

TUGAS PBO MINGGU 3

Memenuhi Tugas Mata Kuliah Pemrograman Berbasis Objek

Dosen : Odhitya Desta Triswidrananta, S.Pd, M.Pd



Nama : Raden Dimas Erlangga

Kelas : D-III Manajemen Informatika 2E

Nim : 2031710121

PROGRAM STUDI D-III MANAJEMEN INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

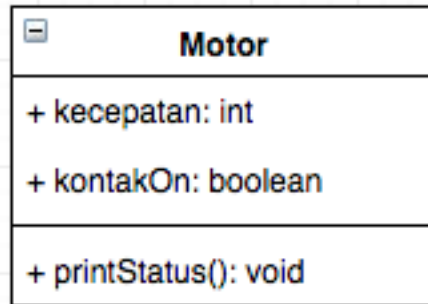
2021

JOBSHEET 3

Percobaan 1

Didalam percobaan enkapsulasi, buatlah class Motor yang memiliki atribut kecepatan dan kontakOn, dan memiliki method printStatus() untuk menampilkan status motor. Seperti berikut

bentuk UML class diagram class Motor adalah sebagai berikut:

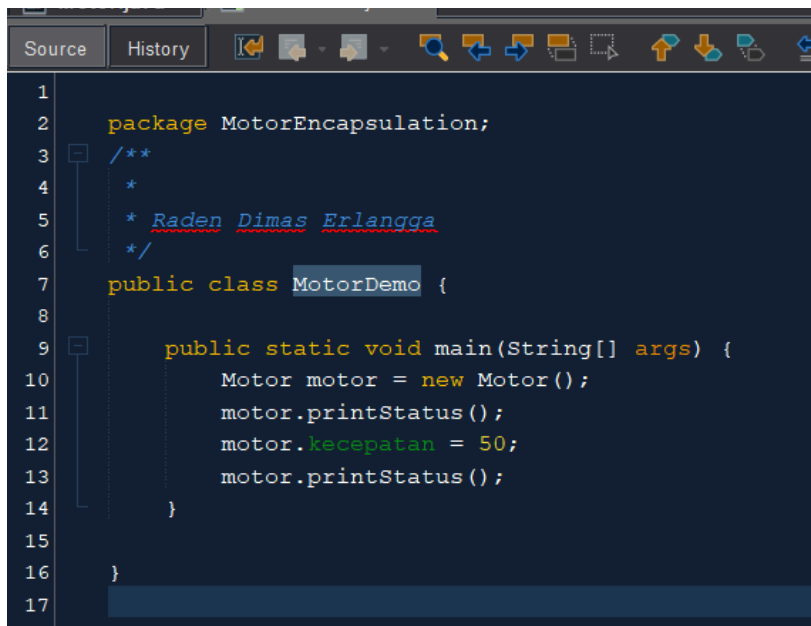


Program Java :

A. Class Motor

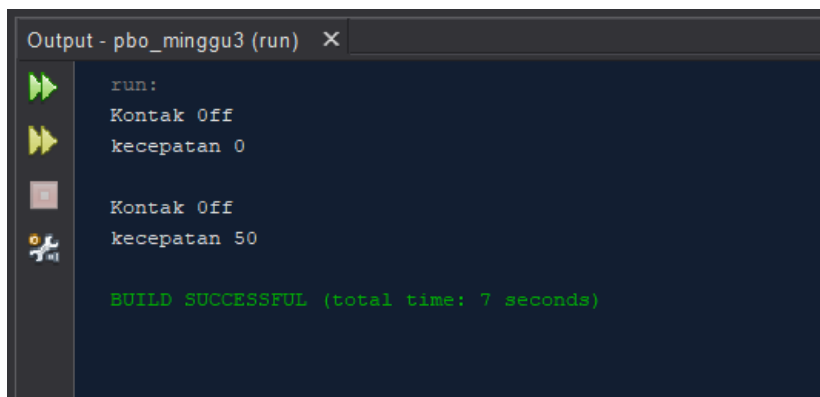
```
Source History
1
2 package MotorEncapsulation;
3
4 /**
5  * Raden Dimas Erlangga
6  */
7 public class Motor {
8     public int kecepatan = 0;
9     public boolean kontakOn = false;
10
11     public void printStatus(){
12         if (kontakOn == true){
13             System.out.println("Kontak On");
14         }
15         else {
16             System.out.println("Kontak Off");
17         }
18         System.out.println("kecepatan "+kecepatan+"\n");
19     }
20 }
21
```

