LAPORAN PRAKTIKUM MINGGU 1

Topik: Setup Hello World dalam 3 Paradigma (Prosedural, OOP, Fungsional)

A. IDENTITAS

Nama : Sri Wahyuningsih

Nim : 240202844

Kelas : 3IKRA

B. TUJUAN

Praktikum ini memiliki tujuan sebagai berikut:

1. Untuk memahami perbedaan paradigma pemrograman (Prosedural, OOP, dan Fungsional).
2. Agar mahasiswaa dapat membuat program sederhana dalam ketiga paradigma.
3. Mahasiswa mampu menggunakan Git untuk version control dan upload hasil praktikum.

C. DASAR TEORI

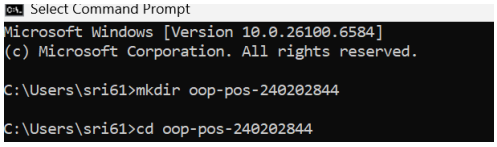
1. Paradigma Prosedural: program ditulis berurutan dalam fungsi utama (main), cocok untuk program kecil.
2. Paradigma OOP: menggunakan konsep class & object untuk modularitas dan skalabilitas.
3. Paradigma Fungsional: memanfaatkan fungsi/lambda yang bersifat deklaratif dan ringkas.
4. Perbedaan paradigma membantu mahasiswa memahami cara berpikir dan menyelesaikan masalah dengan pendekatan berbeda.

D. LANGKAH PRAKTIKUM

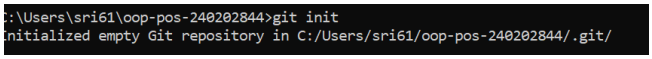
A. Setup Project

1) Pastikan sudah menginstall JDK (Java Development Kit), IDE (misal: IntelliJ IDEA, VS Code, NetBeans), Git, PostgreSQL, dan JavaFX di komputer/laptop.

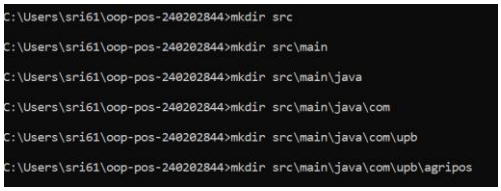
2) Buat folder project oop-pos-<nim>.



3) Inisialisasi repositori Git.



4) Buat struktur awal src/main/java/com/upb/agripos/.



5) Pastikan semua tools dapat berjalan (uji dengan membuat dan menjalankan program Java sederhana).

- Uji Instalasi Buat file HelloWorld.java:

```java
package com.upb.agripos;

public class HelloWorld {
    Run | Debug
    public static void main(String[] args) {
        System.out.println(x:"Hello, Agripos!");
    }
}
```

- Jalankan dan pastikan output Hello Agri-POS! muncul.

```
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloWorld.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloWorld
Hello, Agripos!
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java>
```

B. Buat program "Hello World, I am [nama]-[nim]" dalam 3 paradigma:
(nama = sriwaa, nim = 240202844)

1. Paradigma Prosedural
   a. Buat file baru di com/upb/agripos dengan nama HelloProcedural.java.
   b. Isi kode berikut:

```java
package com.upb.agripos;

public class HelloProcedural {
    Run | Debug
    public static void main(String[] args) {
        String nama = "sriwaa";
        String nim = "240202844";
        System.out.println("Hello World, I am " + nama + "-" + nim);
    }
}
```

   c. Compile dan jalankan, lalu akan muncul outputnya.

```
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloProcedural.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloProcedural
Hello World, I am sriwaa-240202844
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java>
```

2. Paradigma OOP
   a. Buat file baru dengan nama HelloOOP.java.
   b. Isi kode berikut:

```java
package com.upb.agripos;

class Person {
    String nama;
    String nim;

    Person(String nama, String nim) {
        this.nama = nama;
        this.nim = nim;
    }

    void sayHello() {
        System.out.println("Hello World, I am " + nama + "-" + nim);
    }
}

public class HelloOOP {
    Run | Debug
    public static void main(String[] args) {
        Person p = new Person(nama:"sriwaa", nim:"240202844");
        p.sayHello();
    }
}
```

   c. Compile dan jalankan, maka akan muncul hasil eksekusinya.

```
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloOOP.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloOOP
Hello World, I am sriwaa-240202844
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java>
```

3. Paradigma Fungsional
   a. Buat file baru dengan nama HelloFunctional.java.
   b. Isi kode berikut:

```
src > C:\Users\sri61\Documents\oop-pos-240202844\src\main\java\com\upb\agripos\HelloWorld.java > main(String[])
import java.util.function.Supplier;

public class HelloFunctional {
    Run | Debug
    public static void main(String[] args) {
        Supplier<String> hello = () -> "Hello World, I am sriwaa-240202844";

        System.out.println(hello.get());
    }
}
```

c. Compile dan jalankan, maka akan muncul hasil eksekusinya.

```
Hello World, I am sriwaa-240202844
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloFunctional.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloFunctional
Hello World, I am sriwaa-240202844
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java>
```

