LAPORAN TUGAS

Enkapsulasi pada Pemrograman Berorientasi Objek

Disusun sebagai

Mata Kuliah :

Praktikum Pemrograman Berbasis Object

Oleh :

Siska Nuri Aprilia

Sib 2F

2341760038

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

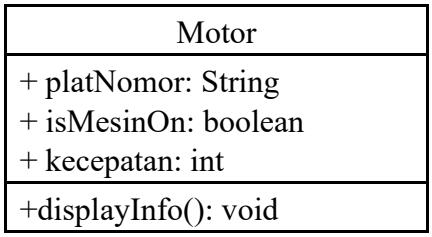
POLITEKNIK NEGERI MALANG

TAHUN 2024/2025

3. Percobaan

3.1 Percobaan 1 – Tanpa Enkapsulasi

Didalam percobaan enkapsulasi, buatlah class Motor yang memiliki atribut platNomor, isMesinOn (bernilai true jika mesin sedang menyala dan false jika tidak menyala), dan kecepatan serta method displayInfo() untuk menampilkan status motor. UML class diagram class Motor adalah sebagai berikut:



1. Buat folder baru dengan nama **Praktikum03**
2. Buat class **Motor**

```
public class Motor {
    public String platNomor;
    public boolean isMesinOn;
    public int kecepatan;

    public void displayInfo(){
        System.out.println("Plat Nomor: " + this.platNomor);
        System.out.println("Status Mesin: " + (this.isMesinOn ? "On" : "Off"));
        System.out.println("Kecepatan: " + this.kecepatan);
        System.out.println("============================");
    }
}
```

3. Kemudian buat class MotorDemo, ketikkan kode berikut ini.

```
public class MotorDemo {
    Run | Debug
    public static void main(String[] args) {
        Motor motor1 = new Motor();
        motor1.displayInfo();

        motor1.platNomor = "B 0838 XZ";
        motor1.kecepatan = 50;
        motor1.displayInfo();
    }
}
```

4. Hasilnya adalah sebagai berikut:

```
Plat Nomor: null
Status Mesin: Off
Kecepatan: 0
============================
Plat Nomor: B 0838 XZ
Status Mesin: Off
Kecepatan: 50
============================
```

5. Selanjutnya tambahkan 2 objek motor lagi di class MotorDemo.java

```
Motor motor2 = new Motor();
motor2.platNomor = "N 9840 AB";
motor2.isMesinOn = true;
motor2.kecepatan = 40;
motor2.displayInfo();

Motor motor3 = new Motor();
motor3.platNomor = "D 8343 CV";
motor3.kecepatan = 60;
motor3.displayInfo();
```

6. Hasilnya sebagai berikut

```
Plat Nomor: B 0838 XZ
Status Mesin: Off
Kecepatan: 50
============================
Plat Nomor: N 9840 AB
Status Mesin: On
Kecepatan: 40
============================
Plat Nomor: D 8343 CV
Status Mesin: Off
Kecepatan: 60
============================
```

- **Jawaban**

1. Langkah Praktikum 1

- **Kode Pemrograman Class Motor**

Motor.java > Motor > displayInfo()

```
public class Motor {
    public String platNomor;
    public boolean isMesinOn;
    public int kecepatan;

    public void displayInfo()
    {
        System.out.println("Plat Nomor   : " + this.platNomor);
        System.out.println("Status Mesin : " + (this.isMesinOn ? "On" : "off"));
        System.out.println("Kecepatan    : " + this.kecepatan);
        System.out.println(x:"==================================");
    }
}
```

- **Kode Pemrograman Class MotorDemo**

```
public class MotorDemo {

    Run | Debug
    public static void main(String[] args) {
        Motor motor1 = new Motor();
        motor1.displayInfo();

        motor1.platNomor = "B 0838 XZ";
        motor1.kecepatan = 50;
        motor1.displayInfo();
    }

}
```

- **Output yang dihasilkan**

```
Plat Nomor    : null
Status Mesin : off
Kecepatan     : 0
=====================================
Plat Nomor    : B 0838 XZ
Status Mesin : off
Kecepatan     : 50
=====================================
```

2. Kode Pemrograman ClassDemo (Modifikasi)

```java
public class MotorDemo {

    Run | Debug
    public static void main(String[] args) {
        Motor motor1 = new Motor();
        motor1.displayInfo();

        motor1.platNomor = "B 0838 XZ";
        motor1.kecepatan = 50;
        motor1.displayInfo();

        Motor motor2 = new Motor();
        motor2.platNomor = "N 9840 AB";
        motor2.isMesinOn = true;
        motor2.kecepatan = 40;
        motor2.displayInfo();

        Motor motor3 = new Motor();
        motor3.platNomor = "D 8343 CV";
        motor3.kecepatan = 60;
        motor3.displayInfo();
    }

}
```