B. Class MotorDemo



```
1
2 package MotorEncapsulation;
3 /**
4  *
5  * Raden Dimas Erlangga
6  */
7 public class MotorDemo {
8
9     public static void main(String[] args) {
10         Motor motor = new Motor();
11         motor.printStatus();
12         motor.kecepatan = 50;
13         motor.printStatus();
14     }
15
16 }
17
```

C. Hasilnya adalah sebagai berikut:



```
Output - pbo_minggu3 (run) X
run:
Kontak Off
kecepatan 0

Kontak Off
kecepatan 50

BUILD SUCCESSFUL (total time: 7 seconds)
```

Dari percobaan 1 - enkapsulasi, menurut anda, adakah yang janggal?

Yaitu, kecepatan motor tiba-tiba saja berubah dari 0 ke 50. Lebih janggal lagi, posisi kontak motor masih dalam kondisi OFF. Bagaimana mungkin sebuah motor bisa sekejap berkecepatan dari nol ke 50, dan itupun kunci kontaknya OFF?

dalam hal ini, akses ke atribut motor ternyata tidak terkontrol. Padahal, objek di dunia nyata selalu memiliki batasan dan mekanisme bagaimana objek tersebut dapat digunakan. Lalu, bagaimana kita bisa memperbaiki class Motor diatas agar dapat digunakan dengan baik? Kita bisa pertimbangkan beberapa hal berikut ini:

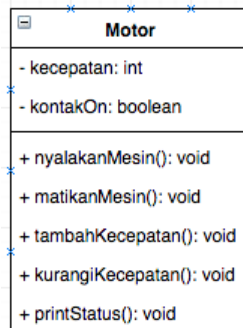
1. Menyembunyikan atribut internal (kecepatan, kontakOn) dari pengguna (class lain)
2. Menyediakan method khusus untuk mengakses atribut.

Untuk itu mari kita lanjutkan percobaan berikutnya tentang Access Modifier.

Percobaan 2 Access Modifier

Pada percobaan ini akan digunakan access modifier untuk memperbaiki cara kerja class Motor pada percobaan ke-1.

1. Ubah cara kerja class motor sesuai dengan UML class diagram berikut.



- a. Berdasarkan UML class diagram tersebut maka class Motor terdapat perubahan, yaitu access modifier kecepatan dan kontakOn menjadi private
- b. Tambahkan method nyalakanMesin, matikanMesin, tambahKecepatan, kurangiKecepatan.

Maka :

