

# **LAPORAN PRAKTIKUM PBO**

## **JOBSHEET 3 – ENKAPSULASI**

Untuk melengkapi tugas mata kuliah Pemrograman Berorientasi Object yang  
dibimbing oleh Ibu Vit Zuraída, S. Kom., M. Kom.



Disusun oleh:

Nama : Hernada Rizka Utami  
Kelas : SIB – 2A  
NIM : 244107060075

**POLITEKNIK NEGERI MALANG**  
**JURUSAN TEKNOLOGI INFORMASI**  
**PROGRAM STUDI D4 SISTEM INFORMASI BISNIS**

## PERCOBAAN 1 – CONSTRUCTOR

Langkah	Percobaan dan Output
2.	<pre>public class user{     public String username;     public String email;     public String password;     public String name;      public void displayInfo(){         System.out.println("Username: " + username);         System.out.println("Email: " + email);         System.out.println("Password: " + password);         System.out.println("Name: " + name);     } }</pre>
3.	<pre>public class UserDemo{     public static void main(String[] args){         user user1 = new user();         user1.displayInfo();     } }</pre>
4.	<pre>UserDemo Username: null Email: null Password: null Name: null</pre>
5.	<pre>public user(String username, String email){     this.username = username;     this.email = email;     this.password = "polinema123"; }</pre>
6.	

	<pre>UserDemo.java:3: error: constructor user in class user cannot es;     user user1 = new user();                   ^     required: String,String     found: no arguments     reason: actual and formal argument lists differ in length 1 error</pre>
7.	<pre>public class UserDemo{     public static void main(String[] args){         user user1 = new user("Annisa.nadya", "annisa.nadya@gmail.com");         user1.displayInfo();      } }</pre>
8.	<pre>UserDemo Username: Annisa.nadya Email: annisa.nadya@gmail.com Password: polinema123 Name: null</pre>

## PERTANYAAN

No.	Pertanyaan
1.	<p>Apa fungsi dari constructor?</p> <p>Constructor berfungsi untuk meng-inisialisasi objek saat pertama kali dibuat. Dipakai untuk memberi nilai awal pada atribut/variabel dalam suatu kelas, constructor dapat dijalankan otomatis ketika objek dibuat dengan new.</p>
2.	<p>Sebutkan keistimewaan constructor dibanding method lain pada suatu class.</p> <ul style="list-style-type: none"> <li>⇒ Nama constructor sama dengan nama kelas.</li> <li>⇒ Tidak memiliki return type</li> <li>⇒ Dipanggil otomatis saat objek dibuat, tanpa harus dipanggil secara eksplisit seperti method biasa.</li> </ul>
3.	<p>Lakukan analisa dan buatlah kesimpulan, apakah constructor bisa memiliki access level modifier private?</p> <p>Iya, bisa. Karena jika constructor diberi private, maka objek tidak bisa dibuat langsung dengan new dari luar kelas. Biasanya dipakai untuk pola desain tertentu.</p>

	Kesimpulannya, constructor bisa dibuat private tapi penggunaanya lebih spesifik untuk membatasi untuk membatasi pembuatan objek agar lebih terkontrol.
--	--

## PERCOBAAN 1 – TANPA ENKAPSULASI

Langkah	Percobaan dan Output
1.	<pre>public class Motor{     public String platNomor;     public boolean statusMesin;     public int kecepatan;      public void displayInfo(){         System.out.println("Plat Motor: " + this.platNomor);         System.out.println("Status Mesin: " + (this.statusMesin ? "On" : "Off"));         System.out.println("Kecepatan: " + this.kecepatan);         System.out.println("=====");     } }</pre>
2.	<pre>public class MotorDemo{     public static void main(String[] args){         Motor motor1 = new Motor();         motor1.platMotor = "B 0338 XZ";         motor1.kecepatan = 50;         motor1.displayInfo();     } }</pre>
3.	 <pre>MotorDemo Plat Motor: B 0338 XZ Status Mesin: Off Kecepatan: 50 =====</pre>
4.	<pre>Motor motor2 = new Motor();     motor2.platNomor = "N 9840 AB";     motor2.statusMesin = true;     motor2.kecepatan = 40;     motor2.displayInfo();</pre>

