

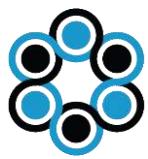
// Setter untuk nama
public void setNama(String nama) {
    this.nama = nama;
}
public static void main(String[] args) {
    // Membuat objek Pegawai
    Pegawai pegawai1 = new Pegawai("Rimuru");
    Pegawai pegawai2 = new Pegawai("Ainz");
    Pegawai pegawai3 = new Pegawai("Saitama");
    Pegawai pegawai4 = new Pegawai("Gon");
    Pegawai pegawai5 = new Pegawai("Asta");
    Pegawai pegawai6 = new Pegawai("Fauzi");

    // Membuat ArrayList untuk menampung objek Pegawai
    ArrayList<Pegawai> daftarPegawai = new ArrayList<>();

    // Menambahkan objek Pegawai ke dalam ArrayList
    daftarPegawai.add(pegawai1);
    daftarPegawai.add(pegawai2);
    daftarPegawai.add(pegawai3);
    daftarPegawai.add(pegawai4);
    daftarPegawai.add(pegawai5);
    daftarPegawai.add(pegawai6);

    // Mencetak nama-nama Pegawai menggunakan for-each loop
    System.out.println("Daftar Nama Pegawai:");
    for (Pegawai pegawai : daftarPegawai) {
        System.out.println(pegawai.getNama());
    }
}
```

Penjelasan



TUGAS PRAKTIKUM

Getter dan Setter digunakan untuk mengakses dan memodifikasi atribut nama, dan Loop for-each digunakan untuk mencetak nama-nama pegawai dari ArrayList

Output

The screenshot shows a terminal window titled "Praktikum p-2". The terminal output is as follows:

```
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> cd "g:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2"
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> if ($?) { javac tempCodeRunnerFile.java } ; if (?) { java tempCodeRunnerFile }
Error: Could not find or load main class tempCodeRunnerFile
Caused by: java.lang.ClassNotFoundException: tempCodeRunnerFile
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> cd "g:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2"
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> if (?) { javac Pegawai.java } ; if (?) { java Pegawai }
Daftar Nama Pegawai:
Rimuru
Ainz
Saitama
Gon
Asta
Faizi
```



TUGAS PRAKTIKUM

Soal Praktikum

4. Merujuk pada soal sebelumnya, buatlah sebuah program Java untuk menambahkan Objek Pegawai dari class Pegawai yang telah dibuat tadi ke dalam class Departemen! Note : Buatlah Class Departemen yang memiliki method untuk menambahkan daftar pegawai.

Note: Buatlah Class Departemen yang memiliki method untuk menambahkan daftar pegawai.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;

// Class Pegawai
class Pegawai {
    // Atribut private
    private String nama;

    // Constructor
    public Pegawai(String nama) {
        this.nama = nama;
    }

    // Getter untuk nama
    public String getNama() {
        return nama;
    }

    // Setter untuk nama
    public void setNama(String nama) {
        this.nama = nama;
    }
}
```



TUGAS PRAKTIKUM

```
// Class Departemen
class Departemen {
    // Atribut untuk menyimpan nama departemen dan daftar
    pegawai
    private String namaDepartemen;
    private ArrayList<Pegawai> daftarPegawai;

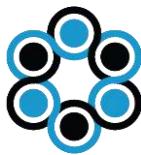
    // Constructor
    public Departemen(String namaDepartemen) {
        this.namaDepartemen = namaDepartemen;
        this.daftarPegawai = new ArrayList<>();
    }

    // Method untuk menambahkan pegawai ke dalam daftar pegawai
    departemen
    public void tambahPegawai(Pegawai pegawai) {
        daftarPegawai.add(pegawai);
    }

    // Method untuk mencetak daftar pegawai
    public void cetakDaftarPegawai() {
        System.out.println("Daftar Pegawai di Departemen " +
namaDepartemen + ":");

        for (Pegawai pegawai : daftarPegawai) {
            System.out.println("- " + pegawai.getNama());
        }
    }
}

// Main Class
public class Main {
    public static void main(String[] args) {
        // Membuat objek Pegawai
        Pegawai pegawai1 = new Pegawai("Rimuru");
        Pegawai pegawai2 = new Pegawai("Ainz");
        Pegawai pegawai3 = new Pegawai("Saitama");
        Pegawai pegawai4 = new Pegawai("Gon");
        Pegawai pegawai5 = new Pegawai("Asta");
```



TUGAS PRAKTIKUM

```
Pegawai pegawai6 = new Pegawai("Fauzi");

// Membuat objek Departemen
Departemen departemenIT = new Departemen("IT");
Departemen departemenHR = new Departemen("HR");

// Menambahkan pegawai ke dalam departemen IT
departemenIT.tambahPegawai(pegawai1);
departemenIT.tambahPegawai(pegawai2);
departemenIT.tambahPegawai(pegawai3);

// Menambahkan pegawai ke dalam departemen HR
departemenHR.tambahPegawai(pegawai4);
departemenHR.tambahPegawai(pegawai5);
departemenHR.tambahPegawai(pegawai6);

// Mencetak daftar pegawai di masing-masing departemen
departemenIT.cetakDaftarPegawai();
System.out.println(); // Pemisah
departemenHR.cetakDaftarPegawai();

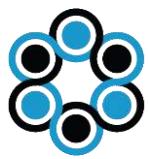
}
```

Penjelasan

Class Departemen:

- Memiliki atribut namaDepartemen untuk menyimpan nama departemen.
- Atribut daftarPegawai berupa ArrayList untuk menyimpan objek-objek Pegawai.
- Method tambahPegawai() digunakan untuk menambahkan objek Pegawai ke dalam daftar pegawai departemen.
- Method cetakDaftarPegawai() digunakan untuk mencetak nama-nama pegawai yang ada di dalam departemen.

Class Main:



TUGAS PRAKTIKUM

- Membuat dua objek departemen, yaitu departemenIT dan departemenHR.
- Menambahkan beberapa pegawai ke masing-masing departemen menggunakan method tambahPegawai().
- Mencetak daftar pegawai di setiap departemen dengan method cetakDaftarPegawai().

Output

```
File Edit Selection View Go Run ... ← → 🔍 Praktikum p-2
EXPLORER PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PRAKTIKUM P-2
J Pegawai.class
J Pegawai.java
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> cd "g:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2"
-2V ; if ($?) { javac tempCodeRunnerFile.java } ; if ($?) { java tempCodeRunnerFile }
Error: Could not find or load main class tempCodeRunnerFile
Caused by: java.lang.ClassNotFoundException: tempCodeRunnerFile
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> cd "g:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2"
-2V ; if ($?) { javac Pegawai.java } ; if ($?) { java Pegawai }
Daftar Nama Pegawai:
Rimuru
Afnan
Sigitama
Gon
Astha
Fauzzi
PS G:\kuliah\PRAKTIKKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2>
```



TUGAS PRAKTIKUM

Soal Praktikum

5. Buatlah sebuah program Java yang mengimplementasikan relasi Aggregation (agregasi) antar kelas. Pilihlah studi kasus yang relevan dimana satu kelas "memiliki" kelas lain, tetapi kelas yang dimiliki dapat berdiri secara independen tanpa kelas yang memiliki. (Berikan studi kasus selain yang terdapat di modul)!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;
import java.util.List;

// Class Mahasiswa
class Mahasiswa {
    private String nama;
    private String nim; // Nomor Induk Mahasiswa

    // Constructor
    public Mahasiswa(String nama, String nim) {
        this.nama = nama;
        this.nim = nim;
    }

    // Getter untuk nama
    public String getNama() {
        return nama;
    }

    // Getter untuk NIM
    public String getNim() {
        return nim;
    }
}
```



TUGAS PRAKTIKUM

```
}

// Menampilkan info mahasiswa
public void tampilanInfo() {
    System.out.println("Nama Mahasiswa: " + nama + ", NIM: "
+ nim);
}
}

// Class Universitas
class Universitas {
    private String namaUniversitas;
    private List<Mahasiswa> daftarMahasiswa;

    // Constructor
    public Universitas(String namaUniversitas) {
        this.namaUniversitas = namaUniversitas;
        this.daftarMahasiswa = new ArrayList<>();
    }

    // Method untuk menambahkan mahasiswa ke universitas
    public void tambahMahasiswa(Mahasiswa mahasiswa) {
        daftarMahasiswa.add(mahasiswa);
    }

    // Method untuk menampilkan daftar mahasiswa
    public void tampilanDaftarMahasiswa() {
        System.out.println("Daftar Mahasiswa di Universitas " +
namaUniversitas + ":");

        for (Mahasiswa mahasiswa : daftarMahasiswa) {
            mahasiswa.tampilanInfo();
        }
    }
}

// Main Class
public class Main {
    public static void main(String[] args) {
        // Membuat objek Mahasiswa
```



TUGAS PRAKTIKUM

```
Mahasiswa mahasiswa1 = new Mahasiswa("Rudi", "12345");
Mahasiswa mahasiswa2 = new Mahasiswa("Siti", "67890");
Mahasiswa mahasiswa3 = new Mahasiswa("Ali", "54321");

// Membuat objek Universitas
Universitas universitas1 = new Universitas("Universitas
Indonesia");

// Menambahkan mahasiswa ke dalam universitas1
universitas1.tambahMahasiswa(mahasiswa1);
universitas1.tambahMahasiswa(mahasiswa2);

// Menampilkan daftar mahasiswa di universitas1
universitas1.tampilkanDaftarMahasiswa();

// Mahasiswa3 tetap bisa ada meskipun tidak dimasukkan
ke universitas
System.out.println("\nMahasiswa yang belum terdaftar di
universitas:");
mahasiswa3.tampilkanInfo();
}

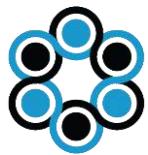
}
```

Penjelasan

Class Main:

- Membuat beberapa objek Mahasiswa dan Universitas.
- Menambahkan beberapa mahasiswa ke universitas dengan method tambahMahasiswa().
- Menampilkan daftar mahasiswa yang tergabung dalam universitas.
- Juga menunjukkan bahwa objek Mahasiswa tetap bisa eksis meskipun tidak terhubung dengan universitas.

Output

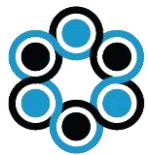


TUGAS PRAKTIKUM

The screenshot shows a terminal window titled "Praktikum p-2". The left pane displays a file tree for a project named "PRAKTIKUM P-2". The "EXPLORER" tab is selected, showing files under "Agresion" and "Pegawai1" packages, and a "Main.java" file in the root. The right pane shows the terminal output:

```
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2\Agresion"
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2\Agresion> ; if ($?) { javac Main.java } ; if ($?) { java Main }
Daftar Mahasiswa di Universitas Universitas Indonesia:
Nama Mahasiswa: Rudi, NIM: 12345
Nama Mahasiswa: Siti, NIM: 67890

Mahasiswa yang belum terdaftar di universitas:
Nama Mahasiswa: Ali, NIM: 54321
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-2\Agresion>
```



TUGAS PRAKTIKUM

Soal Praktikum

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Sanusi
NPM : 06.2023.107748
Modul : 1

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Apa yang dimaksud dengan List, Set, Queue dan Map?

Jawaban

- **List** adalah tipe *data collection* yang memungkinkan menyimpan elemen-elemen dalam urutan tertentu (berdasarkan indeks).
- **Set** adalah tipe *data collection* yang menyimpan kumpulan elemen-elemen unik, artinya tidak ada elemen duplikat dalam Set.
- **Queue** adalah tipe *data collection* yang mewakili antrian (queue) di mana elemen pertama yang dimasukkan adalah elemen pertama dikeluarkan (konsep FIFO -First-In-First-Out).
- **Map** adalah tipe *data collection* yang terdiri atas kunci (key) ke nilai (value), mirip dengan konsep *dictionary*. Setiap kunci dalam Map haruslah bersifat unik atau tidak ada duplikat key tetapi untuk setiap value boleh terdapat duplikat data

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

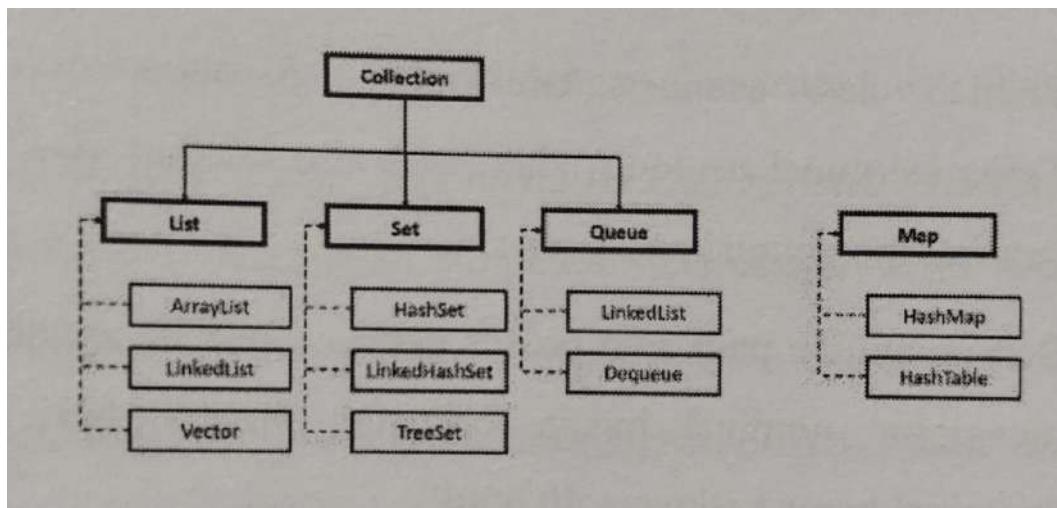
Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Perhatikan ilustrasi dibawah ini!



Mengapa Map tidak termasuk ke dalam interface Collection?

Jawaban

Map tidak tergabung dalam interface collection karena mendukung operasi yang berbeda dan menyimpan data dalam bentuk pasangan kunci (key) nilai, sedangkan interface collection berurusan dengan kumpulan element tunggal.

Source Code

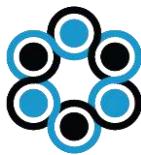
Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output



TUGAS & EVALUASI

Masukan screenshot output disini

Soal Tugas & Evaluasi

3. Jelaskan apa bedanya ArrayList dan LinkedList!

Jawaban

- ArrayList menggunakan array dinamis untuk menyimpan elemen. Ketika array penuh, ArrayList memperbesar ukurannya dengan membuat array baru yang lebih besar dan menyalin elemen lama ke dalam array baru.

Sedangkan,

LinkedList menggunakan struktur data linked list, dimana setiap elemen (node) menyimpan referensi ke dalam berikutnya (dan sebelumnya dalam DoublyLinkedList). Ini memungkinkan elemen untuk disimpan secara non-sekuensial di memori.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Sebuah bank ingin mengelola antrian pelanggan yang datang untuk mendapatkan layanan. Setiap kali seorang pelanggan selesai dilayani, pelanggan tersebut akan keluar dari antrian, dan pelanggan berikutnya akan dilayani. Bank juga ingin memungkinkan pelanggan VIP untuk masuk ke antrian di posisi tertentu. Data Collection apa yang sebaiknya digunakan sesuai studi kasus tersebut?

Jawaban

- Data collection yang cocok untuk studi kasus ini adalah ‘LinkedList’ dengan implementasi ‘Queue’, karena antrian di bank mengikuti pola FIFO (First In, First Out) dimana pelanggan pertama yang datang adalah yang pertama dilayani. Selain itu, ‘LinkedList’ memungkinkan penambahan elemen di posisi tertentu dengan mudah, sehingga memudahkan bank untuk memasukkan pelanggan VIP ke dalam antrian tanpa harus menggeser elemen lain, seperti yang terjadi pada ‘ArrayList’.

Source Code

Tulis kode program dikotak ini...

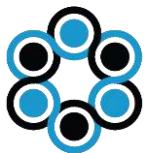
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

5. Buatlah program berdasarkan dari studi kasus soal sebelumnya!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.LinkedList;
import java.util.Queue;

public class Bank{

    private Queue<String> customerQueue = new LinkedList<>(); // Inisialisasi Queue menggunakan LinkedList

    // Menambah pelanggan reguler di akhir antrian
    public void addCustomer(String customerName) {
        customerQueue.offer(customerName); // Menambahkan pelanggan di akhir antrian
        System.out.println(customerName + " masuk antrian.");
    }

    // Melayani pelanggan terdepan di antrian
    public void serveCustomer() {
        String servedCustomer = customerQueue.poll(); // Mengambil dan menghapus pelanggan terdepan
        if (servedCustomer != null) {
            System.out.println(servedCustomer + " sedang dilayani.");
        } else {
            System.out.println("Antrian kosong, tidak ada pelanggan yang dilayani.");
        }
    }
}
```



TUGAS & EVALUASI

```
        }

    }

    // Menyisipkan pelanggan VIP di posisi tertentu dalam
    antrian
    public void addVIPCustomer(String vipCustomerName, int
position) {
        LinkedList<String> tempQueue = new
LinkedList<>(customerQueue); // Konversi Queue ke LinkedList
untuk mengakses posisi
        if (position < 0 || position > tempQueue.size()) {
            System.out.println("Posisi tidak valid. Pelanggan
VIP tidak dapat ditambahkan.");
        } else {
            tempQueue.add(position, vipCustomerName); // Menambahkan
pelanggan VIP di posisi tertentu
            customerQueue = tempQueue; // Menetapkan kembali
tempQueue sebagai customerQueue
            System.out.println(vipCustomerName + " masuk antrian
sebagai pelanggan VIP di posisi " + position);
        }
    }

    // Menampilkan seluruh antrian pelanggan
    public void showQueue() {
        System.out.println("Antrian saat ini: " +
customerQueue);
    }

    // Metode utama untuk menjalankan program
    public static void main(String[] args) {
        Bank bank = new Bank();

        // Menambah beberapa pelanggan reguler
        bank.addCustomer("Pelanggan A");
        bank.addCustomer("Pelanggan B");
        bank.addCustomer("Pelanggan C");

        // Menampilkan antrian setelah menambah pelanggan
        reguler
```



TUGAS & EVALUASI

```
bank.showQueue();

// Menambah pelanggan VIP di posisi tertentu
bank.addVIPCustomer("VIP Pelanggan X", 1);

// Menampilkan antrian setelah menambah pelanggan VIP
bank.showQueue();

// Melayani beberapa pelanggan
bank.serveCustomer();
bank.serveCustomer();

// Menampilkan antrian setelah melayani beberapa
pelanggan
bank.showQueue();
}

}
```

Penjelasan

Pada program ini mengelola antrian bank menggunakan `LinkedList` sebagai Queue yang mendukung pola FIFO (First In, First Out) dan memungkinkan penambahan pelanggan VIP diposisi tertentu. Pada Deklarasi `customerQueue` adalah Queue yang diimplementasikan menggunakan `LinkedList`, memudahkan penambahan pelanggan reguler di akhir dan VIP diposisi tertentu. Pada program ini terdapat 4 method yaitu ada `addCustomer` untuk menambahkan pelanggan reguler di akhir antrian, `addVIPCustomer` untuk menambah pelanggan VIP pada posisi tertentu, `serveCustomer` untuk mengambil dan melayani pelanggan terdepan, dan `showQueue` untuk menampilkan isi antrian saat ini.



TUGAS & EVALUASI

Output

```
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tug
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tug
as Evaluasi\Evaluasi Bab 4\" ; if ($?) { javac Bank.java } ; i
si\Evaluasi Bab 4> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Be
rorientas B masuk antrian.
i Objek\Tugas Evaluasi\Evaluasi Bab 4\" ; if ($?) { javac Bank.java
javac Bank.java } ; if (?) { java Bank }
Pelanggan A masuk antrian.
Pelanggan B masuk antrian.
Pelanggan C masuk antrian.
Antrian saat ini: [Pelanggan A, Pelanggan B, Pelanggan C]
VIP Pelanggan X masuk antrian sebagai pelanggan VIP di posisi 1
Antrian saat ini: [Pelanggan A, VIP Pelanggan X, Pelanggan B, Pelanggan C]
Pelanggan A sedang dilayani.
VIP Pelanggan X sedang dilayani.
Antrian saat ini: [Pelanggan B, Pelanggan C]
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 4> []
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

6. Vincent diminta dosenya untuk membuat sebuah sistem manajemen perpustakaan sederhana. Sistem ini harus mampu mengelola data buku, anggota perpustakaan, dan transaksi peminjaman buku. Tidak hanya satu Vincent diwajibkan menggunakan berbagai jenis data collection seperti ArrayList, HashMap, dan Queue. Buatlah program untuk Vincent!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.Queue;

class Buku {
    String judul;
    String pengarang;
    int idBuku;

    public Buku(int idBuku, String judul, String pengarang) {
        this.idBuku = idBuku;
        this.judul = judul;
        this.pengarang = pengarang;
    }

    public String toString() {
        return "ID Buku: " + idBuku + ", Judul: " + judul + ", Pengarang: " + pengarang;
    }
}
```



TUGAS & EVALUASI

```
class Anggota {  
    String nama;  
    int idAnggota;  
  
    public Anggota(int idAnggota, String nama) {  
        this.idAnggota = idAnggota;  
        this.nama = nama;  
    }  
  
    public String toString() {  
        return "ID Anggota: " + idAnggota + ", Nama: " + nama;  
    }  
}  
  
class SistemPerpustakaan {  
    ArrayList<Buku> daftarBuku = new ArrayList<>();  
    HashMap<Integer, Anggota> daftarAnggota = new HashMap<>();  
    HashMap<Integer, Queue<Buku>> transaksiPinjam = new  
    HashMap<>();  
  
    public void tambahBuku(int idBuku, String judul, String  
pengarang) {  
        Buku buku = new Buku(idBuku, judul, pengarang);  
        daftarBuku.add(buku);  
        System.out.println("Buku \'"+ judul + "\' berhasil  
ditambahkan.");  
    }  
  
    public void tambahAnggota(int idAnggota, String nama) {  
        Anggota anggota = new Anggota(idAnggota, nama);  
        daftarAnggota.put(idAnggota, anggota);  
        System.out.println("Anggota \'"+ nama + "\' berhasil  
ditambahkan.");  
    }  
  
    public void tampilkanDaftarBuku() {  
        if (daftarBuku.isEmpty()) {  
            System.out.println("Tidak ada buku dalam  
perpustakaan.");  
        }  
    }  
}
```



TUGAS & EVALUASI

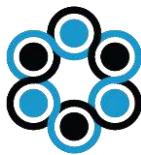
```
        } else {
            System.out.println("Daftar Buku:");
            for (Buku buku : daftarBuku) {
                System.out.println(buku);
            }
        }
    }

public void tampilkanDaftarAnggota() {
    if (daftarAnggota.isEmpty()) {
        System.out.println("Tidak ada anggota terdaftar.");
    } else {
        System.out.println("Daftar Anggota:");
        for (Anggota anggota : daftarAnggota.values()) {
            System.out.println(anggota);
        }
    }
}

public void pinjamBuku(int idAnggota, int idBuku) {
    Buku bukuPinjam = null;
    for (Buku buku : daftarBuku) {
        if (buku.idBuku == idBuku) {
            bukuPinjam = buku;
            break;
        }
    }

    if (bukuPinjam == null) {
        System.out.println("Buku dengan ID " + idBuku + " tidak ditemukan.");
        return;
    }

    if (!daftarAnggota.containsKey(idAnggota)) {
        System.out.println("Anggota dengan ID " + idAnggota +
+ " tidak ditemukan.");
        return;
    }
}
```



TUGAS & EVALUASI

```
        transaksiPinjam.putIfAbsent(idAnggota, new
LinkedList<>());
        Queue<Buku> antrianPinjam =
transaksiPinjam.get(idAnggota);
        antrianPinjam.offer(bukuPinjam);
        System.out.println("Buku \"" + bukuPinjam.judul + "\""
berhasil dipinjam oleh anggota \"" +
daftarAnggota.get(idAnggota).nama + "\".");
    }

    public void tampilkanPeminjaman(int idAnggota) {
        if (!transaksiPinjam.containsKey(idAnggota) ||
transaksiPinjam.get(idAnggota).isEmpty()) {
            System.out.println("Anggota dengan ID " + idAnggota
+ " tidak memiliki peminjaman.");
        } else {
            System.out.println("Daftar Peminjaman untuk Anggota
ID " + idAnggota + ":");

            for (Buku buku : transaksiPinjam.get(idAnggota)) {
                System.out.println(buku);
            }
        }
    }

public class Perpustakaan {
    public static void main(String[] args) {
        SistemPerpustakaan perpustakaan = new
SistemPerpustakaan();

        // Tambah buku
        perpustakaan.tambahBuku(101, "Pemrograman Java",
"Vincent");
        perpustakaan.tambahBuku(102, "Sistem Operasi", "Fauzi");

        // Tambah anggota
        perpustakaan.tambahAnggota(1, "Dicky");
        perpustakaan.tambahAnggota(2, "Agung");
```



TUGAS & EVALUASI

```
// Tampilkan daftar buku dan anggota  
perpustakaan.tampilkanDaftarBuku();  
perpustakaan.tampilkanDaftarAnggota();  
  
// Peminjaman buku  
perpustakaan.pinjamBuku(1, 101);  
perpustakaan.pinjamBuku(1, 102);  
perpustakaan.pinjamBuku(2, 101);  
  
// Tampilkan peminjaman anggota  
perpustakaan.tampilkanPeminjaman(1);  
perpustakaan.tampilkanPeminjaman(2);  
}  
}
```

Penjelasan

Pada program ini mengelola data buku, anggota, dan transaksi peminjaman diperpustakaan dengan menggunakan berbagai data collection seperti ‘ArrayList’, ‘HashMap’, dan ‘Queue’. Pada program tersebut terdapat 4 class, yaitu ada Buku, Anggota, SistemPerpustakaan, dan ManajemenPerpustakaan. Program ini juga memiliki function utama yang berbeda-beda terdapat pada method dicodingan diatas.



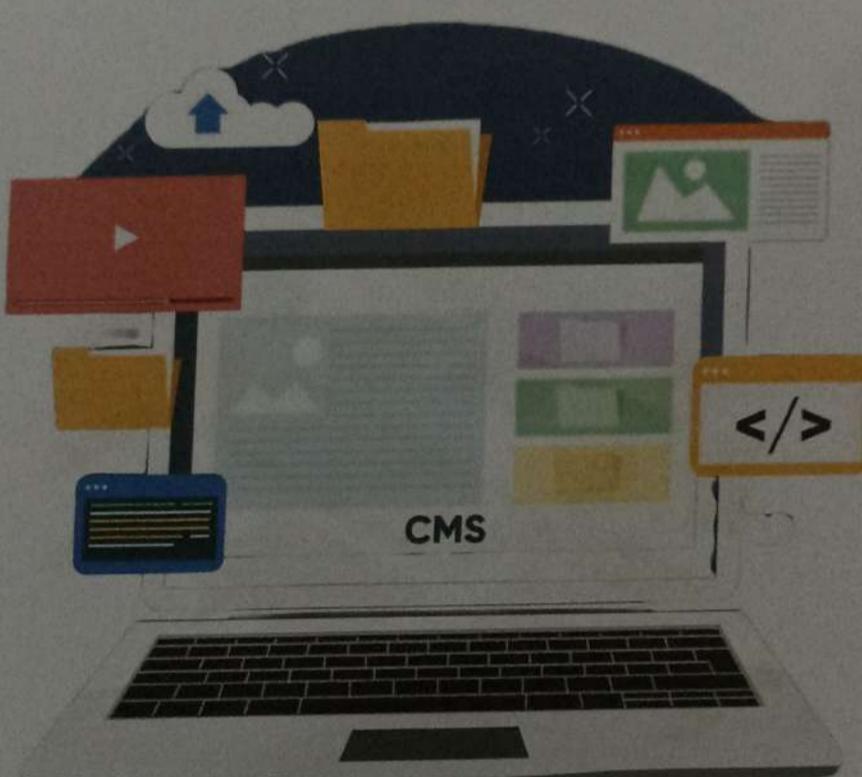
TUGAS & EVALUASI

Output

```
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 4> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 4\Perpustakaan" ; if ($?) { javac Perpustakaan.java } ; if ($?) { java Perpustakaan }
Buku "Pemrograman Java" berhasil ditambahkan.
Buku "Sistem Operasi" berhasil ditambahkan.
Anggota "Dicky" berhasil ditambahkan.
Anggota "Agung" berhasil ditambahkan.
Daftar Buku:
ID Buku: 101, Judul: Pemrograman Java, Pengarang: Vincent
ID Buku: 102, Judul: Sistem Operasi, Pengarang: Fauzi
Daftar Anggota:
ID Anggota: 1, Nama: Dicky
ID Anggota: 2, Nama: Agung
Buku "Sistem Operasi" berhasil dipinjam oleh anggota "Dicky".
Buku "Pemrograman Java" berhasil dipinjam oleh anggota "Agung".
Daftar Peminjaman untuk Anggota ID 1:
ID Buku: 101, Judul: Pemrograman Java, Pengarang: Vincent
ID Buku: 102, Judul: Sistem Operasi, Pengarang: Fauzi
Daftar Peminjaman untuk Anggota ID 2:
ID Buku: 101, Judul: Pemrograman Java, Pengarang: Vincent
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 4\Perpustakaan> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 4\Perpustakaan" ; if ($?) { javac Perpustakaan.java } ; if ($?) { java Perpustakaan }
Buku "Pemrograman Java" berhasil ditambahkan.
Buku "Sistem Operasi" berhasil ditambahkan.
Anggota "Dicky" berhasil ditambahkan.
Anggota "Agung" berhasil ditambahkan.
Daftar Buku:
ID Buku: 101, Judul: Pemrograman Java, Pengarang: Vincent
ID Buku: 102, Judul: Sistem Operasi, Pengarang: Fauzi
Daftar Anggota:
ID Anggota: 1, Nama: Dicky
ID Anggota: 2, Nama: Agung
Buku "Pemrograman Java" berhasil dipinjam oleh anggota "Dicky".
Buku "Sistem Operasi" berhasil dipinjam oleh anggota "Dicky".
Buku "Pemrograman Java" berhasil dipinjam oleh anggota "Agung".
Daftar Peminjaman untuk Anggota ID 1:
ID Buku: 101, Judul: Pemrograman Java, Pengarang: Vincent
ID Buku: 102, Judul: Sistem Operasi, Pengarang: Fauzi
```

Activate Windows
Go to Settings to activate Windows.

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Modul : 5

Fauzi.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Apa itu Access Modifier dan buatlah ilustrasi Batasan Akses dari setiap Access Modifier!

Jawaban

- Access Modifier didalam Object Oriented Programming (OOP) akan menentukan apakah Class lain dapat menggunakan field atau meminta izin untuk mengakses data dari suatu Class.

Ilustrasi Batasan Akses:

Modifier	Class	Package	Subclass	Global
Public	✓	✓	✓	✓
Protected	✓	✓	✓	X
Private	✓	X	X	X
Default	✓	✓	X	X

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Jelaskan perbedaan antara Association, Aggregation, dan Composition dalam hubungan antar class. Berikan contoh kasus untuk setiap jenis relasi tersebut.

Jawaban

- Association adalah relasi global yang istilahnya mencakup hampir semua koneksi logis (hubungan antar Class).
Contoh kasus: Seorang Guru dan Siswa memiliki hubungan asosiasi, dimana seorang guru dapat mengajar beberapa siswa, dan seorang siswa dapat diajar oleh beberapa guru. Namun, keduanya tetap dapat ada secara independen satu sama lain.
- Aggregation merupakan jenis relasi yang terjalin dalam bentuk khusus karenanya adanya pertanyaan kepemilikan dalam sebuah class.
Contoh kasus: Sebuah Perpustakaan dan Buku memiliki hubungan agregasi. Buku bisa saja ada tanpa perpustakaan, tetapi perpustakaan memiliki koleksi buku.
- Composition merupakan relasi yang terjalin apabila sebuah class harus menjadi bagian dari class lainnya untuk dapat berjalan.
Contoh kasus: Sebuah Rumah dan Ruangan memiliki hubungan composition. Jika rumah dihapus, semua ruangan yang ada di dalamnya juga dihapus, karena ruangan hanya ada sebagai bagian dari rumah.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah sebuah class AkunBank yang berisi atribut data-data yang ada di akun seorang nasabah bank dengan menerapkan konsep Encapsulation, kemudian analisis modifier apa saja yang dipakai pada setiap atribut dan tentukan method Getter dan Setter-nya!

