**PEMROGRAMAN BERBASIS OBJEK**

**Praktikum Enkapsulasi**

****

**Lavina/2341760062**

**SIB 2D**

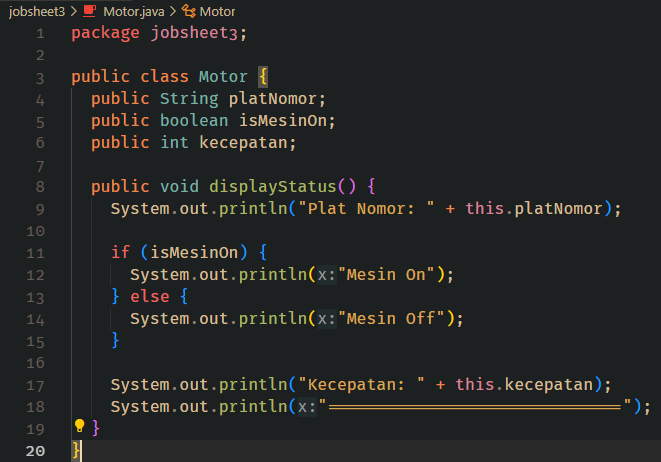
**PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS**

**POLITEKNIK NEGERI MALANG**

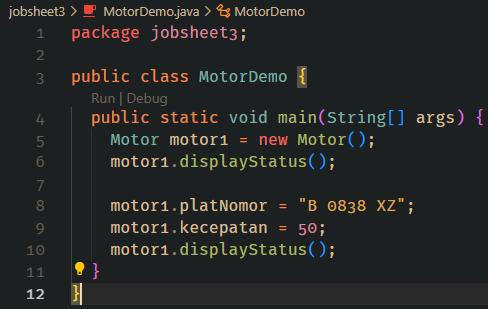
**2024**

**Percobaan 1 – Tanpa Enkapsulasi**

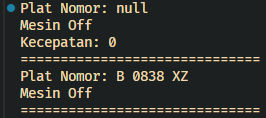
Motor.java



MotorDemo.java



Output:

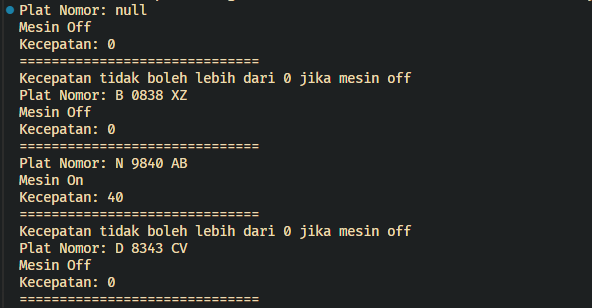


Pada motor1 dengan plat “B 0838 XZ”, kecepatannya dapat berubah dari 0 ke 50 padahal mesin motor masih dalam kondisi Off. Bagaimana mungkin atribut kecepatan bernilai 50 padahal mesin masih Off? Hal ini karena belum tersedia kontrol/batasan terhadap atribut kecepatan. Padahal, objek di dunia nyata selalu memiliki batasan dan mekanisme bagaimana objek tersebut dapat digunakan. 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".

Hasil modifikasi MotorDemo.java



Ouput:



**Percobaan 2 – Enkapsulasi**

Motor.java

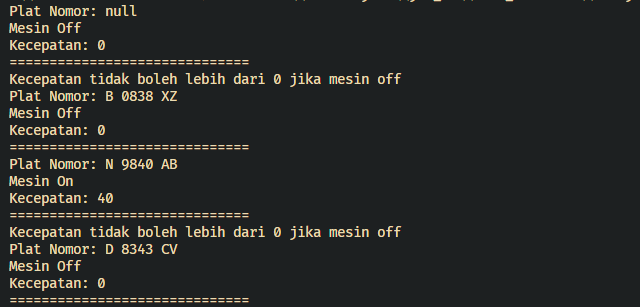




MotorDemo.java

****

Output:



**Pertanyaan**

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

**Jawab:** Karena sudah diatur *conditional statementnya* pada method setKecepatan() dimana jika !isMesinOn (mesin off) bernilai true, maka akan muncul peringatan tersebut.

1. Mengapat atribut merek, kecepatan, dan statusMesin diset private?

**Jawab:** Dibuat private agar tidak dinisialisasi sembarangan dari class lain, harus mengikuti alur algoritma yang sudah dibuat di method-method yang ada di class Motor yang mana hanya di class tersebut atribut-atribut private diatas bisa diakses.

1. Apa fungsi dari setter dan getter?

**Jawab:** Fungsi setter berguna untuk mengisi/menginisialisasi value dari atribut private sedangkan getter berfungsi untuk mendapatkan/mengakses value tersebut.