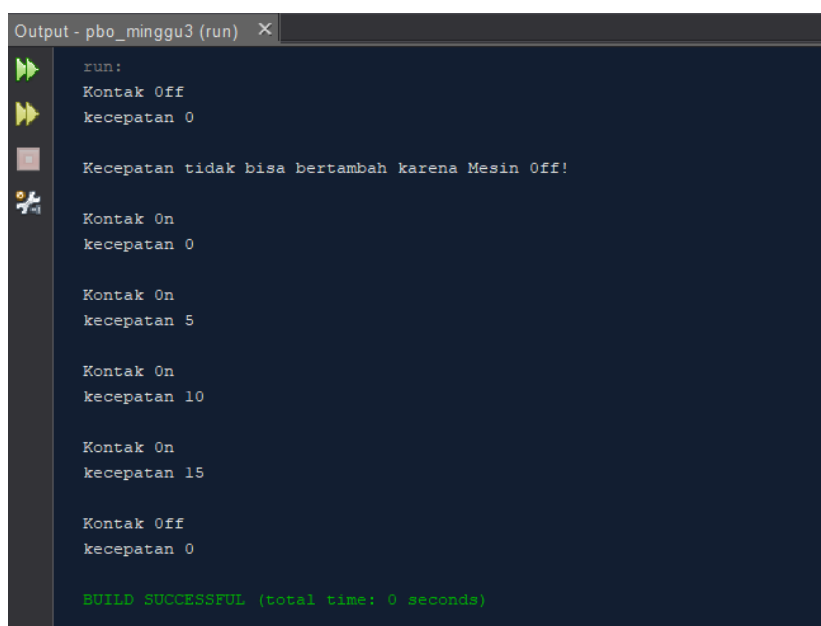
A. Class Motor

```
1 package MotorEncapsulation;
2
3
4 /**
5  * Reden Dimas Erlangga
6  */
7 public class Motor {
8     private int kecepatan = 0;
9     private boolean kontakOn = false;
10    public void nyalakanMesin(){
11        kontakOn = true;
12    }
13    public void matikanMesin(){
14        kontakOn = false;
15        kecepatan = 0;
16    }
17    public void tambahKecepatan(){
18        if (kontakOn == true){
19            kecepatan += 5;
20        }
21        else {
22            System.out.println("Kecepatan tidak bisa bertambah karena Mesin Off! \n");
23        }
24    }
25    public void kurangiKecepatan(){
26        if (kontakOn == true){
27            kecepatan -= 5;
28        }
29        else {
30            System.out.println("Kecepatan tidak bisa berkurang karena mesin Off");
31        }
32    }
33
34
35    public void printStatus(){
36        if (kontakOn == true){
37            System.out.println("Kontak On");
38        }
39        else {
40            System.out.println("Kontak Off");
41        }
42        System.out.println("kecepatan "+kecepatan+"\n");
43    }
44 }
45
```

B. Class MotorDemo

```
1
2  package MotorEncapsulation;
3  /**
4   *
5   * Raden Dimas Erlangga
6   */
7  public class MotorDemo {
8
9      public static void main(String[] args) {
10         Motor motor = new Motor();
11         motor.printStatus();
12         motor.tambahKecepatan();
13
14         motor.nyalakanMesin();
15         motor.printStatus();
16
17         motor.tambahKecepatan();
18         motor.printStatus();
19
20         motor.tambahKecepatan();
21         motor.printStatus();
22
23         motor.tambahKecepatan();
24         motor.printStatus();
25
26         motor.matikanMesin();
27         motor.printStatus();
28     }
29
30 }
31
```

C. Hasil Running



```
Output - pbo_minggu3 (run) X
run:
Kontak Off
kecepatan 0

Kecepatan tidak bisa bertambah karena Mesin Off!

Kontak On
kecepatan 0

Kontak On
kecepatan 5

Kontak On
kecepatan 10

Kontak On
kecepatan 15

Kontak Off
kecepatan 0

BUILD SUCCESSFUL (total time: 0 seconds)
```

Dari percobaan diatas, dapat kita amati sekarang atribut kecepatan tidak bisa diakses oleh pengguna dan diganti nilainya secara sembarangan. Bahkan ketika mencoba menambah kecepatan saat posisi kontak masih OFF, maka akan muncul notifikasi bahwa mesin OFF. Untuk mendapatkan kecepatan yang diinginkan, maka harus dilakukan secara gradual, yaitu dengan memanggil method tambahKecepatan() beberapa kali. Hal ini mirip seperti saat kita mengendarai motor.

Pertanyaan

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

Jawaban :

Karena pada class Motor, terdapat if statement, yang mencegah kecepatan mesin bertambah jika keadaan mesin mati.

2. Mengapa atribut kecepatan dan kontakOn diset private?

Jawaban :