Jawaban

Ketik jawaban disini ...

Source Code

```
public class AkunBank {  
    // Atribut data akun dengan modifier 'private' untuk  
    melindungi data  
    private String nomorRekening;  
    private String namaPemilik;  
    private double saldo;  
  
    // Constructor untuk menginisialisasi objek  
    public AkunBank(String nomorRekening, String namaPemilik,  
    double saldo) {  
        this.nomorRekening = nomorRekening;  
        this.namaPemilik = namaPemilik;  
        this.saldo = saldo;  
    }  
  
    // Getter untuk nomorRekening  
    public String getNomorRekening() {  
        return nomorRekening;  
    }  
  
    // Setter untuk nomorRekening  
    public void setNomorRekening(String nomorRekening) {  
        this.nomorRekening = nomorRekening;  
    }  
}
```



TUGAS & EVALUASI

```
// Getter untuk namaPemilik
public String getNamaPemilik() {
    return namaPemilik;
}

// Setter untuk namaPemilik
public void setNamaPemilik(String namaPemilik) {
    this.namaPemilik = namaPemilik;
}

// Getter untuk saldo
public double getSaldo() {
    return saldo;
}

// Setter untuk saldo
public void setSaldo(double saldo) {
    // Validasi untuk memastikan saldo tidak diatur ke angka negatif
    if (saldo >= 0) {
        this.saldo = saldo;
    } else {
        System.out.println("Saldo tidak bisa negatif.");
    }
}

// Method untuk menambahkan saldo
public void setorTunai(double jumlah) {
    if (jumlah > 0) {
        saldo += jumlah;
        System.out.println("Deposit berhasil. Saldo saat ini: " + saldo);
    } else {
        System.out.println("Jumlah deposit harus positif.");
    }
}

// Method untuk menarik uang
```



TUGAS & EVALUASI

```
public void tarikTunai(double jumlah) {
    if (jumlah > 0 && jumlah <= saldo) {
        saldo -= jumlah;
        System.out.println("Penarikan berhasil. Saldo saat ini: " + saldo);
    } else {
        System.out.println("Penarikan gagal. Jumlah tidak valid atau saldo tidak mencukupi.");
    }
}

public static void main(String[] args) {
    // Membuat objek AkunBank
    AkunBank akun1 = new AkunBank("5200483645" , "Fauzi" ,
5000.0);

    // Mengakses data menggunakan getter
    System.out.println("Nomor Rekening: " +
akun1.getNomorRekening());
    System.out.println("Nama Pemilik: " +
akun1.getNamaPemilik());
    System.out.println("Saldo Awal: " + akun1.getSaldo());

    // Menambahkan saldo
    akun1.setorTunai(2000.0);

    // Menarik uang
    akun1.tarikTunai(1500.0);

    // Menarik jumlah yang melebihi saldo
    akun1.tarikTunai(6000.0);

    // Mengatur nama pemilik menggunakan setter
    akun1.setNamaPemilik("Saputra");
    System.out.println("Nama Pemilik Baru: " +
akun1.getNamaPemilik());
}
```



TUGAS & EVALUASI

Penjelasan

Pada program ini saya menggunakan Access Modifer Private yang terletak pada atribut nomorRekening, namaPemilik, dan saldo. Access Modifier Public digunakan pada method getter dan setter serta method lainnya. Saya method Getter dan Setter yang berfungsi untuk mengakses nilai atribut private dari luar class dan untuk mengatur atau memperbarui nilai atribut private dari luar class. Atribut nomorRekening, namaPemilik, dan saldo bersifat private hanya bisa diakses melalui method getter dan setter.

Output

```
PS G:\Kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi Bab 5> cd "g:\Kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\" ; if ($?) { javac AkunBank.java } ; if ($?) { java AkunBank }
Nomor Rekening: 5200305279
Nama Pemilik: Fauzi
Saldo Awal: 5000.0
Deposit berhasil. Saldo saat ini: 7000.0
Nama Pemilik: Fauzi
Saldo Awal: 5000.0
Deposit berhasil. Saldo saat ini: 7000.0
Penarikan berhasil. Saldo saat ini: 5500.0
Penarikan gagal. Jumlah tidak valid atau saldo tidak mencukupi.
Nama Pemilik Baru: Fauzi
PS G:\Kuliah\PRAKTIKUM\Semester 3\Kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5>
```

Activate Windows
Go to Settings to activate Windows.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatlah dua class, Mahasiswa dan MataKuliah, yang menerapkan relasi Asosiasi Tidak Berarah. Implementasikan method yang memungkinkan seorang mahasiswa untuk mendaftar ke mata kuliah dan mata kuliah menerima pendaftaran mahasiswa.

Jawaban

Ketik jawaban disini ...

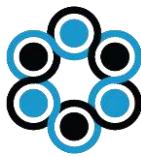
Source Code

```
import java.util.ArrayList;
import java.util.List;

class Mahasiswa {
    public String nama;
    public String npm;
    public List<MataKuliah> mataKuliahList; // Daftar mata
    kuliah yang diikuti oleh mahasiswa

    // Constructor
    public Mahasiswa(String nama, String npm) {
        this.nama = nama;
        this.npm = npm;
        this.mataKuliahList = new ArrayList<>();
    }

    // Method untuk mendaftar ke mata kuliah
    public void daftarMataKuliah(MataKuliah mataKuliah) {
        if (!mataKuliahList.contains(mataKuliah)) {
            mataKuliahList.add(mataKuliah);
            mataKuliah.tambahMahasiswa(this); // Menambahkan
            mahasiswa ini ke daftar mahasiswa pada mata kuliah
            System.out.println(nama + " berhasil mendaftar ke
            mata kuliah " + mataKuliah.getNamaMataKuliah());
        }
    }
}
```



TUGAS & EVALUASI

```
        } else {
            System.out.println(nama + " sudah terdaftar di mata
kuliah " + mataKuliah.getNamaMataKuliah());
        }
    }

// Getter untuk nama mahasiswa
public String getNama() {
    return nama;
}

// Menampilkan daftar mata kuliah yang diikuti mahasiswa
public void tampilanMataKuliah() {
    System.out.println("Mata kuliah yang diikuti " + nama +
":");
    for (MataKuliah mk : mataKuliahList) {
        System.out.println("- " + mk.getNamaMataKuliah());
    }
}
}

class MataKuliah {
    public String kodeMataKuliah;
    public String namaMataKuliah;
    public List<Mahasiswa> mahasiswaList; // Daftar mahasiswa
    yang mendaftar ke mata kuliah ini

    // Constructor
    public MataKuliah(String kodeMataKuliah, String
    namaMataKuliah) {
        this.kodeMataKuliah = kodeMataKuliah;
        this.namaMataKuliah = namaMataKuliah;
        this.mahasiswaList = new ArrayList<>();
    }

    // Method untuk menerima pendaftaran mahasiswa
    public void tambahMahasiswa(Mahasiswa mahasiswa) {
        if (!mahasiswaList.contains(mahasiswa)) {
            mahasiswaList.add(mahasiswa);
        }
    }
}
```



TUGAS & EVALUASI

```
}

// Getter untuk nama mata kuliah
public String getNamaMataKuliah() {
    return namaMataKuliah;
}

// Menampilkan daftar mahasiswa yang terdaftar di mata
kuliah
public void tampilanMahasiswa() {
    System.out.println("Daftar mahasiswa di mata kuliah " +
namaMataKuliah + ":");

    for (Mahasiswa mhs : mahasiswaList) {
        System.out.println("- " + mhs.getNama());
    }
}
}

public class Main {
    public static void main(String[] args) {
        // Membuat objek mahasiswa
        Mahasiswa mahasiswa1 = new Mahasiswa("Fauzi", "A001");
        Mahasiswa mahasiswa2 = new Mahasiswa("Saputra", "B001");

        // Membuat objek mata kuliah
        MataKuliah matkul1 = new MataKuliah("MK01", "Pemrograman
Berorientasi Objek");
        MataKuliah matkul2 = new MataKuliah("MK02", "Sistem
Operasi");

        // Mahasiswa mendaftar ke mata kuliah
        mahasiswa1.daftarMataKuliah(matkul1);
        mahasiswa1.daftarMataKuliah(matkul2);
        mahasiswa2.daftarMataKuliah(matkul1);

        // Menampilkan daftar mahasiswa yang terdaftar di setiap
mata kuliah
        matkul1.tampilanMahasiswa();
        matkul2.tampilanMahasiswa();
    }
}
```



TUGAS & EVALUASI

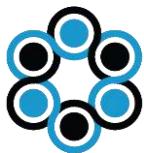
```
// Menampilkan daftar mata kuliah yang diikuti oleh  
setiap mahasiswa  
    mahasiswa1.tampilkanMataKuliah();  
    mahasiswa2.tampilkanMataKuliah();  
}  
}
```

Penjelasan

Pada program ini memiliki 2 Class yaitu Class Mahasiswa dan Class MataKuliah. Kode ini menerapkan asosiasi tidak berarah antara Class Mahasiswa dan Class MataKuliah, dimana Class Mahasiswa memiliki daftar MataKuliah yang diikuti, Class MataKuliah memiliki daftar Mahasiswa yang terdaftar. Didalam method daftarMataKuliah terdapat method tambahMahasiswa di Class MataKuliah untuk menambahkan mahasiswa ke daftar mahasiswa di mata kuliah tersebut. Dan juga di dalam method daftarMataKuliah terdapat atribut mahasiswaList berfungsi untuk daftar mahasiswa yang terdaftar di mata kuliah tersebut (bertipe List <Mahasiswa>).

Output

```
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mahasiswa dan MataKuliah> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mahasiswa dan MataKuliah" ; if ($?) { javac Main.java } ; if ($?) { java Main }  
Fauzi berhasil mendaftar ke mata kuliah Pemrograman Java  
Fauzi berhasil mendaftar ke mata kuliah Struktur Data  
Saputra berhasil mendaftar ke mata kuliah Pemrograman Java  
Daftar mahasiswa di mata kuliah Pemrograman Java:  
- Fauzi  
- Saputra  
Daftar mahasiswa di mata kuliah Struktur Data:  
- Fauzi  
Mata kuliah yang diikuti Fauzi:  
- Pemrograman Java  
- Struktur Data  
Mata kuliah yang diikuti Saputra:  
- Pemrograman Java  
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mahasiswa dan MataKuliah>■ tivate Windows
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

5. Buatlah class Mobil dan class Mesin yang menerapkan relasi Aggregation.

Dalam program tersebut, Mobil memiliki sebuah method pasangMesin() yang menghubungkan class Mesin dengan class Mobil sebelum mesin dapat dinyalakan.

Jawaban

Ketik jawaban disini ...

Source Code

```
class Mobil {  
    public String merk;  
    public Mesin mesin; // Referensi ke Mesin (Aggregation)  
  
    // Constructor  
    public Mobil(String merk) {  
        this.merk = merk;  
    }  
  
    // Method untuk memasang mesin ke mobil  
    public void pasangMesin(Mesin mesin) {  
        this.mesin = mesin;  
        System.out.println("Mesin " + mesin.getTipeMesin() + "  
telah dipasang pada mobil " + merk + ".");  
    }  
  
    // Method untuk menyalakan mesin  
    public void nyalakanMobil() {  
        if (mesin != null) {  
            System.out.println("Mobil " + merk + " siap  
dinyalakan.");  
            mesin.nyalakanMesin();  
        } else {  
        }  
    }  
}
```



TUGAS & EVALUASI

```
        System.out.println("Mobil " + merk + " tidak
memiliki mesin. Pasang mesin terlebih dahulu!");
    }

}

// Getter untuk merk mobil
public String getMerk() {
    return merk;
}

class Mesin {
    public String tipeMesin;

    // Constructor
    public Mesin(String tipeMesin) {
        this.tipeMesin = tipeMesin;
    }

    // Method untuk menyalakan mesin
    public void nyalakanMesin() {
        System.out.println("Mesin " + tipeMesin + " sedang
dinyalakan...");
    }

    // Getter untuk tipe mesin
    public String getTipeMesin() {
        return tipeMesin;
    }
}

public class Main {
    public static void main(String[] args) {
        // Membuat objek mesin
        Mesin mesin1 = new Mesin("Boxer 4 silinder 2.0L
turbocharged");
        Mesin mesin2 = new Mesin("V12 6,5L naturally
aspirated");

        // Membuat objek mobil
        Mobil mobil1 = new Mobil("Porsche");
```



TUGAS & EVALUASI

```
Mobil mobil2 = new Mobil("Lamboghirni");

// Memasang mesin pada mobil
mobil1.pasangMesin(mesin1);
mobil2.pasangMesin(mesin2);

// Mencoba menyalakan mobil
mobil1.nyalakanMobil();
mobil2.nyalakanMobil();

// Mencoba menyalakan mobil tanpa mesin
Mobil mobilTanpaMesin = new Mobil("Pindad Maung EV");
mobilTanpaMesin.nyalakanMobil();

}
```

Penjelasan

Pada program ini memiliki 2 Class, Class Mobil dan Class Mesin. Class Mobil memiliki referensi ke object Mesin, yang menunjukkan relasi Aggregation. Didalam Class Mobil memiliki method pasangMesin() digunakan untuk menghubungkan object Mesin dengan object Mobil. Program ini menunjukkan penggunaan Aggregation dengan object Mobil yang memiliki hubungan dengan object Mesin, dimana object Mesin dapat tetap ada meskipun object Mobil dihapus dari memori.

Output

```
Program Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mobil dan Mesin\ ; if ($?) { javac Main.java } ; if ($?) { java Main }
Mesin Boxer 4 silinder 2.0L turbocharged telah dipasang pada mobil Porsche.
Mesin V12 6,5L naturally aspirated telah dipasang pada mobil Lamboghirni.
Mobil Porsche siap dinyalakan.
Mesin V12 6,5L naturally aspirated telah dipasang pada mobil Lamboghirni.
Mobil Porsche siap dinyalakan.
Mobil Lamboghirni siap dinyalakan.
Mesin V12 6,5L naturally aspirated sedang dinyalakan...
Mobil Pindad Maung EV tidak memiliki mesin. Pasang mesin terlebih dahulu!
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mobil dan Mesin> cd "g:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 5\Mobil dan Mesin"
```

Activate Windows

Go to Settings to activate Windows.



PERTEMUAN 3

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Pertemuan : 3

Fauzi



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 3

RPL-MF6T7-B

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
"PertemuanX_NPM AKHIR"
Ganti "X" menjadi Pertemuan yang sedang berlangsung.
 - 2) Pada saat Praktikum, Jawablah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar "**Laporan Praktikum**".
 - 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
 - 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
 - 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
"PertemuanX_NPM AKHIR.pdf"
 - 6) Upload File **Laporan Praktikum [PDF]** pada form yang sudah disediakan.
-

TUGAS PRAKTIKUM

1. Jelaskan perbedaan antara **Override** dan **Overload**. [Wajib]
2. Jelaskan fungsi dari kata kunci super dalam konsep inheritance. [Wajib]
3. Perusahaan pengiriman "Paket Mabur" mengelola pengiriman paket domestik dan internasional. Setiap paket memerlukan penanganan berbeda, terutama terkait biaya dan regulasi.

Buatlah class **Paket** dengan atribut resi, berat, dan tujuan, lalu method **tampilkanInfoPaket()**. Kemudian buatlah sub-class **PaketDomestik** dan **PaketInternasional** yang mewarisi class **Paket**.

Pada class **PaketDomestik**, tambahkan atribut wilayah, dan opsiPengiriman(seperti same day, hemat, reguler, cargo).

Pada class **PaketInternasional**, tambahkan atribut negara, dan biayaCukai.

Override metode **tampilkanInfoPaket()** di setiap sub-class untuk menampilkan informasi tambahan sesuai dengan class-nya.

Buat objek dari class **PaketDomestik** dan **PaketInternasional**, lalu tampilkan informasi mereka menggunakan **tampilkanInfo()**. [Wajib]

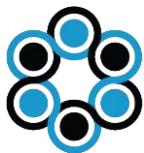


SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

4. Dari soal nomor 3, tambahkan method **hitungBiayaPengiriman()** di class induk yang menerapkan **Overload** untuk menghitung biaya pengiriman dalam berbagai situasi berikut:
 - Berdasarkan berat paket dan tarif dasar.
 - Dengan tambahan parameter untuk biaya asuransi.
 - Dengan tambahan parameter untuk potongan ongkos kirim.
5. Terdapat dua jenis karyawan di sebuah pabrik: **Karyawan Administrasi** dan **Karyawan Produksi**. Setiap karyawan memiliki atribut umum serta atribut khusus sesuai peran mereka di pabrik. Buatlah kelas dan atribut yang sesuai dalam dua package berbeda serta buatlah method untuk menampilkan informasi tentang masing-masing karyawan!

PETUNJUK:

- Package karyawan untuk class **KaryawanUmum** sebagai Class Induk.
- Package jenispegawai untuk class **KaryawanAdministrasi** dan **KaryawanProduksi** sebagai Class Anak.



TUGAS PRAKTIKUM

Soal Praktikum

1. Jelaskan perbedaan antara **Override** dan **Overload**.

Jawaban

- Override terjadi ketika sebuah subclass mendefinisikan ulang metode yang sudah ada di superclass-nya dengan implementasi yang berbeda. Metode yang *di-override* harus memiliki nama, parameter, dan tipe pengembalian yang sama persis dengan metode di superclass.
- Overload terjadi ketika dua atau lebih metode dalam satu kelas memiliki nama yang sama tetapi parameter yang berbeda (baik dalam jumlah, tipe, atau urutannya).

Source Code

Tulis kode program dikotak ini...

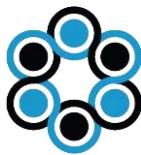
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Jelaskan fungsi dari kata kunci super dalam konsep inheritance.

Jawaban

- Saat subclass memiliki metode yang sama dengan superclass (misalnya, metode di subclass yang meng-*override* metode di superclass), super memungkinkan Anda untuk tetap memanggil versi metode dari superclass. Ini berguna ketika Anda ingin menambahkan fungsionalitas di subclass tanpa menghilangkan perilaku dari superclass.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

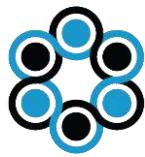
3. Perusahaan pengiriman "Paket Mabur" mengelola pengiriman paket domestik dan internasional. Setiap paket memerlukan penanganan berbeda, terutama terkait biaya dan regulasi. Buatlah class Paket dengan atribut resi, berat, dan tujuan, lalu method tampilkanInfoPaket(). Kemudian buatlah sub-class PaketDomestik dan PaketInternasional yang mewarisi class Paket. Pada class PaketDomestik, tambahkan atribut wilayah, dan opsiPengiriman(seperti same day, hemat, reguler, cargo). Pada class PaketInternasional, tambahkan atribut negara, dan biayaCukai. Override metode tampilkanInfoPaket() di setiap sub-class untuk menampilkan informasi tambahan sesuai dengan class-nya. Buat objek dari class PaketDomestik dan PaketInternasional, lalu tampilkan informasi mereka menggunakan tampilkanInfo().

Jawaban

Ketik jawaban disini ...

Source Code

```
class Paket {  
    protected String resi;  
    protected double berat; // dalam kg  
    protected String tujuan;  
  
    public Paket(String resi, double berat, String tujuan) {  
        this.resi = resi;  
        this.berat = berat;  
        this.tujuan = tujuan;  
    }  
    public void tampilkanInfoPaket() {  
        System.out.println("Resi: " + resi);  
        System.out.println("Berat: " + berat + " kg");  
        System.out.println("Tujuan: " + tujuan);  
    }  
}
```



TUGAS PRAKTIKUM

```
}

class PaketDomestik extends Paket {
    private String wilayah;
    private String opsiPengiriman;

    public PaketDomestik(String resi, double berat, String tujuan, String wilayah, String opsiPengiriman) {
        super(resi, berat, tujuan);
        this.wilayah = wilayah;
        this.opsiPengiriman = opsiPengiriman;
    }

    @Override
    public void tampilkanInfoPaket() {
        super.tampilkanInfoPaket();
        System.out.println("Wilayah: " + wilayah);
        System.out.println("Opsi Pengiriman: " +
opsiPengiriman);
    }
}

class PaketInternasional extends Paket {
    private String negara;
    private double biayaCukai;

    public PaketInternasional(String resi, double berat, String tujuan, String negara, double biayaCukai) {
        super(resi, berat, tujuan);
        this.negara = negara;
        this.biayaCukai = biayaCukai;
    }

    @Override
    public void tampilkanInfoPaket() {
        super.tampilkanInfoPaket();
        System.out.println("Negara: " + negara);
        System.out.println("Biaya Cukai: $" + biayaCukai);
    }
}

public class Main {
```



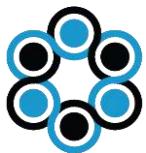
TUGAS PRAKTIKUM

```
public static void main(String[] args) {
    PaketDomestik paketDomestik = new
PaketDomestik("RESI12345", 2.5, "Padang", "Sumatera Barat",
"Same Day");
    System.out.println("Informasi Paket Domestik:");
    paketDomestik.tampilkanInfoPaket();
    double biayaDomestik =
paketDomestik.hitungBiayaPengiriman(5000);
    System.out.println("Biaya Pengiriman Domestik (tanpa
asuransi dan potongan): Rp" + biayaDomestik);
    System.out.println();

    PaketInternasional paketInternasional = new
PaketInternasional("RESI67890", 5.0, "New York", "Amerika
Serikat", 50.0);
    System.out.println("Informasi Paket Internasional:");
    paketInternasional.tampilkanInfoPaket();
    double biayaInternasional =
paketInternasional.hitungBiayaPengiriman(10000, 100, 200);
    System.out.println("Biaya Pengiriman Internasional
(dengan asuransi dan potongan): $" + biayaInternasional);
}
```

Penjelasan

Pada induk class Paket menambahkan method `tampilkanInfoPaket()`. Membahkan sub-class untuk induk class Paket yaitu PaketDosmetik dan PaketInternasional dengan 2 versi Override. Membuat objek pada class Main `paketDomestik` dan `paketInternasional`, menghitung biaya pengiriman, dan menampilkan hasilnya.



TUGAS PRAKTIKUM

Output

```
objek\Praktikum p-3\Paket\` ; if ($?) { javac Main.java } ; if ($?) { java Main }

Informasi Paket Domestik:
Resi: RESI12345
Berat: 2.5 kg
Tujuan: Padang
Wilayah: Sumatera Barat
Opsi Pengiriman: Same Day
Biaya Pengiriman Domestik (tanpa asuransi dan potongan): Rp12500.0

Informasi Paket Internasional:
Resi: RESI67890
Berat: 5.0 kg
Tujuan: New York
Negara: Amerika Serikat
Biaya Cukai: $50.0
Biaya Pengiriman Internasional (dengan asuransi dan potongan): $49900.0
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-3\Paket> [
```



TUGAS PRAKTIKUM

Soal Praktikum

4. Dari soal nomor 3, tambahkan method hitungBiayaPengiriman() di class induk yang menerapkan Overload untuk menghitung biaya pengiriman dalam berbagai situasi berikut:
 - Berdasarkan berat paket dan tarif dasar.
 - Dengan tambahan parameter untuk biaya asuransi.
 - Dengan tambahan parameter untuk potongan ongkos kirim.

Jawaban

Ketik jawaban disini ...

Source Code

```
class Paket {  
    protected String resi;  
    protected double berat; // dalam kg  
    protected String tujuan;  
  
    public Paket(String resi, double berat, String tujuan) {  
        this.resi = resi;  
        this.berat = berat;  
        this.tujuan = tujuan;  
    }  
  
    public void tampilanInfoPaket() {  
        System.out.println("Resi: " + resi);  
        System.out.println("Berat: " + berat + " kg");  
        System.out.println("Tujuan: " + tujuan);  
    }  
  
    public double hitungBiayaPengiriman(double tarifDasarPerKg)  
    {  
        return berat * tarifDasarPerKg;  
    }  
  
    public double hitungBiayaPengiriman(double tarifDasarPerKg,  
    double biayaAsuransi) {
```



TUGAS PRAKTIKUM

```
        return (berat * tarifDasarPerKg) + biayaAsuransi;
    }

    public double hitungBiayaPengiriman(double tarifDasarPerKg,
double biayaAsuransi, double potongan) {
        double totalBiaya = (berat * tarifDasarPerKg) +
biayaAsuransi;
        return totalBiaya - potongan;
    }

}

class PaketDomestik extends Paket {
    private String wilayah;
    private String opsiPengiriman;

    public PaketDomestik(String resi, double berat, String
tujuan, String wilayah, String opsiPengiriman) {
        super(resi, berat, tujuan);
        this.wilayah = wilayah;
        this.opsiPengiriman = opsiPengiriman;
    }

    @Override
    public void tampilkanInfoPaket() {
        super.tampilkanInfoPaket();
        System.out.println("Wilayah: " + wilayah);
        System.out.println("Opsi Pengiriman: " +
opsiPengiriman);
    }
}

class PaketInternasional extends Paket {
    private String negara;
    private double biayaCukai;

    public PaketInternasional(String resi, double berat, String
tujuan, String negara, double biayaCukai) {
        super(resi, berat, tujuan);
        this.negara = negara;
```



TUGAS PRAKTIKUM

```
        this.biayaCukai = biayaCukai;
    }

    @Override
    public void tampilanInfoPaket() {
        super.tampilanInfoPaket();
        System.out.println("Negara: " + negara);
        System.out.println("Biaya Cukai: $" + biayaCukai);
    }
}

public class Main {
    public static void main(String[] args) {
        // Membuat objek PaketDomestik
        PaketDomestik paketDomestik = new
PaketDomestik("RESI12345", 2.5, "Padang", "Sumatera Barat",
"Same Day");
        System.out.println("Informasi Paket Domestik:");
        paketDomestik.tampilanInfoPaket();
        double biayaDomestik =
paketDomestik.hitungBiayaPengiriman(5000);
        System.out.println("Biaya Pengiriman Domestik (tanpa
asuransi dan potongan): Rp" + biayaDomestik);
        System.out.println();

        // Membuat objek PaketInternasional
        PaketInternasional paketInternasional = new
PaketInternasional("RESI67890", 5.0, "New York", "Amerika
Serikat", 50.0);
        System.out.println("Informasi Paket Internasional:");
        paketInternasional.tampilanInfoPaket();
        double biayaInternasional =
paketInternasional.hitungBiayaPengiriman(10000, 100, 200);
        System.out.println("Biaya Pengiriman Internasional
(dengan asuransi dan potongan): $" + biayaInternasional);
    }
}
```



TUGAS PRAKTIKUM

Penjelasan

Melanjutkan soal no 3, menambah method pada induk class Paket dengan 3 versi method OverLoad. Versi pertama: Menghitung biaya berdasarkan berat dan tarif dasar per kg. Versi kedua: Menghitung biaya dengan tambahan biaya asuransi. Versi ketiga: Menghitung biaya dengan tambahan biaya asuransi dan potongan ongkos.

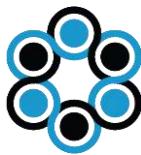
Output

```
tasi Objek\Praktikum p-3\Paket\" ; if ($?) { javac Main.java } ; if (?) { java Main }

Informasi Paket Domestik:
Resi: RESI12345
Berat: 2.5 kg
Tujuan: Padang
Wilayah: Sumatera Barat
Opsi Pengiriman: Same Day
Biaya Pengiriman Domestik (tanpa asuransi dan potongan): Rp12500.0

Informasi Paket Internasional:
Resi: RESI67890
Berat: 5.0 kg
Tujuan: New York
Negara: Amerika Serikat
Biaya Cukai: $50.0
Biaya Pengiriman Internasional (dengan asuransi dan potongan): $49900.0
Biaya Pengiriman Domestik (tanpa asuransi dan potongan): Rp12500.0

Informasi Paket Internasional:
Resi: RESI67890
Berat: 5.0 kg
Tujuan: New York
Negara: Amerika Serikat
Biaya Cukai: $50.0
Biaya Pengiriman Internasional (dengan asuransi dan potongan): $49900.0
Biaya Pengiriman Domestik (tanpa asuransi dan potongan): Rp12500.0
```



TUGAS PRAKTIKUM

Soal Praktikum

5. Terdapat dua jenis karyawan di sebuah pabrik: Karyawan Administrasi dan Karyawan Produksi. Setiap karyawan memiliki atribut umum serta atribut khusus sesuai peran mereka di pabrik. Buatlah kelas dan atribut yang sesuai dalam dua package berbeda serta buatlah method untuk menampilkan informasi tentang masing-masing karyawan! PETUNJUK: - Package karyawan untuk class KaryawanUmum sebagai Class Induk. - Package jenispegawai untuk class KaryawanAdministrasi dan KaryawanProduksi sebagai Class Anak

Jawaban

Ketik jawaban disini ...

Source Code

```
package karyawan;

public class KaryawanUmum {
    // Atribut umum untuk semua karyawan
    protected String nama;
    protected String idKaryawan;
    protected int umur;
    protected String alamat;

    // Constructor untuk KaryawanUmum
    public KaryawanUmum(String nama, String idKaryawan, int
    umur, String alamat) {
        this.nama = nama;
        this.idKaryawan = idKaryawan;
        this.umur = umur;
        this.alamat = alamat;
    }

    // Method untuk menampilkan informasi umum karyawan
    public void tampilanInfo() {
```



TUGAS PRAKTIKUM

```
        System.out.println("Nama: " + nama);
        System.out.println("ID Karyawan: " + idKaryawan);
        System.out.println("Umur: " + umur);
        System.out.println("Alamat: " + alamat);
    }
}

package jenispegawai;

import karyawan.KaryawanUmum;

public class KaryawanAdministrasi extends KaryawanUmum {
    private String departemen;
    private String posisi;

    public KaryawanAdministrasi(String nama, String id, int
umur, String departemen, String posisi) {
        super(nama, id, umur);
        this.departemen = departemen;
        this.posisi = posisi;
    }

    @Override
    public void tampilanInfo() {
        super.tampilanInfo();
        System.out.println("Departemen: " + departemen);
        System.out.println("Posisi: " + posisi);
    }
}

package jenispegawai;

import karyawan.KaryawanUmum;

public class KaryawanProduksi extends KaryawanUmum {
    private String lokasiKerja;
    private String shiftKerja;

    public KaryawanProduksi(String nama, String id, int umur,
String lokasiKerja, String shiftKerja) {
```



TUGAS PRAKTIKUM

```
super(nama, id, umur);
this.lokasiKerja = lokasiKerja;
this.shiftKerja = shiftKerja;
}

@Override
public void tampilanInfo() {
    super.tampilanInfo();
    System.out.println("Lokasi Kerja: " + lokasiKerja);
    System.out.println("Shift Kerja: " + shiftKerja);
}
}

// Main Program
import jenispegawai.KaryawanAdministrasi;
import jenispegawai.KaryawanProduksi;

import jenispegawai.KaryawanAdministrasi;
import jenispegawai.KaryawanProduksi;

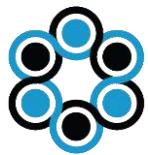
public class Main {
    public static void main(String[] args) {
        KaryawanAdministrasi admin = new
KaryawanAdministrasi("Fauzi", "07748", 20 , "Keuangan", "Staff");
        KaryawanProduksi produksi = new
KaryawanProduksi("Linda", "07749", 50 , "Pabrik", "Shift
Siang");

        System.out.println("Informasi Karyawan Administrasi:");
        admin.tampilanInfo();

        System.out.println("\nInformasi Karyawan Produksi:");
        produksi.tampilanInfo();
    }
}
```

Penjelasan

Pada program ini saya membuat folder yang bernama ‘src’, lalu berisikan folder karyawan, folder jenispegawai dan file Main program utama. class



TUGAS PRAKTIKUM

KaryawanUmum didalam folder karyawan sebagai super/induk class untuk karyawan umum dengan atribut nama, id, umur. class KaryawanProduksi dan class KaryawanAdministrasi di dalam folder jenispegawai sebagai sub class yang memiliki atribut tambahan. Pada Main program utama untuk menguji dengan membuat objek KaryawanAdministrasi dan KaryawanProduksi yaitu admin dan produksi, lalu menampilkan informasi dari masing-masing objek

Output

```
Informasi Karyawan Administrasi:  
Nama: Fauzi  
ID: 07748  
Umur: 20  
Alamat: Jl. Klampis Sachrosa No. 17  
Departemen: Keuangan  
Posisi: Staff  
  