	<pre> Motor motor3 = new Motor(); motor1.platNomor = "D 8343 CV"; motor1.kecepatan = 60; motor1.displayInfo(); </pre>
5.	<pre> Plat Motor: B 0338 XZ Status Mesin: Off Kecepatan: 50 ===== Plat Motor: N 9840 AB Status Mesin: On Kecepatan: 40 ===== Plat Motor: D 8343 CV Status Mesin: Off Kecepatan: 60 ===== </pre>
6.	<pre> int kecepatanBaru = 50;  if(!motor1.statusMesin &amp;&amp; kecepatanBaru &gt; 0){     System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off"); } else{     motor1.kecepatan = kecepatanBaru; } motor1.displayInfo(); </pre>
7.	<pre> Motor motor2 = new Motor(); motor2.platNomor = "N 9840 AB"; motor2.statusMesin = true; kecepatanBaru = 40;  if(!motor2.statusMesin &amp;&amp; kecepatanBaru &gt; 0){     System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off"); } else{     motor2.kecepatan = kecepatanBaru; } motor2.displayInfo();  Motor motor3 = new Motor(); motor3.platNomor = "D 8343 CV"; </pre>

	<pre> kecepatanBaru = 60;  if(!motor3.statusMesin &amp;&amp; kecepatanBaru &gt; 0){     System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off"); } else{     motor3.kecepatan = kecepatanBaru; } motor3.displayInfo(); </pre>
8.	<pre> Kecepatan tidak boleh lebih dari 0 jika mesin off Plat Motor: B 0338 XZ Status Mesin: Off Kecepatan: 0 ===== Plat Motor: N 9840 AB Status Mesin: On Kecepatan: 40 ===== Kecepatan tidak boleh lebih dari 0 jika mesin off Plat Motor: D 8343 CV Status Mesin: Off Kecepatan: 0 ===== </pre>

## PERCOBAAN 2 – ENKAPSULASI

Langkah	Percobaan dan Output
3.	<pre> private String platMotor; private boolean statusMesin; private int kecepatan; </pre>
4.	<pre> MotorDemo Exception in thread "main" java.lang.IllegalAccessException: tried to access field Motor. platMotor from class MotorDemo     at MotorDemo.main(MotorDemo.java:4) </pre>
5.	<pre> public String getPlatNomor(){     return platNomor; }  public void setPlatNomor(String platNomor){     this.platNomor = platNomor; } </pre>

	<pre>     }     public boolean getStatusMesin(){         return statusMesin;     }     public void setStatusMesin(boolean statusMesin){         this.statusMesin = statusMesin;     }     public int getKecepatan(){         return kecepatan;     }     public void setKecepatan(int kecepatan){         this.kecepatan = kecepatan;     } </pre>
6.	<pre> Motor motor1 = new Motor(); motor1.setPlatNomor = ("B 0338 XZ"); motor1.setKecepatan = (50); motor1.displayInfo();  Motor motor2 = new Motor(); motor2.setPlatNomor = ("N 9849 AB"); motor2.setStatusMesin = (true); motor2.setKecepatan = (40); motor2.displayInfo();  Motor motor3 = new Motor(); motor3.setPlatNomor = ("D 8343 CV"); motor3.setKecepatan = (60); motor3.displayInfo(); </pre>
7.	<pre> public void setKecepatan(int kecepatan){     if(!this.statusMesin &amp;&amp; kecepatan &gt; 0){         System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");     } else{         this.kecepatan = kecepatan;     } } </pre>
8.	

	<pre> Kecepatan tidak boleh lebih dari 0 jika mesin off Plat Motor: B 0338 XZ Status Mesin: Off Kecepatan: 0 ===== Plat Motor: N 9849 AB Status Mesin: On Kecepatan: 40 ===== Kecepatan tidak boleh lebih dari 0 jika mesin off Plat Motor: D 8343 CV Status Mesin: Off Kecepatan: 0 ===== </pre>	
--	---	--

