LAPORAN HASIL PRAKTIKUM

MATA KULIAH :

TEKNIK PEMROGRAMAN – PERTEMUAN 3

Disusun Oleh :

NIM	NAMA	KELAS
221524041	Ferdi Ahmad Ariesta	1B

SARJANA TERAPAN PROGRAM STUDI TEKNIK INFORMATIKA JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA POLITEKNIK NEGERI BANDUNG

Daftar Isi

A. Studi Kasus 1

Barang dan Inventori

1.1 Hasil Program

Barang.java

```
public class Barang {
	String kode_barang;
	String nama_barang;
	private int stok;

	public int getStok() {
		return stok;
	}

	public void setStok(int stok) {
		this.stok += stok;
	}

	public Barang (String kode, String nama, int stk) {
		kode_barang = kode;
		nama_barang = nama;
		stok = stk;
	}
}
```

Inventori.java

```
public class Inventori {
	Barang[] barangs;
```

```
     void initBarang() {
          barangs = new Barang[2];
          barangs[0] = new Barang("001", "Baju", 10);
          barangs[1] = new Barang("002", "Celana", 20);
     }

     void showBarang() {
          System.out.println(barangs[0].nama_barang + "("
+ barangs[0].stok + ")");
          System.out.println(barangs[1].nama_barang + "("
+ barangs[1].stok + ")");
     }

     void pengadaan() {
          initBarang();

          barangs[0].setStok(10);
          barangs[1].setStok(20);
          showBarang();
     }

     public static void main(String[] args) {
          Inventori beli = new Inventori();
          beli.pengadaan();
     }
}
```

Output Program

```
Baju(20)
Celana(40)
```

1.2 Permasalahan yang dihadapi

Untuk menghindari manipulasi penambahan stok, maka menggunakan encapsulation, dan di dalam stok diatur agar hanya memiliki aritmatika penambahan saja.

1.3 Solusi permasalahan yang dihadapi

Solusi untuk permasalahan ini adalah dengan cara mengubah variable stok menjadi private dan menambahkan setter dan getter untuk variable stok. Lalu mengubah algoritma di setter menjadi "**`this.stok += stok;`**". Metode ini digunakan agar pengguna tidak bisa memanipulasi variable dari main sehingga stok tidak dapat dimanipulasi dan otomatis akan bertambah.

1.4 Nama teman yang membantu

Muhammad Agim 221524046.

B. Studi Kasus 2

Upin Ipin dan Item

2.1 Hasil Program

UpinIpin.java

```
public class UpinIpin {
	public static void main(String[] args) {
		Item name = new Item("upin");
	}
}
```

Item.java

```
public class Item {
	private String name;
	private Item() {
		name = "lpin";
	}

	public Item(String name) {
		this();
		System.out.println(this.name);
	}
}
```

Output Program

```
lpin
```

2.2 Permasalahan yang dihadapi

Di dalam kode yang tersedia kita diharuskan melakukan 1 pengubahan agar output yang dihasilkan adalah Ipin.

2.3 Solusi permasalahan yang dihadapi

Menambahkan instruksi this() di dalam constructor Item(String) karena di dalam itu kita akan mengeksekusi instruksi this tersebut sehingga constructor() akan di eksekusi dan mengubah variabel name yang kosong dan diisi “ipin”.

2.4 Nama teman yang membantu

Muhammad Agim 221524046.

C. Studi Kasus 3

KelasSatu dan KelasDua

3.1 Hasil Program

KelasSatu.java

```
class KelasSatu{

	//initializer block 2
	{
		System.out.println(11);
	}

	//static block 1
	static{
		System.out.println(2);
	}

	//constructor with argument
	public KelasSatu(int i){
		System.out.println(3);
	}

	//constructor
	public KelasSatu(){
		System.out.println(4);
	}
}
```

KelasDua.java

```
public class KelasDua {
	{
		System.out.println(5);
```

```
	}

	public static void main(String[] args) {
		System.out.println(6);
		KelasSatu satu = new KelasSatu();
		KelasSatu dua = new KelasSatu(3);
	}
}
```

Output Program

```
6
2
11
4
11
3
```

3.2 Permasalahan yang dihadapi

Bagaimana urutan konstruksi objek tersebut, dan mengapa urutannya seperti itu?

3.3 Solusi Permasalahan yang dihadapi

Urutan eksekusi yang di dahulukan antara static block, non-static block, dan construct, pahami juga cara kejra static block. Disini yang diutamakan adalah static.

3.4 Nama teman yang membantu

Muhammad Agim 221524046.