Informasi Karyawan Produksi:  
Nama: Linda  
ID: 07749  
Umur: 59  
Alamat: Jl. Manyar Tompotika No. 11  
Lokasi Kerja: Pabrik  
Shift Kerja: Shift Siang  
PS 6:\Kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Praktikum p-3>
```

Activate Windows
Go to Settings to activate Windows.



TUGAS PRAKTIKUM

Soal Praktikum

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

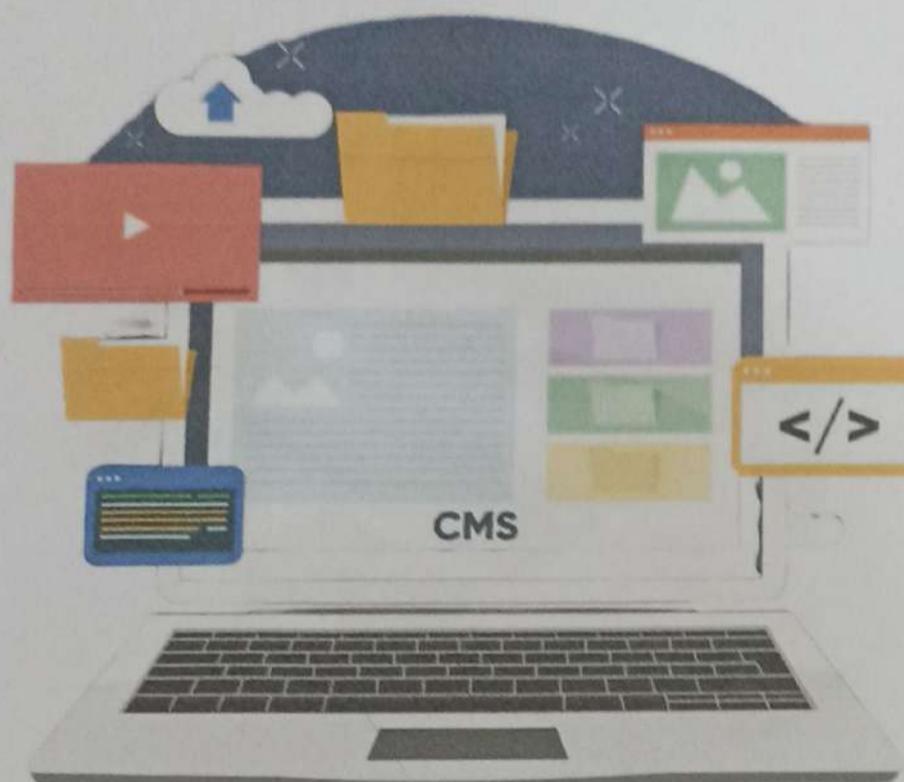
Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Modul : 6

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Apa yang dimaksud dengan Inheritance serta sebutkan macam-macam Inheritance!

Jawaban

- Inheritance merupakan suatu konsep yang mewariskan/menurunkan suatu karakteristik Class ke Class yang lain.

Inheritance dalam java terdiri dari beberapa macam yaitu Single Inheritance, Multilevel Inheritance, Hierarchical Inheritance

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

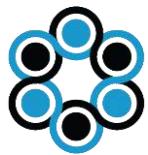
Soal Tugas & Evaluasi

2. Apa maksud dari Generalisasi dan Spesialisasi dalam Inheritance? Buatlah ilustrasi dari kedua konsep tersebut!

Jawaban

- Generalisasi adalah proses dimana kita mengidentifikasi atribut dan pelaku yang sama dari beberapa kelas yang lebih spesifik, lalu mengabstraksikannya ke dalam satu kelas induk yang lebih umum.
- Spesialisasi adalah kebalikan dari generalisasi. Spesialisasi adalah proses membuat kelas turunan dari kelas induk yang lebih umum dengan menambahkan atribut atau perilaku khusus yang hanya dimiliki oleh kelas turunan tersebut.

Konsep	Pengertian	Contoh
Generalisasi	Mengabstraksi fitur umum dari kelas-kelas spesifik ke dalam satu kelas induk yang lebih umum	Class hewan untuk harimau dan singa
Spesialisasi	Membuat kelas yang lebih spesifik dari kelas umum, dengan menambahkan atribut atau metode khusus	Class elang sebagai spesialisasi dari hewan



TUGAS & EVALUASI

Source Code

Tulis kode program dikotak ini...

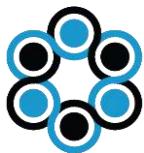
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Terdapat tiga jenis film: Animasi, Dokumenter, dan Pendek. Setiap jenis film memiliki atribut umum serta atribut khusus untuk masing-masing jenis film. Buatlah class dan atribut yang sesuai serta buatlah method untuk menampilkan informasi tentang masing-masing film!

PETUNJUK:

Class Film sebagai Class Induk;

Class Animasi, Dokumenter dan Pendek sebagai Class Anak;

Jawaban

Ketik jawaban disini ...

Source Code

```
class Film {  
    protected String judul;  
    protected String sutradara;  
    protected int durasi;  
  
    public Film(String judul, String sutradara, int durasi) {  
        this.judul = judul;  
        this.sutradara = sutradara;  
        this.durasi = durasi;  
    }  
    public void tampilanInfoFilm(){  
        System.out.println("Judul: " + judul);  
        System.out.println("Sutradara: " + sutradara);  
        System.out.println("Durasi: " + durasi + "Menit");  
    }  
}  
class Animasi extends Film {  
    private String tipeAnimasi;  
  
    public Animasi(String judul, String sutradara, int durasi,  
String tipeAnimasi){
```



TUGAS & EVALUASI

```
        super (judul, sutradara, durasi);
        this.tipeAnimasi = tipeAnimasi;
    }
    @Override
    public void tampilkanInfoFilm(){
        super.tampilkanInfoFilm();
        System.out.println("tipeAnimasi: " + tipeAnimasi);
    }
}
class Dokumenter extends Film {
    private String topikUtama;

    public Dokumenter (String judul, String sutradara, int
durasi, String topikUtama){
        super (judul, sutradara, durasi);
        this.topikUtama = topikUtama;
    }
    @Override
    public void tampilkanInfoFilm(){
        super.tampilkanInfoFilm();
        System.out.println("Topik Utama: " + topikUtama);
    }
}
class Pendek extends Film {
    private String pesanUtama;

    public Pendek(String judul, String sutradara, int durasi,
String pesanUtama) {
        super(judul, sutradara, durasi);
        this.pesanUtama = pesanUtama;
    }
    @Override
    public void tampilkanInfoFilm(){
        super.tampilkanInfoFilm();
        System.out.println("Pesanan Utama: " + pesanUtama);
    }
}
public class Main {
    public static void main(String[] args) {
```



TUGAS & EVALUASI

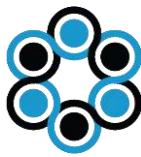
```
Film animasi = new Animasi ("Haikyuu", "Susumu  
Mitsunaka", 110, "3D" );  
System.out.println("Informasi Film Animasi: ");  
animasi.tampilkanInfoFilm();  
  
Film dokumenter = new Dokumenter("Free Solo", "Elizabeth  
Chai Vasarhelyi", 100, "Mengisahkan pendaki tebing bebas");  
System.out.println("\nInformasi Film Dokumenter: ");  
dokumenter.tampilkanInfoFilm();  
  
Film pendek = new Pendek("Sepanjang Jalan Satu Arah",  
"Ardhira Anugrah", 13, "Kesederhanaan dalam hubungan keluarga");  
System.out.println("\nInformasi Film Pendek: ");  
pendek.tampilkanInfoFilm();  
}  
}
```

Penjelasan

Pada program ini memiliki 1 class super dan 3 sub class. 1 class induk yang bernama class Film dan 3 sub class yang bernama class Animasi, Dokumenter, Pendek. Masing-masing class memiliki atribut masing-masing. Class Film yang merupakan class super/induk mempunyai atribut judul, sutradara, dan durasi. Atribut tersebut menggunakan modifier protected untuk bisa diakses oleh sub class

Output

```
Informasi Film Animasi:  
Judul: Haikyuu  
Sutradara: Susumu Mitsunaka  
Durasi: 110Menit  
tipeAnimasi: 3D  
  
Informasi Film Dokumenter:  
Judul: Free Solo  
Sutradara: Elizabeth Chai Vasarhelyi  
Durasi: 100Menit  
Topik Utama: Mengisahkan pendaki tebing bebas  
  
Informasi Film Pendek:  
Judul: Sepanjang Jalan Satu Arah  
Sutradara: Ardhira Anugrah  
Durasi: 13Menit  
Pesan Utama: Kesederhanaan dalam hubungan keluarga  
PS G:\kuliah\PRAKTIKUM\Semester 3\Pemrograman Berorientasi Objek\Tugas Evaluasi\Evaluasi Bab 6\Film>
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Apa yang dimaksud dengan Polimorfisme, Polimorfisme Statis dan Polimorfisme Dinamis?

Jawaban

- Polimorfisme adalah prinsip bahwa suatu kelas dapat memiliki banyak “*bentuk*” metode yang berbeda walaupun memiliki nama yang sama. Arti dari “*bentuk*” adalah konten yang berbeda, tetapi tipe dan parameter data berbeda.
- Polimorfisme Statis, Polimorfisme ini juga dikenal sebagai *Compile Time Polymorphism* karena method ini dipanggil selama kompilasi objek, implementasi polimorfisme ini adalah Overloading.
- Polimorfisme Dinamis, Polimorfisme ini disebut juga *Runtime Polymorphism*, hal ini karena method ini dipanggil selama objek dijalankan, implementasi dari polimorfisme ini yaitu Overriding.

Source Code

Tulis kode program dikotak ini...

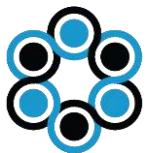
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

5. Dalam sebuah Restoran terdapat 3 jenis pegawai, yaitu **Pegawai Tetap**, **Pegawai Part-Time**, dan **Pegawai Magang**. Ketiga pegawai tersebut memiliki kriteria gaji yang berbeda, untuk pegawai tetap gajinya berasal dari gaji pokok, tunjungan khusus, dan bonus, pegawai part-time berasal dari jumlah jam kerja dan tarif per jam nya, sedangkan pegawai magang hanya berasal dari gaji tetap tanpa tunjangan dan bonus. Berdasarkan kasus tersebut Implementasikan metode **Overloading** dalam bahasa Java!

Jawaban

Ketik jawaban disini ...

Source Code

```
class Pegawai {  
    protected String nama;  
  
    public Pegawai(String nama){  
        this.nama = nama;  
    }  
    public void tampilanInfo(){  
        System.out.println("Nama Pegawai: " + nama);  
    }  
    public double hitungGaji(double gajiPokok, double tunjangan,  
double bonus){  
        return gajiPokok + tunjangan + bonus;  
    }  
    public double hitungGaji(int jamKerja, double tarifPerJam){  
        return jamKerja * tarifPerJam;  
    }  
    public double hitungGaji(double gajiTetap){  
        return gajiTetap;  
    }  
}
```



TUGAS & EVALUASI

```
    }
}

class PegawaiTetap extends Pegawai {
    private double gajiPokok;
    private double tunjangan;
    private double bonus;

    public PegawaiTetap(String nama, double gajiPokok, double tunjangan, double bonus){
        super(nama);
        this.tunjangan = tunjangan;
        this.tunjangan = tunjangan;
        this.bonus = bonus;
    }
    @Override
    public void tampilanInfo(){
        super.tampilanInfo();
        System.out.println("Pegawai Tetap: " + nama);
        System.out.println("Total Gaji: " + hitungGaji(gajiPokok, tunjangan ,bonus));
    }
}
class PegawaiPartTime extends Pegawai{
    private int jamKerja;
    private double tarifPerJam;

    public PegawaiPartTime(String nama, int jamKerja, double tarifPerJam){
        super(nama);
        this.jamKerja = jamKerja;
        this.tarifPerJam = tarifPerJam;
    }
    public void tampilanInfo(){
        super.tampilanInfo();
        System.out.println("Pegawai Part-Time: " + nama);
        System.out.println("Total Gaji: " + hitungGaji(jamKerja, tarifPerJam));
    }
}
```



TUGAS & EVALUASI

```
class PegawaiMagang extends Pegawai{
    private double gajiTetap;

    public PegawaiMagang (String nama, double gajiTetap){
        super(nama);
        this.gajiTetap = gajiTetap;
    }
    public void tampilanInfo(){
        super.tampilanInfo();
        System.out.println("Pegawai Magang: " + nama);
        System.out.println("Total Gaji: " +
hitungGaji(gajiTetap));
    }
}
public class Main {
    public static void main(String[] args) {
        Pegawai pegawaiTetap = new PegawaiTetap("Fauzi",
1500000, 500000, 250000);
        Pegawai pegawaiPartTime = new PegawaiPartTime("Fizo", 6,
200000);
        Pegawai pegawaiMagang = new PegawaiMagang("Aceel's",
3500000);

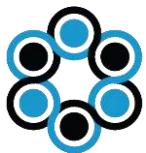
        System.out.println("Informasi Pegawai Tetap: ");
        pegawaiTetap.tampilanInfo();

        System.out.println("\nInformasi Pegawai Part-Time: ");
        pegawaiPartTime.tampilanInfo();

        System.out.println("\nInformasi Pegawai Magang");
        pegawaiMagang.tampilanInfo();
    }
}
```

Penjelasan

Pada program ini class Pegawai merupakan kelas super/induk yang mendefinisikan method hitungGaji dengan overloading untuk menghitung gaji sesuai dengan jenis pegawai. Class PegawaiTetap, PegawaiPartTime, PegawaiMagang adalah kelas sub/anak yang mewarisi class Pegawai. Masing-



TUGAS & EVALUASI

masing kelas memiliki atribut dan logika yang sesuai untuk menghitung gaji berdasarkan jenisnya. Class Main digunakan untuk menjalankan program dan menampilkan informasi dari setiap objek pegawai.

Output

```
Informasi Pegawai Tetap:
```

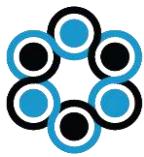
```
Nama Pegawai: Fauzi  
Pegawai Tetap: Fauzi  
Total Gaji: 750000.0
```

```
Informasi Pegawai Part-Time:
```

```
Nama Pegawai: Fizo  
Pegawai Part-Time: Fizo  
Total Gaji: 1200000.0
```

```
Informasi Pegawai Magang
```

```
Nama Pegawai: Aceel's  
Pegawai Magang: Aceel's  
Total Gaji: 3500000.0
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

6. Dalam Sistem Informasi Akademik terdapat beberapa jenis pengguna, yaitu **Admin**, **Dosen**, dan **Mahasiswa**. Ketika ingin menggunakan sistem informasi tersebut, pastinya pengguna perlu melakukan proses login. Setiap jenis pengguna memiliki proses login yang berbeda sesuai dengan peran mereka dalam system. Berdasarkan kasus tersebut Implementasikan metode **Overriding** dalam bahasa Java!

Jawaban

Ketik jawaban disini ...

Source Code

```
class Pengguna {  
    protected String username;  
    protected String password;  
  
    public Pengguna(String username, String password){  
        this.username = username;  
        this.password = password;  
    }  
    public void login(){  
        System.out.println("Proses login untuk pengguna...");  
    }  
}  
class Admin extends Pengguna {  
  
    public Admin(String username, String password){  
        super(username, password);  
    }  
    public void login(){  
        System.out.println("Proses login untuk Admin...");  
        System.out.println("Username: " + username);  
        System.out.println("Password: " + password);  
    }  
}
```



TUGAS & EVALUASI

```
        System.out.println("Anda berhasil login ke sistem!");
    }
}

class Dosen extends Pengguna {

    public Dosen(String username, String password){
        super(username, password);
    }
    public void login(){
        System.out.println("Proses login untuk Dosen...");
        System.out.println("Username: " + username);
        System.out.println("Password: " + password);
        System.out.println("Anda berhasil login ke sistem!");
    }
}
class Mahasiswa extends Pengguna {

    public Mahasiswa(String username, String password){
        super(username, password);
    }
    public void login(){
        System.out.println("Proses login untuk Mahasiswa...");
        System.out.println("Username: " + username);
        System.out.println("Password: " + password);
        System.out.println("Anda berhasil login ke sistem!");
    }
}
public class Main{
    public static void main(String[] args) {
        Pengguna admin = new Admin("Admin07748", "1230");
        Pengguna dosen = new Dosen("Dosen07748", "4560");
        Pengguna mahasiswa = new Mahasiswa("Mahasiswa07748",
        "7890");

        System.out.println("Login untuk Admin: ");
        admin.login();

        System.out.println("\nLogin untuk Dosen: ");
        dosen.login();
    }
}
```



TUGAS & EVALUASI

```
        System.out.println("\nLogin untuk Mahasiswa: ");
        mahasiswa.login();
    }
}
```

Penjelasan

Pada program ini class Pengguna adalah kelas super/induk yang memiliki atribut username dan password. Class Admin, Dosen, Masing-masing kelas turunan (Admin, Dosen, Mahasiswa) mengoverride method login() dari kelas super/induk Pengguna, yang memungkinkan setiap jenis pengguna untuk memiliki implementasi login yang sesuai dengan pengguna.

Output

```
Login untuk Admin:
Proses login untuk Admin...
Username: Admin07748
Password: 1230
Anda berhasil login ke sistem!

Login untuk Dosen:
Proses login untuk Dosen...
Username: Dosen07748
Password: 4560
Anda berhasil login ke sistem!

Login untuk Mahasiswa:
Proses login untuk Mahasiswa...
Username: Mahasiswa07748
Password: 7890
Anda berhasil login ke sistem!
```

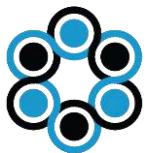
TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Modul : 7

Fauzi.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Apa yang dimaksud dengan Package?

Jawaban

- Package adalah sejenis folder untuk mengelompokkan suatu Class, Interface, dan file-file lainnya ke dalam satu folder.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Sebutkan keuntungan dan syarat-syarat dalam menerapkan konsep package!

Jawaban

Keuntungan menerapkan konsep Package adalah:

- Mengelompokkan Class dan File Java lainnya yang membuat aplikasi kita lebih mudah dipelihara.
- Mencegah terjadinya konflik penamaan yang sama pada suatu Class.
- Mengatur hak akses (enkapsulasi).

Beberapa syarat untuk menerapkan konsep Package adalah sebagai berikut:

- Pada bahasa Java penamaan package disarankan menggunakan huruf kecil.
- Penamaan package menggambarkan tujuan dari Class yang dibungkusnya.
- Penamaan tidak boleh sama (unique) antar package.
- Menuliskan keyword *package* pada baris paling atas tepat sebelum keyword *import*.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Sebutkan dan berikan Screenshot langkah-langkah dalam membuat sebuah Package baru!

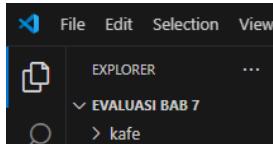
Jawaban

Membuat Package dalam Java menggunakan teks editor Visual Studio Code

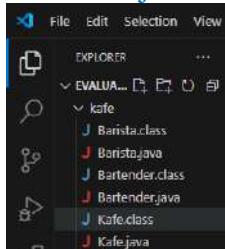
- Bikin folder baru pada explorer anda.
- Buka VS Code, lalu buka folder proyek anda dengan cara klik File > Open Folder... dan pilih proyek anda.



- Di dalam proyek anda, bikinlah folder.
- Masukkan nama folder yang mengikuti struktur package yang anda inginkan. Misalnya, jika package yang diinginkan adalah **kafe** maka buat folder bernama **kafe**.



- Setelah membuat struktur folder package, anda dapat membuat file java di dalam folder kafe.
- Klik kanan pada folder kafe (atau package yang sesuai), lalu pilih New File, dan beri nama file tersebut, misalnya Kafe.java, Barista.java, dan Bartender.java



- Pastikan deklarasi package di bagian atas file Java Anda sesuai dengan struktur folder yang sudah Anda buat.

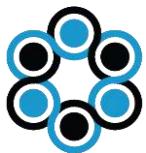
Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...



TUGAS & EVALUASI

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatlah 3 Class dalam package yang anda buat dari soal no 3 lalu akses Class yang ada pada package tersebut dengan menggunakan metode:
 - a. Metode Import All
 - b. Metode Import Each dan
 - c. Metode Fulley Qualified Name

Jawaban

Ketik jawaban disini ...

Source Code

```
package kafe;
//super class
public class Kafe {
    protected String nama;
    protected String jobDesk;

    public Kafe (String nama, String jobDesk){
        this.nama = nama;
        this.jobDesk = jobDesk;
    }
    public void tampilkanInfo(){
        System.out.println("Nama: " + nama);
        System.out.println("Job Desk: " + jobDesk);
    }
}

package kafe;
//sub-class
public class Barista extends Kafe {

    public Barista (String nama, String jobDesk){
```



TUGAS & EVALUASI

```
        super(nama, jobDesk);
    }
    @Override
    public void tampilanInfo(){
        super.tampilanInfo();
    }
}
package kafe;
//sub-class
public class Bartender extends Kafe {

    public Bartender (String nama, String jobDesk){
        super(nama, jobDesk);
    }
    @Override
    public void tampilanInfo(){
        super.tampilanInfo();
    }
}
//Metode Import All
import kafe.*;

public class Main {
    public static void main(String[] args) {
        Kafe barista = new Barista("Fauzi", "Barista");
        barista.tampilanInfo();

        Kafe bartender = new Bartender("Putra", "Bartender");
        bartender.tampilanInfo();
    }
}
//Metode Import Each
import kafe.Barista;
import kafe.Bartender;

public class Main {
    public static void main(String[] args) {
        Barista barista = new Barista("Fauzi", "Barista");
        barista.tampilanInfo();
```



TUGAS & EVALUASI

```
Bartender bartender = new Bartender("Putra",
"Bartender");
    bartender.tampilkanInfo();
}
}

//Metode Fully Qualified
public class Main {
    public static void main(String[] args) {
        kafe.Barista barista= new kafe.Barista("Fauzi",
"Barista");
        barista.tampilkanInfo();

        kafe.Bartender bartender = new kafe.Bartender("Putra",
"Bartender");
        bartender.tampilkanInfo();
    }
}
```

Penjelasan

Pada program ini saya membuat 3 class, 1 super class yang bervariabel Kafe dan 2 sub-class yang bervariabel Barista dan Bartender. Masing-masing pada class Main menggunakan import yang berbeda-beda. Yang pertama menggunakan **metode import all** cara ini akan meng-import seluruh class yang ada didalam package(menggunakan symbol *). Yang kedua menggunakan **metode import each** cara ini hanya meng-import class yang dipilih saja. Keyword: import package_name.ClassName. Yang ketiga menggunakan **metode fully qualified** metode ini memungkinkan memanggil class secara langsung tanpa perlu mengetik keyword import pada bagian atas, hanya saja penulisan kode kita akan menjadi sedikit lebih panjang jika class yang kita panggil ada didalam sebuah sub-package. Keyword package_name.ClassName object = new package_name.ClassName.

Output

```
Nama: Fauzi
Job Desk: Barista
Nama: Putra
Job Desk: Bartender
```



PERTEMUAN 4

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



**PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X**

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Pertemuan : 4

A handwritten signature in black ink, reading "Fauzi", enclosed within a thin blue rectangular border.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 4

RPL-MF8T9-A

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
PertemuanX_NPM AKHIR
Ganti "X" menjadi Pertemuan yang sedang berlangsung.
- 2) Pada saat Praktikum, Jawabanlah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar "**Laporan Praktikum**".
- 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
- 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
- 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
PertemuanX_NPM AKHIR.pdf
- 6) Upload File **Laporan Praktikum [PDF]** pada form yang sudah disediakan.

TUGAS PRAKTIKUM

1. Jelaskan apa bedanya **Abstraction** dan **Interfaces!** [wajib]
2. Jelaskan apa yang dimaksud **Exception Handling!** [wajib]
3. Budi Arye membuat sedang belajar tentang abstraction dan exception handling
Ia membuat program sebagai berikut :

```
abstract class Animal {  
    string nama;  
  
    Animal(String naMa) {  
        this.nama = nama;  
    }  
  
    abstract void bersuara();  
  
    void sleep() {  
        System.out.println(nama + " lagi tidur.");  
    }  
}
```



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

```
class Wolf implements Animal {  
    Wolf(String name) {  
        super(String name);  
    }  
  
    @Override  
    void bersuara() {  
        System.out.println(name + " says: Howl Howl");  
    }  
}
```

```
class Dog extends Wolf {  
  
    Dog(String name) {  
  
        super(name);  
    }  
  
    @Override  
  
    void bersuara() {  
  
        System.out.println(name + " says: Woof Woof");  
    }  
  
    public class Main {  
        public static void main(String[] args) {  
            try {  
                Wolf seringila = new Wolf("Budyeah");  
                seringila.bersuara();  
                seringila.sleep();  
                Dog anjing = new Dog("Aryeah");  
                anjing.bersuara();  
                anjing.sleep();  
                int[] numbers = {1, 2, 3};  
                System.out.println(numbers[5]);  
            } catch (ArrayBoundsException e) {  
                System.out.println("Error: " + e.getMessage());  
            } finally {  
                System.out.println("end.");  
            }  
        }  
    }  
}
```



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

Bantu Budi menyelesaikan kodingannya agar mendapatkan output kurang lebih sebagai berikut: [wajib]

```
Budyeah says: Woof Woof
Budyeah mending tidur.
Aryeah says: Howl Howl
Aryeah mending tidur.
Error: Index 5 out of bounds for length 3
End.
```

4. Dari soal praktikum sebelumnya kalian telah membuat class **Paket** yang di extend atau diturunkan ke kelas **PaketDomestik** dan **PaketInternasional**. Buatlah interface **Pembayaran** dengan metode **prosesPembayaran(double jumlah)**. Kemudian buatlah class **PembayaranTunai** dan **PembayaranKartu** yang mengimplementasikan interface **Pembayaran**, dengan syarat sebagai berikut:
 - Pada class **PembayaranKartu**, tambahkan atribut nomorkartu dan namapemilik.
 - Tambahkan metode bayar pada class **PaketDomestik** yang menerima objek Pembayaran sebagai parameter dan memanggil metode prosesPembayaran.
 - Buatlah objek **PembayaranTunai** dan **PembayaranKartu**, lalu gunakan metode bayar pada objek PaketDomestik untuk melakukan pembayaran.
5. Berdasarkan soal No.4 Buatlah interface **ValidasiPembayaran** dengan metode **validasi (double jumlah)**. Buatlah class **ValidasiSaldo** dan **ValidasiKartu** yang mengimplementasikan interface **ValidasiPembayaran**, dengan ketentuan sebagai berikut:



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

- Pada class **ValidasiSaldo**, tambahkan atribut saldo dan metode validasi untuk memeriksa apakah saldo mencukupi.
- Pada class **ValidasiKartu**, tambahkan atribut nomorkartu dan metode validasi untuk memeriksa validnya nomor kartu.
- Tambahkan metode **validasiPembayaran** pada class PaketDomestik yang menerima objek **ValidasiPembayaran** sebagai parameter dan memanggil metode validasi.
- Buat objek ValidasiSaldo dan ValidasiKartu, lalu gunakan metode validasiPembayaran pada objek PaketDomestik untuk melakukan validasi sebelum pembayaran.



TUGAS PRAKTIKUM

Soal Praktikum

1. Jelaskan apa bedanya Abstraction dan Interfaces!

Jawaban

- Abstraction merupakan fitur yang dapat mengabaikan fungsionalnya secara detail dengan tidak menghilangkan fungsionalnya secara umum.
Sedangkan,
- Interfaces kontrak yang hanya mendeklarasikan metode, dan kelas yang mengimplementasikan interface wajib menyediakan implementasi untuk semua metode.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Jelaskan apa yang dimaksud Exception Handling!

Jawaban

- Exception Handling merupakan permasalahan yang terjadi ketika mengeksekusi program. Pada saat terjadi sebuah kesalahan pada kode program kita, alur jalannya program akan terganggu dan bisa jadi berakhir secara tiba-tiba (crash).

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

3. Budi Arye membuat sedang belajar tentang abstraction dan exception handling Ia membuat program sebagai berikut :

```
abstract class Animal {
    String nama;

    Animal(String nama) {
        this.nama = nama;
    }

    abstract void bersuara();

    void sleep() {
        System.out.println(nama + " lagi tidur.");
    }
}

class Wolf implements Animal {
    Wolf(String name) {
        super(String name);
    }

    @Override
    void bersuara() {
        System.out.println(nama + " says: Howl Howl");
    }
}

class Dog extends Wolf {

    Dog(String name) {
        super(name);
    }

    @Override
    void bersuara() {
        System.out.println(nama + " says: Woof Woof");
    }
}

public class Main {
    public static void main(String[] args) {
        try {
            Wolf seringila = new Wolf("Budyeh");
            seringila.bersuara();
            seringila.sleep();
            Dog anjing = new Dog("Aryeah");
            anjing.bersuara();
            anjing.sleep();
            int[] numbers = {1, 2, 3};
            System.out.println(numbers[5]);
        } catch (ArrayBoundsException e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            System.out.println("end.");
        }
    }
}
```



TUGAS PRAKTIKUM

Bantu Budi menyelesaikan kodingannya agar mendapatkan output kurang lebih sebagai berikut:

```
Budyeah says: Woof Woof
Budyeah mending tidur.
Aryeah says: Howl Howl
Aryeah mending tidur.
Error: Index 5 out of bounds for length 3
End.
```

Jawaban

Ketik jawaban disini ...

Source Code

```
abstract class Animal {
    String nama;

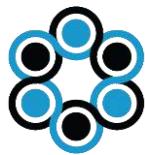
    Animal(String nama) {
        this.nama = nama;
    }

    abstract void bersuara();

    void tidur() {
        System.out.println(nama + " mending tidur");
    }
}

class Wolf extends Animal {
    Wolf(String nama) {
        super(nama);
    }

    @Override
    void bersuara() {
        System.out.println(nama + " says: Howl Howl");
    }
}
```



TUGAS PRAKTIKUM

```
    }
}

class Dog extends Wolf {
    Dog(String nama) {
        super(nama);
    }

    @Override
    void bersuara() {
        System.out.println(nama + " says: Woof Woof");
    }
}

public class Main {
    public static void main(String[] args) {
        try {
            Wolf serigala = new Dog("Budyeah");
            serigala.bersuara();
            serigala.tidur();

            Dog anjing = new Dog("Aryeah");
            anjing.bersuara();
            anjing.tidur();
            int[] numbers = {1, 2, 3};
            System.out.println(numbers[5]);

        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            System.out.println("end.");
        }
    }
}
```

Penjelasan

Kelas Wolf meng-extend Animal: Sekarang Wolf adalah subclass dari Animal, dan mengimplementasikan metode bersuara(). Perbaikan konstruktor:

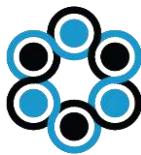


TUGAS PRAKTIKUM

Konstruktor Wolf sekarang memanggil konstruktor dari Animal dengan parameter yang benar. Perbaikan penanganan exception: Mengganti ArrayBoundsException dengan ArrayIndexOutOfBoundsException, yang lebih tepat untuk menangani kesalahan indeks array.

Output

```
Budyeah says: Woof Woof
Budyeah mending tidur
Aryeah says: Woof Woof
Aryeah mending tidur
Error: Index 5 out of bounds for length 3
end.
```



TUGAS PRAKTIKUM

Soal Praktikum

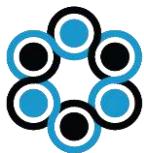
4. Dari soal praktikum sebelumnya kalian telah membuat class Paket yang di extend atau diturunkan ke kelas PaketDomestik dan PaketInternasional. Buatlah interface Pembayaran dengan metode prosesPembayaran(double jumlah). Kemudian buatlah class PembayaranTunai dan PembayaranKartu yang mengimplementasikan interface Pembayaran, dengan syarat sebagai berikut:
- Pada class PembayaranKartu, tambahkan atribut nomorkartu dan namapemilik.
 - Tambahkan metode bayar pada class PaketDomestik yang menerima objek Pembayaran sebagai parameter dan memanggil metode prosesPembayaran.
 - Buatlah objek PembayaranTunai dan PembayaranKartu, lalu gunakan metode bayar pada objek PaketDomestik untuk melakukan pembayaran.

Jawaban

Ketik jawaban disini ...

Source Code