- **Output yang dihasilkan**

```
Plat Nomor   : null
Status Mesin : off
Kecepatan    : 0
====================================
Plat Nomor   : B 0838 XZ
Status Mesin : off
Kecepatan    : 50
====================================
Plat Nomor   : N 9840 AB
Status Mesin : On
Kecepatan    : 40
====================================
Plat Nomor   : D 8343 CV
Status Mesin : off
Kecepatan    : 60
====================================
```

7. Dari hasil di atas, adakah yang janggal?

   Pada motor1 dengan plat “B 0838 XZ”, kecepatannya dapat berubah dari 0 ke 50 padahal mesin motor masih dalam kondisi Off. Bagaimana mungkin atribut kecepatan bernilai 50 padahal mesin masih Off? Hal ini karena belum tersedia kontrol/batasan terhadap atribut kecepatan. Padahal, objek di dunia nyata selalu memiliki batasan/aturan dalam memberi nilai atribut serta mekanisme bagaimana objek tersebut dapat berfungsi/melakukan sesuatu. Misalnya motor yang harus dalam keadaan menyala ketika kecepatan lebih dari 0.

   Kejanggalan ini juga terjadi pada motor ketiga dengan plat nomor "D 8343 CV".

8. Untuk mengatasi hal tersebut, nilai kecepatan baru perlu dicek terlebih dahulu sebelum diassign ke nilai atribut kecepatan

```
motor1.platNomor = "B 0838 XZ";

int kecepatanBaru = 50;

if (!motor1.isMesinOn && kecepatanBaru > 0) {
    System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");
}
else {
    motor1.kecepatan = kecepatanBaru;
}

motor1.displayInfo();
```

9. Lakukan pengecekan yang sama untuk motor2 dan motor3

```
Motor motor2 = new Motor();
motor2.platNomor = "N 9840 AB";
motor2.isMesinOn = true;

kecepatanBaru = 40;

if (!motor2.isMesinOn && kecepatanBaru > 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";
kecepatanBaru = 60;

if (!motor1.isMesinOn && kecepatanBaru > 0) {
    System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");
}
else {
    motor1.kecepatan = kecepatanBaru;
}
motor3.displayInfo();
```

10. Run MotorDemo.java dan perhatikan bahwa sudah terdapat validasi nilai kecepatan terhadap status mesin untuk setiap objek motor

```
Plat Nomor: null
Status Mesin: Off
Kecepatan: 0
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: B 0838 XZ
Status Mesin: Off
Kecepatan: 0
============================
Plat Nomor: N 9840 AB
Status Mesin: On
Kecepatan: 40
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: D 8343 CV
Status Mesin: Off
Kecepatan: 0
============================
```

➢ Kode Pemrograman Class MotorDemo (Modifikasi)

```
public class MotorDemo {

    public static void main(String[] args) {
        // Motor 1
        Motor motor1 = new Motor();
        motor1.displayInfo();

        motor1.platNomor = "B 0838 XZ";
        int kecepatanBaru = 50;

        if (!motor1.isMesinOn && kecepatanBaru > 0) {
            System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin
off");  } else {
            motor1.kecepatan = kecepatanBaru;
        }
        motor1.displayInfo();

        // Motor 2
        Motor motor2 = new Motor();
        motor2.platNomor = "N 9840 AB";
        motor2.isMesinOn = true;

        kecepatanBaru = 40;

        if (!motor2.isMesinOn && kecepatanBaru > 0) {
            System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin
off");  } else {
            motor2.kecepatan = kecepatanBaru;
        }
        motor2.displayInfo();

        // Motor 3
        Motor motor3 = new Motor();
        motor3.platNomor = "D 8343 CV";
        kecepatanBaru = 60;

        if (!motor3.isMesinOn && kecepatanBaru > 0) {
            System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin
off");  } else {
            motor3.kecepatan = kecepatanBaru;
        }
        motor3.displayInfo();
    }
}
```

- Output yang dihasilkan

```
Plat Nomor   : null
Status Mesin : off
Kecepatan    : 0
================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : B 0838 XZ
Status Mesin : off
Kecepatan    : 0
================================================
Plat Nomor   : N 9840 AB
Status Mesin : On
Kecepatan    : 40
================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : D 8343 CV
Status Mesin : off
Kecepatan    : 0
================================================
```

3.2 Percobaan 2 – Enkapsulasi

1. Bayangkan bahwa developer baru saja ingat bahwa seharusnya kecepatan tidak boleh lebih dari 0 jika status mesin tidak menyala setelah membuat 20 objek motor di MotorDemo.java, 10 objek motor di MotorDemo2.java, 25 objek MotorDemo3.java? Pengecekan harus dilakukan 55 kali.
2. Lalu, bagaimana kita bisa memperbaiki class Motor diatas agar dapat digunakan dengan baik? Di sinilah pentingnya melakukan enkapsulasi dalam pemrograman berorientasi objek sehingga perubahan requirement hanya perlu dihandle pada "gerbang" yang dikelola di dalam class tersebut.

