LAPORAN HASIL PRAKTIKUM 03

PEMROMGRAMAN BERBASIS OBJEK

ATHAULLA HAFIZH

244107020030

TI 2A

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

3. Percobaan

Percobaan 1 Enkapsulasi

Kode program class Motor

```
package MotorEncapsulation;


public class Motor {
    public int kecepatan = 0;
    public boolean kontakOn = false;


    void printStatus() {
        if (kontakOn == true) {
            System.out.println("Kontak On");
        } else {
            System.out.println("Kontak Off");
        }
        System.out.println("Kecepatan " + kecepatan + "\n");
    }
}
```

Kode program class MotorDemo

```
package MotorEncapsulation;


public class MotorDemo {
    public static void main(String[] args) {
        Motor motor = new Motor();
        motor.printStatus();
        motor.kecepatan = 50;
        motor.printStatus();
    }
}
```

Percobaan 2 Access Modifier

Kode program class Motor

```
package MotorEncapsulation;


public class Motor {
    private int kecepatan = 0;
    private boolean kontakOn = false;


    public void nyalakanMesin() {
        kontakOn = true;
    }


    public void matikanMesin() {
        kontakOn = false;
        kecepatan = 0;
    }


    public void tambahKecepatan() {
        if (kontakOn == true) {
            kecepatan += 5;
        } else {
            System.out.println("Kecepatan tidak bisa bertambah
karena Mesin Off! \n");
        }
    }


    public void kurangiKecepatan() {
        if (kontakOn == true && kecepatan > 0) {
            kecepatan -= 5;
        } else {
            System.out.println("Kecepatan tidak bisa berkurang
karena Mesin Off! \n");
        }
    }


    void printStatus() {
```

```
        if (kontakOn == true) {
            System.out.println("Kontak On");
        } else {
            System.out.println("Kontak Off");
        }
        System.out.println("Kecepatan " + kecepatan + "\n");
    }
}
```

Kode program class MotorDemo

```
package MotorEncapsulation;

public class MotorDemo {
    public static void main(String[] args) {
        Motor motor = new Motor();
        motor.printStatus();
        motor.tambahKecepatan();

        motor.nyalakanMesin();
        motor.printStatus();

        motor.tambahKecepatan();
        motor.printStatus();

        motor.tambahKecepatan();
        motor.printStatus();

        motor.matikanMesin();
        motor.printStatus();
    }
}
```

Pertanyaan

1. Pada class TestMobil, saat kita menambah kecepatan untuk pertama kalinya, mengapa muncul peringatan “Kecepatan tidak bisa bertambah karena Mesin Off!”?
   Hal tersebut terjadi karena nilai dari variable kontakOn masih bernilai false, itu artinya mesin motor belum diaktifkan (masih off).

2. Mengapa atribut kecepatan dan kontakOn diset private?
   Kode tersebut menerapkan pilar Enkapsulasi, artinya menyembunyikan detail implementasi dari suatu proses atau information hiding, serta membatasi hak akses terhadap variable tersebut. Hal tersebut mencegah resource dari luar mengganti atau mengakses value dari variable tersebut secara bebas.

3. Kode program

```
package MotorEncapsulation;


public class Motor {
    private int kecepatan = 0;
    private boolean kontakOn = false;
    private final int maxKecepatan = 100;


    public void nyalakanMesin() {
        kontakOn = true;
    }


    public void matikanMesin() {
        kontakOn = false;
        kecepatan = 0;
    }


    public void tambahKecepatan() {
        if (kontakOn == true) {
            if (kecepatan < maxKecepatan) {
                kecepatan += 5;
                if (kecepatan > maxKecepatan) {
                    kecepatan = maxKecepatan;
                }
            } else {
```

```
                System.out.println("Kecepatan sudah mencapai batas
maksimal " + maxKecepatan + "! \n");
            }
        } else {
            System.out.println("Kecepatan tidak bisa bertambah
karena Mesin Off! \n");
        }
    }

    public void kurangiKecepatan() {
        if (kontakOn == true && kecepatan > 0) {
            kecepatan -= 5;
        } else {
            System.out.println("Kecepatan tidak bisa berkurang
karena Mesin Off! \n");
        }
    }

    void printStatus() {
        if (kontakOn == true) {
            System.out.println("Kontak On");
        } else {
            System.out.println("Kontak Off");
        }
        System.out.println("Kecepatan " + kecepatan + "\n");
    }
}
```