```
class Paket {  
    protected String resi;  
    protected double berat; // dalam kg  
    protected String tujuan;  
  
    public Paket(String resi, double berat, String tujuan) {  
        this.resi = resi;  
        this.berat = berat;  
        this.tujuan = tujuan;  
    }  
    public void tampilanInfoPaket() {  
        System.out.println("Resi: " + resi);  
        System.out.println("Berat: " + berat + " kg");  
    }  
}
```



TUGAS PRAKTIKUM

```
        System.out.println("Tujuan: " + tujuan);
    }
    public double hitungBiayaPengiriman(double tarifDasarPerKg)
{
    return berat * tarifDasarPerKg;
}
public double hitungBiayaPengiriman(double tarifDasarPerKg,
double biayaAsuransi) {
    return (berat * tarifDasarPerKg) + biayaAsuransi;
}
public double hitungBiayaPengiriman(double tarifDasarPerKg,
double biayaAsuransi, double potongan) {
    double totalBiaya = (berat * tarifDasarPerKg) +
biayaAsuransi;
    return totalBiaya - potongan;
}
}
class PaketDomestik extends Paket {
    private String wilayah;
    private String opsiPengiriman;

    public PaketDomestik(String resi, double berat, String
tujuan, String wilayah, String opsiPengiriman) {
        super(resi, berat, tujuan);
        this.wilayah = wilayah;
        this.opsiPengiriman = opsiPengiriman;
    }

    @Override
    public void tampilanInfoPaket() {
        super.tampilanInfoPaket();
        System.out.println("Wilayah: " + wilayah);
        System.out.println("Opsi Pengiriman: " +
opsiPengiriman);
    }
    public void bayar (Pembayaran pembayaran, double jumlah){
        System.err.println("Melakukan pembayaran untuk paket
dosmetik...");  

        pembayaran.prosesPembayaran(jumlah);
    }
}
```



TUGAS PRAKTIKUM

```
}

class PaketInternasional extends Paket {
    private String negara;
    private double biayaCukai;

    public PaketInternasional(String resi, double berat, String tujuan, String negara, double biayaCukai) {
        super(resi, berat, tujuan);
        this.negara = negara;
        this.biayaCukai = biayaCukai;
    }

    @Override
    public void tampilanInfoPaket() {
        super.tampilanInfoPaket();
        System.out.println("Negara: " + negara);
        System.out.println("Biaya Cukai: $" + biayaCukai);
    }
}

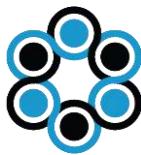
interface Pembayaran {
    void prosesPembayaran(double jumlah);
}

class PembayaranTunai implements Pembayaran{
    @Override
    public void prosesPembayaran(double jumlah){
        System.out.println("Pembayaran sebesar Rp" + jumlah + " " +
"dilakukan secara tunai.");
    }
}

class PembayaranKartu implements Pembayaran {
    private String nomorKartu;
    private String namaPemilik;

    public PembayaranKartu(String nomorKartu, String
namaPemilik){
        this.nomorKartu = nomorKartu;
        this.namaPemilik = namaPemilik;
    }

    @Override
```



TUGAS PRAKTIKUM

```
public void prosesPembayaran(double jumlah){  
    System.out.println("Pembayaran sebesar Rp" + jumlah + "  
" + "dilakukan secara kartu.");  
    System.out.println("Nomor Kartu : " + nomorKartu);  
    System.out.println("Nama Pemilik : " + namaPemilik);  
}  
}  
  
public class Main {  
    public static void main(String[] args) {  
  
        PaketDomestik paketDomestik = new  
PaketDomestik("RESI12345", 2.5, "Padang", "Sumatera Barat",  
"Same Day");  
        System.out.println("Informasi Paket Domestik :");  
        paketDomestik.tampilkanInfoPaket();  
        double biayaDomestik =  
paketDomestik.hitungBiayaPengiriman(5000);  
        System.out.println("Biaya Pengiriman Domestik (tanpa  
asuransi dan potongan): Rp" + biayaDomestik);  
        System.out.println();  
  
        PembayaranTunai pembayaranTunai = new PembayaranTunai();  
        PembayaranKartu pembayaranKartu = new  
PembayaranKartu("5800-1234-5678", "Fauzi");  
  
        paketDomestik.bayar(pembayaranTunai, biayaDomestik);  
        paketDomestik.bayar(pembayaranKartu, biayaDomestik);  
  
        PaketInternasional paketInternasional = new  
PaketInternasional("RESI67890", 5.0, "New York", "Amerika  
Serikat", 50.0);  
        System.out.println("\nInformasi Paket Internasional :");  
        paketInternasional.tampilkanInfoPaket();  
        double biayaInternasional =  
paketInternasional.hitungBiayaPengiriman(10000, 100, 200);  
        System.out.println("Biaya Pengiriman Internasional  
(dengan asuransi dan potongan): $" + biayaInternasional);  
    }  
}
```



TUGAS PRAKTIKUM

Penjelasan

Program ini melanjutkan program praktikum sebelumnya yaitu program Paket. Pada program ini hanya menambahkan metode pembayaran pada class PaketDosmetik. Menambahkan 1 interface yaitu Pembayaran dan mempunyai 2 class yang akan di implementasikan yaitu ada class PembayaranTunai dan PembayaranKartu. Sedikit tambahan method bayar() pada class PaketDosmetik .

Output

```
Biaya Pengiriman Domestik (tanpa asuransi dan potongan): Rp12500.0
Melakukan pembayaran untuk paket dosmetik...
Pembayaran sebesar Rp12500.0 dilakukan secara tunai.
Melakukan pembayaran untuk paket dosmetik...
Pembayaran sebesar Rp12500.0 dilakukan secara kartu.
Nomor Kartu : 5800-1234-5678
Nama Pemilik : Fauzi
```



TUGAS PRAKTIKUM

Soal Praktikum

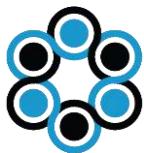
5. Berdasarkan soal No.4 Buatlah interface ValidasiPembayaran dengan metode validasi (double jumlah). Buatlah class ValidasiSaldo dan ValidasiKartu yang mengimplementasikan ValidasiPembayaran, dengan ketentuan sebagai berikut :
- Pada class ValidasiSaldo, tambahkan atribut saldo dan metode validasi untuk memeriksa apakah saldo mencukupi.
 - Pada class ValidasiKartu, tambahkan atribut nomorkartu dan metode validasi untuk memeriksa validnya nomor kartu.
 - Tambahkan metode validasiPembayaran pada class PaketDomestik yang menerima objek ValidasiPembayaran sebagai parameter dan memanggil metode validasi.
 - Buat objek ValidasiSaldo dan ValidasiKartu, lalu gunakan metode validasiPembayaran pada objek PaketDomestik untuk melakukan validasi sebelum pembayaran.

Jawaban

Ketik jawaban disini ...

Source Code

```
class Paket {  
    protected String resi;  
    protected double berat; // dalam kg  
    protected String tujuan;  
  
    public Paket(String resi, double berat, String tujuan) {  
        this.resi = resi;  
        this.berat = berat;  
        this.tujuan = tujuan;  
    }  
    public void tampilkanInfoPaket() {
```



TUGAS PRAKTIKUM

```
        System.out.println("Resi: " + resi);
        System.out.println("Berat: " + berat + " kg");
        System.out.println("Tujuan: " + tujuan);
    }
    public double hitungBiayaPengiriman(double tarifDasarPerKg)
{
    return berat * tarifDasarPerKg;
}
    public double hitungBiayaPengiriman(double tarifDasarPerKg,
double biayaAsuransi) {
        return (berat * tarifDasarPerKg) + biayaAsuransi;
    }
    public double hitungBiayaPengiriman(double tarifDasarPerKg,
double biayaAsuransi, double potongan) {
        double totalBiaya = (berat * tarifDasarPerKg) +
biayaAsuransi;
        return totalBiaya - potongan;
    }
}
class PaketDomestik extends Paket {
    private String wilayah;
    private String opsiPengiriman;

    public PaketDomestik(String resi, double berat, String
tujuan, String wilayah, String opsiPengiriman) {
        super(resi, berat, tujuan);
        this.wilayah = wilayah;
        this.opsiPengiriman = opsiPengiriman;
    }

    @Override
    public void tampilkanInfoPaket() {
        super.tampilkanInfoPaket();
        System.out.println("Wilayah: " + wilayah);
        System.out.println("Opsi Pengiriman: " +
opsiPengiriman);
    }
    public void validasiPembayaran(ValidasiPembayaran validasi,
double jumlah) {
        System.out.println("Melakukan validasi pembayaran...");
```



TUGAS PRAKTIKUM

```
        if (validasi.validasi(jumlah)) {
            System.out.println("Pembayaran dapat dilakukan.");
        } else {
            System.out.println("Pembayaran ditolak.");
        }
    }
}

class PaketInternasional extends Paket {
    private String negara;
    private double biayaCukai;

    public PaketInternasional(String resi, double berat, String tujuan, String negara, double biayaCukai) {
        super(resi, berat, tujuan);
        this.negara = negara;
        this.biayaCukai = biayaCukai;
    }

    @Override
    public void tampilanInfoPaket() {
        super.tampilanInfoPaket();
        System.out.println("Negara: " + negara);
        System.out.println("Biaya Cukai: $" + biayaCukai);
    }
}

interface ValidasiPembayaran {
    boolean validasi(double jumlah);
}

class ValidasiSaldo implements ValidasiPembayaran {
    private double saldo;

    public ValidasiSaldo(double saldo){
        this.saldo = saldo;
    }

    @Override
    public boolean validasi(double jumlah){
        if (saldo >= jumlah) {
            System.out.println("Validasi saldo berhasil: Saldo mencukupi.");
        }
    }
}
```



TUGAS PRAKTIKUM

```
        return true;
    } else {
        System.out.println("Validasi saldo gagal: Saldo
tidak mencukupi.");
        return false;
    }
}
class ValidasiKartu implements ValidasiPembayaran{
    private String nomorKartu;

    public ValidasiKartu(String nomorKartu){
        this.nomorKartu = nomorKartu;
    }
    @Override
    public boolean validasi(double jumlah){
        if (nomorKartu != null && nomorKartu.matches("\\d{16}"))
{
            System.out.println("Validasi kartu berhasil: Nomor
kartu valid.");
            return true;
        }else{
            System.out.println("Validasi kartu gagal: Nomor
kartu tidak valid.");
            return false;
        }
    }
    public class Main {
        public static void main(String[] args) {
            PaketDomestik paketDomestik = new
PaketDomestik("RESI12345", 2.5, "Padang", "Sumatera Barat",
"Same Day");
            System.out.println("Informasi Paket Domestik:");
            paketDomestik.tampilkanInfoPaket();
            double biayaDomestik =
paketDomestik.hitungBiayaPengiriman(5000);
            System.out.println("Biaya Pengiriman Domestik (tanpa
asuransi dan potongan): Rp" + biayaDomestik);
            System.out.println();
        }
    }
}
```



TUGAS PRAKTIKUM

```
    ValidasiSaldo validasiSaldo = new ValidasiSaldo(150000);
    ValidasiKartu validasiKartu = new
ValidasiKartu("5600123456789012");

    System.out.println("Validasi dengan Saldo:");
    paketDomestik.validasiPembayaran(validasiSaldo,
biayaDomestik);

    System.out.println("\nValidasi dengan Kartu:");
    paketDomestik.validasiPembayaran(validasiKartu,
biayaDomestik);

    ValidasiSaldo validasiSaldoTidakCukup = new
ValidasiSaldo(5000); // Saldo tidak cukup
    System.out.println("\nValidasi dengan Saldo (Gagal):");
    paketDomestik.validasiPembayaran(validasiSaldoTidakCukup
, biayaDomestik);

    PaketInternasional paketInternasional = new
PaketInternasional("RESI67890", 5.0, "New York", "Amerika
Serikat", 50.0);
    System.out.println("\nInformasi Paket Internasional:");
    paketInternasional.tampilkanInfoPaket();
    double biayaInternasional =
paketInternasional.hitungBiayaPengiriman(10000, 100, 200);
    System.out.println("Biaya Pengiriman Internasional
(dengan asuransi dan potongan): $" + biayaInternasional);
}
}
```

Penjelasan

Interface `ValidasiPembayaran`, dibuat interface dengan metode `validasi(double jumlah)` yang mengembalikan boolean untuk menentukan apakah validasi berhasil atau gagal. Class `ValidasiSaldo`, menambahkan atribut saldo, method `validasi()` memeriksa apakah saldo mencukupi jumlah yang harus dibayarkan. Class `ValidasiKartu`, menambahkan atribut nomorKartu, Metode `validasi` memeriksa apakah nomor kartu valid (menggunakan pola regex sederhana: harus 16 digit angka). Metode `validasiPembayaran` pada `PaketDomestik`, menerima



TUGAS PRAKTIKUM

parameter bertipe ValidasiPembayaran dan memanggil metode validasi, memberikan pesan apakah validasi berhasil atau gagal. Pada main program, Membuat objek ValidasiSaldo dan ValidasiKartu dengan data yang valid dan tidak valid, memanggil metode validasiPembayaran pada objek PaketDomestik untuk menunjukkan hasil validasi.

Output

```
Validasi dengan Saldo:  
Melakukan validasi pembayaran...  
Validasi saldo berhasil: Saldo mencukupi.  
Pembayaran dapat dilakukan.  
  
Validasi dengan Kartu:  
Melakukan validasi pembayaran...  
Validasi kartu berhasil: Nomor kartu valid.  
Pembayaran dapat dilakukan.  
  
Validasi dengan Saldo (Gagal):  
Melakukan validasi pembayaran...  
Validasi saldo gagal: Saldo tidak mencukupi.  
Pembayaran ditolak.
```



TUGAS PRAKTIKUM

Soal Praktikum

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

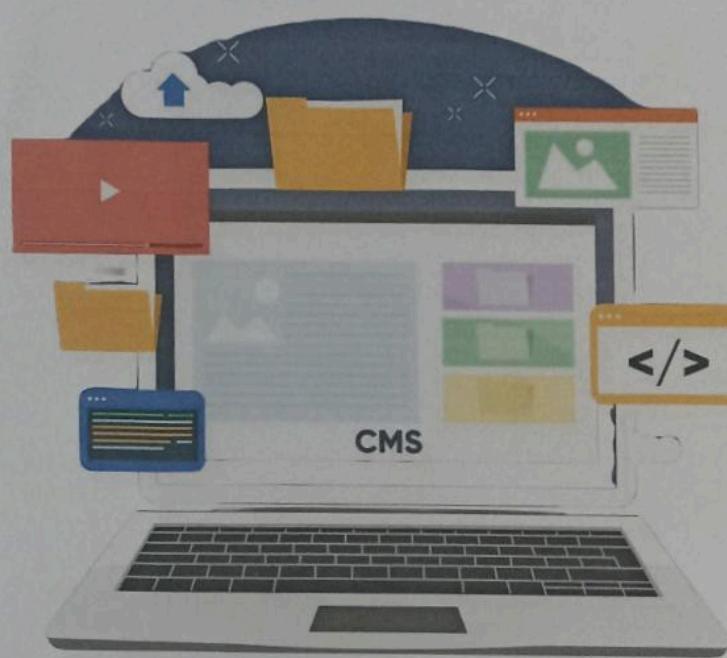
Penjelasan

Tulis Penjelasan disini ...

Output

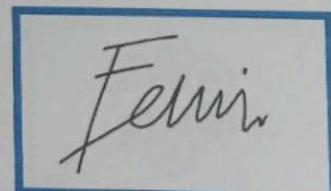
Masukan screenshot output disini

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Salutra
NPM : 06.2023.1.07748
Modul : 8



Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Jelaskan secara singkat tentang Abstraction dan Interfaces serta perbedaannya!

Jawaban

- Abstraction merupakan fitur yang dapat mengabaikan fungsionalnya secara detail dengan tidak menghilangkan fungsionalnya secara umum.
- Interfaces adalah sebuah kontrak atau template yang berisi kumpulan metode abstrak (tanpa implementasi) yang harus diimplementasikan oleh kelas yang menggunakan.

Abstract	Interfaces
Bisa berisi abstract dan non abstract method	Hanya boleh berisi abstract method
Bisa mendeklarasikan constant dan instance variable	Hanya bisa mendeklarasikan constant
Method boleh bersifat static	Method tidak boleh bersifat static
Method boleh bersifat final	Method tidak boleh bersifat final

Source Code

Tulis kode program dikotak ini...

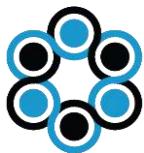
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

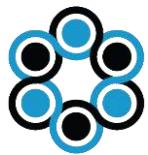
2. Buatlah class abstract dengan nama Mahasiswa, kemudian tambahkan method nama(), npm(), isMhsAktif(). Kemudian buatlah method non-abstract untuk menampilkan data mahasiswa tersebut. Pada method non abstract, gunakan operator ternary. Setelahnya, buatlah 2 sub-class dari Mahasiswa tersebut dengan nama Pagi dan Malam!

Jawaban

Ketik jawaban disini ...

Source Code

```
abstract class Mahasiswa {  
    public abstract String nama();  
    public abstract String npm();  
    public abstract boolean isMhsAktif();  
  
    public void tampilanData(){  
        System.out.println("Nama Mahasiswa : " + nama());  
        System.out.println("Nomor Pokok Mahasiswa : " + npm());  
        System.out.println("Status Mahasiswa : " + (isMhsAktif()  
? "Aktif" : "Tidak Aktif" ));  
    }  
}  
class Pagi extends Mahasiswa {  
    private String nama;  
    private String npm;  
    private boolean statusAktif;  
  
    public Pagi(String nama, String npm, boolean statusAktif) {  
        this.nama = nama;  
        this.npm = npm;
```



TUGAS & EVALUASI

```
        this.statusAktif = statusAktif;
    }
    @Override
    public String nama(){
        return nama;
    }
    @Override
    public String npm(){
        return npm;
    }
    @Override
    public boolean isMhsAktif(){
        return statusAktif;
    }
}
class Malam extends Mahasiswa{
    private String nama;
    private String npm;
    private boolean statusAktif;

    public Malam(String nama, String npm, boolean statusAktif) {
        this.nama = nama;
        this.npm = npm;
        this.statusAktif = statusAktif;
    }
    @Override
    public String nama(){
        return nama;
    }
    @Override
    public String npm(){
        return npm;
    }
    @Override
    public boolean isMhsAktif(){
        return statusAktif;
    }
}
```



TUGAS & EVALUASI

```
public class Main {  
    public static void main(String[] args) {  
        Mahasiswa pagi = new Pagi("Fauzi", "07748", true);  
        System.out.println("Data Mahasiswa Pagi ");  
        pagi.tampilanData();  
  
        Mahasiswa malam = new Malam("Saputra", "07749", false);  
        System.out.println("\nData Mahasiswa Malam ");  
        malam.tampilanData();  
    }  
}
```

Penjelasan

Pada program ini menggunakan abstract class yang bervariabel Mahasiswa,. Mempunyai method abstract nama(), npm(), isMhsAktif(). Dan menggunakan method non abstract untuk menampilkan data. Pada method tampilData/non abstract di method isMhsAktif() terdapat operator ternary untuk menggunakan if else secara singkat, jika mahasiswa aktif ia akan bernilai true, dan jika mahasiswa tidak aktif ia akan bernilai false.

Output

```
Data Mahasiswa Pagi  
Nama Mahasiswa : Fauzi  
Nomor Pokok Mahasiswa : 07748  
Status Mahasiswa : Aktif  
  
Data Mahasiswa Malam  
Nama Mahasiswa : Saputra  
Nomor Pokok Mahasiswa : 07749  
Status Mahasiswa : Tidak Aktif
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah class abstract dengan nama Transportasi, yang di dalamnya terdapat atribut nama dan kapasitas serta method abstract untuk menampilkan cara memesan dan juga menghitung tarif transportasinya. Setelahnya, buat sub-class Taksi, Bus, dan KeretaApi yang meng-implementasikan method serta atribut dari superclass. Note : Masing-masing sub-class harus memiliki perilaku pemesanan dan penghitungan tarif yang berbeda sesuai jenisnya

Jawaban

Ketik jawaban disini ...

Source Code

```
abstract class Transportasi {  
    protected String nama;  
    protected int kapasitas;  
  
    public Transportasi(String nama, int kapasitas) {  
        this.nama = nama;  
        this.kapasitas = kapasitas;  
    }  
    public abstract void caraMemesan();  
  
    public abstract double hitungTarif(int jarak, int  
penumpang);  
  
    public String getNama(){  
        return nama;  
    }  
    public int getKapasitas(){  
        return kapasitas;  
    }  
}
```



TUGAS & EVALUASI

```
class Taksi extends Transportasi {  
    private double tarifPerKm;  
  
    public Taksi(String nama , int kapasitas, double  
tarifPerKm){  
        super(nama, kapasitas);  
        this.tarifPerKm = tarifPerKm;  
    }  
    @Override  
    public void caraMemesan(){  
        System.out.println("Cara memesan" + nama + ": Hubungi  
melalui aplikasi" );  
    }  
    @Override  
    public double hitungTarif(int jarak, int penumpang){  
        return jarak * tarifPerKm;  
    }  
}  
class Bus extends Transportasi {  
    private double tarifPenumpang;  
  
    public Bus(String nama , int kapasitas, double  
tarifPenumpang){  
        super(nama, kapasitas);  
        this.tarifPenumpang = tarifPenumpang;  
    }  
    @Override  
    public void caraMemesan(){  
        System.out.println("Cara memesan" + nama + ": Datang ke  
terminal");  
    }  
    @Override  
    public double hitungTarif( int jarak, int penumpang){  
        return jarak * tarifPenumpang;  
    }  
}  
class KeretaApi extends Transportasi{  
    private double tarifKmPerPenumpang;
```



TUGAS & EVALUASI

```
public KeretaApi(String nama , int kapasitas, double tarifPerKmPerPenumpang){  
    super(nama, kapasitas);  
    this.tarifPerKmPerPenumpang = tarifPerKmPerPenumpang;  
}  
@Override  
public void caraMemesan(){  
    System.out.println("Cara memesan" + nama + ": Pesan tiket melalui web resmi");  
}  
@Override  
public double hitungTarif(int jarak, int penumpang){  
    return jarak * tarifPerKmPerPenumpang * penumpang;  
}  
}  
public class Main {  
    public static void main(String[] args) {  
  
        Transportasi taksi = new Taksi(" Rainbow Taxi", 3, 50000);  
        taksi.caraMemesan();  
        System.out.println("Tarif Taksi untuk 5 km : " + taksi.hitungTarif(3, 1));  
  
        Transportasi bus = new Bus(" Trans Jatim", 10, 5000);  
        bus.caraMemesan();  
        System.out.println("Tarif Bus untuk 10 penumpang : " + bus.hitungTarif(7, 10));  
  
        Transportasi kereta = new KeretaApi(" KA Kertajaya", 300, 1000);  
        kereta.caraMemesan();  
        System.out.println("Tarif Kereta untuk 210 penumpang dengan jarak 100 km : " + kereta.hitungTarif(50, 100));  
    }  
}
```



TUGAS & EVALUASI

Penjelasan

Class abstract Transportasi mempunyai atribut nama dan kapasitas. Method abstract caraMemesan() untuk menampilkan cara memesan transportasi. Implementasi tergantung pada subclass, method hitungTarif(int jarak, int penumpang) untuk menghitung tarif perjalanan. Setiap subclass mengimplementasikan logika penghitungan tarifnya sendiri. Pada subclass Taksi tarif dihitung berdasarkan jarak perjalanan (dalam kilometer) menggunakan atribut tarifPerKm, cara memesannya melalui aplikasi. Subclass Bus tarif dihitung berdasarkan jumlah penumpang menggunakan atribut tarifPerPenumpang, cara memesannya melalui datang ke terminal. Subclass KeretaApi tarif dihitung berdasarkan jarak perjalanan (dalam kilometer) dan jumlah penumpang menggunakan atribut tarifPerKmPerPenumpang, cara memesannya melalui website resmi.

Output

```
Cara memesan Rainbow Taxi: Hubungi melalui aplikasi
Tarif Taksi untuk 5 km : 150000.0 IDR

Cara memesan Trans Jatim: Datang ke terminal
Tarif Bus untuk 10 penumpang : 35000.0 IDR

Cara memesan KA Kertajaya: Pesan tiket melalui web resmi
Tarif Kereta untuk 210 penumpang dengan jarak 100 km : 5000000.0 IDR
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatlah class interface Pembayaran dan Transportasi berdasarkan studi kasus dari soal no 3. Kemudian implementasikan kedua class tersebut pada class Taksi, Bus, dan KeretaApi yang baru! PETUNJUK :
- Interface Transportasi berisi method void bergerak() dan double hitungTarif().
 - Interface Pembayaran berisi method void bayar().

Jawaban

Ketik jawaban disini ...

Source Code

```
interface Transportasi {  
    void bergerak();  
    double hitungTarif(int jarak, int penumpang);  
}  
interface Pembayaran {  
    void bayar(double jumlah);  
}  
class Taksi implements Transportasi, Pembayaran {  
    private String nama;  
    private double tarifPerKm;  
  
    public Taksi(String nama, double tarifPerKm) {  
        this.nama = nama;  
        this.tarifPerKm = tarifPerKm;  
    }  
  
    @Override  
    public void bergerak() {  
        System.out.println(nama + " bergerak di jalan raya.");  
    }  
  
    @Override
```



TUGAS & EVALUASI

```
public double hitungTarif(int jarak, int penumpang) {
    return jarak * tarifPerKm;
}

@Override
public void bayar(double jumlah) {
    System.out.println("Pembayaran sebesar " + jumlah + " IDR untuk " + nama + " berhasil.");
}
}

class Bus implements Transportasi, Pembayaran {
    private String nama;
    private double tarifPerPenumpang;

    public Bus(String nama, double tarifPerPenumpang) {
        this.nama = nama;
        this.tarifPerPenumpang = tarifPerPenumpang;
    }

    @Override
    public void bergerak() {
        System.out.println(nama + " bergerak di jalur bus.");
    }

    @Override
    public double hitungTarif(int jarak, int penumpang) {
        return penumpang * tarifPerPenumpang;
    }

    @Override
    public void bayar(double jumlah) {
        System.out.println("Pembayaran sebesar " + jumlah + " IDR untuk " + nama + " berhasil.");
    }
}

class KeretaApi implements Transportasi, Pembayaran {
    private String nama;
    private double tarifPerKmPerPenumpang;
```



TUGAS & EVALUASI

```
public KeretaApi(String nama, double tarifPerKmPerPenumpang)
{
    this.nama = nama;
    this.tarifPerKmPerPenumpang = tarifPerKmPerPenumpang;
}

@Override
public void bergerak() {
    System.out.println(nama + " bergerak di atas rel kereta
api.");
}

@Override
public double hitungTarif(int jarak, int penumpang) {
    return jarak * tarifPerKmPerPenumpang * penumpang;
}

@Override
public void bayar(double jumlah) {
    System.out.println("Pembayaran sebesar " + jumlah + "
IDR untuk " + nama + " berhasil.");
}
}

public class Main {
    public static void main(String[] args) {
        Taksi taksi = new Taksi("Taksi Redbird", 5000);
        Bus bus = new Bus("Bus Joyoboyo", 10000);
        KeretaApi kereta = new KeretaApi("KA Jayabaya", 15000);

        taksi.bergerak();
        double tarifTaksi = taksi.hitungTarif(4, 1);
        System.out.println("Tarif Taksi: " + tarifTaksi + "
IDR");
        taksi.bayar(tarifTaksi);
        System.out.println();

        bus.bergerak();
        double tarifBus = bus.hitungTarif(10, 20);
        System.out.println("Tarif Bus: " + tarifBus + " IDR");
```



TUGAS & EVALUASI

```
bus.bayar(tarifBus);
System.out.println();

kereta.bergerak();
double tarifKereta = kereta.hitungTarif(100, 300);
System.out.println("Tarif Kereta: " + tarifKereta + "
IDR");
kereta.bayar(tarifKereta);
}
}
```

Penjelasan

Interface Transportasi berisi method void bergerak() menunjukkan cara transportasi bergerak, contoh : jalan raya atau rel kereta. method double hitungTarif(int jarak, int penumpang) untuk menghitung tarif perjalanan sesuai jarak dan jumlah penumpang. Interface Pembayaran berisi method void bayar (double jumlah) untuk melakukan pembayaran setelah tarif dihitung. Terdapat 3 sub-class yaitu Taksi, Bus, KeretaApi. Masing-masing sub class mengimplementasikan kedua interface Transportasi dan Pembayaran.

Output

```
Taksi Redbird bergerak di jalan raya.
Tarif Taksi: 20000.0 IDR
Pembayaran sebesar 20000.0 IDR untuk Taksi Redbird berhasil.

Bus Joyoboyo bergerak di jalur bus.
Tarif Bus: 200000.0 IDR
Pembayaran sebesar 200000.0 IDR untuk Bus Joyoboyo berhasil.

KA Jayabaya bergerak di atas rel kereta api.
Tarif Kereta: 4.5E8 IDR
Pembayaran sebesar 4.5E8 IDR untuk KA Jayabaya berhasil.
```

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Safutra
NPM : 06.2023.1.07746
Modul : g

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Jelaskan apa yang dimaksud dengan Exception Handling dan sebutkan tujuannya!

Jawaban

Exception Handling adalah mekanisme dalam pemrograman yang digunakan untuk menangani kesalahan (error) atau pengecualian (exception) yang terjadi selama program berjalan.

Tujuan utama dari Exception Handling adalah untuk :

- Mengidentifikasi situasi tidak normal
- Menghindari penghentian mendadak
- Memberikan informasi yang berguna
- Menangani kondisi tidak terduga
- Meningkatkan keamanan

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Sebutkan kategori Exception Handling beserta penjelasannya

Jawaban

- Checked Exception

Exception yang terjadi pada saat compile program (Compile Time Exceptions).

- Unchecked Exception

Exception ini terjadi pada saat program menemukan sebuah bug seperti kesalahan logika program.

- User-defined Exception

Exception yang dibuat sendiri oleh pengembang dengan cara mendefinisikan sebuah kelas khusus yang memperluas (extend) kelas Exception atau RuntimeException.

Source Code

Tulis kode program dikotak ini...

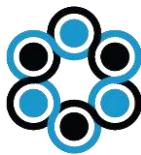
1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah kode program yang menunjukkan implementasi kategori Exception Handling berikut :
- Checked Exception : Buat program yang menangani FileNotFoundException dan IOException.
 - Unchecked Exception : Buat program yang menunjukkan penanganan ArithmeticException dan NullPointerException

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.io.*;

public class ExceptionHandlingExample {
    public static void main(String[] args) {
        // Bagian Checked Exception
        System.out.println("==== Checked Exception ====");
        try {
            readFile("nonexistentfile.txt");
        } catch (FileNotFoundException e) {
            System.out.println("FileNotFoundException ditangani:
File tidak ditemukan.");
        } catch (IOException e) {
            System.out.println("IOException ditangani: Terjadi
kesalahan saat membaca file.");
        }

        // Bagian Unchecked Exception
        System.out.println("\n==== Unchecked Exception ====");
        try {
            int result = divide(10, 0);
            System.out.println("Hasil pembagian: " + result);
        }
    }
}
```



TUGAS & EVALUASI

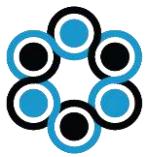
```
        } catch (ArithmaticException e) {
            System.out.println("ArithmaticException ditangani:
Tidak bisa membagi dengan nol.");
        }

        try {
            String text = null;
            printTextLength(text);
        } catch (NullPointerException e) {
            System.out.println("NullPointerException ditangani:
Objek bernilai null.");
        }
    }

    // Checked Exception: FileNotFoundException dan IOException
    public static void readFile(String fileName) throws
FileNotFoundException, IOException {
        FileReader fileReader = new FileReader(fileName);
        BufferedReader bufferedReader = new
BufferedReader(fileReader);
        System.out.println("Isi file: " +
bufferedReader.readLine());
        bufferedReader.close();
    }

    // Unchecked Exception: ArithmaticException
    public static int divide(int a, int b) {
        return a / b;
    }

    // Unchecked Exception: NullPointerException
    public static void printTextLength(String text) {
        System.out.println("Panjang teks: " + text.length());
    }
}
```



TUGAS & EVALUASI

Penjelasan

Checked Exception method `readFile(String fileName)` ini mencoba membaca file menggunakan `FileReader` dan `BufferedReader`, jika file tidak ditemukan akan memicu `FileNotFoundException`, jika terjadi kesalahan saat membaca file akan memicu `IOException`. Unchecked Exception method `divide(int a, int b)` jika `b` bernilai nol akan memicu `ArithmetricException`. Method `printTextLength (String text)` jika `text` bernilai null akan memicu `NullPointerException`.

