MODUL PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

ABSTRACT CLASS, INTERFACE

Deskripsi Singkat

Praktikum pemrograman berorientasi objek adalah praktikum yang menggunakan bahasa Java sebagai bantuan dalam memahami konsep pemrograman berorientasi objek. Materi praktikum berisi teori, latihan dan soal pemrograman.

Tujuan

1. Memahami konsep abstract class dan interface.
2. Menggunakan konsep abstract class dan interface pada pemrograman berorientasi objek.
3. Menggunakan class Random dan Math

Prasyarat

Siswa telah melakukan praktikum 1-7.

Materi 1 : Abstract Class

Pada pewarisan, jika super-kelas mempunyai method yang harus di override oleh sub-kelasnya, maka class dan method tersebut kita buat sebagai abstract. Hal ini berguna jika ada isi method di dalam super-kelas yang tidak dapat digeneralisasi pada sub-kelas sub-kelasnya.

Abstract class tidak dapat di instantiate (dicipta dengan new) namun konsep sub-tipe tetap berlaku sehingga polimorfisme tetap berlaku. Abstract class dapat memiliki method abstract dan method bukan abstract. Abstract class akan memaksa sub-kelas untuk mengoverride method abstractnya, jika tidak maka sub-kelas harus juga bertipe abstract.

Abstract method diawali dengan kata abstract dan tidak memiliki implementasi serta langsung diakhiri oleh semicolon (;). Abstract method digunakan agar di sub-kelasnya di override.

Materi 2 : Interface

Interface merupakan cara standar dalam menetapkan sifat-sifat umum dari class. Interface juga bersifat polimorfisme. Interface digunakan untuk mengatasi kemiripan-kemiripan pada kelas yang tidak berhubungan.

Interface diawali dengan kata kunci **interface**, bukan class. Method pada interface adalah abstract karena tidak memerlukan implementasi (namun tidak perlu kata kunci abstract).

Interface tidak mengandung method constructor. Semua method berakses public secara otomatis tanpa harus dideklarasikan. Field pada interface secara otomatis menjadi public static final.

Pada interface digunakan kata kunci **implements** untuk mengimplementasikan interface yang telah dibuat. Interface dapat mengimplement lebih dari 1 interface, sehingga seolah-olah suatu kelas menerapkan **multiple polimorfisme**.

Materi 3 : Class Random

Banyak aplikasi yang memerlukan nilai acak. Contohnya aplikasi permainan kartu, yang memerlukan proses mengacak kartu pemain. Nilai acak tersebut bisa didapatkan dengan menggunakan class Random. Class Random terletak di dalam package java.util. Class Random memiliki method untuk mengembalikan nilai acak yang bertipe integer, double, Boolean, float dan long.

Detil class Random dapat dilihat pada Java API.

Contoh:

```
Random rndNum = new Random();

int randomNum = rndNum.nextInt();
```

Materi 4 : Class Math

Class Math mengumpulkan banyak method-method yang terkait dengan operasi matematika seperti untuk mencari akar, pangkat, pembulatan, logaritma, trigonometri. Class Math adalah static dan method-method yang ada di dalamnya juga static. Sehingga method pada class Math dapat digunakan tanpa harus menciptakan objek dari class Math terlebih dahulu.

Detil class Math dapat dilihat pada Java API.

LATIHAN 1

Apakah ada kaitan antara abstract class dengan inheritance? Apakah ada kaitan antara interface dan inheritance?

LATIHAN 2

Praktikum 8
Abstract Class, Interface

Buka software BlueJ, dan download projek makhluk-hidup pada alamat di bawah.

http://informatika.unsyiah.ac.id/viska/pbo/makhluk-hidup.zip

Coba buat method berjalan pada kelas Manusia menjadi comment, seperti di bawah:

```
    /*public void berjalan()
    {
        System.out.println("Manusia berjalan dengan 2 kaki");
    }*/
```

Sekarang compile projek makhluk-hidup. Error apa yang muncul? Kenapa error tersebut muncul?

Coba buat kelas CobaMakhlukHidup (tetap dalam projek makhluk-hidup). Codingnya seperti di bawah:

```
public class CobaMakhlukHidup
{

    /**
     * Method main untuk mencoba kelas makhluk hidup, Manusia dan
Hewan
     */
    public static void main (String args[])
    {
        Manusia mnsia = new Manusia();
        Hewan hwan = new Hewan();

        mnsia.bernapas();
        mnsia.makan();
        mnsia.berjalan();

        hwan.bernapas();
        hwan.makan();
        hwan.berjalan();
    }
}
```

Apakah hasil outputnya ketika coding di atas dijalankan? Bagaimana jalannya pemanggilan method pada coding di atas.

Coba tambahkan coding di bawah pada method main kelas CobaMakhlukHidup.

```
     MakhlukHidup mh = new MakhlukHidup();
```

Sekarang compile kembali kelas CobaMakhlukHidup. Error apa yang muncul? Kenapa error tersebut muncul? Bisakah kita mencipta objek dari kelas abstract?

