VERSION 1.0

FEBRUARI, 2021

PEMOGRAMAN BERORIENTASI OBJEK

MEMBANGUN APLIKASI OBJECT ORIENTED SEDERHANA. MODUL 2

TIM PENYUSUN:

- GALIH WASIS W S.KOM,.M.CS.
- FARLI NAHRUL JAVIER
- MUHAMMAD NUR ICHSAN

PRESENTED BY: LAB. INFORMATIKA

UNIVERSITAS MUHAMMADIYAH MALANG

PEMOGRAMAN BERORIENTASI OBJEK

CAPAIAN PEMBELAJARAN MATA KULIAH

1. Mahasiswa mampu membangun aplikasi sederhana menggunakan paradigma pemrograman terstruktur.
2. Mahasiswa mampu membangun aplikasi sederhana menggunakan paradigma object oriented.

SUB CAPAIAN PEMBELAJARAN MATA KULIAH

1. Mahasiswa dapat membandingkan kelebihan dan kekurangan masing-masing paradigma.

KEBUTUHAN HARDWARE & SOFTWARE

1. Compiler java (JDK), JRE.
2. Editor Java (NetBeans, Gel, Eclipse, Jcreator, dll).

MATERI POKOK

Pemrograman Terstruktur

Pemrograman Terstruktur merupakan suatu proses untuk mengimplementasikan urutan langkah penyelesaian suatu masalah dalam bentuk program dan merupakan suatu aktifitas pemrograman yang dilakukan dengan memperhatikan setiap urutan dari setiap langkah perintah yang dikerjakan secara sistematis, logis, dan tersusun berdasarkan algoritma yang sederhana dan dapat dengan mudah dipahami. Prinsip dari pemrograman terstruktur adalah Jika suatu proses telah sampai pada suatu titik ataupun langkah tertentu, maka proses selanjutnya tidak boleh mengeksekusi langkah sebelumnya ataupun kembali ke baris sebelumnya, kecuali pada langkah - langkah untuk proses pengulangan atau berulang (Loop).

Pemrograman Berorientasi Objek

Pemrograman Berorientasi Objek (PBO) atau dalam bahasa inggris disebut Object Oriented Programming (OOP) merupakan sebuah paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalamnya dibungkus dalam suatu kelas-kelas atau objek-objek. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

Perbedaan Pemrograman Terstruktur dan Pemrograman Berorientasi Objek

Perbedaan mendasar antara Pemrograman Berorientasi Objek dan Pemrograman Terstruktur adalah dengan menggunakan Pemrograman Berorientasi Objek maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesaikan suatu masalah tersebut (terstruktur) tetapi objek-objek apa yang dapat melakukan pemecahan masalah tersebut. Sedangkan untuk Pemrograman Terstruktur, menggunakan prosedur/tata cara yang teratur untuk mengoperasikan data struktur.

Untuk program yang simpel/sederhana biasanya menggunakan pemrograman terstruktur karena masih mudah dan tidak banyak dilakukan perubahan yang berarti, sedangkan untuk line lebih dari 100

atau bisa dikatakan rumit, maka digunakan pemrograman berorientasi objek. Pemrograman Terstruktur terdiri dari pemecahan masalah yang besar menjadi masalah yang lebih kecil dan seterusnya, sedangkan untuk pemrograman berorientasi objek terdiri dari pengkelompokan kode dengan data yang mana setiap objek berfungsi secara independen sehingga untuk setiap perubahan kode tidak tergantung pada kode yang lainnya, atau lebih dikenal dengan modular. Terdapat juga perbedaan secara spesifik antara Pemrograman Berorientasi Objek dengan Pemrograman Terstruktur, yaitu pada kelas dan objek. Pada Pemrograman Terstruktur tidak terdapat kelas dan objek.

Pada Terstruktur terdapat "function", di OOP terdapat "method". Kalau di Terstruktur terdapat "modules", di OOP terdapat "objects". Pada di Terstruktur terdapat "argument", di OOP terdapat "message". Begitu juga dengan "variabel" yang terdapat di Terstruktur, di OOP lebih dikenal dengan nama "atribut".

MATERI PRAKTIKUM

Pemrograman Terstruktur

Contoh program terstruktur dengan studi kasus konfersi suhu Celcius ke Reamur, Celcius ke Fahrenheit, Celcius ke Kelvin.

```
public class Suhu {
    public static void main(String[] args) {
        int C = 100;
        double F;
        int K;
        double R;

        F = C*9/5+32;
        K = C+273;
        R = C*4/5;

        System.out.println("Hasil Konversi Suhu");
        System.out.println("Celcius ke Reamur\t:"+R);
        System.out.println("Celcius ke Fahrenheit\t:"+F);
        System.out.println("Celcius ke Kelvin\t:"+K);
    }
}
```

Output

```
run:
Hasil Konversi Suhu
Celcius ke Reamur       :80.0
Celcius ke Fahrenheit   :212.0
Celcius ke Kelvin       :373
BUILD SUCCESSFUL (total time: 0 seconds)
```

Pemrograman Berorientasi Objek

Dengan studi kasus yang sama dirubah ke pemrograman berbasis objek, terdapat 4 class yaitu Fahrenheit, Reamur, Kelvin dan Main.

Pada setiap class terdapat metod untuk menghitung konversi suhu.

Class Fahrenheit

```
public class Fahrenheit {
    int C = 100;
    int F;
    public void Fahrenheit(){
        F = C*9/5+32;
        System.out.println("Celcius ke Fahrenheit\t:"+F);
    }
}
```

Class Reamur

```
public class Reamur {
    int C = 100;
    int R;
    public void Reamur(){
        R = C*4/5;
        System.out.println("Celcius ke Reamur\t:"+R);
    }
}
```

Class Kelvin

```
public class Kelvin {
    int C = 100;
    int K;
    public void Kelvin(){
        K = C+273;
        System.out.println("Celcius ke Kelvin\t:"+K);
    }
}
```

Class Main

```
public class Main {
    public static void main(String[] args) {
        Fahrenheit fahrenheit = new Fahrenheit();
        Kelvin kelvin = new Kelvin();
        Reamur reamur = new Reamur();

        reamur.Reamur();
        fahrenheit.Fahrenheit();
        kelvin.Kelvin();

    }
}
```

Berikut referensi untuk menambah pemahaman kalian mengenai OOP :

https://www.studytonight.com/java/object-and-classes.php

LEMBAR KERJA

KEGIATAN 1

Buat aplikasi sederhana dengan menggunakan paradigma pemrograman terstruktur dan paradigma object oriented dengan studi kasus sebagai berikut :

1. Konversi panjang (km, hm, dam, m, dm, cm, mm). – easy
2. Kalkulator menghitung keliling / luas lingkaran. – hard
   - Terdapat 2 Inputan (nilai jarijari dan inputMenu)
   - Tersedia pilihan menu untuk menentukan hasil keliling atau luas.

Pilih salah satu dari kedua studi kasus tersebut untuk dikerjakan.

Catatan : Aplikasi yang dibangun adalah aplikasi yang sama

KEGIATAN 2

Jelaskan kepada asisten terkait kelebihan dan kekurangan masing-masing paradigma.

*Jika ada source code yang identik, maka akan dilakukan pengurangan nilai.

RUBRIK PENILAIAN

1. Kegiatan 1
   a. Menyelesaikan soal kegiatan 1 sesuai dengan ketentuan dan dapat menjelaskan alur program yang dibuat.
      - Studi kasus 1 (20).; atau
      - Studi kasus 2 (40).

   b. Pemahaman dan penerapan OOP (40).
2. Kegiatan 2 (20).