Output

```
==== Checked Exception ====
FileNotFoundException ditangani: File tidak ditemukan.

==== Unchecked Exception ====
ArithmetricException ditangani: Tidak bisa membagi dengan nol.
NullPointerException ditangani: Objek bernilai null.
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buat sebuah program yang melakukan operasi pembagian angka dengan penanganan menggunakan blok try-catchfinally. Program ini harus dapat menangani input yang tidak valid, termasuk pembagian dengan nol.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.Scanner;

public class DivisionWithLoop {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int jumlahOperasi = 2;

        System.out.println("Program Operasi Pembagian (Maksimal " + jumlahOperasi + " kali)");

        for (int i = 1; i <= jumlahOperasi; i++) {
            System.out.println("\nOperasi ke-" + i + ":");

            try {
                System.out.print("Masukkan angka pertama: ");
                int num1 = Integer.parseInt(scanner.nextLine());

                System.out.print("Masukkan angka kedua: ");
                int num2 = Integer.parseInt(scanner.nextLine());

                int hasil = num1 / num2;
                System.out.println("Hasil pembagian: " + hasil);
            } catch (NumberFormatException e) {

```



TUGAS & EVALUASI

```
        System.out.println("Error: Input harus berupa
angka.");}

    } catch (ArithmetricException e) {
        System.out.println("Error: Tidak bisa membagi
dengan nol.");

    } finally {
        System.out.println("Operasi ke-" + i + "
selesai.");
    }
}

System.out.println("\nProgram selesai. Terima kasih
telah menggunakan program ini.");
scanner.close();
}
}
```

Penjelasan

Blok try mengambil input dari pengguna menggunakan Scanner, melalukan operasi pembagian pembagian dengan angka yang dimasukkan. Di program ini mempunyai 2 blok catch NumberFormatException dan ArithmetricException untuk jika input pengguna bukan angka, jika pengguna mencoba membagi dengan nol. Blok finally bagian ini selalu dijalankan, terlepas dari apakah terjadi pengecualian atau tidak. Saya menambahkan looping for digunakan untuk mengulangi operasi. Membahkan variable jumlahOperasi untuk mengulangi operasi pembagian hingga ke-2 sesuai kebutuhan.

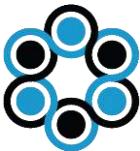
Output

```
Program Operasi Pembagian (Maksimal 2 kali)

Operasi ke-1:
Masukkan angka pertama: 10
Masukkan angka kedua: 2
Hasil pembagian: 5
Operasi ke-1 selesai.

Operasi ke-2:
Masukkan angka pertama: 10
Masukkan angka kedua: dua
Error: Input harus berupa angka.
Operasi ke-2 selesai.

Program selesai. Terima kasih telah menggunakan program ini.
```



TUGAS & EVALUASI



PERTEMUAN 5

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Pertemuan : 5

Fauzi.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 5

RPL-MF10T11-B

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
PertemuanX_NPM AKHIR
Ganti "X" menjadi Pertemuan yang sedang berlangsung.
 - 2) Pada saat Praktikum, Jawabanlah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar "**Laporan Praktikum**".
 - 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
 - 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
 - 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
PertemuanX_NPM AKHIR.pdf
 - 6) Simpan **Project Java** kalian kedalam bentuk **RAR**, nama file RAR mengikuti nama project "***PertemuanX_NPM AKHIR.rar***".
 - 7) Upload File **Laporan Praktikum [PDF]** dan **Project Java [RAR]** pada form yang sudah disediakan.
-

TUGAS PRAKTIKUM

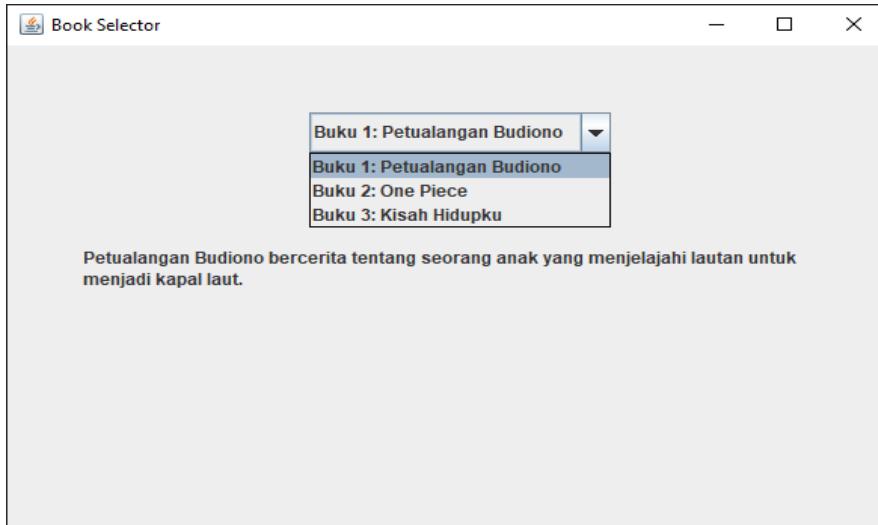
1. Dalam mengelola file, apa saja yang bisa di lakukan oleh Java Files? [Wajib]
2. Buatlah program Java File untuk membuat file 'Biodata.txt'! [Wajib]
3. Jelaskan apa itu **Jframe** pada **Component Java Swing** serta berikan contohnya penggunaannya dengan ukuran frame 600 x 400 [Wajib]
4. Buatlah program Java Swing yang memiliki input berupa JTextField untuk memasukkan username dan JPasswordField untuk memasukkan password. Tambahkan tombol "Login". Ketika tombol "Login" ditekan, lakukan validasi dengan ketentuan berikut:
 - Panjang password minimal 8 karakter.
 - Password harus mengandung huruf besar, huruf kecil, dan angka.
 - Jika validasi berhasil, tampilkan pesan "Login Berhasil" di JLabel.
 - Jika validasi gagal, tampilkan pesan "Password tidak valid" di JLabel.
5. Melanjutkan dari soal no 3, tambahkan komponen JComboBox dengan beberapa pilihan judul buku. Ketika pengguna memilih judul buku dari

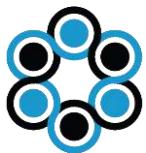


SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

JComboBox, maka akan menampilkan sinopsis dari judul buku yang dipilih.

Seperti contoh berikut:





TUGAS PRAKTIKUM

Soal Praktikum

1. Dalam mengelola file, apa saja yang bisa di lakukan oleh Java Files?

Jawaban

- Bisa membuat, membaca, memperbarui, dan menghapus file.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Buatlah program Java File untuk membuat file ‘Biodata.txt’!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

public class BiodataFile {
    public static void main(String[] args) {

        String fileName = "Biodata.txt";

        String biodata = """
            Nama      : Fauzi Saputra
            Umur      : 20 tahun
            Alamat    : Jl. Menur Pumpungan no 11
            Pekerjaan: Mahasiswa Semester 3
            Hobi      : Membaca, Coding, Badminton
            """;

        try {
            File file = new File(fileName);

            if (file.exists()) {
                System.out.println("File sudah ada. Menimpa file
lama...");}
            } else {
                System.out.println("Membuat file baru...");}
        }
    }
```



TUGAS PRAKTIKUM

```
        FileWriter fileWriter = new FileWriter(file);
        BufferedWriter bufferedWriter = new
BufferedWriter(fileWriter);

        bufferedWriter.write(biodata);

        bufferedWriter.close();

        System.out.println("File '" + fileName + "' berhasil
dibuat dan diisi!");
    } catch (IOException e) {
        System.out.println("Terjadi kesalahan saat membuat
file: " + e.getMessage());
    }
}
}
```

Penjelasan

File, membuat objek file untuk nama file Biodata.txt. FileWriter, digunakan untuk menulis teks ke file. BufferedWriter, membungkus FileWriter untuk meningkatkan efisiensi penulisan. Exception Handling, menggunakan blok try-catch untuk menangani potensi error seperti kegagalan membuat file.

Output

```
Membuat file baru...
File 'Biodata.txt' berhasil dibuat dan diisi!
```

```
File sudah ada. Menimpa file lama...
File 'Biodata.txt' berhasil dibuat dan diisi!
```

```
Biodata.txt - Notepad
File Edit Format View Help
Nama : Fauzi Saputra
Umur : 20 tahun
Alamat : Jl. Menur Pumpungan no 11
Pekerjaan: Mahasiswa Semester 3
Hobi : Membaca, Coding, Badminton
```



TUGAS PRAKTIKUM

Soal Praktikum

3. Jelaskan apa itu JFrame pada Component Java Swing serta berikan contohnya penggunaannya dengan ukuran frame 600 x 400

Jawaban

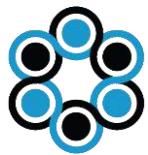
- JFrame adalah bingkai utama dalam aplikasi Swing. Ini mewakili jendela yang dapat diubah ukurannya dan berfungsi sebagai wadah untuk menampung komponen-komponen lain seperti tombol, panel, dan lainnya.

Source Code

```
import javax.swing.*;
import java.awt.*;
public class JavaSwing {
    JFrame frame = new JFrame();
    public void JavaSwing(){
        frame.setSize (600, 400);
        frame.setTitle("ini adalah judul");
        frame.setVisible(true);
    }
    public static void main(String[] args) {
        JavaSwing java = new JavaSwing();
        java.JavaSwing();
    }
}
```

Penjelasan

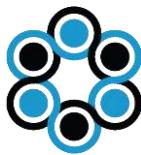
Pada kode diatas, kita akan menggunakan frame berukuran 600 x 400



TUGAS PRAKTIKUM

Output

```
JFrame > Javaswing.java > Javaswing
1 import javax.swing.*;
2 import java.awt.*;
3 public class JavaSwing {
4     JFrame frame = new JFrame();
5     public void JavaSwing(){
6         frame.setSize (width:600, height:480);
7         frame.setTitle(title:"ini adalah judul");
8         frame.setVisible(b:true);
9     }
10    Run|Debug
11    public static void main(String[] args) {
12        JavaSwing java = new JavaSwing();
13        java.JavaSwing();
14    }
}
```



TUGAS PRAKTIKUM

Soal Praktikum

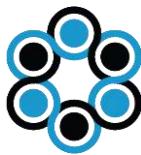
4. Buatlah program Java Swing yang memiliki input berupa JTextField untuk memasukkan username dan JPasswordField untuk memasukkan password. Tambahkan tombol "Login". Ketika tombol "Login" ditekan, lakukan validasi dengan ketentuan berikut:
 - Panjang password minimal 8 karakter.
 - Password harus mengandung huruf besar, huruf kecil, dan angka.
 - Jika validasi berhasil, tampilkan pesan "Login Berhasil" di JLabel.
 - Jika validasi gagal, tampilkan pesan "Password tidak valid" di JLabel.

Jawaban

-

Source Code

```
import javax.swing.*;  
import java.awt.*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class LoginValidationApp {  
    public static void main(String[] args) {  
  
        JFrame frame = new JFrame("Login Validation");  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.setSize(400, 200);  
        frame.setLayout(new GridLayout(4, 2, 10, 10));  
  
        JLabel lblUsername = new JLabel("Username:");  
        JTextField txtUsername = new JTextField();  
  
        JLabel lblPassword = new JLabel("Password:");  
        JPasswordField txtPassword = new JPasswordField();
```



TUGAS PRAKTIKUM

```
 JButton btnLogin = new JButton("Login");
 JLabel lblMessage = new JLabel("", SwingConstants.CENTER);

 frame.add(lblUsername);
 frame.add(txtUsername);

 frame.add(lblPassword);
 frame.add(txtPassword);

 frame.add(new JLabel());
 frame.add(btnLogin);

 frame.add(new JLabel());
 frame.add(lblMessage);

 btnLogin.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e) {
         String username = txtUsername.getText();
         String password = new String(txtPassword.getPassword());

         if (validatePassword(password)) {
             lblMessage.setText("Login Berhasil");
             lblMessage.setForeground(Color.GREEN);
         } else {
             lblMessage.setText("Password tidak valid");
             lblMessage.setForeground(Color.RED);
         }
     }
 });

 private boolean validatePassword(String password) {
     if (password.length() < 8) {
         return false;
     }
     boolean hasUpperCase = false;
     boolean hasLowerCase = false;
     boolean hasDigit = false;
```



TUGAS PRAKTIKUM

```
        for (char c : password.toCharArray()) {
            if (Character.isUpperCase(c)) {
                hasUpperCase = true;
            }
            if (Character.isLowerCase(c)) {
                hasLowerCase = true;
            }
            if (Character.isDigit(c)) {
                hasDigit = true;
            }
        }

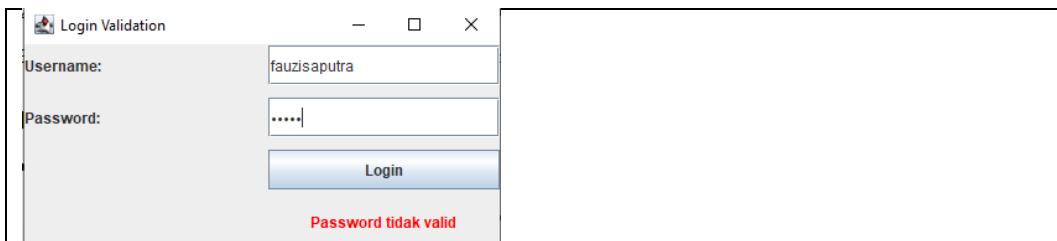
        return hasUpperCase && hasLowerCase && hasDigit;
    }
});

frame.setLocationRelativeTo(null);
frame.setVisible(true);
}
}
```

Penjelasan

Panjang password harus minimal 8 karakter. Password harus memiliki minimal satu huruf besar, satu huruf kecil, dan satu angka. Validasi dilakukan di fungsi validatePassword(). JTextField: Untuk memasukkan username. JPasswordField: Untuk memasukkan password. JButton: Tombol "Login". JLabel: Menampilkan pesan hasil validasi.

Output





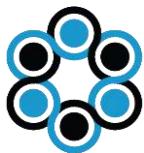
TUGAS PRAKTIKUM

Login Validation

Username: fauzisaputra

Password:
Login

Login Berhasil



TUGAS PRAKTIKUM

Soal Praktikum

5. Melanjutkan dari soal no 3, tambahkan komponen JComboBox dengan beberapa pilihan judul buku. Ketika pengguna memilih judul buku dari JComboBox, maka akan menampilkan sinopsis dari judul buku yang dipilih. Seperti contoh berikut:

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

Ketik soal disini ...

Jawaban

Ketik jawaban disini ...

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

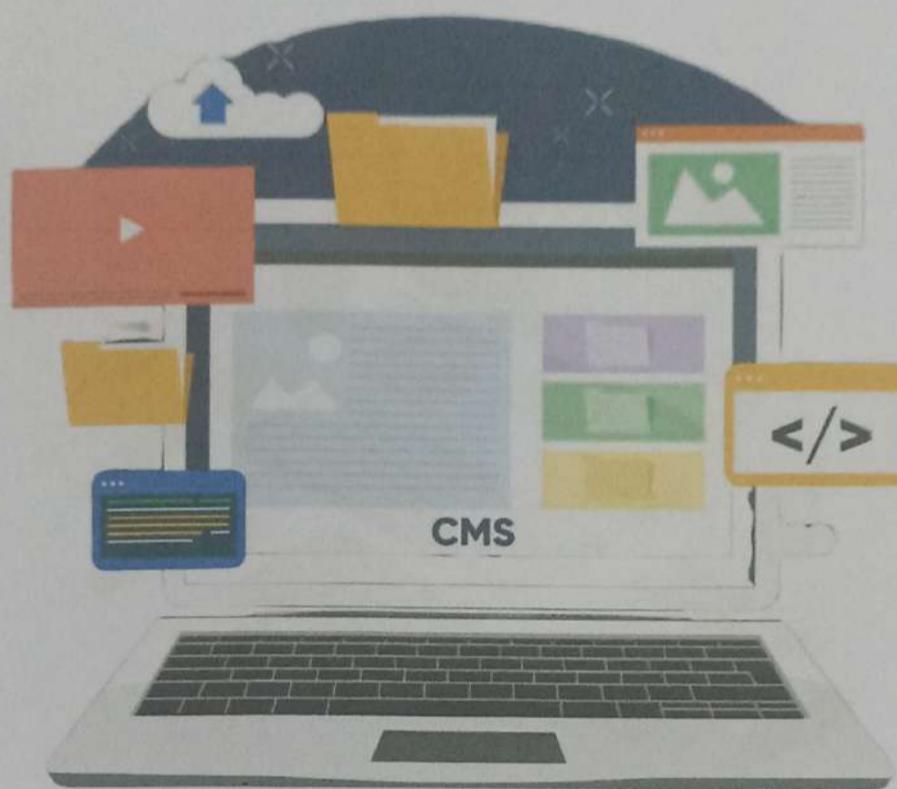
Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini

TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Modul : 10

Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Buatlah ArrayList dari Class Transaksi dengan atribut tanggal (string) dan nominal (float) lalu inputkan 5 object ke dalam ArrayList tersebut!

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;

class Transaksi {
    String tanggal;
    float nominal;

    public Transaksi(String tanggal, float nominal) {
        this.tanggal = tanggal;
        this.nominal = nominal;
    }
    @Override
    public String toString() {
        return "Tanggal: " + tanggal + ", Nominal: " + nominal;
    }
}

public class Main {
    public static void main(String[] args) {
        ArrayList<Transaksi> daftarTransaksi = new
        ArrayList<>();

        daftarTransaksi.add (new Transaksi("2024-12-06",
100000));
        daftarTransaksi.add (new Transaksi("2024-12-07
",150000));
        daftarTransaksi.add (new Transaksi("2024-12-08
",200000));
```



TUGAS & EVALUASI

```
daftarTransaksi.add (new Transaksi("2024-12-09  
",250000));  
daftarTransaksi.add (new Transaksi("2024-12-10  
",300000));  
  
System.out.println(" Daftar Transaksi: ");  
for (Transaksi tranksasi : daftarTransaksi){  
    System.out.println(tranksasi);  
}  
}  
}
```

Penjelasan

Class Transaksi memiliki atribut tanggal (string) dan nominal (float), menggunakan method override `toString()` untuk menampilkan informasi objek dengan format yang mudah dibaca. Pada main program, membuat `ArrayList<Transaksi>` untuk menyimpan daftar transaksi, menambahkan 5 object Transaksi ke dalam `ArrayList` menggunakan method `add`, menggunakan for-each untuk mencetak detail setiap transaksi di dalam `ArrayList`.

Output

```
Daftar Transaksi:  
Tanggal: 2024-12-06, Nominal: 100000.0  
Tanggal: 2024-12-07 , Nominal: 150000.0  
Tanggal: 2024-12-08 , Nominal: 200000.0  
Tanggal: 2024-12-09 , Nominal: 250000.0  
Tanggal: 2024-12-10 , Nominal: 300000.0
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Lalu buatkan method “exportToTxt” yang dimana method tersebut akan:
- Membuat file text baru
 - Menuliskan semua value yang terdapat pada ArrayList ke dalam file baru yang telah dibuat,

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;

class Transaksi {
    String tanggal;
    float nominal;

    public Transaksi(String tanggal, float nominal) {
        this.tanggal = tanggal;
        this.nominal = nominal;
    }
    @Override
    public String toString() {
        return "Tanggal: " + tanggal + ", Nominal: " + nominal;
    }
}
public class Main {

    public static void exportToTxt(ArrayList<Transaksi>
daftarTransaksi, String fileName){
```

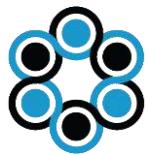


TUGAS & EVALUASI

```
try (BufferedWriter writer = new BufferedWriter( new  
FileWriter(fileName))){  
    for (Transaksi transaksi : daftarTransaksi) {  
        writer.write(transaksi.toString());  
        writer.newLine();  
    }  
    System.out.println("Data berhasil diekspor ke file:  
" + fileName);  
} catch (IOException e){  
    System.out.println("Terjadi kesalahan saat menulis  
file: " + e.getMessage());  
}  
}  
  
public static void main(String[] args) {  
    ArrayList<Transaksi> daftarTransaksi = new  
ArrayList<>();  
  
    daftarTransaksi.add (new Transaksi("2024-12-06",  
100000));  
    daftarTransaksi.add (new Transaksi("2024-12-07",  
150000));  
    daftarTransaksi.add (new Transaksi("2024-12-08",  
200000));  
    daftarTransaksi.add (new Transaksi("2024-12-09",  
250000));  
    daftarTransaksi.add (new Transaksi("2024-12-10",  
300000));  
  
    exportToTxt(daftarTransaksi, "transaksi.txt");  
}  
}
```

Penjelasan

Method `exportToTxt` berparameter `ArrayList<Transaksi>` dan nama file (`String`), menggunakan class `BufferedWriter` untuk menulis ke file teks, setiap object transaksi dalam `ArrayList` diubah menjadi string menggunakan `toString()` dan dituliskan ke file, jika terjadi kesalahan saat menulis file, akan ditangani oleh



TUGAS & EVALUASI

blok catch. Main method memanggil exportTxt untuk menyimpan file data file ke bernama transaksi.txt.

Output

```
Data berhasil dieksport ke file: transaksi.txt
Main.java    transaksi.txt
transaksi.txt
1 Tanggal: 2024-12-06, Nominal: 100000.0
2 Tanggal: 2024-12-07 , Nominal: 150000.0
3 Tanggal: 2024-12-08 , Nominal: 200000.0
4 Tanggal: 2024-12-09 , Nominal: 250000.0
5 Tanggal: 2024-12-10 , Nominal: 300000.0
```



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah sebuah program untuk menampilkan sebuah gambar yang ada pada directory anda menggunakan Image Class!

Jawaban

Ketik jawaban disini ...

Source Code

```
import javax.swing.ImageIcon;  
  
import javax.swing.JFrame;  
  
import javax.swing.JLabel;  
  
import javax.swing.SwingUtilities;  
  
  
public class Main {  
  
    public static void main(String[] args) {  
  
        SwingUtilities.invokeLater(() -> {  
  
            JFrame frame = new JFrame ("Display Image");  
  
            frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);  
  
            frame.setSize(600, 400);  
  
  
            ImageIcon     imageIcon      =      new      ImageIcon  
("G:/kuliah/PRAKTIKUM/Semester 3/Pemrograman Berorientasi Objek/Tugas  
Evaluasi/Evaluasi Bab 10/image/fz.png");  
  
  
            JLabel label = new JLabel (imageIcon);  
    }  
}
```



TUGAS & EVALUASI

```
frame.getContentPane().add(label);

frame.setVisible(true);

});

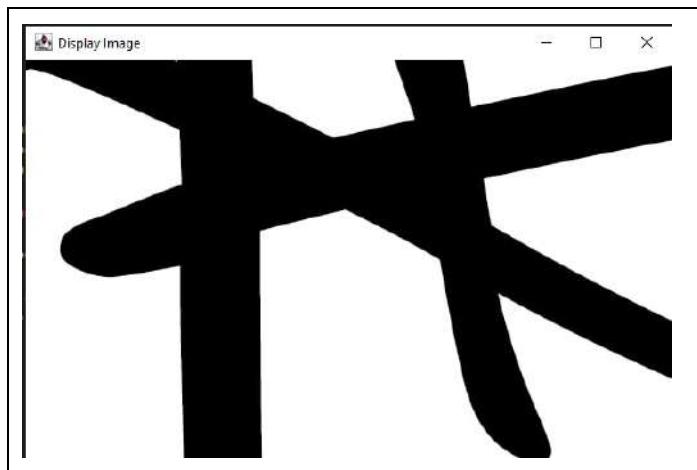
}

}
```

Penjelasan

javax.swing.ImageIcon digunakan untuk memuat gambar dari file sistem ke dalam aplikasi, javax.swing.JFrame merupakan sebuah jendela utama (frame) tempat elemen GUI ditampilkan, javax.swing.JLabel komponen untuk menampilkan teks atau gambar, javax.swing.SwingUtilities kelas utility untuk menangani pekerjaan GUI di Event Dispatch Thread (EDT). SwingUtilities.invokeLater logika untuk membuat GUI. frame.getContentPane().add(label) untuk menambahkan (yang berisi gambar) ke dalam konten utama dari jendela frame. frame.setVisible(true) untuk menampilkan jendela GUI di layar. Tanpa perintah ini jendela tidak akan terlihat meskipun telah dibuat.

Output





TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatlah program untuk menampilkan sebuah persegi panjang menggunakan BufferedImage Class!

Jawaban

Ketik jawaban disini ...

Source Code

```
import javax.swing.*;  
import java.awt.*;  
import java.awt.image	BufferedImage;  
  
public class persegiPanjang {  
  
    public static void main(String[] args) {  
        SwingUtilities.invokeLater(() -> {  
  
            JFrame frame = new JFrame("Gambar Persegi Panjang");  
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
            frame.setSize(500, 500);  
  
            JPanel panel = new JPanel() {  
                @Override  
                protected void paintComponent(Graphics g) {
```



TUGAS & EVALUASI

```
super.paintComponent(g);

BufferedImage image = new BufferedImage(500, 500,
BufferedImage.TYPE_INT_ARGB);

Graphics2D g2d = image.createGraphics();

g2d.setColor(Color.BLUE);
g2d.fillRect(100, 100, 300, 200); // x, y, width, height

g2d.setColor(Color.BLACK);
g2d.drawRect(100, 100, 300, 200);

g.drawImage(image, 0, 0, null);

g2d.dispose();

}

};

frame.add(panel);

frame.setVisible(true);

});

}

}
```

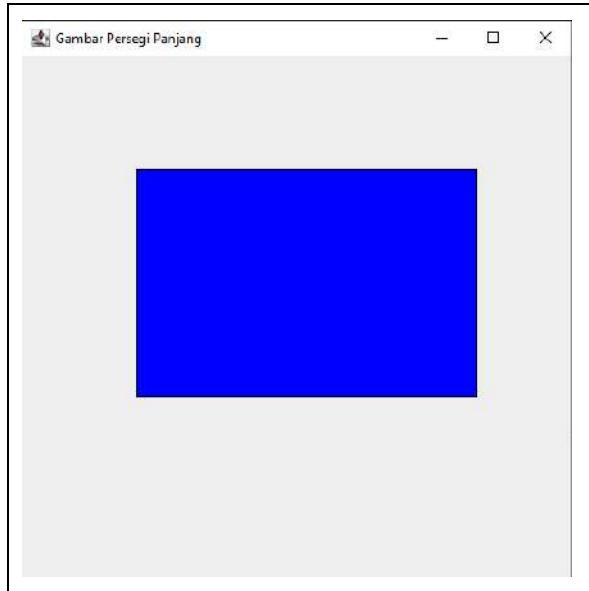


TUGAS & EVALUASI

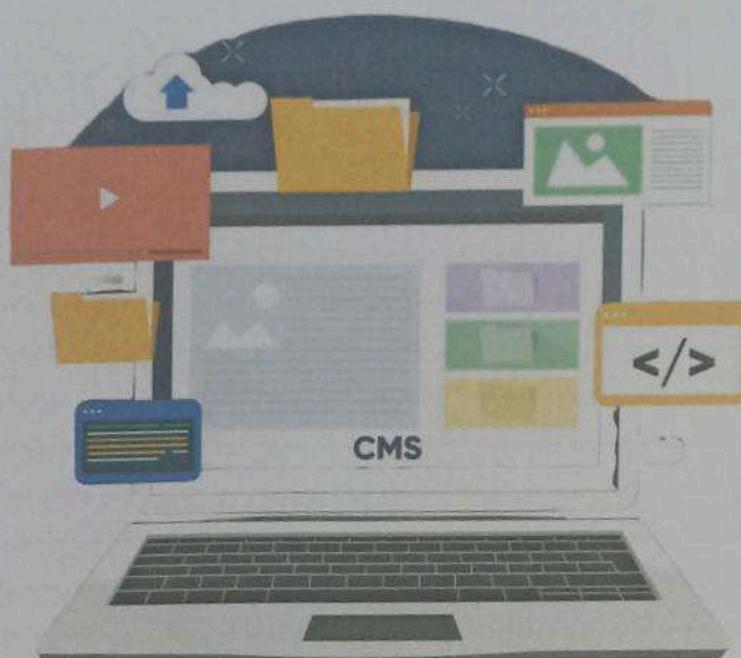
Penjelasan

BufferedImage digunakan untuk menggambar secara programatik. Class ini membuat gambar di memori. BufferedImage.TYPE_INT_ARGB memungkinkan gambar memiliki transparasi dan warna 32-bit. Method fillReact(x, y, width, height) digunakan untuk menggambar persegi panjang yang terisi penuh dengan warna. Method drawReact(x, y, width, height) digunakan untuk menggambarkan outline persegi panjang. Class Graphics2D digunakan untuk menggambar ke dalam objek BufferedImage. Class ini adalah subclass dari Graphics yang memberikan kontrol lebih baik atas rendering. Method paintComponent di-override untuk menggambar gambar menggunakan Graphics.

Output

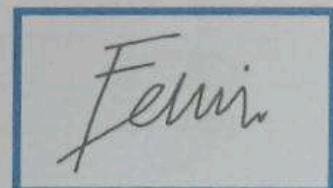


TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07748
Modul : 11



Fauzi



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Apa perbedaan utama antara AWT dan Swing dalam Java, Jelaskan kelebihan dan kekurangan masing-masing?

Jawaban

- AWT memungkinkan komponen-komponen AWT bergantung pada elemen native dari sistem operasi (OS) tempat aplikasi dijalankan, sehingga sering disebut sebagai heavyweight components.

Kelebihan AWT: mudah digunakan untuk membuat GUI dasar, menggunakan komponen native OS sehingga lebih ringan dibandingkan Swing.

Kekurangan AWT: komponen GUI tergantung pada sistem operasi, sehingga tampilan dan perilaku bisa berbeda-beda, tidak memiliki komponen GUI kompleks seperti tabel atau pohon.

- Swing menyediakan komponen-komponen grafis untuk untuk membangun antarmuka pengguna dalam aplikasi Java.

Kelebihan Swing: menyediakan lebih banyak komponen GUI seperti JTable, JTree, JTabbedPane, dll, tampilan GUI sama di semua platform karena tidak bergantung pada elemen native OS.

Kekurangan Swing: sedikit lebih lambat dibandingkan AWT, karena rendering dilakukan oleh JVM, aplikasi berbasis Swing cenderung membutuhkan lebih banyak memori.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class



TUGAS & EVALUASI

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Bagaimana cara kerja event listener dalam Java Swing?

Jawaban

- Listener harus didaftarkan ke sumber event menggunakan metode seperti addActionListener, addMouseListener, atau addKeyListener.
- Setelah listener didaftarkan, Java akan memantau interaksi pengguna.
- Ketika event terjadi, metode di listener (seperti actionPerformed) dipanggil secara otomatis untuk menangani event tersebut.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Apa fungsi dari JFrame dan bagaimana cara menggunakannya untuk membuat jendela utama dalam aplikasi Swing?

Jawaban

JFrame berfungsi sebagai wadah untuk menampung komponen-komponen lain seperti tombol, panel, dll.