E. KODE PROGRAM

1. Paradigma Prosedural

```
HelloWorld.java | HelloProcedural.java | HelloProcedural.class | HelloFunctional.java
src > main > java > com > upb > agripos > HelloProcedural.java
package com.upb.agripos;

public class HelloProcedural {
    Run | Debug
    public static void main(String[] args) {
        String nama = "sriwaa";
        String nim = "240202844";
        System.out.println("Hello World, I am " + nama + "-" + nim);
    }
}
```

2. Paradigma OOP

```
HelloWorld.java | HelloProcedural.java | HelloProcedural.class | HelloFunctional.java
src > main > java > com > upb > agripos > HelloOOP.java > ...
package com.upb.agripos;

class Person {
    String nama;
    String nim;

    Person(String nama, String nim) {
        this.nama = nama;
        this.nim = nim;
    }

    void sayHello() {
        System.out.println("Hello World, I am " + nama + "-" + nim);
    }
}

public class HelloOOP {
    Run | Debug
    public static void main(String[] args) {
        Person p = new Person(nama:"sriwaa", nim:"240202844");
        p.sayHello();
    }
}
```

3. Paradigma Fungsional

```
HelloWorld.java | HelloProcedural.java | HelloProcedural.class | HelloFunctional.java
src > main > java > com > upb > agripos > HelloFunctional.java > HelloFunctional > main(String[])
package com.upb.agripos;

import java.util.function.Supplier;

public class HelloFunctional {
    Run | Debug
    public static void main(String[] args) {
        Supplier<String> hello = () -> "Hello World, I am sriwaa-240202844";

        System.out.println(hello.get());
    }
}
```

F. HASIL EKSEKUSI

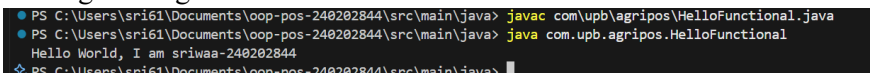
1) Paradigma prosedural

```
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloProcedural.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloProcedural
Hello World, I am sriwaa-240202844
```

2) Paradigma OOP

```
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> javac com\upb\agripos\HelloOOP.java
PS C:\Users\sri61\Documents\oop-pos-240202844\src\main\java> java com.upb.agripos.HelloOOP
Hello World, I am sriwaa-240202844
```

3) Paradigma fungsional



G. ANALISIS

1. Jelaskan bagaimana kode berjalan!

   Jawab:

   - Pada paradigma prosedural, program berjalan secara urut dari atas ke bawah. Variabel nama dan nim didefinisikan, lalu dicetak menggunakan System.out.println.
   - Pada paradigma OOP, program membuat objek dari class Person. Objek tersebut menyimpan atribut nama dan nim, kemudian method sayHello() dipanggil untuk mencetak output.
   - Pada paradigma fungsional, program menggunakan Supplier dengan *lambda expression* untuk mendefinisikan fungsi yang menghasilkan string. Fungsi tersebut dipanggil dengan .get() untuk mencetak hasil.

2. Apa perbedaan pendekatan minggu ini dibanding minggu sebelumnya

   Jawab:

   Minggu ini (minggu pertama) menekankan perbandingan *tiga paradigma* dalam satu bahasa (Java)yaitu langkah berurutan (prosedural), berbasis objek (OOP), dan berbasis fungsi (fungsional).

3. Kendala yang dihadapi dan cara mengatasinya

   Jawab:

   - Kendala: Error Could not find or load main class, karena file .java tidak sesuai dengan package dan struktur folder.
     Solusi: Menjalankan javac dan java dari folder root src/main/java serta memastikan nama package sama dengan struktur folder.
   - Kendala: Typo nama file/class (contoh: HelloWorldl.java dengan huruf l berlebih).
     Solusi: Mengecek kembali nama file agar sama dengan nama class public.
   - Kendala: Bingung membedakan sintaks antar paradigma.
     Solusi: Membuat file terpisah untuk setiap paradigma sehingga lebih jelas perbedaannya.

Jelaskan Kelebihan dan Kekurangan tiap pendekatan

a. Paradigma Prosedural

Kelebihan:

1) Sederhana dan mudah dipahami untuk pemula.
2) Cocok untuk program kecil dengan alur langsung.
3) Eksekusi lebih cepat karena tidak ada struktur tambahan.

Kekurangan:

1) Sulit dikelola jika program sudah besar.
2) Tidak ada konsep enkapsulasi, sehingga variabel bisa bercampur.
3) Susah untuk melakukan *reuse* kode.

b. Paradigma OOP

Kelebihan:

1) Struktur kode lebih terorganisir dengan class dan objek.
2) Mendukung prinsip encapsulation, inheritance, polymorphism sehingga mudah dikembangkan.
3) Lebih mudah *maintenance* untuk program besar.

