Muhammad Sheva Kurnia Meazza

50424947

1IA16

Soal

1. Apa yang dimaksud paradigma Pemrograman dan java masuk kedalam jenis bahasa pemrograman apa.
2. Bagaimana konsep bahasa pemrograman java dan uraikan komponen atau elemen yang diperlukan dalam bahasa Pemrogaraman Java.
3. Apa yang harus diperhatikan object Orientid, PBO adalah object, class, atribute, method, inheritens, encapsul, polymorphism, uraikan masing-masing dan buat programnya.
4. Ilustrasikan dalam kasus sehari-hari yang dimaksud dengan Object, Class, atribute, method, Inheritens, polymorphism
5. buat program dengan tampilan inputan npm. GU

Jawaban

1. Paradigma secara bahasa memiliki makna yaitu kerangka berpikir atau model dalam teori ilmu pengetahuan. Kata paradigma banyak digunakan dalam berbagai disiplin ilmu termasuk pemrograman. Paradigma pemrograman merupakan gaya, klasifikasi, dan pendekatan dalam penulisan program untuk memecahkan masalah dengan menggunakan bahasa pemrograman yang digunakan.
   https://dosenit.com/ilmu-komputer/paradigma-pemrograman

   Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (general purpose), dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi Java mampu berjalan di beberapa platform sistem operasi yang berbeda. Java merupakan salah satu bahasa pemrograman dengan konsep OOP Diarsipkan 2017-01-18 di Wayback Machine.. Di mana program yang dibangun berorientasikan kepada Object. Aplikasi yang dibangun dengan konsep OOP terdiri atas object-object yang saling berhubungan
   https://id.wikipedia.org/wiki/Java

2. Berikut adalah konsep dasar pada java :

   1. Sintaks Dasar
   Seorang programmer pemula perlu mempelajari syntax dasar pemrograman Java jika ingin menguasai bahasa pemrograman itu dengan baik. Sebagai informasi, syntax merupakan aturan menulis kode dalam sebuah bahasa pemrograman supaya lebih terstruktur untuk mencapai suatu tujuan.

Syntax bukan hanya sekadar gabungan dari angka, simbol, dan kata, tetapi juga tentang struktur kelas, variabel deklarasi, tipe data, operator, dan penggunaan titik koma (;) sebagai penanda akhir server di Java. Seorang programmer pemula wajib menguasai hal itu.

2. Variabel dan Tipe Data

Pemahaman tentang variabel dan tipe data juga sangat penting dalam pemrograman Java. Seorang programmer pemula harus mengetahui cara mendeklarasikan variabel dan menginisialisasi nilai. Selain itu, programmer pemula juga harus mengetahui cara menggunakan tipe data, seperti integer, float, boolean, dan string.

3. Struktur Kontrol

Struktur kontrol adalah blok pemprograman yang berfungsi untuk menganalisis variabel. Dan struktur kontrol juga dapat digunakan untuk mengendalikan alur program. Dalam hal ini, programmer pemula harus memahami konsep, seperti pernyataan if-else, perulangan (loop), dan switch-case. Ini mungkin yang bersangkutan untuk mengambil keputusan berdasarkan kondisi tertentu dan mengulang bagian kode yang sama berulang kali.

4. OOP

Java merupakan bahasa pemrograman yang berbasis objek. Sementara itu, OOP merupakan suatu metode pemrograman yang berorientasi pada objek. Oleh karena itu, agar bisa menguasai pemprograman Java, programmer pemula perlu memahami OOP dengan baik. Pemula harus mempelajari konsep kelas, objek, pewarisan, enkapsulasi, dan polimorfisme.

https://biztechacademy.id/10-konsep-dasar-dalam-pemrograman-java-yang-harus-diketahui-untuk-pemula/

3. OOP (*Object Oriented Programming*) atau dalam bahasa indonesia dikenal dengan pemrograman berorientasikan objek (PBO) merupakan sebuah paradigma atau teknik pemrograman yang **berorientesikan Objek**.

    a. Class adalah cetak biru atau blueprint dari object. Class digunakan hanya untuk membuat kerangka dasar. Yang akan kita pakai nanti adalah hasil cetakan dari class, yakni object.Dalam bahasa Java, penulisan class diawali dengan keyword class, kemudian diikuti dengan nama dari class tersebut. Aturan penulisan nama class sama seperti aturan penulisan variabel di Java (lebih tepatnya aturan identifier), yakni tidak boleh diawali angka dan tidak boleh mengandung spasi.

```
class Laptop {
  // isi dari class Laptop...
  // isi dari class Laptop...
}
```

b. Property (atau kadang juga dengan atribut atau field) adalah data yang terdapat dalam sebuah class. Melanjutkan analogi tentang laptop, property dari laptop bisa berupa merk, warna, jenis processor, ukuran layar, dan lain-lain.

```
class Laptop {
  String pemilik;
  String merk;
  double ukuranLayar;
}
```

c. Method adalah tindakan yang bisa dilakukan di dalam class. Jika menggunakan analogi class Laptop, maka contoh method adalah: menghidupkan laptop, mematikan laptop, atau mengganti cover laptop. Method pada dasarnya adalah function yang berada di dalam class. Seluruh sifat function bisa diterapkan ke dalam method seperti bisa di isi argument/parameter, mengembalikan nilai (dengan keyword return), dan lain-lain.

```
class Laptop {
  void hidupkanLaptop() {
    //... isi dari method hidupkanLaptop
  }

  String matikanLaptop() {
    //... isi dari method matikanLaptop
  }
}
```

