TUGAS & EVALUASI

Soal Tugas & Evaluasi

1. Jelaskan secara singkat tentang abstraction dan interfaces serta perbedaanya!

Jawaban

Abstraction merupakan fitur yang dapat mengabaikan fungsionalnya secara detail dengan tidak meghilangkan fungsionalnya secara umum

Interfaces memiliki pengertian dan fungsi yang hamper sama dengan class abstract, walaupun fungsi dari keduanya sama,tetapi ada perbedaan

Perbedaan:

Abstract : bisa berisi abstract dan non abstract method, kita harus menuliskan sendiri modifiednya

Interface : hanya boleh berisi abstract method, hanya bisa mendeklarasikan constant

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 maasiswa tersebut. Pada method non abstract gunakan operator ternary

Source Code

```
abstract class mahasiswa{
    abstract String nama();
    abstract int npm();
    abstract boolean isMhsAktif();

    public void tampilkanData(){
        System.out.println("Nama: " + nama());
        System.out.println("NPM: " + npm());
        System.out.println("MhsStatus: " + (isMhsAktif() ?
"Aktif" : "Non Aktif"));
    }
}

class MahasiswaAktif extends mahasiswa{
    private String nama;
    private int npm;
    private boolean isMhsAktif;

    MahasiswaAktif(String nama, int npm, boolean isMhsAktif){
        this.nama = nama;
        this.npm = npm;
        this.isMhsAktif = isMhsAktif;
    }

    String nama(){
        return nama;
    }
    int npm(){
        return npm;
    }
    boolean isMhsAktif(){
        return isMhsAktif;
    }
}

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in
the gutter.
public class Main {
    public static void main(String[] args) {
```

```
        mahasiswa mhs = new MahasiswaAktif("Adam", 123,
false);
        mhs.tampilkanData();
    }
}
```

awalnya membuat class abstract dengan nama mahasiswa dengan atribut nama, npm, isMhsAktif, setelah itu membuat tampilannya, seelah itu membuat class lagi dengan nama mahasiswaaktif dengan extends mahsiswa dengan string nama, npm, isMhsAktif, dan memanggil abstract nama, dan npm, dan bool mhsAktif, yg terakhit membuat objek di main untuk di inisialisasi

Output

```
Nama: Adam
NPM: 123
MhsStatus: Non Aktif
```

TUGAS & EVALUASI

Soal Tugas & Evaluasi

3. Buatlah abstract dengan nama transportasi, yang didalammnya terdapat atribnut nama dan kapasitas serta metod abstract untuk menampilkan cara memesan dan juga untuk menghitung tarif transportasinya

   Setelahnya, buat sub-class taksi, bus, dan keretaApi yang mengimplementasikan metod serta atribut superclass

   Note : masin -masing sub-class harus memiliki perilaku pemesanan dan penghitung tarif yang berbeda sesuai jenisnya