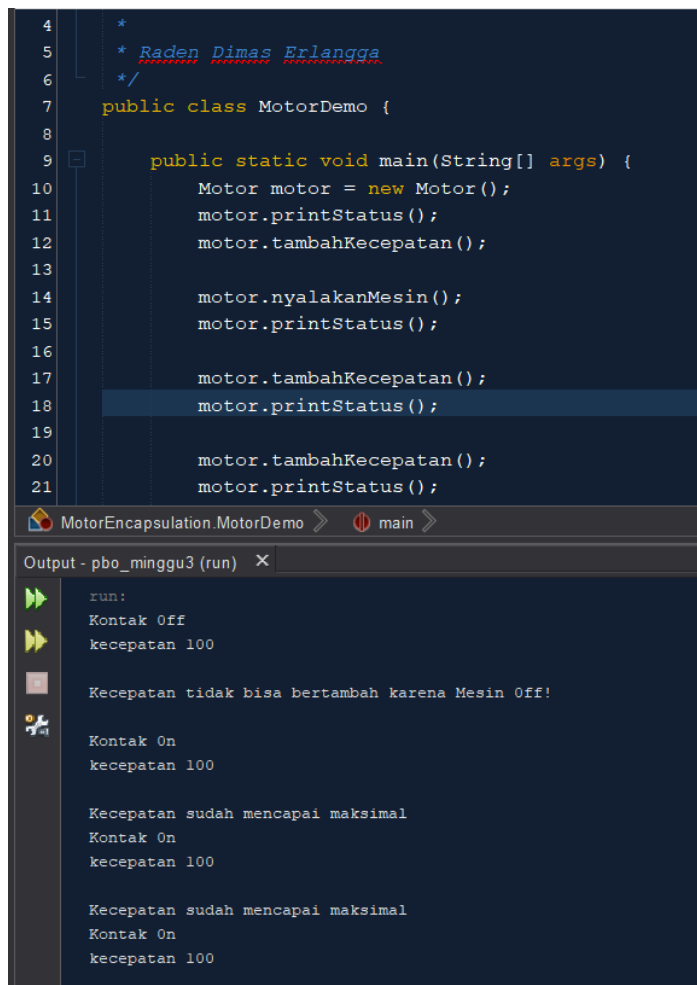
Karena class motor berada di satu package yang sama, maka bisa menggunakan atribut private untuk melakukan access modifier

3. Ubah class Motor sehingga kecepatan maksimalnya adalah 100!

Jawaban :

```
7      public class Motor {
8          private int kecepatan = 100;
9          private boolean kontakOn = false;
10         public void nyalakanMesin() {
11             kontakOn = true;
12         }
13         public void matikanMesin() {
14             kontakOn = false;
15             kecepatan = 0;
16         }
17         public void tambahKecepatan() {
18             if (kontakOn == true) {
19                 if (kecepatan == 100) {
20                     System.out.println("Kecepatan sudah mencapai maksimal");
21                 } else {
22                     kecepatan += 5;
23                 }
24             }
25         }
26     }
27 }
```

Disini saya mendeklarasi kecepatan = 100 untuk melakukan test apakah if statement bekerja, saat di run hasil nya adalah sebagai berikut :



```
4  *
5  * Raden Dimas Erlangga
6  */
7  public class MotorDemo {
8
9      public static void main(String[] args) {
10         Motor motor = new Motor();
11         motor.printStatus();
12         motor.tambahKecepatan();
13
14         motor.nyalakanMesin();
15         motor.printStatus();
16
17         motor.tambahKecepatan();
18         motor.printStatus();
19
20         motor.tambahKecepatan();
21         motor.printStatus();
22     }
```

MotorEncapsulation.MotorDemo main

Output - pbo_minggu3 (run)

```
run:
Kontak Off
kecepatan 100

Kecepatan tidak bisa bertambah karena Mesin Off!

Kontak On
kecepatan 100

Kecepatan sudah mencapai maksimal
Kontak On
kecepatan 100

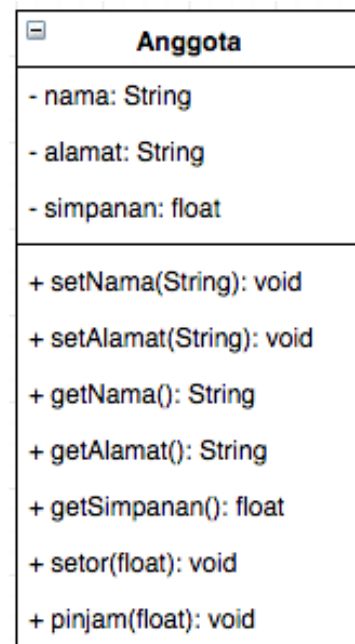
Kecepatan sudah mencapai maksimal
Kontak On
kecepatan 100
```

Method tambah kecepatan menampilkan print “kecepatan sudah mencapai maksimal”

Percobaan 3 Getter & Setter

Misalkan di sebuah sistem informasi koperasi, terdapat class Anggota. Anggota memiliki atribut nama, alamat dan simpanan, dan method setter, getter dan setor dan pinjam. Semua atribut pada anggota tidak boleh diubah sembarangan, melainkan hanya dapat diubah melalui method setter, getter, setor dan tarik. Khusus untuk atribut simpanan tidak terdapat setter karena simpanan akan bertambah ketika melakukan transaksi setor dan akan berkurang ketika melakukan peminjaman/tarik.