Cara menggunakan JFrame dalam aplikasi Swing bisa dilihat contoh program dibawah ini

Source Code

```
Import javax.swing.*;
Import java.awt.*;

public class JavaSwing {
    JFrame frame = new JFrame();
    public void JavaSwing(){
        frame.setSize (400,600);
        frame.setTitle("Hello World!");
        frame.setVisible(true);
    }
    public static void main(String[] args) {
        JavaSwing java = new JavaSwing();
        Java.JavaSwing();
    }
}
```

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatlah sebuah aplikasi sederhana menggunakan Java Swing yang memiliki JFrame dengan ukuran 400x600, JLabel untuk menampilkan teks, JTextField untuk input teks, dan JButton yang ketika diklik akan menampilkan teks dari JTextField ke JLabel.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.awt.event.*;
import javax.swing.*;
import java.awt.event.ActionListener;

public class MahasiswaApp {
    @SuppressWarnings("Convert2Lambda")
    public static void main(String[] args) {
        JFrame frame = new JFrame("Aplikasi Data Mahasiswa");
        frame.setSize(400, 600);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);

        JLabel nameLabel = new JLabel("Nama Mahasiswa:");
        nameLabel.setBounds(50, 50, 300, 30);
        frame.add(nameLabel);

        JTextField nameField = new JTextField();
        nameField.setBounds(50, 90, 300, 30);
        frame.add(nameField);

        JLabel npmLabel = new JLabel("NPM Mahasiswa:");
        npmLabel.setBounds(50, 140, 300, 30);

        frame.add(npmLabel);
```



TUGAS & EVALUASI

```
JTextField npmField = new JTextField();
npmField.setBounds(50, 180, 300, 30);
frame.add/npmField;

JButton button = new JButton("Tampilkan Data");
button.setBounds(150, 230, 120, 30);
frame.add(button);

JLabel outputNameLabel = new JLabel("", SwingConstants.CENTER);
outputNameLabel.setBounds(50, 280, 300, 30);
frame.add(outputNameLabel);

JLabel outputNpmLabel = new JLabel("", SwingConstants.CENTER);
outputNpmLabel.setBounds(50, 320, 300, 30);
frame.add(outputNpmLabel);

button.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String nama = nameField.getText();
        String npm = npmField.getText();
        outputNameLabel.setText("Nama: " + nama);
        outputNpmLabel.setText("NPM: " + npm);
    }
});
frame.setVisible(true);
}
```

Penjelasan

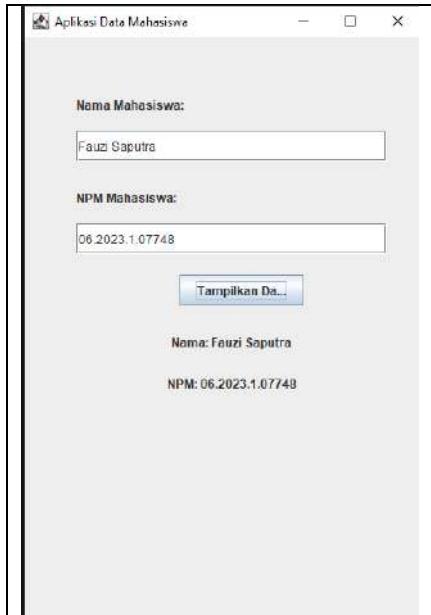
JFrame membuat jendela utama dengan ukuran 400x600, method setLayout(null) digunakan untuk manual position komponen. JLabel lokasi dan ukuran diatur menggunakan method setBounds (x, y, width, height). JTextField sebagai area input teks dari pengguna. JButton ketika diklik, mengambil teks dari JTextField dan menampilkan teks tersebut di JLabel. ActionListener untuk



TUGAS & EVALUASI

menangkap aksi tombol. `@SuppressWarnings("Convert2Lambda")` digunakan untuk menginstruksikan kompiler agar tidak memberikan peringatan terkait peluang mengganti anonymous inner class menjadi lambda expression.

Output

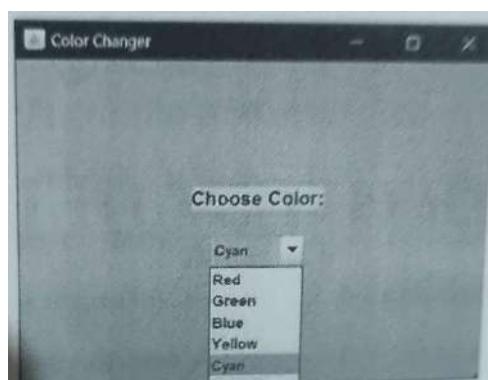




TUGAS & EVALUASI

Soal Tugas & Evaluasi

5. Buatlah sebuah aplikasi Java Swing yang memiliki JComboBox dengan beberapa pilihan warna. Ketika pengguna memilih warna dari JComboBox, latar belakang JFrame harus berubah sesuai dengan warna yang dipilih. Seperti contoh berikut:



Jawaban

Ketik jawaban disini ...

Source Code

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ColorChanger extends JFrame {

    public ColorChanger() {
        setTitle("Color Changer");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout());
```



TUGAS & EVALUASI

```
panel.setOpaque(false);

JLabel label = new JLabel("Choose Color:");
label.setFont(new Font("Arial", Font.BOLD, 18));
panel.add(label);

String[] colors = {"Red", "Green", "Blue", "Yellow",
"Cyan", "Magenta"};
JComboBox<String> colorComboBox = new
JComboBox<>(colors);
panel.add(colorComboBox);

colorComboBox.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String pilihWarna = (String)
colorComboBox.getSelectedItem();

        switch (pilihWarna) {
            case "Red":
                getContentPane().setBackground(Color.RED);
                break;
            case "Green":
                getContentPane().setBackground(Color.GREEN);
                break;
            case "Blue":
                getContentPane().setBackground(Color.BLUE);
                break;
            case "Yellow":
                getContentPane().setBackground(Color.YELLOW);
                break;
            case "Cyan":
                getContentPane().setBackground(Color.CYAN);
                break;
            case "Magenta":
                break;
        }
    }
});
```



TUGAS & EVALUASI

```
        getContentPane().setBackground(Color.MAGENTA);
    }
}
});

add(panel);

getContentPane().setBackground(Color.WHITE);

setVisible(true);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> new ColorChanger());
}
}
```

Penjelasan

JComboBox berisi daftar warna seperti “Red”, “Green”, “Blue”, dll. Ketika pengguna memilih warna, ActionListener dipanggil untuk menangani event tersebut. Saat item dipilih, method getSelectedItem() mengambil pilihan pengguna, dan warna latar belakang JFrame diubah menggunakan metode setBackground(). Warna latar belakang diubah sesuai pilihan menggunakan SwitchCase. Background awal diatur menjadi putih menggunakan getContentPane(). setBackground (Color.WHITE).

Output





PERTEMUAN 6

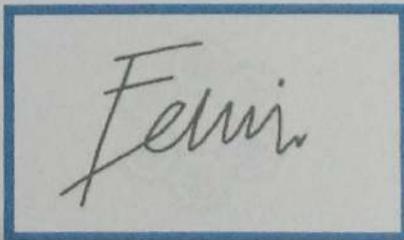
**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Favzi Saputra
NPM : 06.2023.1.07748
Pertemuan : 6

A handwritten signature in black ink, reading "Favzi", enclosed within a thin blue rectangular border.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE X
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 6

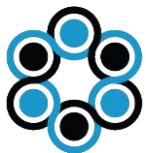
RPL-M12-A

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
"PertemuanX_NPM AKHIR"
Ganti "X" menjadi Pertemuan yang sedang berlangsung.
 - 2) Pada saat Praktikum, Jawablah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar "**Laporan Praktikum**".
 - 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
 - 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
 - 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
"PertemuanX_NPM AKHIR.pdf"
 - 6) Upload File **Laporan Praktikum [PDF]** pada form yang sudah disediakan.
-

TUGAS PRAKTIKUM

1. Apa yang kalian ketahui tentang arsitektur MVC pada Java [wajib].
2. Apa keuntungan penggunaan MVC pada program Java [wajib].
3. Sebutkan dan jelaskan komponen-komponen apa saja yang terdapat pada MVC.
4. Allen ingin membuat sebuah program sim sekolah sederhana dengan entitas Siswa, Nilai, Mata kuliah. dimana Nilai berelasi dengan id Siswa dan id Matkul. Buatlah Model untuk ke 3 entitas tersebut. [wajib].
5. Dari soal sebelumnya buatlah Controller untuk Create 3 entitas tersebut baik Siswa, Mata Kuliah, ataupun Nilai.
6. Buatlah View untuk input nilai mata kuliah siswa



TUGAS PRAKTIKUM

Soal Praktikum

1. Apa yang kalian ketahui tentang arsitektur MVC pada Java

Jawaban

MVC adalah singkatan dari Model-View-Controller, merupakan sebuah pola desain (design pattern) yang digunakan dalam pengembangan perangkat lunak untuk memisahkan komponen utama dari aplikasi.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

2. Apa keuntungan penggunaan MVC pada program Java

Jawaban

Dengan adanya MVC, aplikasinya menjadi lebih mudah untuk dikelola, dikembangkan oleh tim yang berbeda, dan juga memungkinkan adanya perubahan tanpa harus mengubah seluruh sistem.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

3. Sebutkan dan jelaskan komponen-komponen apa saja yang terdapat pada MVC

Jawaban

MVC dibagi menjadi 3 komponen yaitu Model, View, Controller

- Model, bertugas untuk memproses suatu data (kotor) yang dikirim oleh Controller dan mengembalikkan kembali ke Controller dalam bentuk Entitas Objek Data.
- View, bertugas untuk menampilkan apapun (informasi) ke User baik itu dalam bentuk Console maupun Graphical User Interface (GUI).
- Controller, bertugas menangani segala macam bentuk logika dalam suatu aplikasi. Terkadang Controller dapat menjalankan tugasnya pada saat pertama kali aplikasi berjalan, Jadi pada saat aplikasi kita RUN maka akan ada Controller yang menangani dan mengarahkan aplikasi kita ke alur yang benar.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

4. Allen ingin membuat sebuah program sim sekolah sederhana dengan entitas Siswa, Nilai, Mata kuliah. dimana Nilai berelasi dengan id Siswa dan id Matkul. Buatlah Model untuk ke 3 entitas tersebut

Jawaban

Ketik jawaban disini ...

Source Code

```
package Models;

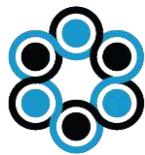
public class SiswaModel {
    private int id;
    private String nama;
    private String kelas;

    public SiswaModel(int id, String nama, String kelas) {
        this.id = id;
        this.nama = nama;
        this.kelas = kelas;
    }

    public int getId() {
        return id;
    }

    public String getNama() {
        return nama;
    }

    public String getKelas() {
        return kelas;
    }
}
```



TUGAS PRAKTIKUM

```
public void setId(int id) {
    this.id = id;
}

public void setNama(String nama) {
    this.nama = nama;
}

public void setKelas(String kelas) {
    this.kelas = kelas;
}

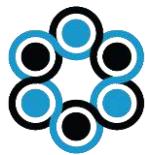
@Override
public String toString() {
    return "Siswa{id=" + id + ", nama='" + nama + "'",
kelas='" + kelas + "'}";
}
}

package Models;

public class NilaiModel {
    private int id;
    private int idSiswa;
    private int idMatkul;
    private double nilai;

    public Nilai(int id, int idSiswa, int idMatkul, double
nilai) {
        this.id = id;
        this.idSiswa = idSiswa;
        this.idMatkul = idMatkul;
        this.nilai = nilai;
    }

    public int getId() {
        return id;
    }
}
```



TUGAS PRAKTIKUM

```
public int getIdSiswa() {
    return idSiswa;
}

public int getIdMatkul() {
    return idMatkul;
}

public double getNilai() {
    return nilai;
}

@Override
public String toString() {
    return "Nilai{id=" + id + ", idSiswa=" + idSiswa + ", idMatkul=" + idMatkul + ", nilai=" + nilai + "}";
}
}

package Models;

public class MataKuliahModel {
    private int id;
    private String namaMatkul;

    public MataKuliahModel(int id, String namaMatkul) {
        this.id = id;
        this.namaMatkul = namaMatkul;
    }

    public int getId() {
        return id;
    }

    public String getNamaMatkul() {
        return namaMatkul;
    }
}
```



TUGAS PRAKTIKUM

```
@Override  
public String toString() {  
    return "MataKuliah{id=" + id + ", namaMatkul='" +  
    namaMatkul + "'}";  
}  
}
```

Penjelasan

Class SiswaModel mempunyai 3 atribut id, nama, kelas, menggunakan method getter&setter, dan menggunakan method `toString()` untuk representasi string dari data nilai. Class NilaiModel mempunyai 4 atribut id, idSiswa, idMatkul, nilai, menggunakan method getter&setter, dan menggunakan method `toString()` untuk representasi string dari data nilai. Class MataKuliahModel mempunyai 2 atribut id dan namaMatkul, menggunakan method getter, dan menggunakan method `toString()` untuk memberikan representasi string dari data nilai.

Output

Masukan screenshot output disini



TUGAS PRAKTIKUM

Soal Praktikum

5. Dari soal sebelumnya buatlah Controller untuk Create 3 entitas tersebut baik Siswa, Mata Kuliah, ataupun Nilai

Jawaban

Ketik jawaban disini ...

Source Code

```
package Controller;

import java.util.ArrayList;
import java.util.List;

public class SiswaController {
    private List<Siswa> siswaList = new ArrayList<>();

    public void createSiswa(int id, String nama, String kelas) {
        Siswa siswa = new Siswa(id, nama, kelas);
        siswaList.add(siswa);
        System.out.println("Siswa berhasil ditambahkan: " +
siswa);
    }

    public List<Siswa> getSiswaList() {
        return siswaList;
    }
}

package Controller;

import java.util.ArrayList;
import java.util.List;

public class NilaiController {
    private List<Nilai> nilaiList = new ArrayList<>();
```



TUGAS PRAKTIKUM

```
public void createNilai(int id, int idSiswa, int idMatkul,
double nilai) {
    NilaiController newNilai = new NilaiController(id,
idSiswa, idMatkul, nilai);
    nilaiList.add(newNilai);
    System.out.println("Nilai berhasil ditambahkan: " +
newNilai);
}

public List<Nilai> getNilaiList() {
    return nilaiList;
}
}

package Controller;

import java.util.ArrayList;
import java.util.List;

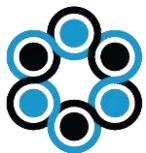
public class MataKuliahController {
    private List<MataKuliah> mataKuliahList = new ArrayList<>();

    public void createMataKuliah(int id, String namaMatkul) {
        MataKuliah matkul = new MataKuliah(id, namaMatkul);
        mataKuliahList.add(matkul);
        System.out.println("Mata Kuliah berhasil ditambahkan: " +
+ matkul);
    }

    public List<MataKuliah> getMataKuliahList() {
        return mataKuliahList;
    }
}
```

Penjelasan

Class SiswaController mempunyai 1 atribut bervariabel siswaList untuk menyimpan objek Siswa. Method createSiswa berfungsi untuk membuat objek baru



TUGAS PRAKTIKUM

Siswa berdasarkan parameter dan menggunakan getter untuk mengembalikan daftar semua siswa yang ada di siswaList. Class Nilai Controller dan MataKuliahController mempunyai logika sama seperti Class SiswaController hanya saja beda variabel. Jadi, fungsi utama controller ini untuk menambahkan data baru ke dalam daftar, dan mengakses semua data yang telah ditambahkan.

Output

```
| Siswa berhasil ditambahkan: Siswa{idSiswa=1, nama='Fauzi', kelas='Pagi'}  
| Siswa berhasil ditambahkan: Siswa{idSiswa=2, nama='Saputra', kelas='Malam'}  
| Mata kuliah berhasil ditambahkan: MataKuliah{idMatkul=1, namaMatkul='Pemrograman Berorientasi Objek', sks=5}  
| Mata kuliah berhasil ditambahkan: MataKuliah{idMatkul=2, namaMatkul='Sistem Operasi', sks=3}
```



TUGAS PRAKTIKUM

Soal Praktikum

6. Buatlah View untuk input nilai mata kuliah siswa

Jawaban

Ketik jawaban disini ...

Source Code

```
package Views;

import Controllers.NilaiController;
import Controllers.SiswaController;
import Controllers.MataKuliahController;

import java.util.Scanner;

public class NilaiView {
    protected Scanner scanner = new Scanner(System.in);
    protected NilaiController nilaiController;
    protected SiswaController siswaController;
    protected MataKuliahController mataKuliahController;

    public NilaiView(NilaiController nilaiController,
SiswaController siswaController, MataKuliahController
mataKuliahController) {
        this.nilaiController = nilaiController;
        this.siswaController = siswaController;
        this.mataKuliahController = mataKuliahController;
    }

    public void inputNilai() {
        System.out.println("\n==== Input Nilai Mata Kuliah ===");

        // Input ID Siswa
        System.out.print("Masukkan ID Siswa: ");
        int idSiswa = scanner.nextInt();
    }
}
```



TUGAS PRAKTIKUM

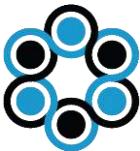
```
// Input ID Mata Kuliah  
System.out.print("Masukkan ID Mata Kuliah: ");  
int idMatkul = scanner.nextInt();  
  
// Input Nilai  
System.out.print("Masukkan Nilai: ");  
double nilai = scanner.nextDouble();  
  
// Tambahkan Nilai melalui Controller  
nilaiController.tambahNilai(idSiswa, idMatkul, nilai);  
  
System.out.println("Nilai berhasil ditambahkan!");  
scanner.close();  
}  
}
```

Penjelasan

Class NilaiView meminta input dari atribut Model yang terdiri idSiswa, idMataKuliah, Nilai. Mengirim data yang diinput ke NilaiController untuk diproses. Method inputNilai menampilkan output, menerima input dari Model, semua input dibaca menggunakan Scanner. Method tambahNilai pada NilaiController dipanggil dengan parameter dari pengguna, kemudian data tersebut ditambahkan ke dalam daftar nilai di NilaiController.

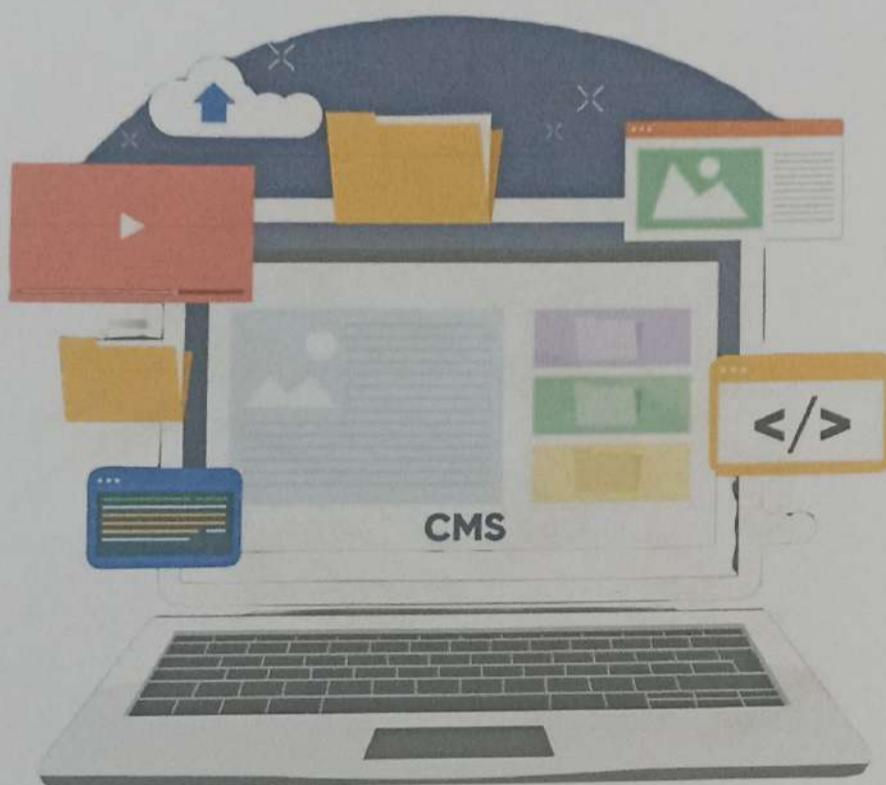
Output

```
== Sistem Sekolah ==  
1. Tampilkan Siswa  
2. Tampilkan Mata Kuliah  
3. Input Nilai  
4. Tampilkan Nilai  
0. Keluar  
Pilih menu: 3  
  
== Input Nilai ==  
  
== Input Nilai Mata Kuliah ==  
Masukkan ID Siswa: 1  
Masukkan ID Mata Kuliah: 1  
Masukkan Nilai: 80  
Nilai berhasil ditambahkan: Nilai{idSiswa=1, idMatkul=1, nilai=80.0}  
Nilai berhasil ditambahkan!
```



TUGAS PRAKTIKUM

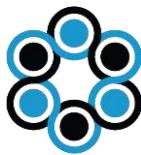
TUGAS DAN EVALUASI



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.07740
Modul : R2

A handwritten signature of "Fauzi" enclosed within a blue rectangular border.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Jelaskan alur kerja arsitektur MVC dan bagaimana interaksi antara User, View, Controller, dan Model dalam mengelola permintaan pengguna!

Jawaban

- User berinteraksi dengan bagian View
- View memanggil sebuah Controller untuk melakukan sesuatu (Event)
- Controller mengelola request dari User dan memanggil sebuah Model untuk mendapatkan data (jika dibutuhkan , jika tidak membutuhkan data maka langsung ke step 5)
- Model mengelola permintaan dari Controller dan mengembalikan data yang telah diolah, Kembali lagi ke Controller
- Controller mengirim data ke View untuk ditampilkan/disajikan ke User

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

2. Mengapa di dalam MVC, input/output tidak diperbolehkan berada di dalam Controller?

Jawaban

- Karena untuk menjaga pemisahan tanggung jawab (separation of concerns) yang jelas antara komponen-komponen dalam aplikasi. Proses yang dapat terjadi di dalam Controller adalah mengelola hasil input yang telah dilakukan pada View, mengirim data (kotor) ke Model untuk di proses lebih lanjut, serta memilih View yang tepat supaya aplikasi dapat berjalan dengan semestinya.

Source Code

Tulis kode program dikotak ini...

1 kotak dan 1 Penjelasan untuk 1 Class

Penjelasan

Tulis Penjelasan disini ...

Output

Masukan screenshot output disini



TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Pemilik toko ingin mengelola inventaris barang dan catatan penjualan secara efisien. Mereka memerlukan sebuah sistem yang dapat mencatat informasi tentang barang yang ada ditoko dan transaksi penjualan yang terjadi.

Entity yang harus dibuat:

Barang

- idBarang - Penjualan - totalHarga - quantitas - tanggal
- nama - idPenjualan - harga - stock

Tugas:

Terapkan arsitektur MVC untuk mengelola data barang dan penjualan

Buat Model, Controller, dan View yang mendukung:

Daftar barang:

- Penambahan dan pembaruan barang
- Pencatatan transaksi penjualan
- Daftar transaksi penjualan

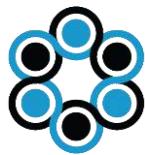
Jawaban

Ketik jawaban disini ...

Source Code

```
package models;

public class Barang {
    private String idBarang;
    private String nama;
    private double harga;
    private int stock;
```



TUGAS & EVALUASI

```
public Barang(String idBarang, String nama, double harga,  
int stock) {  
    this.idBarang = idBarang;  
    this.nama = nama;  
    this.harga = harga;  
    this.stock = stock;  
}  
  
public String getIdBarang() {  
    return idBarang;  
}  
  
public void setIdBarang(String idBarang) {  
    this.idBarang = idBarang;  
}  
  
public String getNama() {  
    return nama;  
}  
  
public void setNama(String nama) {  
    this.nama = nama;  
}  
  
public double getHarga() {  
    return harga;  
}  
  
public void setHarga(double harga) {  
    this.harga = harga;  
}  
  
public int getStock() {  
    return stock;  
}  
  
public void setStock(int stock) {  
    this.stock = stock;
```



TUGAS & EVALUASI

```
}

public void updateStock(int quantitySold) {
    this.stock -= quantitySold;
}

}

package models;

public class Penjualan {
    private String idPenjualan;
    private String idBarang;
    private int quantitas;
    private double totalHarga;
    private String tanggal;

    public Penjualan(String idPenjualan, String idBarang, int
        quantitas, double totalHarga, String tanggal) {
        this.idPenjualan = idPenjualan;
        this.idBarang = idBarang;
        this.quantitas = quantitas;
        this.totalHarga = totalHarga;
        this.tanggal = tanggal;
    }

    public String getIdPenjualan() {
        return idPenjualan;
    }

    public void setIdPenjualan(String idPenjualan) {
        this.idPenjualan = idPenjualan;
    }

    public String getIdBarang() {
        return idBarang;
    }

    public void setIdBarang(String idBarang) {
        this.idBarang = idBarang;
    }
}
```



TUGAS & EVALUASI

```
}

public int getQuantitas() {
    return quantitas;
}

public void setQuantitas(int quantitas) {
    this.quantitas = quantitas;
}

public double getTotalHarga() {
    return totalHarga;
}

public void setTotalHarga(double totalHarga) {
    this.totalHarga = totalHarga;
}

public String getTanggal() {
    return tanggal;
}

public void setTanggal(String tanggal) {
    this.tanggal = tanggal;
}

}

package controllers;

import models.*;
import java.util.ArrayList;
import java.util.List;

public class inventoryController {
    private List<Barang> barangList = new ArrayList<>();
    private List<Penjualan> penjualanList = new ArrayList<>();

    public void tambahBarang(Barang barang) {
```



TUGAS & EVALUASI

```
        barangList.add(barang);
    }

    public void updateBarang(String idBarang, double hargaBaru,
int stockBaru) {
        for (Barang barang : barangList) {
            if (barang.getIdBarang().equals(idBarang)) {
                barang.setHarga(hargaBaru);
                barang.setStock(stockBaru);
                break;
            }
        }
    }

    public void catatPenjualan(Penjualan penjualan) {
        for (Barang barang : barangList) {
            if
(barang.getIdBarang().equals(penjualan.getIdBarang())) {
                if (barang.getStock() >=
penjualan.getQuantitas()) {
                    barang.updateStock(penjualan.getQuantitas());
                ;
                    penjualanList.add(penjualan);
                } else {
                    System.out.println("Stok barang tidak
mencukupi!");
                }
                break;
            }
        }
    }

    public List<Barang> getDaftarBarang() {
        return barangList;
    }

    public List<Penjualan> getDaftarPenjualan() {
        return penjualanList;
    }
}
```



TUGAS & EVALUASI

```
}

package views;

import models.*;
import java.util.List;

public class inventoryView {
    public void tampilkanDaftarBarang(List<Barang> barangList) {
        System.out.println("Daftar Barang:");
        for (Barang barang : barangList) {
            System.out.println("ID: " + barang.getIdBarang() +
", Nama: " + barang.getNama() + ", Harga: " + barang.getHarga() +
", Stock: " + barang.getStock());
        }
    }

    public void tampilkanDaftarPenjualan(List<Penjualan>
penjualanList) {
        System.out.println("Daftar Penjualan:");
        for (Penjualan penjualan : penjualanList) {
            System.out.println("ID Penjualan: " +
penjualan.getIdPenjualan() + ", ID Barang: " +
penjualan.getIdBarang() + ", Quantitas: " +
penjualan.getQuantitas() + ", Total Harga: " +
penjualan.getTotalHarga() + ", Tanggal: " +
penjualan.getTanggal());
        }
    }
}

import controllers.*;
import models.*;
import views.*;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
```



TUGAS & EVALUASI

```
inventoryController controller = new
inventoryController();
inventoryView view = new inventoryView();
Scanner scanner = new Scanner(System.in);

while (true) {
    System.out.println("\nMenu:");
    System.out.println("1. Tambah Barang");
    System.out.println("2. Update Barang");
    System.out.println("3. Catat Penjualan");
    System.out.println("4. Lihat Daftar Barang");
    System.out.println("5. Lihat Daftar Penjualan");
    System.out.println("6. Keluar");
    System.out.print("Pilih: ");
    int pilihan = scanner.nextInt();
    scanner.nextLine();

    switch (pilihan) {
        case 1:
            System.out.print("ID Barang: ");
            String idBarang = scanner.nextLine();
            System.out.print("Nama Barang: ");
            String nama = scanner.nextLine();
            System.out.print("Harga: ");
            double harga = scanner.nextDouble();
            System.out.print("Stock: ");
            int stock = scanner.nextInt();
            controller.tambahBarang(new Barang(idBarang,
nama, harga, stock));
            break;
        case 2:
            System.out.print("ID Barang: ");
            idBarang = scanner.nextLine();
            System.out.print("Harga Baru: ");
            double hargaBaru = scanner.nextDouble();
            System.out.print("Stock Baru: ");
            int stockBaru = scanner.nextInt();
            controller.updateBarang(idBarang, hargaBaru,
stockBaru);
    }
}
```



TUGAS & EVALUASI

```
        break;
    case 3:
        System.out.print("ID Penjualan: ");
        String idPenjualan = scanner.nextLine();
        System.out.print("ID Barang: ");
        idBarang = scanner.nextLine();
        System.out.print("Quantitas: ");
        int quantitas = scanner.nextInt();
        System.out.print("Total Harga: ");
        double totalHarga = scanner.nextDouble();
        scanner.nextLine(); // Clear buffer
        System.out.print("Tanggal: ");
        String tanggal = scanner.nextLine();
        controller.catatPenjualan(new
Penjualan(idPenjualan, idBarang, quantitas, totalHarga,
tanggal));
        break;
    case 4:
        view.tampilkanDaftarBarang(controller.getDaft
tarBarang());
        break;
    case 5:
        view.tampilkanDaftarPenjualan(controller.get
DaftarPenjualan());
        break;
    case 6:
        System.out.println("Keluar...");
        scanner.close();
        return;
    default:
        System.out.println("Pilihan tidak valid!");
    }
}
}
```



TUGAS & EVALUASI

Penjelasan

Models berfungsi representasi data. Pada Models mempunyai 2 Class yaitu Barang dan Penjualan. Masing-masing class memiliki atribut, menyediakan method updateStock(int quantitySold) pada class Barang untuk mengurangi stok saat penjualan. Controller berfungsi mengelola data barang dan penjualan, mempunyai class yang bervariabel inventoryController, memiliki atribut barangList dan penjualanList untuk menyimpan daftar ke dalam inventaris. Memiliki 4 method utama yang berfungsi untuk menambahkan, memperbarui, memvalidasi, dan mengembalikan daftar barang dan daftar penjualan. View berfungsi menampilkan data kepada pengguna, memiliki method tampilkanDaftarBarang dan tampilkanDaftarPenjualan. Main program berfungsi sebagai titik program dan menyatukan komponen MVC, pada main program membuat switch case untuk menyediakan opsi bagi pengguna. Membaca input dari pengguna menggunakan Scanner.

Output

```
Menu:
1. Tambah Barang
2. Update Barang
3. Catat Penjualan
4. Lihat Daftar Barang
5. Lihat Daftar Penjualan
6. Keluar
Pilih: 4
Daftar Barang:
ID: 1, Nama: Laptop, Harga: 175000.0, Stock: 3

Menu:
1. Tambah Barang
2. Update Barang
3. Catat Penjualan
4. Lihat Daftar Barang
5. Lihat Daftar Penjualan
6. Keluar
Pilih: 3
ID Penjualan: 2
ID Barang: 1
Quantitas: 3
Total Harga: 100000
Tanggal: 24

Menu:
1. Tambah Barang
2. Update Barang
3. Catat Penjualan
4. Lihat Daftar Barang
5. Lihat Daftar Penjualan
6. Keluar
Pilih: 5
Daftar Penjualan:
ID Penjualan: 2, ID Barang: 1, Quantitas: 3, Total Harga: 100000.0, Tanggal: 24
```



PERTEMUAN 7

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA**

LAPORAN PRAKTIKUM



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

Nama : Fauzi Saputra
NPM : 06.2023.1.0748
Pertemuan : 7

A handwritten signature in black ink, reading "Fauzi", enclosed within a thin blue rectangular border.