Source Code

```
package bab8No3;

import java.util.Scanner;

abstract class Transportasi {
    String nama;
    int kapasitas;

    public Transportasi(String nama, int kapasitas) {
        this.nama = nama;
        this.kapasitas = kapasitas;
    }

    abstract void caraMemesan();
    abstract double hitungTarif(int jarak);

    public void tampilkanInfo() {
        System.out.println("Nama Transportasi : " + nama);
        System.out.println("Kapasitas         : " + kapasitas
+ " orang");
    }
}

class Taksi extends Transportasi {
    double tarifPerKm;

    public Taksi(String nama, int kapasitas, double
tarifPerKm) {
        super(nama, kapasitas);
        this.tarifPerKm = tarifPerKm;
    }

    @Override
```

```
    void caraMemesan() {
        System.out.println("Cara Memesan: Hubungi melalui
aplikasi BlueBird atau panggil dari pinggir jalan.");
    }

    @Override
    double hitungTarif(int jarak) {
        return tarifPerKm * jarak;
    }
}

class Bus extends Transportasi {
    double tarifFlat;

    public Bus(String nama, int kapasitas, double tarifFlat) {
        super(nama, kapasitas);
        this.tarifFlat = tarifFlat;
    }

    @Override
    void caraMemesan() {
        System.out.println("Cara Memesan: Datang ke terminal
atau naik di halte bus.");
    }

    @Override
    double hitungTarif(int jarak) {
        return tarifFlat;
    }
}

class KeretaApi extends Transportasi {
    double tarifPerKm;

    public KeretaApi(String nama, int kapasitas, double
tarifPerKm) {
        super(nama, kapasitas);
        this.tarifPerKm = tarifPerKm;
    }

    @Override
    void caraMemesan() {
        System.out.println("Cara Memesan: Beli tiket secara
online atau di stasiun.");
    }

    @Override
    double hitungTarif(int jarak) {
        return tarifPerKm * jarak;
    }
}


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

TUGAS & EVALUASI

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

        Taksi taksi = new Taksi("Taksi Kota", 4, 5000);
        Bus bus = new Bus("Bus Trans", 40, 12000);
        KeretaApi kereta = new KeretaApi("Kereta Luxury", 200,
20000);

        System.out.println("=== PILIH TRANSPORTASI ===");
        System.out.println("1. Taksi");
        System.out.println("2. Bus");
        System.out.println("3. Kereta Api");
        System.out.print("Masukkan pilihan Anda: ");
        int pilihan = scanner.nextInt();

        System.out.print("Masukkan jarak perjalanan (dalam
km): ");
        int jarak = scanner.nextInt();

        System.out.println();

        switch (pilihan) {
            case 1:
                taksi.tampilkanInfo();
                taksi.caraMemesan();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + taksi.hitungTarif(jarak));
                break;
            case 2:
                bus.tampilkanInfo();
                bus.caraMemesan();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + bus.hitungTarif(jarak));
                break;
            case 3:
                kereta.tampilkanInfo();
                kereta.caraMemesan();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + kereta.hitungTarif(jarak));
                break;
            default:
                System.out.println("Pilihan tidak valid.");
                break;
        }

        scanner.close();
    }
}
```

Penjelasan

Program ini mendefinisikan sebuah sistem transportasi menggunakan konsep pemrograman berorientasi objek di Java, dengan menggunakan pewarisan (inheritance) dan polimorfisme. Ada kelas abstrak bernama `Transportasi`, yang

memiliki dua atribut, yaitu `nama` dan `kapasitas`, serta dua metode abstrak: `caraMemesan()` dan `hitungTarif(int jarak)`. Kelas ini bertindak sebagai dasar untuk transportasi umum lainnya. Kemudian, kelas `Taksi`, `Bus`, dan `KeretaApi` mewarisi kelas `Transportasi` dan mengimplementasikan metode-metode yang sudah didefinisikan di kelas induk, namun dengan perilaku yang berbeda sesuai jenis transportasi masing-masing.

Kelas `Taksi` menambahkan atribut `tarifPerKm` yang digunakan untuk menghitung tarif berdasarkan jarak yang ditempuh, sementara kelas `Bus` memiliki atribut `tarifFlat`, yang menetapkan tarif tetap tanpa memperhitungkan jarak. Di sisi lain, `KeretaApi` juga menggunakan tarif per kilometer seperti taksi. Masing-masing kelas juga mengimplementasikan metode `caraMemesan()`, yang menjelaskan cara memesan transportasi tersebut, apakah lewat aplikasi, terminal, atau stasiun.

Program utama memungkinkan pengguna untuk memilih jenis transportasi, memasukkan jarak yang akan ditempuh, dan kemudian menampilkan informasi terkait transportasi tersebut, termasuk nama, kapasitas, cara memesan, serta tarif yang dihitung berdasarkan jarak. Jika pengguna memilih transportasi yang tidak valid, program akan menampilkan pesan kesalahan. Dengan menggunakan objek-objek transportasi ini, program menggambarkan konsep pewarisan di mana kelas turunan mewarisi properti dan metode dari kelas induk, dan polimorfisme di mana metode yang sama (`caraMemesan()` dan `hitungTarif()`) memiliki implementasi yang berbeda di setiap kelas turunan.