1. Berikut ini UML class buatlah class Mahasiswa pada program:



2. class Anggota :

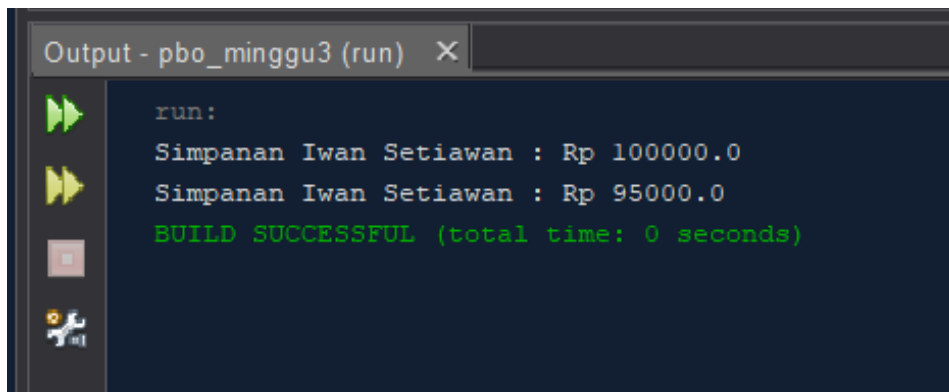
```
1 package KoperasiGetterSetter;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class Anggota {
6     private String nama;
7     private String alamat;
8     private float simpanan;
9
10    public void setNama(String nama){
11        this.nama = nama;
12    }
13    public void setAlamat(String alamat){
14        this.alamat = alamat;
15    }
16    public String getNama(){
17        return nama;
18    }
19    public String getAlamat(){
20        return alamat;
21    }
22    public float getSimpanan(){
23        return simpanan;
24    }
25    public void setor(float uang){
26        simpanan += uang;
27    }
28    public void pinjam(float uang){
29        simpanan -= uang;
30    }
31 }
```

Jika diperhatikan pada class Anggota, atribut nama dan alamat memiliki masing-masing 1 getter dan setter. Sedangkan atribut simpanan hanya memiliki getSimpanan() saja, karena seperti tujuan awal, atribut simpanan akan berubah nilainya jika melakukan transaksi setor() dan pinjam/tarik().

3. class KoperasiDemo

```
1 package KoperasiGetterSetter;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class KoperasiDemo {
6     public static void main(String[] args) {
7         Anggota anggotat1 = new Anggota();
8         anggotat1.setNama("Iwan Setiawan");
9         anggotat1.setAlamat("Jalan Sukarno Hatta no 10");
10        anggotat1.setor(100000);
11        System.out.println("Simpanan " + anggotat1.getNama() + " : Rp " + anggotat1.getSimpanan());
12
13        anggotat1.pinjam(5000);
14        System.out.println("Simpanan " + anggotat1.getNama() + " : Rp " + anggotat1.getSimpanan());
15    }
16 }
17 }
```

Hasil Running :

