Nama : Rio Ferdinan
NIM : 00000070666

LATIHAN PERSIAPAN UAS OOP-LECTURE IF433-I

Jelaskan apa yang anda ketahui terkait dengan abstract class pada konsep OOP! Jawablah dengan selengkap-lengkapnya.

= Abstract class adalah sebuah kelas yang tidak dapat diinstansiasi secara langsung, artinya tidak dapat membuat objek dari kelas ini. Abstract class biasanya berisi satu atau lebih metode abstrak, yang merupakan metode yang dideklarasikan tetapi tidak diimplementasikan dalam abstract class itu sendiri. Kelas-kelas turunan dari abstract class harus mengimplementasikan metode-metode abstrak ini. Meskipun tidak dapat diinstansiasi, abstract class dapat memiliki konstruktor. Konstruktor ini biasanya digunakan untuk inisialisasi atribut atau melakukan beberapa setup yang diperlukan oleh subclass.

Kegunaan abstract class :

- Abstract class menyediakan kerangka kerja atau blueprint untuk kelas-kelas turunan. Ini memastikan bahwa kelas-kelas turunan mengimplementasikan metode-metode penting yang telah didefinisikan dalam abstract class.
- Abstract class membantu dalam mengatur struktur aplikasi dengan memberikan hierarki yang jelas.
- Abstract class memungkinkan penggunaan kembali kode dengan menyediakan metode dan properti yang dapat digunakan oleh berbagai subclass. Hal ini mengurangi duplikasi kode dan meningkatkan konsistensi.

Buatlah contoh kode penggunaan abstract class pada pemrograman Java! Desainlah contoh atau studi kasus anda sendiri!

```java
abstract class Hewan {
    // Metode abstrak
    public abstract void bersuara();

    // Metode dengan implementasi
    public void tidur() {
        System.out.println("Hewan sedang tidur");
    }
}

class Kucing extends Hewan {
    // Implementasi metode abstrak
    public void bersuara() {
        System.out.println("Meow");
    }
}

class Anjing extends Hewan {
    // Implementasi metode abstrak
    public void bersuara() {
        System.out.println("Guk Guk");
    }
}

public class Main {
    public static void main(String[] args) {
        Hewan kucing = new Kucing();
        kucing.bersuara(); // Output: Meow
        kucing.tidur();    // Output: Hewan sedang tidur

        Hewan anjing = new Anjing();
        anjing.bersuara(); // Output: Guk Guk
        anjing.tidur();    // Output: Hewan sedang tidur
    }
}
```

Jelaskan apa yang anda ketahui terkait dengan interface pada konsep OOP! Jawablah dengan selengkap-lengkapnya!

= Interface adalah sebuah kontrak yang mendefinisikan sekumpulan metode yang harus diimplementasikan oleh kelas-kelas yang mengimplementasikan interface tersebut. Interface tidak dapat memiliki implementasi metode sama sekali, semua metode dalam interface bersifat abstrak secara default (tidak memiliki tubuh/implementasi). Interface juga dapat mendefinisikan properti (atribut) dan konstanta, tetapi tidak dapat menyertakan implementasi dari metode atau properti tersebut.

Kegunaan interface :

- Interface mendefinisikan kontrak yang harus dipatuhi oleh kelas yang mengimplementasikannya.
- Interface memungkinkan polimorfisme, yaitu kemampuan untuk menggunakan objek dari kelas yang berbeda melalui referensi yang sama (tipe interface). Ini membuat kode lebih fleksibel dan mudah untuk dikembangkan.
- Interface mempromosikan penggunaan komposisi dan pengurangan ketergantungan antar kelas.

Apa perbedaan interface dan abstract class? Jelaskan dan berikan pros & cons penggunaan keduanya!