1. Ubah class Motor sehingga kecepatan maksimalnya adalah 100.

**Jawab:** Ditambahkan conditional statementnya seperti berikut:

  public void setKecepatan(int kecepatan) {

    if (!this.isMesinOn) {

      System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");

    } else if (kecepatan > 100) {

      System.out.println("Kecepatan tidak boleh lebih dari 100");

    } else {

      this.kecepatan = kecepatan;

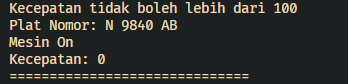
    }

  }

Lalu, ketika kecepatannya diset lebih dari 100:

motor2.setKecepatan(140);

Hasilnya :



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

**Jawab:** Ditambahkan conditional statementnya seperti berikut:

public void setKecepatan(int kecepatan) {

    if (!this.isMesinOn) {

      System.out.println("Kecepatan tidak boleh lebih dari 0 jika mesin off");

    } else if (kecepatan > 100) {

      System.out.println("Kecepatan tidak boleh lebih dari 100");

    } else if (kecepatan < 0) {

      System.out.println("Kecepatan tidak boleh kurang dari 0");

    } else {

      this.kecepatan = kecepatan;

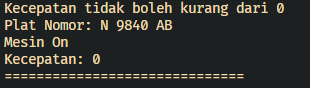
    }

  }

Jika kecepatannya dibuat negatif:

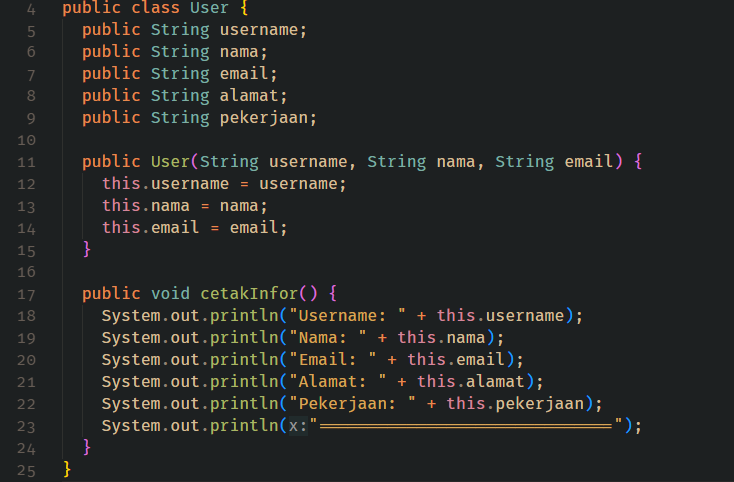
motor2.setKecepatan(-140);

Hasilnya:



**Percobaan 3 – Constructor**

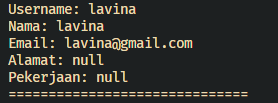
User.java



DemoUser.java

****

Output:



**Pertanyaan**

1. Apa yang dimaksud constructor?

**Jawab:** constructor biasa digunakan jika kita ingin membuat banyak objek dengan tipe dan atribut yang sama sehingga mempermudah penginisialisasian atribut.

1. Sebutkan aturan dalam membuat constructor?

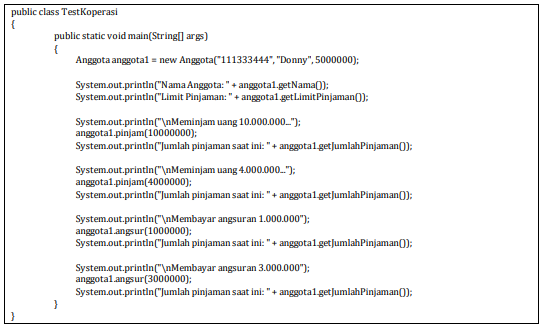
**Jawab:** Best practicenya, constructor dibuat dengan *upper-case first latter*, constructor bisa dibuat lebih dari satu tetapi tidak boleh memiliki parameter yang sama.

1. Lakukan analisa dan buat kesimpulan apakah constructor bisa bertipe private?

**Jawab:** Jika constructor dibuat private, maka constructor tersebut jadi tidak bisa digunakan, karena parameternya jadi tidak visible saat digunakan di class lain sedangkan saat instansiasi objek harus menyertakan constructor.

**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().

****

**Anggota.java**

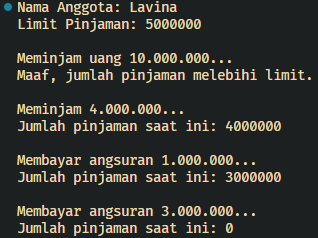
****

****

**TestKoperasi.java**

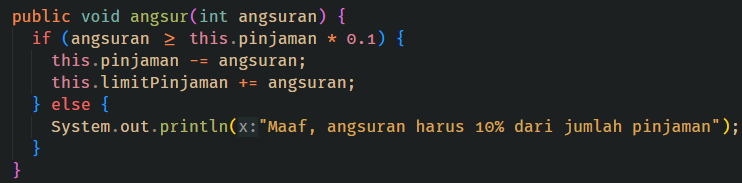
****

**Output:**

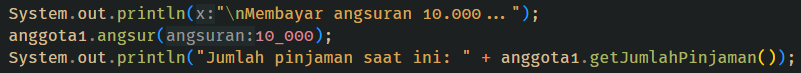
****

1. 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”

**Hasil modifikasi method angsur() :**

****

Untuk validasi, dibuat angsurannya menjadi hanya 10.000 :

****

Hasilnya :

