MODUL PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

CUSTOM CLASS

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. Menggunakan enum.
2. Menggunakan static.
3. Membuat dan menggunakan nested class.

Prasyarat

Siswa telah melakukan praktikum 1-9.

Materi 1 : Enum

Enum mulai diperkenal pada Java versi 5 (J2SE 5). Enum merupakan tipe data yang mengandung kumpulan konstan yang tetap. Enum juga dapat dianggap seperti class yang memiliki beberapa nilai konstan yang tetap. Secara implicit, *modifier* enum adalah static dan final. Enum secara bawaan merupakan turunan dari class Enum sehingga Enum tidak bisa extends class yang lain.

Enum sering dipakai untuk suatu kumpulan data yang bersifat tetap contohnya hari dalam seminggu (SENIN, SELASA, RABU, KAMIS, JUMAT, SABTU, MINGGU).

Enum memiliki kelebihan, seperti:

- Enum memperbaiki type safety
- Enum dapat digunakan pada switch
- Enum dapat dijejaki (*traverse*)
- Enum dapat memiliki field, constructor dan method
- Enum dapat mengimplementasi banyak interface

Contoh:

```
public enum mataAngin {UTARA, SELATAN, BARAT, TIMUR};
```

Materi 2 : Static

Modifier static biasanya dipergunakan untuk membuat method atau variable yang menyatakan milik dari kelas, bukan milik objek (sehingga independen terhadap objek). Yang dapat diberi modifier statis adalah variabel, method, initialization block, inner class dalam class lain. Selain dari empat hal tersebut, maka tidak bisa menggunakan modifier static.

Variabel yang diberi *modifier* static maka variable tersebut menjadi milik class. Pengertian menjadi milik kelas adalah, kita **dapat** mengakses atau menggunakannya tanpa proses instansiasi objek dari kelas tersebut (yang kita butuhkan hanya kelas tersebut dapat diakses). Variabel static akan disharing oleh semua objek pada class yang sama. Variabel static hanya menggunakan memori sekali saat class tersebut diloading sehingga menjadi penggunaan memory menjadi efisien.

Method static adalah method milik class. Method static dapat dipanggil tanpa perlu menciptakan objek dari class tersebut. Method static dapat mengakses data static dan merubah nilainya. Namun method static tidak dapat menggunakan variable non-static dan memanggil method non-static secara langsung.

Materi 3 : Nested class

Nested class merupakan class yang berada di dalam class yang lain. Nested class ada beberapa jenis:

- Non-static nested class (disebut inner class):
  - Member inner class
  - Anonymous inner class
  - Method-local inner class
- Static nested class

Nested class berguna untuk mengelompokkan class yang hanya dipakai pada suatu tempat. Nested class dapat menambah sifat encapsulation sebab inner class dapat mengakses atribut dan method outer class yang private, sedangkan inner class sendiri tersebunyi dari dunia luar.

Penggunaan inner class sering ditemukan pada implementasi event pada aplikasi berbasis Graphical User Interface (GUI).

LATIHAN 1

Ketik dan jalankan coding di bawah:

```
class EnumExample1{
     public enum Season { WINTER, SPRING, SUMMER, FALL}
     public static void main(String[] args) {
        for (Season s : Season.values())
          System.out.println(s);
```

```
      }
}
```

Apakah hasil outputnya ketika coding di atas dijalankan?

LATIHAN 2

Ketik dan jalankan coding di bawah:

```
class EnumExample5{
       enum Day{ SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY,
SATURDAY}
       public static void main(String args[]){
          Day day=Day.MONDAY;
          switch(day){
              case SUNDAY: System.out.println("sunday");
                           break;
              case MONDAY: System.out.println("monday");
                           break;
              default:
                           System.out.println("other day");
          }
       }
}
```

Apakah hasil outputnya ketika coding di atas dijalankan?

LATIHAN 3

Ketik dan jalankan coding di bawah:

```
class Student{
     int NIM;
     String name;
     static String college ="Unsyiah";

     static void change(){
          college = "USK";
     }

     Student(int r,String n){
        NIM = r;
        name = n;
     }

     void display (){
          System.out.println(NIM+" "+name+" "+college);
```

```
    }

    public static void main(String args[]){
       Student s1 = new Student(111,"Karen");
       Student s2 = new Student(222,"Arya");
       s1.display();
       s2.display();

       Student.change();
       s1.display();
       s2.display();
    }
}
```

Apakah hasil outputnya ketika coding di atas dijalankan? Pahami output yang terhasil.

LATIHAN 4

Ketik dan kompilasi coding di bawah:

```
public class MathHelper {

    public static double penambahan(String s1, String s2) {
        double d1 = Double.parseDouble(s1);
        double d2 = Double.parseDouble(s2);
        return d1 + d2;
    }

    public static double pengurangan(String s1, String s2) {
        double d1 = Double.parseDouble(s1);
        double d2 = Double.parseDouble(s2);
        return d1 - d2;
    }

    public static double perkalian(String s1, String s2) {
        double d1 = Double.parseDouble(s1);
        double d2 = Double.parseDouble(s2);
        return d1 * d2;
    }

    public static double pembagian(String s1, String s2) {
        double d1 = Double.parseDouble(s1);
        double d2 = Double.parseDouble(s2);
        return d1 / d2;
    }
}
```

Ketik dan jalankan coding di bawah:

```
import java.util.Scanner;

public class Kalkulator {

    public static void main(String[] args) {
        Kalkulator kalkulator = new Kalkulator();
        kalkulator.perhitungan();
    }

    protected void perhitungan() {
        InputHelper input = new InputHelper();

        String s1 = input.getInput("Masukkan angka: ");
        String s2 = input.getInput("Masukkan angka: ");
        String op = input.getInput("Pilih operasi (+ - * /):");

        double result = 0;

        try {
            switch (op) {
                case "+":
                    result = MathHelper.penambahan(s1, s2);
                    break;
                case "-":
                    result = MathHelper.pengurangan(s1, s2);
                    break;
                case "*":
                    result = MathHelper.perkalian(s1, s2);
                    break;
                case "/":
                    result = MathHelper.pembagian(s1, s2);
                    break;
                default:
                    System.out.println("Operasi tidak dikenal!");
                    return;
            }

            System.out.println("Hasil: " + result);

        } catch (Exception e) {
            System.out.println("Number formatting exception " +
e.getMessage());
        }
    }

    class InputHelper{
        public String getInput(String prompt) {
            System.out.print(prompt);
            Scanner sc = new Scanner(System.in);
            return sc.nextLine();
        }
    }
```

```
}
```

Apakah hasil outputnya ketika coding di atas dijalankan? Pahami output yang terhasil.

SOAL-SOAL

1. Ubah class Kalkulator pada Latihan 4 sehingga proses perhitungan dapat berulang terus hingga pengguna mengetik tombol q atau Q untuk keluar dari aplikasi.