   Pada OOP, konsep enkapsulasi diimplementasikan dengan cara:

   a. Menyembunyikan atribut internal (platNomor, isMesinOn, dan kecepatan) dari luar/class lain dengan mengubah access level modifier menjadi private
   b. Menyediakan setter dan getter untuk memanipulasi dan mengakses nilai atribut tersebut

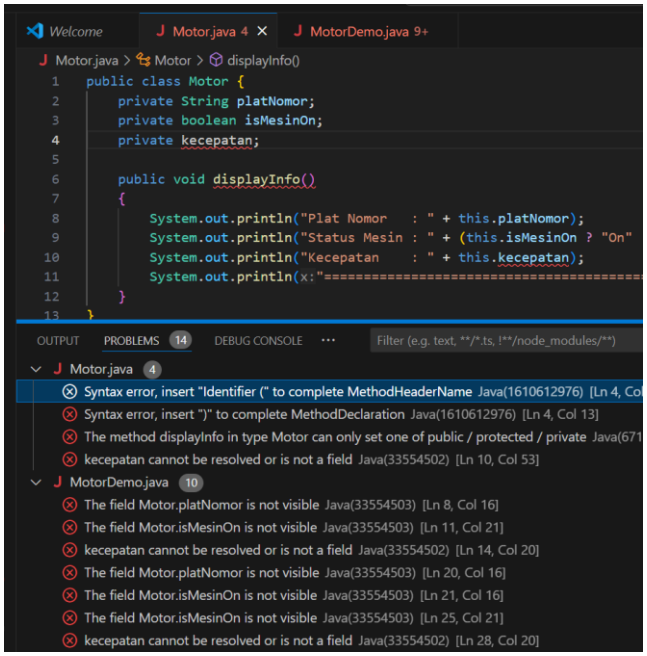
Motor
- platNomor: String - isMesinOn: Boolean - kecepatan: int
+displayStatus(): void +setPlatNomor(platNomor:String): void +getPlatNomor(): String +setIsMesinOn(isMesinOn:boolean): void +getIsMesinOn(): boolean +setKecepatan(kecepatan:int): void

+getKecepatan(): int

3. Ubah access level modifier menjadi private

```
private String platNomor;
private boolean isMesinOn;
private int kecepatan;
```

➢ Eror



4. Setelah berubah menjadi private, atribut platNomor, isMesinOn, dan kecepatan tidak bisa diakses dari luar class (muncul error)

```
Motor motor1 = new Motor();
motor1.displayInfo();

motor1.platNomor = "B 0838 XZ";

int kecepatanBaru = 50;

if (!motor1.isMesinOn && kecepatanBaru > 0) {
    System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");
}
else {
    motor1.kecepatan = kecepatanBaru;
}

motor1.displayInfo();
```

5. Selanjutnya perlu di buat setter dan getter untuk setiap atribut.

```
public String getPlatNomor() {
    return platNomor;
}

public void setPlatNomor(String platNomor) {
    this.platNomor = platNomor;
}

public boolean getIsMesinOn() {
    return isMesinOn;
}

public void setIsMesinOn(boolean isMesinOn) {
    this.isMesinOn = isMesinOn;
}

public int getKecepatan() {
    return kecepatan;
}

public void setKecepatan(int kecepatan) {
    this.kecepatan = kecepatan;
}
```

6. Dengan enkapsulasi, nilai atribut diakses menggunakan getter dan dimanipulasi menggunakan setter sebagai berikut (belum ada validasi nilai kecepatan terhadap status mesin)

```
public static void main(String[] args) {
    Motor motor1 = new Motor();
    motor1.displayInfo();

    motor1.setPlatNomor("B 0838 XZ");
    motor1.setKecepatan(50);
    motor1.displayInfo();

    Motor motor2 = new Motor();
    motor2.setPlatNomor("N 9840 AB");
    motor2.setIsMesinOn(true);
    motor2.setKecepatan(40);
    motor2.displayInfo();

    Motor motor3 = new Motor();
    motor3.setPlatNomor("D 8343 CV");
    motor3.setKecepatan(60);
    motor3.displayInfo();
}
```

7. Dengan menerapkan enkapsulasi, perubahan requirement di tengah implementasi program dapat dilakukan dengan lebih mudah. Pada setter kecepatan, dilakukan validasi nilai kecepatan terhadap status mesin sebagai berikut:

```
public void setKecepatan(int kecepatan) {
    if (!this.isMesinOn && kecepatan > 0) {
        System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");
    }
    else{
        this.kecepatan = kecepatan;
    }
}
```