LATIHAN 3

Buka software BlueJ, dan download projek garis-bulat pada alamat di bawah.

http://informatika.unsyiah.ac.id/viska/pbo/garis-bulat.zip

Coba buat method lebihBesar pada kelas BilanganBulat menjadi comment, seperti di bawah:

```
/*public boolean lebihBesar(Object a, Object b)
{
    BilanganBulat a1 = (BilanganBulat) a;
    BilanganBulat b1 = (BilanganBulat) b;

    if(a1.getNilai() > b1.getNilai())
        return true;
    else
        return false;
}*/
```

Sekarang compile kembali projek garis-bulat. Error apa yang muncul? Kenapa error tersebut muncul?

Coba buat 2 objek BilanganBulat (misalnya bilangan1 dan bilangan2), jangan lupa berikan nilai awal pada objek tersebut. Kemudian coba method lebihBesar, lebihKecil dan samaDengan dengan menghantar parameter objek BilanganBulat yang telah anda buat (misalnya bilangan1 pada Object a dan bilangan2 pada Object b). Apakah hasilnya?

Coba juga membuat 2 objek Garis dan method-methodnya.

LATIHAN 4

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
import java.util.Random;

public class CoinFlip {

    public static void main(String[] args) {

        // 50% chance heads, 50% chance tails
        Random rand = new Random();
        double chance = rand.nextDouble();
        if (chance < 0.5) {
            System.out.println("heads!");
        }
```

```
        else {
            System.out.println("tails!");
        }
    }
}
```

LATIHAN 5

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
import java.util.Random;

public class RandomRange {

    public static void main(String[] args) {

        Random num = new Random();
        int randomnum = num.nextInt(10)+4;
        System.out.println("Random Number: " + randomnum);
    }
}
```

LATIHAN 6

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
import java.util.Random;

public class RandomSeriesSeed {

    public static void main(String[] args) {

        Random rand = new Random(20);
        System.out.println("Random Number 1: " + rand.nextInt(100));
        System.out.println("Random Number 2: " + rand.nextInt(100));
        System.out.println("Random Number 3: " + rand.nextInt(100));
        System.out.println("Changing seed to change to sequence");
        rand.setSeed(5);
        System.out.println("Random Number 4: " + rand.nextInt(100));
        System.out.println("Random Number 5: " + rand.nextInt(100));
        System.out.println("Random Number 6: " + rand.nextInt(100));
        System.out.println("Setting seed 40 to produce the previous
sequence");
        rand.setSeed(20);
        System.out.println("Random Number 7: " + rand.nextInt(100));
        System.out.println("Random Number 8: " + rand.nextInt(100));
        System.out.println("Random Number 9: " + rand.nextInt(100));

    }
```

```
}
```

LATIHAN 7

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
import java.util.Random;

public class RollingDice {

    public static void main(String[] args) {
        Random rand = new Random();
        int tries = 0;

        int sum = 0;
        while (sum != 7) {
            // roll the dice once
            int roll1 = rand.nextInt(6) + 1;
            int roll2 = rand.nextInt(6) + 1;
            sum = roll1 + roll2;
            System.out.println(roll1 + " + " + roll2 + " = " + sum);
            tries++;
        }

        System.out.println("You won after " + tries + " tries!");
    }
}
```

LATIHAN 8

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
import java.util.Scanner;

public class AgeLimit {
   public static void main(String args[]) {
        int age=0;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter age ");
        age = sc.nextInt();
      //age=Math.max(age,0);
      age=Math.min(40, age);
       System.out.print("Age is "+age);
    }
}
```

LATIHAN 9

Ketik program berikut ini, lalu kompilasi dan jalankan.

```
public class PrintMathResult {

    public static void main(String args[]) {

        Math.sqrt(121.0); //no output
        System.out.println("Square root: " + Math.sqrt(121.0));
        double result = Math.min(3, 7) + Math.abs(-50);
        System.out.println("Result is " + result);
    }
}
```

SOAL-SOAL

1. Buatlah projek baru bernama coba-interface.
   Klik New Class… dan pilih Interface. Ketik coding di bawah:

```
public interface kontrolTv {
 public void hidupkan(boolean hidup);
 public void pindahChannel(int channel);
 public void keraskanVolume(int tambah);
 public void pelankanVolume(int kurangi);
}
```

   Klik New Class… dan buat kelas Televisi yang mengimplement interface kontrolTv. Contoh coding yang belum lengkap seperti di bawah:

```
public class televisi implements kontrolTv {
    private String merek;
    private boolean hidup;
    private String[] saluran={"RCTI","SCTV","INDOSIAR",
                              "TPI","ANTV,TRANSTV","TRANS7"};
    private int volume=0;

    public televisi(String merek) {
        this.merek = merek;
    }

    public void hidupkan(boolean hidup) {
        if(hidup){
          this.hidup=true;
          System.out.println("Televisi dihidupkan");
        }else{
           this.hidup=false;
          System.out.println("Televisi dimatikan");
        }
    }
```

Coding di atas belum lengkap, lengkapi coding tersebut dengan implementasi dari method pindahChannel, keraskanVolume dan pelankanVolume.

Buat kelas baru yang berisi method void main seperti di bawah:

```
public static void main(String[] args) {
kontrolTv tvku=new televisi("SHARP");
tvku.hidupkan(true);
tvku.keraskanVolume(10);
tvku.pelankanVolume(3);
tvku.pindahChannel(5);
tvku.hidupkan(false);
}
```

Coba coding hasil implementasi method-method yang telah anda buat dengan method main di atas.

2. Ubah latihan 7 sehingga menggunakan konsep PBO dengan menciptakan class Dadu.