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

PERTEMUAN 7

RPL-MF1T12-B

MEKANISME PRAKTIKUM

- 1) Buat Sebuah **Project Java Baru** pada IntelliJ IDEA dengan nama Project:
PertemuanX_NPM AKHIR
Ganti "X" menjadi Pertemuan yang sedang berlangsung.
 - 2) Pada saat Praktikum, Jawabanlah Soal Pertanyaan yang memiliki Label **WAJIB** terlebih dahulu Pada Lembar "**Laporan Praktikum**".
 - 3) Segala Bentuk **Soal yang memiliki Jawaban** berupa **Kode Program**, maka kode program tersebut harus disimpan pada **File java Project** yang telah dibuat.
 - 4) Setiap **File Java** yang dibuat harus mencantumkan Pertanyaan pada bagian atas (baris pertama)
 - 5) Simpan **File Laporan Praktikum** yang berupa **DOCX** menjadi **FILE PDF** kemudian ubah nama file PDF menjadi:
PertemuanX_NPM AKHIR.pdf
 - 6) Upload File **Laporan Praktikum [PDF]** pada form yang sudah disediakan.
-

TUGAS PRAKTIKUM

1. Buatlah sebuah program sederhana untuk menginput data menggunakan **inputan dinamis**. Program harus memiliki atribut / method static, trapkan juga Naming Convnetion dengan benar.
2. Buatlah sebuah sistem informasi sederhana untuk sebuah bengkel dengan fitur:
 - Menampilkan, Menambahkan, Menghapus, dan Mengedit data kendaraan, mekanik, atau layanan.
 - Setiap kendaraan harus memiliki mekanik yang menangani, dan setiap kendaraan mendapatkan layanan tertentu.
 - Gunakan konsep encapsulation dan relasi antar kelas yang sesuai. [Wajib]
3. Buatlah program untuk manajemen sistem sewa kendaraan, yang terdiri dari kendaraan listrik dan kendaraan berbahan bakar bensin. Berikut fitur yang harus dimiliki aplikasi:
 - **Kelas abstrak** Kendaraan yang memiliki metode abstrak `hitungHargaSewa()`
 - Dua kelas turunan yaitu



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

- a. KendaraanListrik: Menghitung harga sewa berdasarkan durasi sewa dan tingkat konsumsi daya.
- b. KendaraanBensin: Menghitung harga sewa berdasarkan durasi sewa dan jumlah bahan bakar yang digunakan.
- **Interface Diskon** untuk menghitung diskon khusus untuk KendaraanListrik. [Wajib]
4. Kembangkan Soal 2 sehingga data bengkel yang dikelola dapat disimpan ke dalam file teks.

Peetunjuk:

- Simpan data Kendaraan, Mekanik, dan Layanan ke dalam file **dataBengkel.txt** saat program berakhir.
- Saat program dimulai, baca data dari file tersebut jika file sudah ada.
- Gunakan **exception handling** untuk menangani kasus seperti file tidak ditemukan atau format data yang salah. [Wajib]
- 5. Sebuah toko penyewaan alat camping ingin membuat aplikasi sederhana untuk mempermudah proses penyewaan alat camping. Aplikasi ini akan digunakan oleh kasir untuk mencatat penyewaan alat camping. Berikut fitur yang harus dimiliki aplikasi:
 - Aplikasi memiliki daftar alat camping lengkap dengan nama alat, jenis, dan harga sewa per hari.
 - Kasir dapat menambahkan alat camping yang disewa pelanggan ke dalam daftar penyewaan.
 - Aplikasi akan menghitung total harga dari semua alat yang disewa berdasarkan durasi sewa.
 - Kasir dapat membatalkan salah satu alat yang disewa jika terjadi kesalahan input.

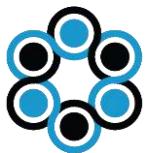
Ketentuan:

- Gunakan **GUI (Swing)** untuk antarmuka aplikasi.
- Gunakan arsitektur **MVC** (Model-View-Controller).



SOAL PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERIODE IX
Laboratorium Rekayasa Perangkat Lunak, ITATS

- Validasi input, seperti jumlah alat yang disewa harus berupa angka yang valid, durasi sewa tidak boleh lebih dari 10 hari.
- Tampilkan pesan konfirmasi untuk pembatalan alat.



TUGAS PRAKTIKUM

Soal Praktikum

1. Buatlah sebuah program sederhana untuk menginput data menggunakan inputan dinamis. Program harus memiliki atribut / method static, terapkan juga Naming Convnetion dengan benar.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.Scanner;

public class UserDataMahasiswa {
    private static String userName;

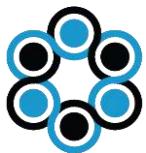
    public static void setUserName(String name) {
        userName = name;
    }

    public static String getUserName() {
        return userName;
    }

    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            System.out.print("Masukkan Nama: ");
            String name = scanner.nextLine();

            setUserName(name);

            System.out.println("Hello Bro " + getUserName() +
                    "!");
        }
    }
}
```



TUGAS PRAKTIKUM

Penjelasan

Pada program ini, atribut `userName` dideklarasikan sebagai static, method getter dan setter digunakan untuk mengatur dan mendapatkan nilai atribut `userName`. Main program mengambil input dinamis dari user dan menggunakan method static untuk mengatur dan mendapatkan nama user.

Output

Masukkan Nama: Fauzi
Hello Bro Fauzi!



TUGAS PRAKTIKUM

Soal Praktikum

2. Buatlah sebuah sistem informasi sederhana untuk sebuah bengkel dengan fitur :
- Menampilkan, Menambahkan, Menghapus, dan Mengedit data kendaraan, mekanik, atau layanan.
 - Setiap kendaraan harus memiliki mekanik yang menangani, dan setiap kendaraan mendapatkan layanan tertentu.
 - Gunakan konsep encapsulation dan relasi antar kelas yang sesuai.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;
import java.util.Scanner;

public class BengkelSistemInformasi {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Bengkel bengkel = new Bengkel();

        while (true) {
            System.out.println("\n==== Sistem Informasi Bengkel
====");
            System.out.println("1. Tambah Kendaraan");
            System.out.println("2. Tampilkan Kendaraan");
            System.out.println("3. Edit Kendaraan");
            System.out.println("4. Hapus Kendaraan");
            System.out.println("5. Keluar");
            System.out.print("Pilih menu: ");
            int choice = scanner.nextInt();
            scanner.nextLine();
```



TUGAS PRAKTIKUM

```
switch (choice) {
    case 1:
        System.out.print("Masukkan Nama Kendaraan:");
        String namaKendaraan = scanner.nextLine();

        System.out.print("Masukkan Nama Mekanik: ");
        String namaMekanik = scanner.nextLine();

        System.out.print("Masukkan Layanan: ");
        String layanan = scanner.nextLine();

        bengkel.tambahKendaraan(new
Kendaraan(namaKendaraan, new Mekanik(namaMekanik), new
Layanan(layanan)));
        System.out.println("Kendaraan berhasil
ditambahkan!");
        break;
    case 2:
        bengkel.tampilkanKendaraan();
        break;
    case 3:
        System.out.print("Masukkan nomor kendaraan
yang ingin diedit: ");
        int editIndex = scanner.nextInt() - 1;
        scanner.nextLine(); // Konsumsi newline

        if (editIndex < 0 || editIndex >=
bengkel.getKendaraanList().size()) {
            System.out.println("Nomor kendaraan
tidak valid!");
            break;
        }

        System.out.print("Masukkan Nama Baru
Kendaraan: ");
        String namaBaruKendaraan =
scanner.nextLine();
```



TUGAS PRAKTIKUM

```
        System.out.print("Masukkan Nama Baru
Mekanik: ");
        String namaBaruMekanik = scanner.nextLine();

        System.out.print("Masukkan Layanan Baru: ");
        String layananBaru = scanner.nextLine();

        bengkel.editKendaraan(editIndex,
namaBaruKendaraan, namaBaruMekanik, layananBaru);
        System.out.println("Kendaraan berhasil
diedit!");
        break;
    case 4:
        System.out.print("Masukkan nomor kendaraan
yang ingin dihapus: ");
        int hapusIndex = scanner.nextInt() - 1;

        if (hapusIndex < 0 || hapusIndex >=
bengkel.getKendaraanList().size()) {
            System.out.println("Nomor kendaraan
tidak valid!");
            break;
        }

        bengkel.hapusKendaraan(hapusIndex);
        System.out.println("Kendaraan berhasil
dihapus!");
        break;
    case 5:
        System.out.println("Terima kasih telah
menggunakan sistem!");
        scanner.close();
        return;
    default:
        System.out.println("Pilihan tidak valid!");
    }
}
}

class Bengkel {
```



TUGAS PRAKTIKUM

```
private ArrayList<Kendaraan> kendaraanList = new
ArrayList<>();

public void tambahKendaraan(Kendaraan kendaraan) {
    kendaraanList.add(kendaraan);
}

public void tampilkanKendaraan() {
    if (kendaraanList.isEmpty()) {
        System.out.println("Tidak ada kendaraan yang
terdaftar.");
        return;
    }

    System.out.println("\n==== Daftar Kendaraan ===");
    for (int i = 0; i < kendaraanList.size(); i++) {
        Kendaraan kendaraan = kendaraanList.get(i);
        System.out.printf("%d. Nama Kendaraan: %s, Mekanik:
%s, Layanan: %s%n",
                           (i + 1), kendaraan.getNama(),
kendaraan.getMekanik().getNama(),
kendaraan.getLayanan().getNama());
    }
}

public void editKendaraan(int index, String namaBaru, String
namaMekanikBaru, String layananBaru) {
    Kendaraan kendaraan = kendaraanList.get(index);
    kendaraan.setNama(namaBaru);
    kendaraan.getMekanik().setNama(namaMekanikBaru);
    kendaraan.getLayanan().setNama(layananBaru);
}

public void hapusKendaraan(int index) {
    kendaraanList.remove(index);
}

public ArrayList<Kendaraan> getKendaraanList() {
    return kendaraanList;
}
```



TUGAS PRAKTIKUM

```
}

class Kendaraan {
    private String nama;
    private Mekanik mekanik;
    private Layanan layanan;

    public Kendaraan(String nama, Mekanik mekanik, Layanan
layanan) {
        this.nama = nama;
        this.mekanik = mekanik;
        this.layanan = layanan;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

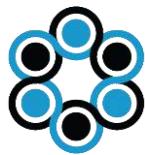
    public Mekanik getMekanik() {
        return mekanik;
    }

    public Layanan getLayanan() {
        return layanan;
    }
}

class Mekanik {
    private String nama;

    public Mekanik(String nama) {
        this.nama = nama;
    }

    public String getNama() {
        return nama;
    }
}
```



TUGAS PRAKTIKUM

```
public void setNama(String nama) {
    this.nama = nama;
}
}
class Layanan {
    private String nama;

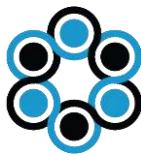
    public Layanan(String nama) {
        this.nama = nama;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }
}
```

Penjelasan

Class Bengkel merangkum detail kendaraan, termasuk nama kendaraan, mekanik, dan layanannya. Class Kendaraan dan Mekanik merangkum detail mekanik dan layanan masing-masing. Class BengkelSistemInformasi berisi method utama untuk menambahkan, menampilkan, mengedit, dan menghapus kendaraan



TUGAS PRAKTIKUM

Output

```
== Sistem Informasi Bengkel ==
1. Tambah Kendaraan
2. Tampilkan Kendaraan
3. Edit Kendaraan
4. Hapus Kendaraan
5. Keluar
Pilih menu: 2
Tidak ada kendaraan yang terdaftar.

== Sistem Informasi Bengkel ==
1. Tambah Kendaraan
2. Tampilkan Kendaraan
3. Edit Kendaraan
4. Hapus Kendaraan
5. Keluar
Pilih menu: 1
Masukkan Nama Kendaraan: Porsche
Masukkan Nama Mekanik: Zay
Masukkan Layanan: Ganti Mesin
Kendaraan berhasil ditambahkan!

== Sistem Informasi Bengkel ==
1. Tambah Kendaraan
2. Tampilkan Kendaraan
3. Edit Kendaraan
4. Hapus Kendaraan
5. Keluar
Pilih menu: 1
Masukkan Nama Kendaraan: Lamboghirni
Masukkan Nama Mekanik: Fauzi
Masukkan Layanan: Pasang Nitro
Kendaraan berhasil ditambahkan!
```



TUGAS PRAKTIKUM

Soal Praktikum

3. Buatlah program untuk manajemen sistem sewa kendaraan, yang terdiri dari kendaraan listrik dan kendaraan berbahan bakar bensin. Berikut fitur yang harus dimiliki aplikasi :
- Kelas Abstrak Kendaraan yang memiliki metode abstrak hitungHargaSewa().
 - Dua kelas turunan yaitu :
 - a. KendaraanListrik: Menghitung harga sewa berdasarkan durasi sewa dan tingkat konsumsi daya.
 - b. KendaraanBensin: Menghitung harga sewa berdasarkan durasi sewa dan jumlah bahan bakar yang digunakan.
 - Interface Diskon untuk menghitung diskon khusus untuk KendaraanListrik.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.util.ArrayList;
import java.util.Scanner;

abstract class Kendaraan {
    private String nama;
    private int durasiSewa;

    public Kendaraan(String nama, int durasiSewa) {
        this.nama = nama;
        this.durasiSewa = durasiSewa;
    }

    public String getNama() {
        return nama;
    }
}
```



TUGAS PRAKTIKUM

```
}

public int getDurasiSewa() {
    return durasiSewa;
}

public abstract double hitungHargaSewa();
}

interface Diskon {
    double hitungDiskon();
}

class KendaraanListrik extends Kendaraan implements Diskon {
    private double konsumsiDaya;
    private static final double TARIF_PER_JAM = 10000;
    private static final double TARIF_PER_KWH = 5000;

    public KendaraanListrik(String nama, int durasiSewa, double konsumsiDaya) {
        super(nama, durasiSewa);
        this.konsumsiDaya = konsumsiDaya;
    }

    @Override
    public double hitungHargaSewa() {
        double harga = (getDurasiSewa() * TARIF_PER_JAM) +
(konsumsiDaya * TARIF_PER_KWH);
        return harga - hitungDiskon();
    }

    @Override
    public double hitungDiskon() {
        if (getDurasiSewa() > 5) {
            return 0.1 * ((getDurasiSewa() * TARIF_PER_JAM) +
(konsumsiDaya * TARIF_PER_KWH));
        }
        return 0;
    }
}
```



TUGAS PRAKTIKUM

```
}

class KendaraanBensin extends Kendaraan {
    private double bahanBakar;
    private static final double TARIF_PER_JAM = 15000;
    private static final double TARIF_PER_LITER = 7000;

    public KendaraanBensin(String nama, int durasiSewa, double bahanBakar) {
        super(nama, durasiSewa);
        this.bahanBakar = bahanBakar;
    }

    @Override
    public double hitungHargaSewa() {
        return (getDurasiSewa() * TARIF_PER_JAM) + (bahanBakar * TARIF_PER_LITER);
    }
}

public class SistemSewaKendaraan {
    private ArrayList<Kendaraan> daftarKendaraan = new ArrayList<>();
    private Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        SistemSewaKendaraan sistem = new SistemSewaKendaraan();
        sistem.menuUtama();
    }

    private void menuUtama() {
        while (true) {
            System.out.println("\n==== Sistem Sewa Kendaraan ===");
            System.out.println("1. Tambah Kendaraan Listrik");
            System.out.println("2. Tambah Kendaraan Bensin");
            System.out.println("3. Tampilkan Semua Kendaraan");
            System.out.println("4. Keluar");
            System.out.print("Pilih menu: ");
        }
    }
}
```



TUGAS PRAKTIKUM

```
int pilihan = scanner.nextInt();

switch (pilihan) {
    case 1:
        tambahKendaraanListrik();
        break;
    case 2:
        tambahKendaraanBensin();
        break;
    case 3:
        tampilkanKendaraan();
        break;
    case 4:
        System.out.println("Terima kasih telah
menggunakan sistem!");
        return;
    default:
        System.out.println("Pilihan tidak valid.");
}
}

private void tambahKendaraanListrik() {
    System.out.print("Masukkan Nama Kendaraan Listrik: ");
    scanner.nextLine();
    String nama = scanner.nextLine();
    System.out.print("Masukkan Durasi Sewa (jam): ");
    int durasi = scanner.nextInt();
    System.out.print("Masukkan Konsumsi Daya (kWh): ");
    double daya = scanner.nextDouble();

    Kendaraan kendaraan = new KendaraanListrik(nama, durasi,
daya);
    daftarKendaraan.add(kendaraan);
    System.out.println("Kendaraan Listrik berhasil
ditambahkan!");
}

private void tambahKendaraanBensin() {
```



TUGAS PRAKTIKUM

```
System.out.print("Masukkan Nama Kendaraan Bensin: ");
scanner.nextLine();
String nama = scanner.nextLine();
System.out.print("Masukkan Durasi Sewa (jam): ");
int durasi = scanner.nextInt();
System.out.print("Masukkan Jumlah Bahan Bakar (liter): ");
double bahanBakar = scanner.nextDouble();

Kendaraan kendaraan = new KendaraanBensin(nama, durasi,
bahanBakar);
daftarKendaraan.add(kendaraan);
System.out.println("Kendaraan Bensin berhasil
ditambahkan!");
}

private void tampilKendaraan() {
if (daftarKendaraan.isEmpty()) {
System.out.println("Tidak ada kendaraan yang
terdaftar.");
return;
}

System.out.println("\n==== Daftar Kendaraan ===");
for (int i = 0; i < daftarKendaraan.size(); i++) {
Kendaraan kendaraan = daftarKendaraan.get(i);
System.out.printf("%d. Nama: %s, Durasi Sewa: %d
jam, Harga Sewa: Rp%.2f%n",
(i + 1), kendaraan.getNama(),
kendaraan.getDurasiSewa(), kendaraan.hitungHargaSewa());
}
}
}
```

Penjelasan

Class Abstrak yaitu Class Kendaraan mendefinisikan atribut dan method abstrak hitungHargaSewa(). Class KendaraanListrik mengimplementasikan Interface Diskon untuk menghitung harga sewa dan konsumsi daya. Class KendaraanBensin untuk menghitung harga sewa berdasarkan durasi sewa dan



TUGAS PRAKTIKUM

konsumsi bahan bakar. Interface Diskon mendefinisikan method hitungDiskon(), untuk menghitung diskon. Class SistemSewaKendaraan berisi main program untuk berinteraksi dengan user dan menghitung harga sewa kendaraan yang dipilih.

Output

```
==> Sistem Sewa Kendaraan ==>
1. Tambah Kendaraan Listrik
2. Tambah Kendaraan Bensin
3. Tampilkan Semua Kendaraan
4. Keluar
Pilih menu: 1
Masukkan Nama Kendaraan Listrik: Doplhin
Masukkan Durasi Sewa (jam): 2
Masukkan Konsumsi Daya (kwh): 200
Kendaraan Listrik berhasil ditambahkan!

==> Sistem Sewa Kendaraan ==>
1. Tambah Kendaraan Listrik
2. Tambah Kendaraan Bensin
3. Tampilkan Semua Kendaraan
4. Keluar
Pilih menu: 2
Masukkan Nama Kendaraan Bensin: Raize
Masukkan Durasi Sewa (jam): 3
Masukkan Jumlah Bahan Bakar (liter): 36
Kendaraan Bensin berhasil ditambahkan!

==> Sistem Sewa Kendaraan ==>
1. Tambah Kendaraan Listrik
2. Tambah Kendaraan Bensin
3. Tampilkan Semua Kendaraan
4. Keluar
Pilih menu: 3

==> Daftar Kendaraan ==
1. Nama: Doplhin, Durasi Sewa: 2 jam, Harga Sewa: Rp1020000.00
2. Nama: Raize, Durasi Sewa: 3 jam, Harga Sewa: Rp297000.00

==> Sistem Sewa Kendaraan ==>
1. Tambah Kendaraan Listrik
2. Tambah Kendaraan Bensin
3. Tampilkan Semua Kendaraan
```



TUGAS PRAKTIKUM

Soal Praktikum

4. Kembangkan Soal 2 sehingga data bengkel yang dikelola dapat disimpan ke dalam file teks.

Petunjuk:

- Simpan data Kendaraan, Mekanik, dan Layanan ke dalam file dataBengkel.txt saat program berakhir.
- Saat program dimulai, baca data dari file tersebut jika file sudah ada.
- Gunakan exception handling untuk menangani kasus seperti file tidak ditemukan atau format data yang salah.

Jawaban

Ketik jawaban disini ...

Source Code

```
import java.io.*;
import java.util.*;

// Kelas Mekanik
class Mekanik {
    private String nama;

    public Mekanik(String nama) {
        this.nama = nama;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }
}
```



TUGAS PRAKTIKUM

```
}

@Override
public String toString() {
    return nama;
}
}

// Kelas Layanan
class Layanan {
    private String nama;

    public Layanan(String nama) {
        this.nama = nama;
    }

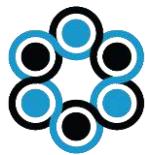
    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    @Override
    public String toString() {
        return nama;
    }
}

// Kelas Kendaraan
class Kendaraan {
    private String nama;
    private Mekanik mekanik;
    private Layanan layanan;

    public Kendaraan(String nama, Mekanik mekanik, Layanan
layanan) {
        this.nama = nama;
```



TUGAS PRAKTIKUM

```
        this.mekanik = mekanik;
        this.layanan = layanan;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public Mekanik getMekanik() {
        return mekanik;
    }

    public void setMekanik(Mekanik mekanik) {
        this.mekanik = mekanik;
    }

    public Layanan getLayanan() {
        return layanan;
    }

    public void setLayanan(Layanan layanan) {
        this.layanan = layanan;
    }

    @Override
    public String toString() {
        return "Kendaraan: " + nama + ", Mekanik: " + mekanik +
", Layanan: " + layanan;
    }
}

// Sistem Informasi Bengkel
public class SistemBengkel {
    private static final String FILE_NAME = "dataBengkel.txt";
    private List<Kendaraan> daftarKendaraan = new ArrayList<>();
```



TUGAS PRAKTIKUM

```
private List<Mekanik> daftarMekanik = new ArrayList<>();
private List<Layanan> daftarLayanan = new ArrayList<>();
private Scanner scanner = new Scanner(System.in);

public static void main(String[] args) {
    SistemBengkel sistem = new SistemBengkel();
    sistem.bacaDataDariFile();
    sistem.menuUtama();
    sistem.simpanDataKeFile();
}

private void menuUtama() {
    while (true) {
        System.out.println("\n==== Sistem Informasi Bengkel
====");
        System.out.println("1. Tambah Kendaraan");
        System.out.println("2. Tampilkan Semua Kendaraan");
        System.out.println("3. Edit Kendaraan");
        System.out.println("4. Hapus Kendaraan");
        System.out.println("5. Keluar");
        System.out.print("Pilih menu: ");
        int pilihan = scanner.nextInt();

        switch (pilihan) {
            case 1:
                tambahKendaraan();
                break;
            case 2:
                tampilanSemuaKendaraan();
                break;
            case 3:
                editKendaraan();
                break;
            case 4:
                hapusKendaraan();
                break;
            case 5:
                System.out.println("Terima kasih telah
menggunakan sistem!");
        }
    }
}
```



TUGAS PRAKTIKUM

```
        return;
    default:
        System.out.println("Pilihan tidak valid.");
    }
}

private void tambahKendaraan() {
    scanner.nextLine(); // Konsumsi newline
    System.out.print("Masukkan Nama Kendaraan: ");
    String namaKendaraan = scanner.nextLine();

    Mekanik mekanik = pilihMekanik();
    Layanan layanan = pilihLayanan();

    Kendaraan kendaraan = new Kendaraan(namaKendaraan,
mekanik, layanan);
    daftarKendaraan.add(kendaraan);
    System.out.println("Kendaraan berhasil ditambahkan!");
}

private Mekanik pilihMekanik() {
    if (daftarMekanik.isEmpty()) {
        System.out.print("Masukkan nama mekanik: ");
        String namaMekanik = scanner.nextLine();
        Mekanik mekanik = new Mekanik(namaMekanik);
        daftarMekanik.add(mekanik);
        return mekanik;
    } else {
        System.out.println("Pilih Mekanik:");
        for (int i = 0; i < daftarMekanik.size(); i++) {
            System.out.printf("%d. %s%n", (i + 1),
daftarMekanik.get(i).getNama());
        }
        System.out.print("Pilih Mekanik (masukkan nomor): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine(); // Konsumsi newline
        return daftarMekanik.get(pilihan - 1);
    }
}
```



TUGAS PRAKTIKUM

```
        }
    }

private Layanan pilihLayanan() {
    if (daftarLayanan.isEmpty()) {
        System.out.print("Masukkan nama layanan: ");
        String namaLayanan = scanner.nextLine();
        Layanan layanan = new Layanan(namaLayanan);
        daftarLayanan.add(layanan);
        return layanan;
    } else {
        System.out.println("Pilih Layanan:");
        for (int i = 0; i < daftarLayanan.size(); i++) {
            System.out.printf("%d. %s%n", (i + 1),
daftarLayanan.get(i).getNama());
        }
        System.out.print("Pilih Layanan (masukkan nomor):");
    });
    int pilihan = scanner.nextInt();
    scanner.nextLine(); // Konsumsi newline
    return daftarLayanan.get(pilihan - 1);
}

private void tampilkanSemuaKendaraan() {
    if (daftarKendaraan.isEmpty()) {
        System.out.println("Tidak ada kendaraan yang
terdaftar.");
    } else {
        System.out.println("\n== Daftar Kendaraan ==");
        for (int i = 0; i < daftarKendaraan.size(); i++) {
            System.out.printf("%d. %s%n", (i + 1),
daftarKendaraan.get(i));
        }
    }
}

private void editKendaraan() {
    tampilkanSemuaKendaraan();
```



TUGAS PRAKTIKUM

```
System.out.print("Pilih kendaraan yang ingin diedit  
(masukkan nomor): ");  
int pilihan = scanner.nextInt();  
scanner.nextLine(); // Konsumsi newline  
  
if (pilihan < 1 || pilihan > daftarKendaraan.size()) {  
    System.out.println("Pilihan tidak valid.");  
    return;  
}  
  
Kendaraan kendaraan = daftarKendaraan.get(pilihan - 1);  
System.out.print("Masukkan Nama Kendaraan Baru: ");  
String namaBaru = scanner.nextLine();  
kendaraan.setNama(namaBaru);  
  
kendaraan.setMekanik(pilihMekanik());  
kendaraan.setLayanan(pilihLayanan());  
  
System.out.println("Kendaraan berhasil diperbarui!");  
}  
  
private void hapusKendaraan() {  
    tampilkanSemuaKendaraan();  
    System.out.print("Pilih kendaraan yang ingin dihapus  
(masukkan nomor): ");  
    int pilihan = scanner.nextInt();  
  
    if (pilihan < 1 || pilihan > daftarKendaraan.size()) {  
        System.out.println("Pilihan tidak valid.");  
        return;  
    }  
  
    daftarKendaraan.remove(pilihan - 1);  
    System.out.println("Kendaraan berhasil dihapus!");  
}  
  
private void simpanDataKeFile() {  
    try (PrintWriter writer = new PrintWriter(new  
FileWriter(FILE_NAME))) {
```



TUGAS PRAKTIKUM

```
        for (Kendaraan kendaraan : daftarKendaraan) {
            writer.printf("%s;%s;%s%n", kendaraan.getNama(),
kendaraan.getMekanik().getNama(),
kendaraan.getLayanan().getNama());
        }
        System.out.println("Data berhasil disimpan ke
file.");
    } catch (IOException e) {
        System.out.println("Terjadi kesalahan saat menyimpan
data ke file: " + e.getMessage());
    }
}

private void bacaDataDariFile() {
    File file = new File(FILE_NAME);
    if (!file.exists()) {
        System.out.println("File data tidak ditemukan.
Membuat file baru saat menyimpan.");
        return;
    }

    try (BufferedReader reader = new BufferedReader(new
FileReader(file))) {
        String line;
        while ((line = reader.readLine()) != null) {
            String[] data = line.split(";");
            String namaKendaraan = data[0];
            String namaMekanik = data[1];
            String namaLayanan = data[2];

            Mekanik mekanik =
daftarMekanik.stream().filter(m ->
m.getNama().equals(namaMekanik)).findFirst().orElseGet(() -> {
                Mekanik m = new Mekanik(namaMekanik);
                daftarMekanik.add(m);
                return m;
            });
        }
    }
}
```



TUGAS PRAKTIKUM

```
Layanan layanan =  
daftarLayanan.stream().filter(l ->  
l.getNama().equals(namaLayanan)).findFirst().orElseGet(() -> {  
    Layanan l = new Layanan(namaLayanan);  
    daftarLayanan.add(l);  
    return l;  
});  
  
Kendaraan kendaraan = new  
Kendaraan(namaKendaraan, mekanik, layanan);  
daftarKendaraan.add(kendaraan);  
}  
System.out.println("Data berhasil dibaca dari  
file.");  
} catch (IOException | ArrayIndexOutOfBoundsException e)  
{  
    System.out.println("Terjadi kesalahan saat membaca  
data dari file: " + e.getMessage());  
}  
}  
}
```

Penjelasan

Memambahkan method simpanKeFile() dan method bacaDariFile() untuk menulis dan membaca data dari file dataBengkel.txt. Menggunakan IOException untuk menangani kesalahan file tidak dapat dibuka/ditulis. Sebelum keluar, program menyimpan data kendaraan ke file dataBengkel.txt menggunakan method simpanKeFile().

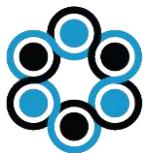
Output



*dataBengkel.txt - Notepad

File Edit Format View Help

1. Nama Kendaraan: Porsche, Mekanik: Zay, Layanan: Ganti Mesin



TUGAS PRAKTIKUM

```
== Daftar Kendaraan ==
1. Nama Kendaraan: Porsche, Mekanik: Zay, Layanan: Ganti Mesin

== Sistem Informasi Bengkel ==
1. Tambah Kendaraan
2. Tampilkan Kendaraan
3. Edit Kendaraan
4. Hapus Kendaraan
5. Keluar
Pilih menu: 3
Masukkan nomor kendaraan yang ingin diedit: 1
Masukkan Nama Baru Kendaraan: Lamboghirni
Masukkan Nama Baru Mekanik: Fauzi
Masukkan Layanan Baru: Cuci Mobil
Kendaraan berhasil diedit!