8. Run MotorDemo.java. Hasilnya sebagai berikut:

```
Plat Nomor: null
Status Mesin: Off
Kecepatan: 0
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: B 0838 XZ
Status Mesin: Off
Kecepatan: 0
============================
Plat Nomor: N 9840 AB
Status Mesin: On
Kecepatan: 40
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: D 8343 CV
Status Mesin: Off
Kecepatan: 0
============================
```

- **Kode Pemrograman Class Motor (Modifikasi)**

```
public class Motor {
    private String platNomor;
    private boolean isMesinOn;
    private int kecepatan;

    public void displayInfo()
    {
        System.out.println("Plat Nomor   : " + this.platNomor);
        System.out.println("Status Mesin : " + (this.isMesinOn ? "On" : "off"));
        System.out.println("Kecepatan    : " + this.kecepatan);
        System.out.println("============================================");
    }

    public String getPlatNomor()
    {
        return platNomor;
    }

    public void setPlatNomor(String platNomor)
    {
        this.platNomor = platNomor;
    }

    public boolean getIsMesinOn()
        {
            return isMesinOn;
        }

    public void setIsMesinOn(boolean isMesinOn)
    {
        this.isMesinOn = isMesinOn;
    }

    public int getKecepatan()
    {
        return kecepatan;
    }

    public void setKecepatan(int kecepatan)
    {
        if (!this.isMesinOn && kecepatan > 0)
        {
            System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin
off");  }

    else
    {
        this.kecepatan = kecepatan;
    }
}
}
```

- **Kode Pemrograman Class MotorDemo (Modifikasi)**

```
public class MotorDemo {

    Run | Debug
    public static void main(String[] args) {
        // Motor 1
        Motor motor1 = new Motor();
        motor1.displayInfo();

        motor1.setPlatNomor(platNomor:"B 0838 XZ");
        motor1.setKecepatan(kecepatan:50);
        motor1.displayInfo();

        // Motor 2
        Motor motor2 = new Motor();
        motor2.setPlatNomor(platNomor:"N 9840 AB");
        motor2.setIsMesinOn(isMesinOn:true);
        motor2.setKecepatan(kecepatan:40);
        motor2.displayInfo();

        // Motor 3
        Motor motor3 = new Motor();
        motor3.setPlatNomor(platNomor:"D 8343 CV");
        motor3.setKecepatan(kecepatan:60);
        motor3.displayInfo();

    }
}
```

- **Output yang dihasilkan**

```
Plat Nomor   : null
Status Mesin : off
Kecepatan    : 0
==================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : B 0838 XZ
Status Mesin : off
Kecepatan    : 0
==================================================
Plat Nomor   : N 9840 AB
Status Mesin : On
Kecepatan    : 40
==================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : D 8343 CV
Status Mesin : off
Kecepatan    : 0
==================================================
```

9. Setter dan getter dipakai sebagai “gerbang” untuk mengakses atau memodifikasi atribut yang bernilai private. Hal ini akan membuat kontrol atau validasi atribut lebih mudah dilakukan. Jika ada perubahan requirement di kemudian hari, misalnya atribut kecepatan tidak boleh bernilai negatif, hanya perlu dilakukan modifikasi pada setKecepatan() tanpa perlu melakukan perubahan berulang kali di seluruh program yang melakukan assignment nilai kecepatan motor. Pengujian terhadap perubahan requirement juga dapat dilakukan pada scope yang lebih kecil, tanpa harus menguji keseluruhan sistem yang dikembangkan.

3.3 Pertanyaan

1. Pada class MotorDemo, saat kita menambah kecepatan untuk pertama kalinya, mengapa muncul peringatan “Kecepatan tidak bisa bertambah karena Mesin Off!”?

- karena ada logika dalam kode yang memeriksa apakah mesin dalam keadaan **mati** (isMesinOn == false) dan kecepatan ingin diatur ke nilai yang lebih dari 0. Contohnya, saat kita mencoba menambah kecepatan (kecepatanBaru = 50) untuk **motor1**, mesin motor tersebut masih dalam kondisi **mati** (isMesinOn == false karena belum dihidupkan). Maka, if tersebut akan bernilai **true**, dan pesan peringatan akan ditampilkan pada output ketika dirunning

- Kode pemrograman tersebut juga sudah terdapat kontrol atau batasan di dalam class Motor ada perintah untuk memeriksa apakah mesin on/off jika mesin off dan kecepatan yang diberikan lebih dari 0 maka output yang akan muncul adalah “Kecepatan tidak bisa bertambah karena Mesin Off!”, jika mesin on nilai yg dimasukkan akan disimpan dan ditampilkan di kecepatan

2. Mengapat atribut merek, kecepatan, dan statusMesin sebaiknya diset private?

- Enkapsulasi adalah prinsip utama dalam **Object-Oriented Programming (OOP)** yang membantu menjaga integritas data. Dengan menetapkan atribut sebagai **private**, sehingga data hanya dapat diakses dan dimodifikasi melalui **getter** dan **setter**.