Percobaan 3 Getter dan Setter

Kode program class Anggota

```
package KoperasiGetterSetter;


public class Anggota {
    private String nama;
    private String alamat;
    private float simpanan;

    public void setNama(String nama){
        this.nama = nama;
    }
    public void setAlamat(String alamat){
        this.alamat = alamat;
    }
    public String getNama(){
        return nama;
    }
    public String getAlamat(){
        return alamat;
    }
    public float getSimpanan(){
        return simpanan;
    }
    public void setor(float uang){
        simpanan += uang;
    }
    public void pinjam(float uang){
        simpanan -= uang;
    }
}
```

Kode program class KoperasiDemo

```
package KoperasiGetterSetter;


public class KoperasiDemo {

    public static void main(String[] args) {

        Anggota anggota1 = new Anggota();

        anggota1.setNama("Iwan Setiawan");

        anggota1.setAlamat("Jalan Soekarno Hatta No. 10");

        anggota1.setor(100000);

        System.out.println("Simpanan " + anggota1.getNama() + " : Rp
" + anggota1.getSimpanan());

        anggota1.pinjam(50000);

        System.out.println("Simpanan " + anggota1.getNama() + " : Rp
" + anggota1.getSimpanan());

    }

}
```

Percobaan 4 Konstruktor, Instansiasi

Kode program class Anggota

```
package KoperasiGetterSetter;


public class Anggota {

    private String nama;

    private String alamat;

    private float simpanan;


    Anggota (String nama, String alamat){

        this.nama = nama;

        this.alamat = alamat;

        this.simpanan = 0;

    }


    public void setNama(String nama){

        this.nama = nama;

    }
```

```
    public String getNama(){
        return nama;
    }
    public String getAlamat(){
        return alamat;
    }
    public float getSimpanan(){
        return simpanan;
    }
    public void setor(float uang){
        simpanan += uang;
    }
    public void pinjam(float uang){
        simpanan -= uang;
    }
}
```

Kode program class KoperasiDemo

```
package KoperasiGetterSetter;


public class KoperasiDemo {
    public static void main(String[] args) {
        Anggota anggota1 = new Anggota("Iwan", "Jalan Mawar");
        System.out.println("Simpanan " + anggota1.getNama() + " : Rp
" + anggota1.getSimpanan());


        anggota1.setNama("Iwan Setiawan");
        anggota1.setAlamat("Jalan Soekarno Hatta No. 10");
        anggota1.setor(100000);
        System.out.println("Simpanan " + anggota1.getNama() + " : Rp
" + anggota1.getSimpanan());
        anggota1.pinjam(50000);
        System.out.println("Simpanan " + anggota1.getNama() + " : Rp
" + anggota1.getSimpanan());
    }
}
```

Pertanyaan Percobaan 3 dan 4

1. Getter adalah public method yang memiliki return dengan tipe data tertentu yang berfungsi untuk mendapatkan nilai dari attribute yang memiliki modifier private. Sedangkan setter adalah public method yang tidak memiliki return value di mana fungsinya adalah untuk memanipulasi variable yang memiliki modifier private.

2. Method getSimpanan() adalah method getter, fungsinya untuk mengambil atau mendapatkan nilai dari variable simpanan dengan tipe float. Attribute ini memiliki modifier private, sehingga tidak bisa untuk mengakses attribute tersebut melalui instance class.

3. Method untuk menambah saldo melalui method setor, di mana operasi aritmatika terjadi di dalam method tersebut untuk menambah value dari attribute simpanan yang ada dengan uang yang masuk melalui parameter uang.

4. Konstruktor adalah method yang pertama kali dipanggil ketika sebuah class diinstansiasi, sehingga bisa menetapkan value atau default value dari sebuah instance /objek melalui method konstruktor.