## PERTANYAAN

No.	Pertanyaan
1.	Pada class MotorDemo, saat kita menambah kecepatan untuk pertama kalinya, mengapa muncul peringatan “Kecepatan tidak boleh lebih dari 0 jika mesin off”? Karena statusMesin awal adalah false (off), dan dalam setKEcepatan ada aturan if, jadi selama mesinnya off, motor tidak boleh jalan meskipun kecepatannya diset.
2.	Mengapa atribut merek, kecepatan, dan statusMesin sebaiknya diset private? Agar tidak bisa diakses langsung dari luar kelas (prinsip enkapsulasi). Mencegah perubahan nilai atribut secara sembarangan, memaksa penggunaan setter / getter sehingga setiap perubahan bisa dijaga aturan tertentu.
3.	<p>Apa fungsi dari setter dan getter?</p> <ul style="list-style-type: none"> <li>⇒ Setter : Memeberi atau mengubah nilai atribut dengan cara yang terkontrol, bisa ditambahkan validasi sebelum nilai disimpan.</li> <li>⇒ Getter : Mengambil atau mengembalikan nilai atribut supaya bisa dibaca oleh class lain tanpa membuka akses langsung.</li> <li>⇒ Keduanya adalah implementasi dari enkapsulasi, yaitu menyembunyikan detail data tetapi tetap menyediakan akses aman.</li> </ul>
4.	<p>Modifikasi class Motor sehingga kecepatan maksimalnya adalah 100.</p> <pre> } else if(this.statusMesin &amp;&amp; kecepatan &gt; 100){     System.out.println("Kecepatan tidak boleh melebihi maksimal jika mesin on"); } </pre> <pre> Motor motor2 = new Motor(); motor2.setPlatNomor("N 9849 AB"); motor2.setStatusMesin(true); </pre>

	<pre>motor2.setKecepatan(120); motor2.displayInfo();</pre> <pre>Kecepatan tidak boleh melebihi maksimal jika mesin on Plat Motor: N 9849 AB Status Mesin: On Kecepatan: 0 =====</pre>
5.	<p>Modifikasi class Motor sehingga kecepatan nya tidak boleh bernilai negatif</p> <pre>else if(kecepatan &lt; 0){     System.out.println("Kecepatan tidak boleh negatif"); }</pre> <pre>Motor motor2 = new Motor(); motor2.setPlatNomor("N 9849 AB"); motor2.setStatusMesin(true); motor2.setKecepatan(120); motor2.displayInfo();  Motor motor3 = new Motor(); motor3.setPlatNomor("D 8343 CV"); motor3.setStatusMesin(true); motor3.setKecepatan(-60); motor3.displayInfo();</pre> <pre>Kecepatan tidak boleh lebih dari 0 jika mesin off Plat Motor: B 0338 XZ Status Mesin: Off Kecepatan: 0 ===== Kecepatan tidak boleh melebihi maksimal jika mesin on Plat Motor: N 9849 AB Status Mesin: On Kecepatan: 0 ===== Kecepatan tidak boleh negatif Plat Motor: D 8343 CV Status Mesin: On Kecepatan: 0 =====</pre>

## TUGAS

1. Pada sebuah sistem informasi koperasi simpan pinjam, terdapat class Anggota yang memiliki atribut nomor KTP, nama, limit peminjaman, dan jumlah pinjaman. Anggota dapat meminjam uang dengan limit peminjaman yang ditentukan. Anggota juga dapat mengangsur pinjaman. Ketika anggota tersebut mengangsur pinjaman, maka jumlah pinjaman akan berkurang sesuai dengan nominal yang diangsur. Buatlah class Anggota tersebut dengan konsep enkapsulasi. Berikan atribut, method, dan constructor sesuai dengan kebutuhan. Uji dengan TestKoperasi.java berikut ini untuk memeriksa apakah class Anggota yang anda buat telah sesuai dengan yang diharapkan. Perhatikan bahwa nilai atribut pinjaman tidak dapat diubah secara random dari luar class, tetapi hanya dapat diubah melalui method pinjam() dan angsur()

### Anggota

```
public class Anggota {  
    private String noKtp;  
    private String nama;  
    private int limitPinjaman;  
    private int jumlahPinjaman;  
  
    public Anggota(String noKtp, String nama, int limitPinjaman) {  
        this.noKtp = noKtp;  
        this.nama = nama;  
        this.limitPinjaman = limitPinjaman;  
        this.jumlahPinjaman = 0;  
    }  
  
    public String getNoKtp() {  
        return noKtp;  
    }  
  
    public String getNama() {  
        return nama;  
    }  
  
    public int getLimitPinjaman() {  
        return limitPinjaman;  
    }  
  
    public int getJumlahPinjaman() {  
        return jumlahPinjaman;  
    }  
  
    public void pinjam(int uang) {  
        if (jumlahPinjaman + uang > limitPinjaman) {  
            System.out.println("Maaf, jumlah pinjaman melebihi limit.");  
        }  
    }  
}
```

```

    } else {
        jumlahPinjaman += uang;
    }
}

public void angsur(int uang) {
    jumlahPinjaman -= uang;
    if (jumlahPinjaman < 0) {
        jumlahPinjaman = 0;
    }
}
}