Perbedaan interface dan abstract class :

- Interface digunakan untuk mendefinisikan fungsi yang harus diimplementasikan oleh kelas yang mengimplementasi interface tersebut, memastikan bahwa semua kelas yang mengimplementasinya memiliki metode yang sama. Sedangkan abstract digunakan untuk menyediakan sebagian implementasi dasar yang dapat digunakan kembali oleh kelas turunan.
- Interface tidak dapat memiliki atribut non-konstan, hanya konstanta (final static variables). Sedangkan, Abstract class dapat memiliki atribut yang dapat digunakan oleh metode-metodenya.
- Interface dapat mengimplementasikan interface lainnya (Nested Interface), memungkinkan untuk menggabungkan beberapa kemampuan atau perilaku. Sedangkan, Abstract class dapat meng-extend class java lain dan mengimplementasikan beberapa interface
- Interface tidak bisa mengimplementasikan abstract class, sedangkan Abstract class dapat mengimplementasikan interface.
- Interface diimplementasikan dengan kata kunci “implements”, sedangkan Abstract class di-extend dengan kata kunci “extends”.

Pros Penggunaan Interface :

1. Multiple Inheritance:
   Memungkinkan sebuah kelas untuk mengimplementasikan beberapa interface, memberikan fleksibilitas dalam menggabungkan berbagai kemampuan.

2. Desain fleksibel:
   Membantu dalam merancang sistem yang sangat fleksibel dan dapat diperluas dengan mudah tanpa terikat pada hierarki kelas tertentu.
3. Kontrak yang Jelas:
   Menyediakan kontrak yang jelas untuk perilaku yang harus diimplementasikan oleh kelas, memastikan konsistensi.

Cons Penggunaan Interface :

1. Tidak ada implementasi:
   Semua metode harus diimplementasikan oleh kelas yang mengimplementasikan interface, yang dapat meningkatkan jumlah kode yang harus ditulis.
2. Tidak ada state:
   Tidak bisa menyimpan state (atribut) selain konstanta, sehingga tidak cocok untuk berbagi data atau status.

Pros Penggunaan Abstract Class :

1. Kode Reusable:
   Abstract class memungkinkan implementasi bersama, mengurangi duplikasi kode di antara subclass.
2. Memiliki Atribut:
   Dapat memiliki atribut dan metode konkret, sehingga lebih cocok untuk berbagi data atau status di antara subclass.
3. Tingkat Abstraksi yang Bervariasi:
   Dapat mendefinisikan metode abstrak dan konkret, memberikan lebih banyak fleksibilitas dalam desain.

Cons Penggunaan Abstract Class :

1. Single Inheritance:
   Kelas hanya dapat mewarisi dari satu abstract class, yang membatasi kemampuan untuk menggabungkan beberapa perilaku dari hierarki kelas yang berbeda.
2. Keterikatan pada Hierarki:
   Menyebabkan keterikatan pada hierarki kelas tertentu, yang dapat mengurangi fleksibilitas desain.

Buatlah contoh kode penggunaan interface pada pemrograman Java! Desainlah contoh atau studi kasus anda sendiri!

```java
// Interface Animal
interface Animal {
    void eat();
    void makeSound();
}

// Class untuk hewan jenis Singa
class Lion implements Animal {
    @Override
    public void eat() {
        System.out.println("Lion is eating meat.");
    }

    @Override
    public void makeSound() {
        System.out.println("Roar!");
    }
}

// Class untuk hewan jenis Gajah
class Elephant implements Animal {
    @Override
    public void eat() {
        System.out.println("Elephant is eating leaves and grass.");
    }

    @Override
    public void makeSound() {
        System.out.println("Trumpet!");
    }
}

// Class untuk hewan jenis Monyet
class Monkey implements Animal {
    @Override
    public void eat() {
        System.out.println("Monkey is eating fruits and insects.");
    }

    @Override
    public void makeSound() {
        System.out.println("Ooh ooh aah aah!");
```

```
    }
}

// Main class
public class Zoo {
    public static void main(String[] args) {
        // Membuat objek hewan
        Animal lion = new Lion();
        Animal elephant = new Elephant();
        Animal monkey = new Monkey();

        // Memanggil metode dari interface
        System.out.println("Lion:");
        lion.eat();
        lion.makeSound();

        System.out.println("\nElephant:");
        elephant.eat();
        elephant.makeSound();

        System.out.println("\nMonkey:");
        monkey.eat();
        monkey.makeSound();
    }
}
```

Apa yang dimaksud dengan reference assignment? Jelaskan dan berikan contohnya pada pemrograman Java!