5. Nama method harus sama dengan nama class, Konstruktor tidak memilik tipe data return, Konstruktor harus memilik modifier public

6. Di dalam Java, konstruktor dapat diberi modifier private. Namun, jika sebuah konstruktor didefinisikan sebagai private, maka kelas tersebut tidak dapat diinstansiasi secara langsung dari luar kelas menggunakan operator new. Dengan kata lain, akses terhadap konstruktor dibatasi hanya di dalam kelas itu sendiri. Meskipun demikian, penggunaan konstruktor private memiliki tujuan khusus, yaitu untuk mengendalikan proses pembuatan objek. Tetapi dalam praktik di dalam jobsheet dan materi enkapsulasi ini, sangat tidak disarankan untuk membuat Konstruktor bersifat private.

7. Passing Parameter digunakan ketika sebuah method membutuhkan input dari luar agar bisa bekerja dengan nilai yang dinamis, misalnya untuk memanipulasi attribute objek atau perhitungan berdasarkan nilai yang diberikan.

8. Attribute class bersifat milik kelas bukan milik objek tertentu, nilainya sama dan dapat diakses bersama oleh semua objek yang dibuat dari kelas tersebut, dan dapat diakses langsung tanpa melalui instance class. Sedangkan instansiasi atribut bersifat milik tiap objek hasil instansiasi dari class, dan setiap objek memiliki copy attributenya sendiri sehingga nilainya dapat berbeda antara objek satu dengan yang lain.

9. Class method disebut juga static method adalah method yang didefinisikan dengan keyword static. Method ini bersifat miliki kelas, sehingga dapat dipanggil langsung melalui nama kelas tanpa perlu membuat instance objek. Sebaliknya, instansiasi method adalah method yang tidak menggunakan keyword static. Method ini hanya bisa dipanggil setelah sebuah instance objek dibuat dari kelas tersebut.

Tugas

1. Kode program class EncapDemo

```
package TugasPraktikum;

public class EncapDemo {
    private String name;
    private int age;

    public String getName() {
        return name;
    }

    public void setName(String newName) {
        name = newName;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int newAge) {
        if (newAge > 30) {
            age = 30;
        } else {
            age = newAge;
        }
    }
}
```

Kode program class EncapTest

```
package TugasPraktikum;

public class EncapTest {
```

```
    public static void main(String[] args) {
        EncapDemo encap = new EncapDemo();
        encap.setName("James");
        encap.setAge(35);

        System.out.println("Name : " + encap.getName());
        System.out.println("Age : " + encap.getAge());
    }
}
```

2. Pada program diatas, pada class EncapTest kita mengeset age dengan nilai 35, namun pada saat ditampilkan ke layar nilainya 30, jelaskan mengapa.
   Hal ini terjadi karena di dalam method setAge() terdapat kondisi validasi. Ketika nilai newAge > 30, maka nilai age akan diset menjadi 30. Jadi ketika kita memanggil setAge(35), karena 35 > 30, maka age akan diset menjadi 30 bukan 35. Ini adalah implementasi enkapsulasi untuk membatasi nilai age maksimal 30.

3. Kode program class EncapDemo

```
package TugasPraktikum;

public class EncapDemo {
    private String name;
    private int age;

    public String getName() {
        return name;
    }

    public void setName(String newName) {
        name = newName;
    }

    public int getAge() {
        return age;
    }
```

```
    public void setAge(int newAge) {
        if (newAge > 30) {
            age = 30;
        } else if (newAge < 18) {
            age = 18;
        } else {
            age = newAge;
        }
    }
}
```

Kode program class EncapTest

```
package TugasPraktikum;

public class EncapTest {
    public static void main(String[] args) {
        EncapDemo encap = new EncapDemo();

        encap.setName("James");

        encap.setAge(35);
        System.out.println("Name : " + encap.getName());
        System.out.println("Age (set 35): " + encap.getAge());

        encap.setAge(15);
        System.out.println("Age (set 15): " + encap.getAge());

        encap.setAge(25);
        System.out.println("Age (set 25): " + encap.getAge());

        encap.setAge(18);
        System.out.println("Age (set 18): " + encap.getAge());

        encap.setAge(30);
        System.out.println("Age (set 30): " + encap.getAge());
```

```
    }
}
```