d. Object atau objek adalah hasil cetak dari class, atau bisa juga disebut hasil konkrit dari class. Masih menggunakan analogi class Laptop, maka object dari class Laptop bisa berupa: laptopRudi, laptopLisa, atau laptopDuniailkom. Sebuah object dari class Laptop akan memiliki seluruh ciri-ciri laptop, termasuk property dan method-nya. Proses mencetak object dari class ini disebut dengan instansiasi (atau instantiation).

```java
class Laptop {
  String pemilik;
  String merk;
  double ukuranLayar;

  void hidupkanLaptop() {
    //... isi dari method hidupkanLaptop
  }

  String matikanLaptop() {
    //... isi dari method matikanLaptop
    return "";
  }
}

class BelajarJava {
  public static void main(String args[]){
    Laptop laptopRudi = new Laptop();
    Laptop laptopLisa = new Laptop();
  }
}
```

e. Maksud dari encapsulation adalah membungkus class dan menjaga apa apa saja yang ada didalam class tersebut, baik method ataupun atribut, agar tidak dapat di akses oleh class lainnya.

f. Inheritance atau Pewarisan/Penurunan adalah konsep pemrograman dimana sebuah class dapat ‘menurunkan’ property dan method yang dimilikinya kepada class lain. Konsep inheritance digunakan untuk memanfaatkan fitur ‘code reuse’ untuk menghindari duplikasi kode program.

```java
public class Bapak {
  private String nama = "Bapak";
  private String alamat = "Sleman";

  public String getNama() {
    return nama;
  }

  private String getNamaPrivate(){
    return nama;
  }

  protected String getAlamat(){
    return alamat;
  }
}
```

- Anak.java

```java
package a;
```

```
public class Anak extends Bapak{

    public String getNamaPublic(){
       return super.getNama();
    }
// public String getNamaPrivate(){
//        return super.getNamaPrivate();
//      INI AKAN ERROR karena hak akses di parent class untuk method ini private
//    }


    @Override
    public String getAlamat(){
       return super.getAlamat();
    }


   public static void main(String args[]) {
      Anak g = new Anak();
      System.out.println("dari method public : " + g.getNamaPublic());
      System.out.println("dari method protected : " + g.getAlamat());

   }
}
```

g. Polymorphism adalah suatu aksi yang memungkinkan pemrogram menyampaikan pesan tertentu keluar dari hirarki obyeknya, dimana obyek yang berbeda memberikan tanggapan/respon terhadap pesan yang sama sesuai dengan sifat masing-masing obyek.

```
class Animal{

protected void hello(){
            System.out.println("Hello");
}

 protected void move(){
    System.out.println("Animals can move");
  }

}

class Dog extends Animal{
 @Override
 public void move(){
    System.out.println("Dogs can walk and run");  }


 // method tambahan untuk analisa
            public void bark(){
    System.out.println("Dogs can bark");
  }
```

```
}

public class Override_Test{

  public static void main(String args[]){
    Animal a = new Animal(); // Animal reference and object
    Animal b = new Dog(); // Animal reference but Dog object

 a.move();
 b.move();


 System.out.println();
 a.hello();

  }
}
```

https://www.duniailkom.com/tutorial-oop-java-pengertian-class-object-property-dan-method/

4. A. class dan object
   Contoh class disini adalah class laptop dengan objeknya laptop a dan laptop b

   B. Atribut
   Atribut masih berkaitan dengan class yang disini contohnya adalah laptop maka atribut dari laptop bisa berupa : **warna, merk, gpu, cpu, dsb.**

   C. Method
   Method ini adalah sebuah pernyataan instruksi untuk program yang disini contohnya adalah laptop maka methodnya adalah : **mengetik, mendesain, dsb.**

   D. Inheritance
   Atau pewarisan adalah hubungan antara dua objek atau lebih. Di mana terdapat sebuah objek utama yang mewariskan attribute maupun *method* yang dimilikinya kepada objek lainnya, baik itu sebagian maupun keseluruhan. Contoh disini : **Laptop Acer Nitro V16 itu mewarisi komponen komponen dari Acer Nitro V15**

   E. Encapsulation
   membungkus class dan menjaga apa apa saja yang ada didalam class tersebut, baik method ataupun atribut, agar tidak dapat di akses oleh class lainnya. Contohnya : **kita tidak perlu tahu bagaimana caranya komputer menerjemahkan bahasa pemrograman menjadi bahasa mesin.**

   F. polymorphism
   adalah suatu aksi yang memungkinkan pemrogram menyampaikan pesan tertentu keluar dari hirarki obyeknya, dimana obyek yang berbeda memberikan tanggapan/respon terhadap pesan yang sama sesuai dengan sifat masing-masing obyek. Contoh : **setiap kelas memiliki method yang sama yaitu info tapi implementasinya beda beda ada info penyimpanan, info baterai, info layar, dsb.**

https://www.duniailkom.com/tutorial-oop-java-pengertian-class-object-property-dan-method/
https://www.dicoding.com/blog/apa-itu-oop-pada-java-beserta-contohnya/

5.

```java
import javax.swing.*;

public class test {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        String npm = JOptionPane.showInputDialog(parentComponent:null, message:"Masukkan NPM Anda:",
        title:"Input NPM", JOptionPane.QUESTION_MESSAGE);

        if (npm != null && !npm.isEmpty()) {
            JOptionPane.showMessageDialog(parentComponent:null, "NPM Anda adalah: " + npm,
            title:"Hasil Input", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(parentComponent:null, message:"Anda tidak memasukkan NPM.
            ", title:"Peringatan", JOptionPane.WARNING_MESSAGE);
        }

        System.exit(status:0);
    }
}
```