TUGAS & EVALUASI

Output

```
=== PILIH TRANSPORTASI ===
1. Taksi
2. Bus
3. Kereta Api
Masukkan pilihan Anda: 3
Masukkan jarak perjalanan (dalam km): 13

Nama Transportasi : Kereta Luxury
Kapasitas         : 200 orang
Cara Memesan: Beli tiket secara online atau di stasiun.
Tarif untuk 13 km: Rp260000.0
```

TUGAS & EVALUASI

Soal Tugas & Evaluasi

4. Buatla class interface pembayaran dan transportasi berdasarkan studi kasus dari sooal no.3. kemudian implementasikan kedua class tersebut pada class taksi, bus, dan kereta api yang baru!
   Petunjuk :
   - Interface transportasi berisi e=methid oid bergerak() dan double hitungtarif()
   - Interface pembayaran berisi metid void bayar()

Source Code

```
package bab8No4;

import java.util.Scanner;

interface Transportasi {
    void bergerak();
    double hitungTarif(int jarak);
}

interface Pembayaran {
    void bayar();
}

class Taksi implements Transportasi, Pembayaran {
    String nama;
    int kapasitas;
    double tarifPerKm;

    public Taksi(String nama, int kapasitas, double
tarifPerKm) {
        this.nama = nama;
        this.kapasitas = kapasitas;
        this.tarifPerKm = tarifPerKm;
    }

    @Override
    public void bergerak() {
        System.out.println(nama + " bergerak menuju tujuan.");
    }

    @Override
    public double hitungTarif(int jarak) {
        return tarifPerKm * jarak;
    }

    @Override
    public void bayar() {
        System.out.println("Pembayaran taksi telah
```

```
berhasil.");
    }

    public void tampilkanInfo() {
        System.out.println("Nama Taksi : " + nama);
        System.out.println("Kapasitas  : " + kapasitas + "
orang");
    }
}

class Bus implements Transportasi, Pembayaran {
    String nama;
    int kapasitas;
    double tarifFlat;

    public Bus(String nama, int kapasitas, double tarifFlat) {
        this.nama = nama;
        this.kapasitas = kapasitas;
        this.tarifFlat = tarifFlat;
    }

    @Override
    public void bergerak() {
        System.out.println(nama + " bergerak dari terminal.");
    }

    @Override
    public double hitungTarif(int jarak) {
        return tarifFlat;
    }

    @Override
    public void bayar() {
        System.out.println("Pembayaran bus telah berhasil.");
    }

    public void tampilkanInfo() {
        System.out.println("Nama Bus   : " + nama);
        System.out.println("Kapasitas  : " + kapasitas + "
orang");
    }
}

class KeretaApi implements Transportasi, Pembayaran {
    String nama;
    int kapasitas;
    double tarifPerKm;

    public KeretaApi(String nama, int kapasitas, double
tarifPerKm) {
        this.nama = nama;
        this.kapasitas = kapasitas;
        this.tarifPerKm = tarifPerKm;
    }
```

```
    @Override
    public void bergerak() {
        System.out.println(nama + " sedang berjalan menuju
stasiun tujuan.");
    }

    @Override
    public double hitungTarif(int jarak) {
        return tarifPerKm * jarak;
    }

    @Override
    public void bayar() {
        System.out.println("Pembayaran kereta api telah
berhasil.");
    }

    public void tampilkanInfo() {
        System.out.println("Nama Kereta : " + nama);
        System.out.println("Kapasitas   : " + kapasitas + "
orang");
    }
}

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

        Taksi taksi = new Taksi("Taksi Kota", 4, 5000);
        Bus bus = new Bus("Bus Trans", 40, 12000);
        KeretaApi kereta = new KeretaApi("Kereta Luxury", 200,
20000);

        System.out.println("=== PILIH TRANSPORTASI ===");
        System.out.println("1. Taksi");
        System.out.println("2. Bus");
        System.out.println("3. Kereta Api");
        System.out.print("Masukkan pilihan Anda: ");
        int pilihan = scanner.nextInt();

        System.out.print("Masukkan jarak perjalanan (dalam
km): ");
        int jarak = scanner.nextInt();

        System.out.println();

        switch (pilihan) {
            case 1:
                taksi.tampilkanInfo();
                taksi.bergerak();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + taksi.hitungTarif(jarak));
                taksi.bayar();
                break;
            case 2:
```