- Atribut dibuat **private** agar tidak dapat diakses langsung dari kelas lain, sehingga kita dapat mencegah perubahan/manipulasi data yang mungkin terjadi yang nantinya bisa tidak terkontrol. Dengan menjadikannya private, kita memaksa/membatasi semua interaksi dengan atribut tersebut dilakukan melalui **setter** dan **getter**, yang memberikan batasan/kontrol atas bagaimana atribut tersebut diubah atau diambil. Ini memastikan bahwa setiap perubahan mengikuti aturan atau logika tertentu yang kita tentukan

3. Apa fungsi dari setter dan getter?

1. Getter

- **Fungsi**: Untuk mendapatkan atau mengakses nilai dari atribut private.
- **Tujuan**: Mengizinkan kelas lain membaca nilai atribut secara aman tanpa memberikan akses langsung ke atribut itu sendiri.

2. Setter

- **Fungsi**: Untuk mengubah atau menetapkan nilai ke atribut private.
- **Tujuan**: Mengontrol cara nilai atribut diubah, memastikan perubahan mengikuti aturan atau logika tertentu.

4. Ubah class Motor sehingga kecepatan maksimalnya adalah 100

- **Kode Pemrograman Class Motor**

```
    // Setter untuk kecepatan dengan batasan maksimal 100
    public void setKecepatan(int kecepatan) {
        if (!this.isMesinOn && kecepatan > 0) {
            System.out.println(x:"Kecepatan tidak boleh lebih dari 0 jika mesin off");
        } else if (kecepatan > 100) {
            System.out.println(x:"Kecepatan tidak boleh lebih dari 100!");
        } else {
            this.kecepatan = kecepatan;
        }
    }
}
```

- **Kode Pemrograman Class MotorDemo**

```
// Motor 2
Motor motor2 = new Motor();
motor2.setPlatNomor(platNomor:"N 9840 AB");
motor2.setIsMesinOn(isMesinOn:true);
motor2.setKecepatan(kecepatan:120); // Akan menampilkan peringatan karena lebih dari 100
motor2.displayInfo();
```

- **Output yang dihasilkan**

```
Plat Nomor   : null
Status Mesin : off
Kecepatan    : 0
================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : B 0838 XZ
Status Mesin : off
Kecepatan    : 0
================================================
Kecepatan tidak boleh lebih dari 100!
Plat Nomor   : N 9840 AB
Status Mesin : On
Kecepatan    : 0
================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor   : D 8343 CV
Status Mesin : off
Kecepatan    : 0
================================================
PS C:\Users\ASUS\Documents\PEMROGRAMAN BERBASIS OBJECT\Jobsheet 3- Enkapsulasi>
```

5. Ubah class Motor sehingga kecepatan nya tidak boleh nilai negatif

- **Kode Pemrograman Class Motor**

```java
// Setter untuk kecepatan dengan batasan negatif dan maksimal 100
public void setKecepatan(int kecepatan) {
    if (!this.isMesinOn && kecepatan > 0) {
        System.out.println(x:"Kecepatan tidak boleh lebih dari 0 jika mesin off");
    } else if (kecepatan > 100) {
        System.out.println(x:"Kecepatan tidak boleh lebih dari 100!");
    } else if (kecepatan < 0) {
        System.out.println(x:"Kecepatan tidak boleh bernilai negatif!");
    } else {
        this.kecepatan = kecepatan;
    }
}
```

➢ Kode Pemrograman Class MotorDemo

```java
    // Motor 3
    Motor motor3 = new Motor();
    motor3.setPlatNomor(platNomor:"D 8343 CV");
    motor3.setKecepatan(-10); // Berhasil karena kecepatan di bawah 100
    motor3.displayInfo();
}
```

➢ Output yang dihasilkan

```
Plat Nomor    : null
Status Mesin : off
Kecepatan     : 0
==================================================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor    : B 0838 XZ
Status Mesin : off
Kecepatan     : 0
==================================================
Kecepatan tidak boleh lebih dari 100!
Plat Nomor    : N 9840 AB
Status Mesin : On
Kecepatan     : 0
==================================================
Kecepatan tidak boleh bernilai negatif!
Plat Nomor    : D 8343 CV
Status Mesin : off
Kecepatan     : 0
==================================================
```

3.4 Percobaan 3 - Constructor

Pada pelajaran sebelumnya, instansiasi objek dari suatu class dilakukan dengan menggunakan syntax **new <NamaClass>();** misalnya motor1 = new Motor();

Dengan baris kode tersebut, kita telah menggunakan constructor default yaitu Motor() tanpa parameter apapun. Oleh karena itu, saat objek motor1 diinstansiasi, setiap nilai atribut pada motor1 akan bernilai default. Atribut merek yang bertipe string bernilai default **null**, atribut isMesinOn yang bertipe boolean bernilai default **false**, dan atribut kecepatan yang bertipe integer bernilai default **0**.

```
Plat Nomor: null
Status Mesin: Off
Kecepatan: 0
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: B 0838 XZ
Status Mesin: Off
Kecepatan: 0
============================
Plat Nomor: N 9840 AB
Status Mesin: On
Kecepatan: 40
============================
Kecepatan tidak boleh lebih dari 0 jika mesin off
Plat Nomor: D 8343 CV
Status Mesin: Off
Kecepatan: 0
============================
```

Pada beberapa kasus, kita menginginkan suatu objek dari class tertentu sudah memiliki nilai untuk beberapa (atau seluruh) atribut pada saat objek tersebut dibuat.