4) Kode lebih mudah digunakan ulang (*reusability*).

Kekurangan:

1) Lebih kompleks daripada prosedural.
2) Membutuhkan perencanaan desain program lebih matang.
3) Kurang efisien untuk program kecil/sederhana.

c. Paradigma Fungsional

Kelebihan:

1) Kode lebih ringkas dan deklaratif (fokus pada *apa* yang dikerjakan, bukan *bagaimana*).
2) Lebih mudah diuji (*testable*) karena fungsi murni tanpa efek samping.
3) Mendukung *parallel processing* dengan lebih aman.

Kekurangan:

1) Masih asing bagi sebagian programmer Java karena paradigma ini lebih populer di bahasa seperti Haskell/Scala.
2) Bisa sulit dipahami bagi pemula yang terbiasa dengan OOP/prosedural.
3) Kurang cocok untuk aplikasi yang butuh manajemen state/objek yang kompleks.

H. KESIMPULAN

1) Paradigma prosedural sederhana dan cocok untuk program kecil, tetapi kurang terstruktur jika program semakin besar.
2) Paradigma OOP membuat program lebih terorganisir melalui class dan objek, sehingga lebih mudah dikembangkan dan dipelihara.
3) Paradigma fungsional menjadikan kode lebih ringkas, deklaratif, dan minim pengulangan kode.
4) Setiap paradigma memiliki kelebihan dan kekurangannya masing-masing, sehingga pemilihan paradigma harus disesuaikan dengan kebutuhan aplikasi.
5) Untuk aplikasi kompleks seperti sistem POS, OOP lebih sesuai karena dapat merepresentasikan entitas nyata (produk, pelanggan, transaksi) dalam bentuk objek yang saling berinteraksi.

QUIZ

1. Apakah OOP selalu lebih baik dari prosedural?

   Jawaban:

   Tidak selalu. OOP lebih baik untuk aplikasi besar yang kompleks karena mendukung struktur dan reusabilitas, tetapi untuk program kecil dan sederhana, prosedural justru lebih cepat, lebih ringan, dan lebih mudah dipahami.

2. Kapan functional programming lebih cocok digunakan dibanding OOP atau prosedural?

   Jawaban:

   Functional programming lebih cocok ketika aplikasi membutuhkan operasi data dalam jumlah besar, pemrosesan paralel, atau manipulasi koleksi dengan cara deklaratif (misalnya analisis data, pemrosesan stream, dan sistem yang membutuhkan minim efek samping).

3. Bagaimana paradigma (prosedural, OOP, fungsional) memengaruhi maintainability dan scalability aplikasi?

   Jawaban:

   a) Prosedural: kurang maintainable untuk aplikasi besar karena kode sulit dikelola dan diperluas.
   b) OOP: sangat mendukung maintainability dan scalability berkat enkapsulasi, inheritance, dan modularitas.
   c) Fungsional: meningkatkan maintainability dengan kode ringkas dan minim efek samping, serta mudah di-*scale* untuk pemrosesan paralelL.

4. Mengapa OOP lebih cocok untuk mengembangkan aplikasi POS disbanding prosedural?

   Jawaban:

   Karena aplikasi POS melibatkan banyak entitas (produk, pelanggan, transaksi, karyawan) yang saling berhubungan. OOP memungkinkan representasi entitas tersebut dalam bentuk objek, sehingga lebih mudah diorganisir, diperluas, dan dipelihara dibandingkan paradigma prosedural yang cepat menjadi rumit untuk sistem besar.

5. Bagaimana paradigma fungsional dapat membantu mengurangi kode berulang (boilerplate code)?

   Jawaban:

   Paradigma fungsional menggunakan konsep *higher-order functions*, *lambda expressions*, dan fungsi murni yang dapat digunakan ulang. Hal ini mengurangi kode berulang dengan menggantikan pola perulangan manual menjadi ekspresi deklaratif, sehingga kode lebih singkat dan bebas dari boilerplate.

CHECKLIST KEBERHASILAN

- ✓ Lingkungan kerja sudah siap (JDK, IDE, Git, PostgreSQL, JavaFX).
- ✓ Repositori Git sudah dibuat dan commit awal berhasil (week1-setup-hello-pos).
- ✓ Program "Hello POS World" berjalan di tiga paradigma dan menampilkan NIM serta nama mahasiswa.
- ✓ Versi OOP dan fungsional menggunakan minimal tiga objek/entri produk.
- ✓ Tangkapan layar hasil eksekusi ketiga program telah disertakan.
- ✓ Laporan singkat telah dilampirkan.
- ✓ Perbedaan paradigma sudah dipahami dan dijelaskan pada laporan.