4. Kode program class Anggota

```
package TugasPraktikum;

public class Anggota {
    private String nomorKTP;
    private String nama;
    private int limitPinjaman;
    private int jumlahPinjaman;

    public Anggota(String nomorKTP, String nama, int limitPinjaman)
{
        this.nomorKTP = nomorKTP;
        this.nama = nama;
        this.limitPinjaman = limitPinjaman;
        this.jumlahPinjaman = 0;
    }

    public String getNomorKTP() {
        return nomorKTP;
    }

    public String getNama() {
        return nama;
    }

    public int getLimitPinjaman() {
        return limitPinjaman;
    }

    public int getJumlahPinjaman() {
        return jumlahPinjaman;
    }
```

```
    public void pinjam(int jumlah) {
        if (jumlahPinjaman + jumlah > limitPinjaman) {
            System.out.println("Maaf, jumlah pinjaman melebihi
limit.");
        } else {
            jumlahPinjaman += jumlah;
        }
    }

    public void angsur(int jumlah) {
        if (jumlah > jumlahPinjaman) {
            System.out.println("Maaf, angsuran melebihi jumlah
pinjaman.");
        } else {
            jumlahPinjaman -= jumlah;
        }
    }
}
```

5. Kode program class Anggota

```
package TugasPraktikum;

public class Anggota {
    private String nomorKTP;
    private String nama;
    private int limitPinjaman;
    private int jumlahPinjaman;

    public Anggota(String nomorKTP, String nama, int limitPinjaman)
{
        this.nomorKTP = nomorKTP;
        this.nama = nama;
        this.limitPinjaman = limitPinjaman;
        this.jumlahPinjaman = 0;
    }
```

```
    public String getNomorKTP() {
        return nomorKTP;
    }

    public String getNama() {
        return nama;
    }

    public int getLimitPinjaman() {
        return limitPinjaman;
    }

    public int getJumlahPinjaman() {
        return jumlahPinjaman;
    }

    public void pinjam(int jumlah) {
        if (jumlahPinjaman + jumlah > limitPinjaman) {
            System.out.println("Maaf, jumlah pinjaman melebihi
limit.");
        } else {
            jumlahPinjaman += jumlah;
        }
    }

    public void angsur(int jumlah) {
        if (jumlahPinjaman == 0) {
            System.out.println("Tidak ada pinjaman untuk
diangsur.");
            return;
        }

        double minimalAngsuran = 0.1 * jumlahPinjaman;
```

```
        if (jumlah < minimalAngsuran) {
            System.out.println("Maaf, angsuran harus 10% dari jumlah
pinjaman");
        } else if (jumlah > jumlahPinjaman) {
            System.out.println("Maaf, angsuran melebihi jumlah
pinjaman.");
        } else {
            jumlahPinjaman -= jumlah;
        }
    }
}
```

6. Kode program class TestKoperasi

```
package TugasPraktikum;
import java.util.Scanner;


public class TestKoperasi {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);


        Anggota donny = new Anggota("111333444", "Donny", 5000000);


        System.out.println("Nama Anggota: " + donny.getNama());
        System.out.println("Limit Pinjaman: " +
donny.getLimitPinjaman());


        System.out.print("\nMasukkan jumlah pinjaman pertama: ");
        int pinjaman1 = input.nextInt();
        System.out.println("Meminjam uang " + pinjaman1 + "...");
        donny.pinjam(pinjaman1);
        System.out.println("Jumlah pinjaman saat ini: " +
donny.getJumlahPinjaman());


        System.out.print("\nMasukkan jumlah pinjaman kedua: ");
        int pinjaman2 = input.nextInt();
        System.out.println("Meminjam uang " + pinjaman2 + "...");
        donny.pinjam(pinjaman2);
```

```
        System.out.println("Jumlah pinjaman saat ini: " +
donny.getJumlahPinjaman());


        System.out.print("\nMasukkan jumlah angsuran pertama: ");
        int angsuran1 = input.nextInt();
        System.out.println("Membayar angsuran " + angsuran1);
        donny.angsur(angsuran1);
        System.out.println("Jumlah pinjaman saat ini: " +
donny.getJumlahPinjaman());


        System.out.print("\nMasukkan jumlah angsuran kedua: ");
        int angsuran2 = input.nextInt();
        System.out.println("Membayar angsuran " + angsuran2);
        donny.angsur(angsuran2);
        System.out.println("Jumlah pinjaman saat ini: " +
donny.getJumlahPinjaman());


        input.close();
    }
}
```