1. Misalkan di sebuah sistem informasi, terdapat class **User** yang memiliki atribut username, nama, email, alamat, dan pekerjaan. Saat suatu objek user dibuat, user tersebut harus sudah memiliki nilai username, nama, dan email. Dengan kebutuhan tersebut, kita harus membuat sebuah constructor baru sebagai berikut:

```java
public class User {
    public String username;
    public String nama;
    public String email;
    public String alamat;
    public String pekerjaan;

    public User(String username, String nama, String email) {
        this.username = username;
        this.nama = nama;
        this.email = email;
    }

    public void cetakInfo()
    {
        System.out.println("Username: " + username);
        System.out.println("Nama: " + nama);
        System.out.println("Email: " + email);
        System.out.println("Alamat: " + alamat);
        System.out.println("Pekerjaan: " + pekerjaan);
        System.out.println("==================");
    }
}
```

2. Setelah kita menyediakan constructor baru secara eksplisit, maka constructor default yaitu User() tidak bisa digunakan lagi kecuali kita buat juga. Multiple constructor akan dibahas pada materi overloading dan overriding.

```java
public class DemoUser {
    public static void main(String[] args) {
        User user1 = new User();
    }
}
```

constructor User in class User cannot be applied to given types;
required: String,String,String
found: no arguments
reason: actual and formal argument lists differ in length

(Alt-Enter shows hints)

3. Instansiasi objek user baru dengan constructor yang telah dibuat pada no 1 bisa dilakukan dengan cara berikut:

```java
public class DemoUser {
    public static void main(String[] args) {
        User user1 = new User("annisa.nadya", "Annisa Nadya", "annisa.nadya@gmail.com");

        user1.cetakInfo();
    }
}
```

4. Hasilnya sebagai berikut:

```
run:
Username: annisa.nadya
Nama: Annisa Nadya
Email: annisa.nadya@gmail.com
Alamat: null
Pekerjaan: null
==================
BUILD SUCCESSFUL (total time: 0 seconds)
```

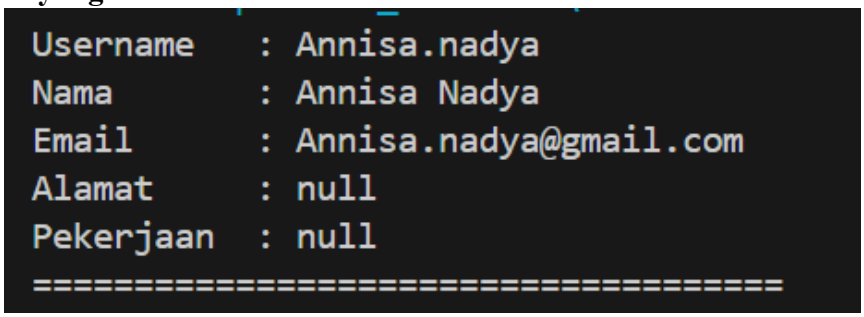
- **Jawaban**
  - **Kode Pemrograman Class User**

```
public class User {
    public String username;
    public String nama;
    public String email;
    public String alamat;
    public String pekerjaan;

    public User(String username, String nama, String email)
    {
        this.username = username;
        this.nama = nama;
        this.email = email;
    }

    public void cetakInfo()
    {
        System.out.println("Username   : " + username);
        System.out.println("Nama       : " + nama);
        System.out.println("Email      : " + email );
        System.out.println("Alamat     : " + alamat);
        System.out.println("Pekerjaan  : " + pekerjaan);
        System.out.println(x:"====================================");
    }
}
```

  - **Kode Pemrograman Class UserDemo**

```
DemoUser.java > DemoUser > main(String[])
public class DemoUser {
    Run | Debug
    public static void main(String[] args) {
        User user1 = new User(username:"Annisa.nadya", nama:"Annisa Nadya", email:"Annisa.nadya@gmail.com");

        user1.cetakInfo();
    }

}
```