TUGAS & EVALUASI

```
                bus.tampilkanInfo();
                bus.bergerak();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + bus.hitungTarif(jarak));
                bus.bayar();
                break;
            case 3:
                kereta.tampilkanInfo();
                kereta.bergerak();
                System.out.println("Tarif untuk " + jarak + "
km: Rp" + kereta.hitungTarif(jarak));
                kereta.bayar();
                break;
            default:
                System.out.println("Pilihan tidak valid.");
                break;
        }

        scanner.close();
    }
}
```

Penjelasan

Program ini menggunakan konsep interface di Java untuk mendefinisikan perilaku yang harus dimiliki oleh berbagai jenis transportasi dan pembayaran. Ada dua interface utama yang digunakan: Transportasi dan Pembayaran. Interface Transportasi mendefinisikan dua metode, yaitu bergerak() untuk menggambarkan bagaimana transportasi tersebut bergerak, dan hitungTarif(int jarak) untuk menghitung tarif berdasarkan jarak yang ditempuh. Sementara itu, interface Pembayaran hanya mendefinisikan satu metode, yaitu bayar(), yang digunakan untuk mengonfirmasi bahwa pembayaran telah berhasil dilakukan.

Kelas Taksi, Bus, dan KeretaApi masing-masing mengimplementasikan kedua interface ini. Setiap kelas memiliki atribut yang sesuai, seperti nama, kapasitas, dan tarif untuk taksi dan kereta api, serta tarif tetap untuk bus. Setiap kelas juga mengimplementasikan metode bergerak() untuk menjelaskan bagaimana transportasi tersebut bergerak (misalnya, taksi bergerak menuju tujuan, bus berangkat dari terminal, dan kereta api berjalan menuju stasiun tujuan), serta hitungTarif() yang menghitung tarif berdasarkan jarak. Kelas-kelas ini juga

TUGAS & EVALUASI

mengimplementasikan metode bayar() untuk mencetak konfirmasi bahwa pembayaran telah berhasil dilakukan.

Program utama meminta pengguna untuk memilih jenis transportasi, memasukkan jarak perjalanan, dan kemudian menampilkan informasi tentang transportasi yang dipilih, termasuk nama, kapasitas, cara bergerak, tarif yang dihitung berdasarkan jarak, dan konfirmasi pembayaran. Dengan menggunakan interface, program ini memastikan bahwa semua jenis transportasi memiliki perilaku yang konsisten meskipun cara mereka bergerak dan menghitung tarif bisa berbeda-beda. Program ini juga menunjukkan penerapan konsep polimorfisme karena metode yang sama, seperti bergerak(), hitungTarif(), dan bayar(), diimplementasikan secara berbeda pada setiap kelas yang mengimplementasikan interface tersebut.

Output

```
=== PILIH TRANSPORTASI ===
1. Taksi
2. Bus
3. Kereta Api
Masukkan pilihan Anda: 2
Masukkan jarak perjalanan (dalam km): 1

Nama Bus   : Bus Trans
Kapasitas  : 40 orang
Bus Trans bergerak dari terminal.
Tarif untuk 1 km: Rp12000.0
Pembayaran bus telah berhasil.
```