== Sistem Informasi Bengkel ==
1. Tambah Kendaraan
2. Tampilkan Kendaraan
3. Edit Kendaraan
4. Hapus Kendaraan
5. Keluar
Pilih menu: 5
Terima kasih telah menggunakan sistem!
Data berhasil disimpan ke file.
```



TUGAS PRAKTIKUM

Soal Praktikum

5. Sebuah toko penyewaan alat camping ingin membuat aplikasi sederhana untuk mempermudah proses penyewaan alat camping. Aplikasi ini akan digunakan oleh kasir untuk mencatat penyewaan alat camping. Berikut fitur yang harus dimiliki aplikasi:
- Aplikasi memiliki daftar alat camping lengkap dengan nama alat, jenis, dan harga sewa per hari.
 - Kasir dapat menambahkan alat camping yang disewa pelanggan ke dalam daftar penyewaan.
 - Aplikasi akan menghitung total harga dari semua alat yang disewa berdasarkan durasi sewa.
 - Kasir dapat membatalkan salah satu alat yang disewa jika terjadi kesalahan input.

Ketentuan:

- Gunakan GUI (Swing) untuk antarmuka aplikasi.
- Gunakan arsitektur MVC (Model-View-Controller).
- Validasi input, seperti jumlah alat yang disewa harus berupa angka yang valid, durasi sewa tidak boleh lebih dari 10 hari.
- Tampilkan pesan konfirmasi untuk pembatalan alat.

Jawaban

Ketik jawaban disini ...

Source Code

```
package model;  
  
public class AlatCamping {
```



TUGAS PRAKTIKUM

```
private String nama;
private String jenis;
private double hargaSewa;

public AlatCamping(String nama, String jenis, double
hargaSewa) {
    this.nama = nama;
    this.jenis = jenis;
    this.hargaSewa = hargaSewa;
}

public String getNama() {
    return nama;
}

public String getJenis() {
    return jenis;
}

public double getHargaSewa() {
    return hargaSewa;
}

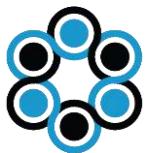
public void setNama(String nama) {
    this.nama = nama;
}

public void setJenis(String jenis) {
    this.jenis = jenis;
}

public void setHargaSewa(double hargaSewa) {
    this.hargaSewa = hargaSewa;
}
}

package model;

import java.util.ArrayList;
```



TUGAS PRAKTIKUM

```
import javax.swing.DefaultListModel;

public class PenyewaanModel {
    private ArrayList<AlatCamping> daftarAlat = new
ArrayList<>();
    private DefaultListModel<String> daftarSewa = new
DefaultListModel<>();

    public void tambahAlat(AlatCamping alat) {
        daftarAlat.add(alat);
    }

    public ArrayList<AlatCamping> getDaftarAlat() {
        return daftarAlat;
    }

    public void tambahSewa(String item) {
        daftarSewa.addElement(item);
    }

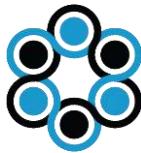
    public void hapusSewa(int index) {
        if (index >= 0 && index < daftarSewa.getSize()) {
            daftarSewa.remove(index);
        }
    }

    public DefaultListModel<String> getDaftarSewa() {
        return daftarSewa;
    }
}

package view;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionListener;

public class PenyewaanView extends JFrame {
    private JComboBox<String> alatComboBox = new JComboBox<>();
```



TUGAS PRAKTIKUM

```
private JTextField durasiField = new JTextField(10);
private JButton tambahButton = new JButton("Tambah");
private JButton hapusButton = new JButton("Hapus");
private JButton hitungButton = new JButton("Hitung Total");
private JLabel totalLabel = new JLabel("Total: Rp 0");
private JList<String> sewaList = new JList<>(new
DefaultListModel<>());

public PenyewaanView() {
    setTitle("Aplikasi Penyewaan Alat Camping");
    setSize(500, 400);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLayout(new BorderLayout());

    // Panel atas
    JPanel atasPanel = new JPanel();
    atasPanel.setLayout(new GridLayout(3, 2));
    atasPanel.add(new JLabel("Pilih Alat:"));
    atasPanel.add(alatComboBox);
    atasPanel.add(new JLabel("Durasi (hari):"));
    atasPanel.add(durasiField);
    atasPanel.add(tambahButton);
    atasPanel.add(hapusButton);

    // Panel tengah
    JPanel tengahPanel = new JPanel(new BorderLayout());
    tengahPanel.add(new JLabel("Daftar Penyewaan:"),
BorderLayout.NORTH);
    tengahPanel.add(new JScrollPane(sewaList),
BorderLayout.CENTER);

    // Panel bawah
    JPanel bawahPanel = new JPanel();
    bawahPanel.add(hitungButton);
    bawahPanel.add(totalLabel);

    // Tambahkan panel ke frame
    add(atasPanel, BorderLayout.NORTH);
    add(tengahPanel, BorderLayout.CENTER);
```



TUGAS PRAKTIKUM

```
        add(bawahPanel, BorderLayout.SOUTH);
    }

    public void setComboBoxItems(String[] items) {
        alatComboBox.setModel(new
DefaultComboBoxModel<>(items));
    }

    public String getSelectedAlat() {
        return (String) alatComboBox.getSelectedItem();
    }

    public String getDurasi() {
        return durasiField.getText();
    }

    public void setTotalLabel(String text) {
        totalLabel.setText(text);
    }

    public void setSewaListModel(DefaultListModel<String> model)
{
    sewaList.setModel(model);
}

    public int getSelectedSewaIndex() {
        return sewaList.getSelectedIndex();
    }

    public void tambahListener(ActionListener listener) {
        tambahButton.addActionListener(listener);
    }

    public void hapusListener(ActionListener listener) {
        hapusButton.addActionListener(listener);
    }

    public void hitungListener(ActionListener listener) {
        hitungButton.addActionListener(listener);
    }
}
```



TUGAS PRAKTIKUM

```
    }

}

package controller;

import model.AlatCamping;
import model.PenyewaanModel;
import view.PenyewaanView;

import javax.swing.*;

public class PenyewaanController {
    private PenyewaanModel model;
    private PenyewaanView view;

    public PenyewaanController(PenyewaanModel model,
PenyewaanView view) {
        this.model = model;
        this.view = view;

        // Set daftar alat camping
        model.tambahAlat(new AlatCamping("Tenda",
"Perlengkapan", 50000));
        model.tambahAlat(new AlatCamping("Kompor", "Peralatan
Masak", 30000));
        model.tambahAlat(new AlatCamping("Matras",
"Perlengkapan", 20000));
        updateComboBox();

        // Set listener
        view.addTambahListener(e -> tambahSewa());
        view.addHapusListener(e -> hapusSewa());
        view.addHitungListener(e -> hitungTotal());
    }

    private void updateComboBox() {
        String[] items = model.getDaftarAlat().stream()
            .map(AlatCamping::getNama)
            .toArray(String[]::new);
    }
}
```



TUGAS PRAKTIKUM

```
view.setComboBoxItems(items);
}

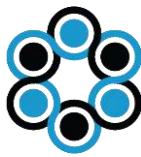
private void tambahSewa() {
    String namaAlat = view.getSelectedAlat();
    String durasiStr = view.getDurasi();

    try {
        int durasi = Integer.parseInt(durasiStr);
        if (durasi <= 0 || durasi > 10) {
            JOptionPane.showMessageDialog(view, "Durasi harus antara 1-10 hari!");
            return;
        }

        AlatCamping alat = model.getDaftarAlat().stream()
            .filter(a -> a.getNama().equals(namaAlat))
            .findFirst()
            .orElse(null);

        if (alat != null) {
            String item = String.format("%s - %s (Rp %.0f x %d hari)", alat.getNama(), alat.getJenis(), alat.getHargaSewa(), durasi);
            model.tambahSewa(item);
            view.setSewaListModel(model.getDaftarSewa());
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(view, "Durasi harus berupa angka!");
    }
}

private void hapusSewa() {
    int index = view.getSelectedSewaIndex();
    if (index < 0) {
        JOptionPane.showMessageDialog(view, "Pilih item yang ingin dihapus!");
        return;
    }
}
```



TUGAS PRAKTIKUM

```
    }

    int confirm = JOptionPane.showConfirmDialog(view,
"Apakah Anda yakin ingin menghapus item ini?");
    if (confirm == JOptionPane.YES_OPTION) {
        model.hapusSewa(index);
        view.setSewaListModel(model.getDaftarSewa());
    }
}

private void hitungTotal() {
    double total = 0;
    for (int i = 0; i < model.getDaftarSewa().getSiz();
i++) {
        String item = model.getDaftarSewa().getElementAt(i);
        String[] parts = item.split(" x ");
        double hargaPerHari =
Double.parseDouble(parts[0].split("Rp ")[1].replaceAll("[^0-9]",
 ""));
        int durasi = Integer.parseInt(parts[1].split(
")[0]);
        total += hargaPerHari * durasi;
    }
    view.setTotalLabel("Total: Rp " + total);
}
}

import controller.PenyewaanController;
import model.*;
import view.*;

public class Main {
    public static void main(String[] args) {
        PenyewaanModel model = new PenyewaanModel();
        PenyewaanView view = new PenyewaanView();
        PenyewaanController controller = new
PenyewaanController(model, view);
        view.setVisible(true);
    }
}
```



TUGAS PRAKTIKUM

}

Penjelasan

AlatCamping pada model memakili data alat camping dengan atribut seperti nama, jenis, dan hargaSewa per hari. PenyewaanModel pada model menangani data penyewaan yang mencakup daftar alat camping pada method daftarAlat dan daftar alat yang telah disewa pada method daftarSewa. PenyewaanView pada view menyediakan antarmuka pengguna menggunakan komponen Swing, ada beberapa juga elemen UI pada PenyewaanView. PenyewaanControl pada controller menghubungkan antara Model dan View. Kasir dapat menghapus alat tersebut dari daftar penyewaan setelah mendapatkan konfirmasi. Input durasi disaring untuk memastikan angka yang dimasukkan valid (antara 1 hingga 10 hari), jika durasi tidak valid atau alat tidak dipilih, aplikasi akan menampilkan pesan kesalahan.

Output

The screenshot shows a Java Swing application window titled 'Aplikasi Penyewaan Alat Camping'. The interface includes:

- A dropdown menu labeled 'Pilih Alat:' containing the option 'Tenda'.
- An input field labeled 'Durasi (hari):' with the value '3'.
- Two buttons: 'Tambah' and 'Hapus'.
- A list titled 'Daftar Penyewaan:' containing three items:
 - Kompor - Peralatan Masak (Rp 30000 x 3 hari)
 - Matras - Perlengkapan (Rp 20000 x 3 hari)
 - Tenda - Perlengkapan (Rp 50000 x 3 hari)
- A button labeled 'Hitung Total'.
- A display area showing 'Total: Rp 300000,0'.

TUGAS AKHIR



PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK PERIODE X

**MANAJEMEN PENYEWAAN ALAT OUTDOOR
(CAMP & HIKING)**

Nama : Fauzi Saputra

NPM : 06.2023.1.07748



TUGAS AKHIR

PENDAHULUAN

Deskripsi Umum



Studi kasus pemrograman dalam manajemen penyewaan alat outdoor adalah analisis mendalam tentang bagaimana teknologi dan pemrograman komputer digunakan untuk meningkatkan efisiensi, dan pengelolaan operasional dalam penyewaan alat outdoor. Dalam studi kasus ini, akan mempelajari berbagai aspek, seperti mengelola data produk, penyewaan secara online, dan integrasi teknologi lainnya.

Konsep Object Oriented Programming yang Digunakan

Object-Oriented Programming (OOP) atau diartikan dalam bahasa indonesia yaitu Pemrograman Berorientasi Objek adalah paradigma pemrograman tingkat lanjut setelah paradigma prosedural yang berfokus pada permodelan dunia nyata ke dalam bentuk objek-objek yang memiliki atribut (data) dan method (fungsi) yang berkaitan satu sama lain. OOP dapat membantu dalam mengorganisasi dan mengelola kode dengan lebih terstruktur dan mudah dimengerti.



TUGAS AKHIR

ANALISIS PERSIAPAN

Topik Bahasan

1. Modul 1: Pengenalan Java OOP
2. Modul 2: Class dan Bagiannya
3. Modul 3: Constructor dan Object

Tujuan Pertemuan

Tujuan dari Pertemuan 1 (satu) ini bertujuan untuk memberikan pemahaman dasar tentang pemrograman berorientasi objek dengan bahasa pemrograman Java, mencakup konsep dasar OOP, pembuatan Class dan bagiannya, serta penggunaan Constructor dan Object dalam pengembangan perangkat lunak.

Isi Pertemuan

```
public class Produk{  
    private String nama;  
    private String merek;  
    private String jenis;  
    private double harga;  
    private String deskripsi;  
    private int stok;  
  
    public Produk (String nama, String merek, String jenis,  
double harga, String deskripsi, int stok) {  
        this.nama = nama;  
        this.merek = merek;  
        this.jenis = jenis;  
        this.harga = harga;  
        this.deskripsi = deskripsi;  
        this.stok = stok;  
    }  
}
```



TUGAS AKHIR

Penjelasan

Kode di atas adalah implementasi dari sebuah kelas Java yang bervariabel “Produk”. Kelas ini memiliki beberapa atribut, yaitu “nama” (untuk menyimpan nama produk), “merek” (untuk menyimpan merek produk), “jenis” (untuk menyimpan jenis produk), “harga” (untuk menyimpan harga produk), “stok” (untuk menyimpan stok ketersediaan produk). Selain itu, kelas ini memiliki sebuah konstruktor yang digunakan untuk membuat objek Produk baru dengan menginisialisasi atribut-atributnya berdasarkan nilai-nilai yang diberikan saat objek dibuat. Dengan demikian, kelas ini digunakan untuk mempresentasikan informasi tentang Produk dalam suatu sistem program.

```
public class CustomerModel {  
    private String nomorHp;  
    private String alamat;  
  
    public CustomerModel(String nomorHp, String alamat) {  
        this.nomorHp = nomorHp;  
        this.alamat = alamat;  
    }  
}
```

Penjelasan

Pada kode di atas merupakan implementasi dari sebuah kelas Java yang bervariabel “CustomerModel”. Kelas ini memiliki beberapa atribut, yaitu “nomorHp” (untuk menyimpan nomor hp customer), “alamat” (untuk menyimpan alamat customer). Kelas ini juga memiliki sebuah konstruktor yang digunakan untuk membuat objek CustomerModel baru dengan menginisialisasi atribut-atributnya berdasarkan nilai-nilai yang diberikan saat objek dibuat.



TUGAS AKHIR

IMPLEMENTASI

Topik Bahasan

1. Modul 4: Input Process Output
2. Modul 5: Data Collection

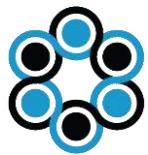
Tujuan Pertemuan

Modul 4: “Input Process Output” pada bahasa pemrograman Java input merupakan suatu proses memasukkan data ke dalam sebuah program komputer baik dilakukan secara statis maupun dinamis (input dari user). Pada process berisi sebuah logika-logika dari program yang kita buat, yaitu seperti pengujian, dan perulangan. Dan untuk membuat sebuah output program dapat menggunakan perintah “System.out” yang biasanya orang-orang menyebutnya dengan “sout” atau “sysout”. Sementara itu, Modul 5: “Data Collection” merujuk pada struktur data yang digunakan untuk mengelompokkan dan menyimpan sejumlah elemen atau objek dalam satu kesatuan. Data Collection memungkinkan kita untuk mengelola, mengakses, dan memanipulasi sejumlah besar data dengan lebih terstruktur dan efisien.

Korelasi antara kedua modul ini terletak pada fakta bahwa “Input Process Input” memberikan dasar untuk memahami bagaimana data diproses dalam bahasa pemrograman Java, sementara “Data Collection” melengkapi pemahaman ini dengan mengajarkan cara mendapatkan data yang dibutuhkan dalam proses tersebut.

Hasil Implementasi

```
public class AdminController {  
    private List<Produk> produks;  
    private List<Penyewaan> rentals;
```



TUGAS AKHIR

```
public AdminController() {  
    produks = new ArrayList<>();  
    rentals = new ArrayList<>();  
}
```

Penjelasan

Pada kode program ini memiliki kelas berupa `ArrayList` yang menyimpan objek dari kelas terkait dengan operasional Admin, yaitu:

1. ‘List<Produk> produks’ : sebuah list yang akan menyimpan objek-objek dari kelas Produk. List ini berfungsi untuk menampung data produk yang ada.
2. ‘produks’ : sebuah objek `ArrayList` kosong dan menginisialisasi variabel produks untuk menampung data produk.



TUGAS AKHIR

Topik Bahasan

1. Modul 6: Encapsulation
2. Modul 7: Inheritance

Tujuan Pertemuan

Modul 6: “Encapsulation” pada bahasa pemrograman Java adalah untuk memahami konsep dasar pemaketan data dan metode kelas dalam kelas, sehingga dapat menjaga keamanan dan integritas data, meningkatkan modularitas, memudahkan keterbacaan kode serta mengendalikan akses terhadap data dan memastikan data hanya bisa dimodifikasi dengan cara yang valid. Sementara itu, tujuan dari Modul 7: “Inheritance” adalah untuk memungkinkan sebuah kelas turunan (subclass) mewarisi atribut dan metode dari kelas lain/induk (superclass).

Korelasi antar kedua modul ini terletak pada fakta bahwa encapsulation membantu melindungi data dalam kelas, sementara inheritance memungkinkan pembuatan hierarki kelas yang dapat memanfaatkan sifat dari metode yang sudah ada, menciptakan hubungan yang kuat antara objek dan mengurangi duplikasi kode dalam pemrograman Java. Dengan encapsulation, meskipun data diwariskan melalui inheritance, akses tetap terkontrol dan aman, keduanya memainkan peran penting dalam pembangunan aplikasi yang efisien dan modular.

Hasil Implementasi

```
public class CustomerModel extends User {  
    private String nomorHp;  
    private String alamat;  
  
    public CustomerModel(String username, String password,  
String nomorHp, String alamat) {  
        super(username, password);  
        this.nomorHp = nomorHp;  
        this.alamat = alamat;
```



TUGAS AKHIR

```
    }
    public String getNomorHp() {
        return nomorHp;
    }
    public void setNomorHp(String nomorHp) {
        this.nomorHp = nomorHp;
    }
    public String getAlamat() {
        return alamat;
    }
    public void setAlamat(String alamat) {
        this.alamat = alamat;
    }
}
```

Penjelasan

Pada kelas “CustomerModel” merupakan turunan dari kelas “User”. Dalam konteks pemarisan ini, kelas “CustomerModel” adalah kelas turunan (subclass) dari kelas induk “User” (superclass), ini berarti bahwa kelas “CustomerModel” mewarisi atribut-atribut dan metode-metode yang ada dalam kelas “User”. Selain itu, kelas “CustomerModel” memiliki atribut tambahan berupa “nomorHp” dan “alamat” yang berkaitan dengan informasi tambahan tentang tamu yang tidak ada dalam kelas “User”.

Poin pentingnya adalah pewarisan ditunjukkan melalui kata kunci extends super() dalam konstruktor memastikan kelas induk diinisialisasi dengan benar. Encapsulation diimplementasikan melalui access modifier private. Struktur ini memungkinkan untuk membuat objek khusus dengan atribut tambahan diluar “username” dan “password”.



TUGAS AKHIR

Topik Bahasan

1. Modul 8: Relasi Antar Class
2. Modul 9: Polymorphism

Tujuan Pertemuan

Tujuan dari Modul 8: “Relasi Antar Class” berfokus pada penyembunyian suatu kelas dan membatasi akses langsung ke data melalui modifier private. Sementara itu, Modul 9: “Polymorphism” memungkinkan objek untuk bereaksi secara berbeda terhadap method yang sama, memberikan fleksibilitas dalam desain sistem.

Korelasi antara kedua modul ini terletak pada fakta bahwa dengan menggabungkan kedua konsep ini, pengembang dapat membuat sistem yang tidak hanya aman dan terlindungi, tetapi juga sangat adaptif terhadap perubahan dan pengembangan lebih lanjut. Keduanya dapat memainkan peran penting dalam kemampuan untuk membuat kode yang lebih abstrak dan modular.

Hasil Implementasi

```
public class Penyewaan {  
    private CustomerModel customer;  
    private Keranjang keranjang;  
    private LocalDate tanggalSewa;  
    private LocalDate tanggalPengembalian;  
    private String metodePengiriman;  
    private String metodePembayaran;  
    private String status;  
    private double denda;  
  
    public Penyewaan(Customer customer, Keranjang keranjang,  
        LocalDate tanggalSewa, LocalDate tanggalPengembalian,  
        String metodePengiriman, String  
        metodePembayaran) {
```



TUGAS AKHIR

```
        this.customer = customer;
        this.keranjang = keranjang;
        this.tanggalSewa = tanggalSewa;
        this.tanggalPengembalian = tanggalPengembalian;
        this.metodePengiriman = metodePengiriman;
        this.metodePembayaran = metodePembayaran;
        this.status = "Diantar";
        this.denda = hitungDenda(tanggalPengembalian,
tanggalSewa);
    }

    public String getCustomer() {
        return customer;
    }
    public double hitungDenda(LocalDate tanggalPengembalian,
LocalDate tanggalSewa) {
        if (tanggalPengembalian.isAfter(tanggalSewa)) {
            long terlambat =
ChronoUnit.DAYS.between(tanggalSewa, tanggalPengembalian);
            return terlambat * 10000; // Rp 10.000 per hari
        }
        return 0;
    }
    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("Penyewaan oleh:
").append(customer).append("\n");
        sb.append("Keranjang: ").append(keranjang).append("\n");
        sb.append("Tanggal Sewa:
").append(tanggalSewa).append("\n");
        sb.append("Tanggal Pengembalian:
").append(tanggalPengembalian).append("\n");
        sb.append("Status: ").append(status).append("\n");
        sb.append("Total Biaya:
").append(totalCost).append("\n");
        sb.append("Daftar Item:\n");
        for (Produk keranjang : keranjang) {
            sb.append("-
").append(keranjang.getCustomer()).append("\n");
        }
    }
}
```



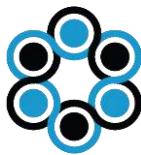
TUGAS AKHIR

```
    }  
    return sb.toString();  
}
```

Penjelasan

Kelas “Penyewaan” memiliki dua atribut yaitu “customer” dan “keranjang” yang masing-masing merupakan objek dari kelas “CustomerModel” dan “Keranjang”, yang berarti kelas “Penyewaan” memiliki refensi ke objek-objek dari kelas lain untuk mempresentasikan data customer dan keranjang yang terkait dengan penyewaan.

Selain itu, kelas ini juga memiliki metode Override “toString” mengoverride metode bawaan dari kelas “Object” untuk memberikan representasi string yang mudah dibaca, dan berisi informasi lengkap tentang penyewaan. Kesuluruhan desain ini mendemonstrasikan fleksibilitas dan hubungan yang kuat antara kelas-kelas dalam suatu sistem.



TUGAS AKHIR

Topik Bahasan

1. Modul 10: Package
2. Modul 11: Abstraction dan Interfaces

Tujuan Pertemuan

Tujuan yang dapat dicapai dari Modul 10: “Package” adalah untuk mengelompokkan kelas-kelas yang memiliki fungsi atau tujuan serupa ke dalam paket, yang juga membantu juga dalam pemeliharaan dan manajemen yang lebih baik. Sedangkan dari Modul 11: “Abstraction dan Interfaces”, tujuannya untuk memungkinkan fleksibilitas dan modularitas dalam sistem, karena berbagai kelas yang berbeda dapat berinteraksi melalui antarmuka (interfaces) atau abstraksi (abstraction) tanpa bergantung pada implementasi spesifik, yang juga dapat diorganisasi secara lebih efisien melalui penggunaan paket.

Korelasi antara kedua modul ini terletak pada integrasi konsep paket dengan penggunaan antarmuka dan abstraksi, membantu menciptakan struktur kode yang lebih terstruktur, mudah dipelihara, dan sesuai prinsip OOP.

Hasil Implementasi

```
package models;

public abstract class User {
    private String username;
    private String password;

    public User(String username, String password) {
        this.username = username;
        this.password = password;
    }
}
```



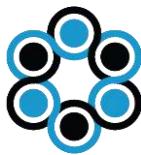
TUGAS AKHIR

Penjelasan

```
▽ models
  J AdminModel.java
  J CustomerModel.cl...
  J CustomerModel.java
  J Penyewaan.class
  J Penyewaan.java
  J Produk.class
  J Produk.java
  J User.class
  J User.java
```

Didalam terdapat banyak kelas-kelas, yaitu ada “AdminModel.java”, “CustomerModel.java”, “Penyewaan.java”, “Produk.java” dan “User.java”.

Kode program ini merupakan implementasi konsep abstraksi dalam Java melalui penggunaan kelas abstrak. Kelas “User” dideklarasikan sebagai abstrak dengan tujuan menjadi kerangka dasar untuk kelas-kelas turunan, seperti kelas “AdminModel” dan “CustomerModel”, yang akan menambahkan atribut spesifik. Kelas ini memiliki dua atribut yaitu “username” dan “password”, yang diatur melalui konstruktor untuk memastikan setiap objek kelas “User” memiliki data yang valid sejak awal. Atribut-atribut tersebut bersifat private, yang memastikan data hanya dapat diakses dan dimodifikasi secara tidak langsung melalui kelas itu sendiri. Dengan pendekatan ini, kode menjadi lebih modular dan terorganisir.



TUGAS AKHIR

Topik Bahasan

1. Modul 12: Exception Handling
2. Modul 13: Pemrograman Multimedia

Tujuan Pertemuan

Tujuan yang dapat dicapai dari Modul 12: “Exception Handling” untuk mengelola dan menangani kondisi kesalahan (error) atau kejadian tak terduga yang dapat terjadi selama eksekusi program tanpa menyebabkan program berhenti secara mendadak, sehingga dapat menciptakan aplikasi yang lebih handal dan dapat diandalkan. Sementara itu, Modul 13: “Pemrograman Multimedia” bertujuan untuk mengintegrasikan berbagai elemen media, seperti gambar, suara, video, animasi, dan teks, dalam satu aplikasi yang interaktif dan menarik.

Korelasi kedua modul ini terletak dalam pengembangan perangkat lunak, dimana exception handling memastikan aplikasi berjalan dengan lancar tanpa gangguan, sementara multimedia memperkaya antarmuka dan interaksi melalui integrasi berbagai elemen media.

Hasil Implementasi

```
import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

public class SimpleMultimediaExceptionHandling {

    public static void main(String[] args) {
        // Path ke gambar yang ingin dimuat
        String imagePath = "path/to/your/image.jpg"; // Ganti
        dengan path file gambar yang valid
    }
}
```



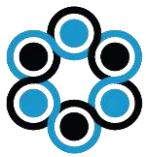
TUGAS AKHIR

```
// Coba memuat gambar dan menangani kemungkinan kesalahan
try {
    // Memuat gambar
    BufferedImage image = loadImage(imagePath);
    System.out.println("Gambar berhasil dimuat!");
    System.out.println("Lebar gambar: " +
image.getWidth() + "px");
    System.out.println("Tinggi gambar: " +
image.getHeight() + "px");
} catch (IOException e) {
    // Menangani kesalahan jika file tidak ditemukan atau format gambar tidak didukung
    System.err.println("Terjadi kesalahan saat memuat gambar: " + e.getMessage());
}
}

// Fungsi untuk memuat gambar dari file
public static BufferedImage loadImage(String path) throws IOException {
    File file = new File(path);
    if (!file.exists()) {
        throw new IOException("File gambar tidak ditemukan di path: " + path);
    }
    return ImageIO.read(file);
}
```

Penjelasan

Pada program ini diatur dengan path menuju file gambar yang dimuat, kemudian mencoba memuat gambar tersebut dengan menggunakan fungsi “loadImage()”. Fungsi ini membaca file gambar dari path yang diberikan menggunakan “ImageIO.read()”. Jika file tidak ditemukan atau ada masalah saat membaca gambar, maka fungsi ini akan melemparkan “IOException”, yang akan ditangkap oleh blok “catch” untuk menangani kesalahan tersebut. Jika gambar berhasil dimuat, program akan menampilkan pesan “Gambar berhasil dimuat!”.



TUGAS AKHIR

Jika terjadi kesalahan, pesan kesalahan yang relevan akan ditampilkan tentang masalah yang terjadi dengan pesan “Terjadi kesalahan saat memuat gambar: Tidak dapat membaca file gambar”.

Dengan demikian, program ini menggabungkan exception handling dengan pemrograman multimedia untuk memastikan aplikasi tetap berjalan dengan lancar dalam membaca dan mengelola file gambar.



TUGAS AKHIR

Topik Bahasan

1. Modul 14: Graphic User Interface (GUI)

Tujuan Pertemuan

Tujuan yang dapat dicapai dalam modul ini untuk membangun antarmuka pengguna dalam aplikasi Java. Swing dirancang untuk memberikan tampilan dan nuansa yang lebih konsisten dan lebih kaya. Dengan menguasai Java GUI, pengembang dapat menciptakan aplikasi yang lebih menarik dan mudah digunakan, yang merupakan kemampuan penting dalam pengembangan perangkat lunak modern.

Hasil Implementasi

```
package views;

import controllers.*;
import java.awt.*;
import javax.swing.*;

public class CustomerView {
    private CustomerController customerController;

    public CustomerView(CustomerController customerController) {
        this.customerController = customerController;
        createGUI();
    }

    private void createGUI() {
        JFrame frame = new JFrame("Customer Login");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);

        JPanel panel = new JPanel();
    }
}
```



TUGAS AKHIR

```
panel.setLayout(new GridLayout(3, 2));

JLabel usernameLabel = new JLabel("Username:");
JTextField usernameField = new JTextField();

JLabel passwordLabel = new JLabel("Password:");
JPasswordField passwordField = new JPasswordField();

JButton loginButton = new JButton("Login");
JButton registerButton = new JButton("Register");

loginButton.addActionListener(e -> {
    String username = usernameField.getText();
    String password = new
String(passwordField.getPassword());

    if (customerController.loginCustomer(username,
password)) {
        JOptionPane.showMessageDialog(frame, "Login
Successful! Welcome, " + username + "!");
    } else {
        JOptionPane.showMessageDialog(frame, "Login
Failed! Silahkan Coba Lagi.", "Error",
JOptionPane.ERROR_MESSAGE);
    }
});

registerButton.addActionListener(e -> {
    String username =
JOptionPane.showInputDialog("Masukkan Username:");
    String password =
JOptionPane.showInputDialog("Masukkan Password:");
    String nomorHp =
JOptionPane.showInputDialog("Masukkan Nomor Hp:");
    String alamat =
JOptionPane.showInputDialog("Masukkan Alamat:");

    customerController.registerCustomer(username,
password, nomorHp, alamat);
});
```



TUGAS AKHIR

```
JOptionPane.showMessageDialog(frame, "Registration  
Successful! Silahkan Login.");  
});  
  
panel.add(usernameLabel);  
panel.add(usernameField);  
panel.add(passwordLabel);  
panel.add(passwordField);  
panel.add(loginButton);  
panel.add(registerButton);  
  
frame.add(panel);  
frame.setVisible(true);  
}  
}
```

Penjelasan

Kode program di atas merupakan implementasi antarmuka pengguna berbasis Graphic User Interfaces (GUI) menggunakan pustaka Swing di bahasa pemrograman Java. Kelas “CustomerView” bertanggung jawab untuk membuat dan mengelola antarmuka pengguna, dalam metode “createGUI()”, elemen-elemen antarmuka seperti label, input teks, dan tombol diatur menggunakan layout grid agar tampil terorganisir. Tombol “Login” memicu proses autentikasi dengan memvalidasi username dan password melalui metode “loginCustomer()”. Jika login berhasil, pesan sukses ditampilkan, sedangkan jika gagal login akan muncul pesan kesalahan. Tombol “Register” memungkinkan pengguna untuk memasukkan data akun baru, yang kemudian diteruskan ke metode “registerCustomer()” untuk menyimpan data tersebut. Program ini memanfaatkan dialog input dan pesan pop-up untuk memberikan notifikasi kepada pengguna, membuat interaksi lebih intuitif dan interaktif.



TUGAS AKHIR

HASIL DAN DOKUMENTASI

Tampilan Login

The screenshot displays a window titled "Customer Login". The window has a standard title bar with minimize, maximize, and close buttons. Inside, there are two input fields: one for "Username" and one for "Password", both represented by large empty rectangles. At the bottom, there are two buttons: "Login" on the left and "Register" on the right, both in a light blue gradient.

Pada halaman login terdapat judul bernama “Customer Login” pada bagian atas, kemudian untuk melakukan proses autentikasi maka pengguna harus memasukkan data akun berupa Username dan Password, sebelum melakukan proses Login, pengguna harus melakukan Register terlebih dahulu untuk membuat akun baru.



TUGAS AKHIR

Tampilan Dashboard Aplikasi

The screenshot shows a window titled "Admin Dashboard". On the left, there is a vertical list of labels: "Nama Barang:", "Merek Barang:", "Jenis Barang:", "Harga Barang:", "Deskripsi Barang:", and "Stok:". To the right of each label is a corresponding input field represented by a horizontal line. At the bottom left of the input area, there is a blue button labeled "Tambah Produk".

Pada halaman dashboard terdapat judul bernama “Admin Dashboard” pada bagian atas. Antarmuka memiliki label seperti “Nama Barang”, “Merek Barang”, “Jenis Barang”, “Deskripsi Barang”, dan “Stok”, yang masing-masing menunjukkan data spesifik produk yang perlu diinput oleh admin, kemudian untuk melakukan proses autentikasi maka pengguna harus memasukkan data berupa nama, merek, jenis, deskripsi, dan jumlah stok. Bagian bawah layar dilengkapi dengan tombol “Tambah Produk”, yang berfungsi untuk memproses dan menyimpan data produk ke dalam sistem setelah semua data diisi.



KRITIK
&
SARAN

**LABORATORIUM REKAYASA PERANGKAT LUNAK
FAKULTAS TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI ADHI TAMA SURABAYA
2023**



Kritik & Saran

Laboratorium Rekayasa Perangkat Lunak (RPL) memiliki peran penting dalam mendukung kegiatan belajar mengajar dan pengembangan keterampilan teknis mahasiswa. Jadi, kritik saya terhadap Lab ini adalah koneksi internet yang sering bermasalah, sehingga menghambat pembelajaran. Selain itu, kebersihan laboratorium perlu ditingkatkan karena area kerja sering kali terlihat kurang terjaga. Pengelolaan jadwal penggunaan laboratorium juga menjadi masalah, karena beberapa kelompok mahasiswa tidak mendapatkan akses yang cukup.



Kritik & Saran

Kritik yang saya sampaikan kepada Aslab adalah kurangnya responsivitas beberapa asisten dalam menjawab pertanyaan mahasiswa, yang membuat proses belajar menjadi kurang optimal. Penjelasan yang terkadang terlalu cepat atau kurang detail juga menjadi kendala bagi mahasiswa dalam memahami konsep yang diajarkan. Selain itu, beberapa aslab terlihat kurang ramah dan profesional saat berinteraksi, sehingga menciptakan suasana yang kurang nyaman.