```

## TestKoperasi

```

public class TestKoperasi {
    public static void main(String[] args) {
        Anggota anggota1 = new Anggota("111333444", "Donny", 5000000);

        System.out.println("Nama Anggota: " + anggota1.getNama());
        System.out.println("Limit Pinjaman: " + anggota1.getLimitPinjaman());

        System.out.println("\nMeminjam uang 10.000.000...");
        anggota1.pinjam(10000000);
        System.out.println("Jumlah Pinjaman saat ini: " + anggota1.getJumlahPinjaman());

        System.out.println("\nMeminjam uang 4.000.000...");
        anggota1.pinjam(4000000);
        System.out.println("Jumlah Pinjaman saat ini: " + anggota1.getJumlahPinjaman());

        System.out.println("\nMembayar angsuran 1.000.000...");
        anggota1.angsur(1000000);
        System.out.println("Jumlah Pinjaman saat ini: " + anggota1.getJumlahPinjaman());

        System.out.println("\nMembayar angsuran 3.000.000...");
        anggota1.angsur(3000000);
        System.out.println("Jumlah Pinjaman saat ini: " + anggota1.getJumlahPinjaman());
    }
}

```

## Output

```
Nama Anggota: Donny
Limit Pinjaman: 5000000

Meminjam uang 10.000.000...
Maaf, jumlah pinjaman melebihi limit.
Jumlah Pinjaman saat ini: 0

Meminjam uang 4.000.000...
Jumlah Pinjaman saat ini: 4000000

Membayar angsuran 1.000.000...
Jumlah Pinjaman saat ini: 3000000

Membayar angsuran 3.000.000...
Jumlah Pinjaman saat ini: 0
```

- ⇒ Pada class testkoperasi, dibuat objek anggota dengan nama Donny. Nilai atribut limipinjaman diset sebesar 5000000, ketika Donny mencoba meminjam uang 10000000, program menolak karena jumlah pinjaman yang diajukan melebihi limit, sehingga jumlah pinjaman tetap 0. Selanjutnya, saat Donny meminjam 4000000 permintaan diterima sehingga pinjaman bertam menjadi 4000000. Kemudian Donny membayar angsuran sebesar 1000000, program menjalankan method angsur(int uang) yang mengurangi jumlah pinjaman dan ketika angsuran berikutnya 3000000, sistem kembali mengurangi pinjaman hingga habis dan pinjaman menjadi 0.

2. Modifikasi class Anggota agar nominal yang dapat diangsur minimal adalah 10% dari jumlah pinjaman saat ini. Jika mengangsur kurang dari 10%, maka muncul peringatan “Maaf, angsuran harus 10% dari jumlah pinjaman”.

## Modifikasi

```
public void angsur(int uang) {
    if (uang >= (0.1 * jumlahPinjaman)) {
        jumlahPinjaman -= uang;
        if (jumlahPinjaman < 0) {
            jumlahPinjaman = 0;
        }
    } else {
        System.out.println("Maaf, angsuran harus 10% dari jumlah pinjaman");
    }
}
```

## Output

```
Nama Anggota: Rere
Limit Pinjaman: 7000000

Meminjam uang 3.000.000...
Jumlah Pinjaman saat ini: 3000000

Meminjam uang 6.000.000...
Maaf, jumlah pinjaman melebihi limit.
Jumlah Pinjaman saat ini: 3000000

Membayar angsuran 2.00.000...
Maaf, angsuran harus 10% dari jumlah pinjaman
Jumlah Pinjaman saat ini: 3000000
```

- ⇒ pada class testkoperasi dibuat objek bernama rere yang hamper sama seperti program sebelumnya, hanya saja pada method angsur ditambahkan logika if. Logika ini berfungsi untuk mengecek apakah jumlah angsuran lebih kecil dari 10% dari total pinjaman. Jika iya, maka perintah ditolak dan program akan menampilkan pesan “Maaf, angsuran harus 10% dari jumlah pinjaman”.