= Reference assignment adalah proses di mana sebuah variabel diberi referensi ke objek atau nilai lain, yang berarti variabel tersebut menunjuk ke alamat memori dari objek atau nilai tersebut. Dengan cara ini, dua atau lebih variabel dapat merujuk pada objek yang sama di memori, sehingga perubahan yang dilakukan pada salah satu variabel akan tercermin pada variabel lainnya.

Ini berbeda dengan nilai assignment, di mana nilai dari satu variabel disalin ke variabel lainnya. Dalam reference assignment, kedua variabel akan merujuk ke alamat memori yang sama, sehingga keduanya akan berbagi nilai yang sama.

Contoh dalam Java :

```java
public class Main {
    public static void main(String[] args) {
        // Membuat objek pertama
        String objek1 = new String("Nilai 1");

        // Menggunakan reference assignment untuk membuat variabel baru
        // yang merujuk pada objek yang sama dengan variabel objek1
        String objek2 = objek1;

        // Mengubah nilai variabel objek1
        objek1 = "Nilai 2";

        // Mencetak nilai kedua variabel
        System.out.println("Nilai objek1: " + objek1); // Output: Nilai 2
        System.out.println("Nilai objek2: " + objek2); // Output: Nilai 1
    }
}
```

Apa yang dimaksud dengan Exception? Jelaskan!
= Exception adalah sebuah objek dalam pemrograman yang mewakili suatu keadaan anomali atau kesalahan yang terjadi selama eksekusi program. Ketika suatu kejadian yang tidak diharapkan terjadi, seperti kesalahan dalam logika program, operasi matematika yang tidak valid, atau masalah saat mengakses sumber daya eksternal seperti berkas atau jaringan, sebuah exception dapat dilemparkan.

Apa saja jenis-jenis exception pada Java? Jelaskan!

- Checked Exception: Checked exception adalah subclass dari kelas Exception yang harus dideklarasikan dalam blok try-catch atau diteruskan ke blok pemanggil dengan menggunakan kata kunci throws. Beberapa contoh checked exception di Java termasuk :
    1. IOException: Terjadi ketika ada kesalahan saat melakukan operasi I/O (Input/Output), seperti membaca atau menulis ke berkas.
    2. SQLException: Terjadi ketika ada kesalahan dalam operasi database.
    3. ClassNotFoundException: Terjadi ketika mencoba mengakses kelas yang tidak ada.
- Unchecked Exception: Unchecked exception adalah subclass dari kelas RuntimeException. Mereka tidak wajib dideklarasikan dalam blok try-catch atau diteruskan menggunakan throws, sehingga sering kali disebut sebagai "unchecked" karena tidak diperiksa oleh kompilator. Beberapa contoh unchecked exception di Java termasuk :
    1. NullPointerException: Terjadi ketika mencoba menggunakan referensi objek yang null.
    2. ArithmeticException: Terjadi ketika terjadi kesalahan matematika, seperti pembagian dengan nol.
    3. ArrayIndexOutOfBoundsException: Terjadi ketika mencoba mengakses indeks yang diluar batas pada sebuah array.

Berikan contoh penanganan exception pada Java beserta kode programnya!

```java
public class Main {
    public static void main(String[] args) {
        try {
            // Potensial terjadi kesalahan
            int hasil = bagi(10, 0);
            System.out.println("Hasil pembagian: " + hasil); // Tidak akan
mencapai baris ini
        } catch (ArithmeticException e) {
            // Exception handling, terjadi ketika pembagian dengan nol
            System.out.println("Terjadi kesalahan aritmatika: " +
e.getMessage());
        }
    }

    // Metode untuk melakukan pembagian
    public static int bagi(int pembilang, int penyebut) {
        return pembilang / penyebut; // Kemungkinan terjadi ArithmeticException
jika penyebut adalah 0
    }
}
```

Pilihlah salah satu studi kasus di bawah dan kemudian desainlah class diagram untuk system yang sesuai dengan studi kasus yang anda pilih!

- **Buatlah beberapa kelas yang relevan beserta atribut dan metode yang relevan untuk kelas tersebut!**
- **Class diagram harus memuat kelas abstract dan kelas interface!**

= Studi kasus yang dipilih adalah : Sistem informasi swalayan (gudang/jual-beli/karyawan).

Class Diagram :