- **Output yang dihasilkan**



3.5 Pertanyaan

1. Apa yand dimaksud constructor?
   - Constructor adalah method khusus yang akan dieksekusi pada saat pembuatan objek (instance).
   - Biasanya method ini digunakan untuk inisialisasi atau mempersiapkan data untuk objek.
   - Constructor digunakan untuk menginisialisasi objek, yaitu untuk menetapkan nilai awal pada atribut objek atau untuk menjalankan kode yang diperlukan sebelum objek dapat digunakan.
2. Sebutkan aturan dalam membuat constructor
   - Nama harus sama dengan class, tidak memiliki type return ataupun void, bisa dengan parameter atau tidak, bisa lebih dari 1 construstor dalam 1 class dengan syarat memiliki parameter yang berbeda.
3. Lakukan analisa dan buat kesimpulan apakah constructor bisa memiliki access level modifier private?
   - Ya, constructor bisa memiliki modifier private Ini digunakan untuk membatasi pembuatan objek dari kelas tersebut dari luar kelas, sehingga pembuatan objek hanya bisa dilakukan dengan cara yang diizinkan oleh class itu sendiri. Dengan menggunakan constructor privat, kita memastikan bahwa objek hanya dapat dibuat sesuai dengan aturan yang ditetapkan di dalam kelas tersebut.

4. Tugas

1. Pada sebuah sistem informasi koperasi simpan pinjam, terdapat class Anggota yang memiliki atribut antara lain 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, berikan atribut, method dan constructor sesuai dengan kebutuhan. Uji dengan TestKoperasi 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()

```
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());
        }
}
```

Hasil yang diharapkan:

```
D:\MyJava>javac TestKoperasi.java

D:\MyJava>java TestKoperasi
Nama Anggota: Donny
Limit Pinjaman: 5000000

Meminjam uang 10.000.000...
Maaf, jumlah pinjaman melebihi limit.

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
```

JAWABAN

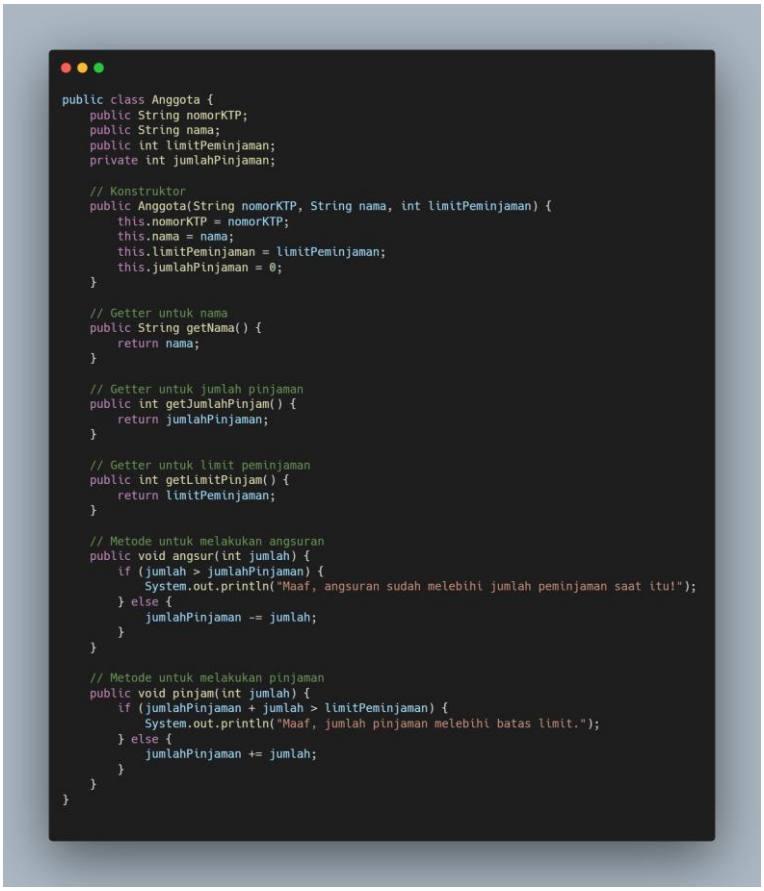
- **Output yang dihasilkan**

```
==========================================================
Nama Anggota        : Siska Nuri Aprilia
Limit Pinjaman      : 5000000
==========================================================

|----------------------------|
| Meminjam uang Rp 10.000.000 |
|----------------------------|
Maaf, jumlah pinjaman melebihi batas limit.
Jumlah Pinjaman Saat ini : 0

|----------------------------|
|  Meminjam uang Rp 4.000.000 |
|----------------------------|
Jumlah Pinjaman Saat ini : 4000000

|----------------------------|
| Bayar angsuran Rp 1.000.000 |
|----------------------------|
Jumlah Pinjaman Saat ini : 3000000

Membayar angsuran 3.000.000 melebihi pinjaman
------------------------------------------
Jumlah pinjaman saat ini : 0
```

➢ Kode Pemrograman Class TestKoperasi

```
public class TestKoperasi {
    public static void main(String[] args) {

        Anggota anggota1 = new Anggota("0304052005", "Siska Nuri Aprilia" ,  5000000);


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

        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("| Meminjam uang Rp 10.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.pinjam(10000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());
        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("|  Meminjam uang Rp 4.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.pinjam(4000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());
        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("| Bayar angsuran Rp 1.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.angsur(1000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());

        System.out.println();
        System.out.println("Membayar angsuran 3.000.000 melebihi pinjaman");
        System.out.println("-------------------------------------------");
        anggota1.angsur(3000000);

        System.out.println("Jumlah pinjaman saat ini : " +
anggota1.getJumlahPinjam());


    }
}
```

- **Kode Pemrograman Class Anggota**



```java
public class Anggota {
    public String nomorKTP;
    public String nama;
    public int limitPeminjaman;
    private int jumlahPinjaman;

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

    // Getter untuk nama
    public String getNama() {
        return nama;
    }

    // Getter untuk jumlah pinjaman
    public int getJumlahPinjam() {
        return jumlahPinjaman;
    }

    // Getter untuk limit peminjaman
    public int getLimitPinjam() {
        return limitPeminjaman;
    }

    // Metode untuk melakukan angsuran
    public void angsur(int jumlah) {
        if (jumlah > jumlahPinjaman) {
            System.out.println("Maaf, angsuran sudah melebihi jumlah peminjaman saat itu!");
        } else {
            jumlahPinjaman -= jumlah;
        }
    }

    // Metode untuk melakukan pinjaman
    public void pinjam(int jumlah) {
        if (jumlahPinjaman + jumlah > limitPeminjaman) {
            System.out.println("Maaf, jumlah pinjaman melebihi batas limit.");
        } else {
            jumlahPinjaman += jumlah;
        }
    }
}
```

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

➢ **Output yang dihasilkan**

```
==================================================
Nama Anggota        : Siska Nuri Aprilia
Limit Pinjaman      : 5000000
==================================================

|----------------------------|
| Meminjam uang Rp 10.000.000 |
|----------------------------|
Maaf, jumlah pinjaman melebihi batas limit.
Jumlah Pinjaman Saat ini : 0

|----------------------------|
|  Meminjam uang Rp 4.000.000 |
|----------------------------|
Jumlah Pinjaman Saat ini : 4000000

|----------------------------|
| Bayar angsuran Rp 1.000.000 |
|----------------------------|
Jumlah Pinjaman Saat ini : 3000000

|----------------------------|
| Bayar angsuran Rp 2.000.000 |
|----------------------------|
Jumlah Pinjaman Saat ini : 1000000

Membayar angsuran 200.000
------------------------------------------
Jumlah pinjaman saat ini : 800000
```

➢ Kode Pemrograman TestKoperasi

```
public class TestKoperasi {
    public static void main(String[] args) {

        Anggota anggota1 = new Anggota("0304052005", "Siska Nuri Aprilia" ,  5000000);


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

        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("| Meminjam uang Rp 10.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.pinjam(10000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());
        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("|  Meminjam uang Rp 4.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.pinjam(4000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());
        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("| Bayar angsuran Rp 1.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.angsur(1000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());

        System.out.println();
        System.out.println("|----------------------------|");
        System.out.println("| Bayar angsuran Rp 2.000.000 |");
        System.out.println("|----------------------------|");
        anggota1.angsur(2000000);
        System.out.println("Jumlah Pinjaman Saat ini : " +
anggota1.getJumlahPinjam());

        System.out.println();
        System.out.println("Membayar angsuran 200.000");
        System.out.println("-----------------------------------------");
        anggota1.angsur(200000);

        System.out.println("Jumlah pinjaman saat ini : " +
anggota1.getJumlahPinjam());


    }
}
```

➢ Kode Pemrograman Class Anggota

```java
public class Anggota {
    public String nomorKTP;
    public String nama;
    public int limitPeminjaman;
    private int jumlahPinjaman;

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

    // Getter untuk nama
    public String getNama() {
        return nama;
    }

    // Getter untuk jumlah pinjaman
    public int getJumlahPinjam() {
        return jumlahPinjaman;
    }

    // Getter untuk limit peminjaman
    public int getLimitPinjam() {
        return limitPeminjaman;
    }

    // Metode untuk melakukan angsuran
    public void angsur(int jumlah) {
        if (jumlah > jumlahPinjaman+ 0.1) {
            System.out.println("Maaf, angsuran harus 10% dari jumlah pinjaman.");
        } else {
            jumlahPinjaman -= jumlah;
        }
    }

    // Metode untuk melakukan pinjaman
    public void pinjam(int jumlah) {
        if (jumlahPinjaman + jumlah > limitPeminjaman) {
            System.out.println("Maaf, jumlah pinjaman melebihi batas limit.");
        } else {
            jumlahPinjaman += jumlah;
        }
    }
}
```