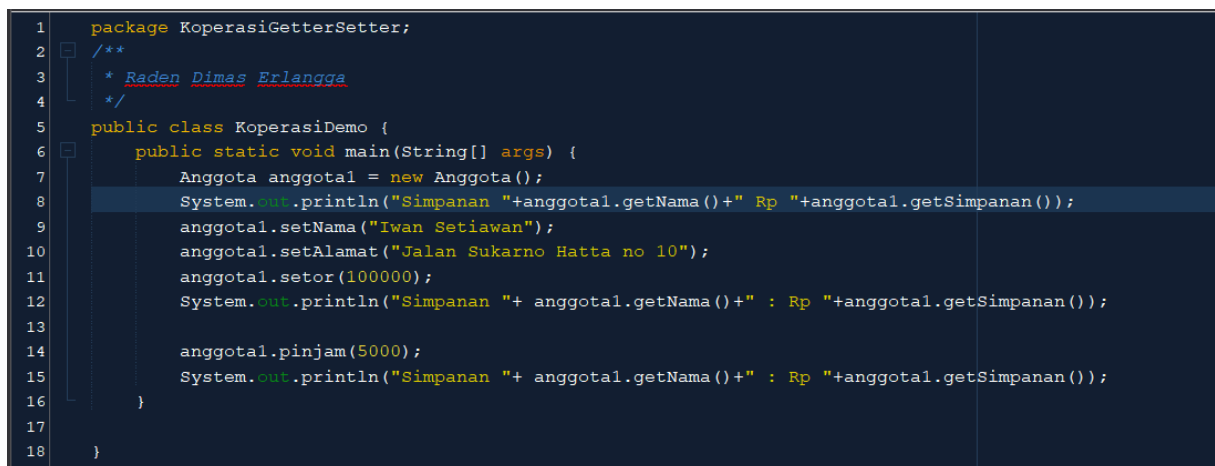


```
Output - pbo_minggu3 (run) X
run:
Simpanan Iwan Setiawan : Rp 100000.0
Simpanan Iwan Setiawan : Rp 95000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Dapat dilihat pada hasil percobaan diatas, untuk mengubah simpanan tidak dilakukan secara langsung dengan mengubah atribut simpanan, melainkan melalui method `setor()` dan `pinjam()`. Untuk menampilkan nama pun harus melalui method `getNama()`, dan untuk menampilkan simpanan melalui `getSimpanan()`.

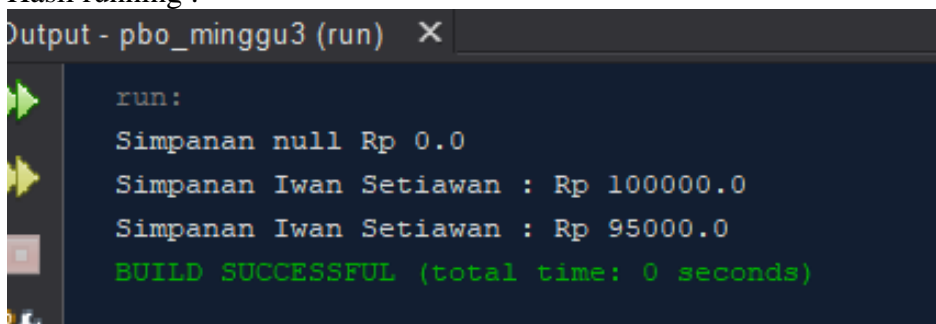
Percobaan 4 Konstruktor, instansiasi

1. ubah class `KoperasiDemo` :



```
1 package KoperasiGetterSetter;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class KoperasiDemo {
6     public static void main(String[] args) {
7         Anggota anggota1 = new Anggota();
8         System.out.println("Simpanan "+anggota1.getNama()+" Rp "+anggota1.getSimpanan());
9         anggota1.setNama("Iwan Setiawan");
10        anggota1.setAlamat("Jalan Sukarno Hatta no 10");
11        anggota1.setor(100000);
12        System.out.println("Simpanan "+ anggota1.getNama()+" : Rp "+anggota1.getSimpanan());
13
14        anggota1.pinjam(5000);
15        System.out.println("Simpanan "+ anggota1.getNama()+" : Rp "+anggota1.getSimpanan());
16    }
17 }
18 }
```

Hasil running :



```
Output - pbo_minggu3 (run) X
run:
Simpanan null Rp 0.0
Simpanan Iwan Setiawan : Rp 100000.0
Simpanan Iwan Setiawan : Rp 95000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Dapat dilihat hasil running program, ketika dilakukan pemanggilan method `getNama()` hasilnya hal ini terjadi karena atribut nama belum diset nilai defaultnya. Hal ini dapat ditangani dengan membuat konstruktor.

3. Ubah Class anggota menjadi seperti berikut :

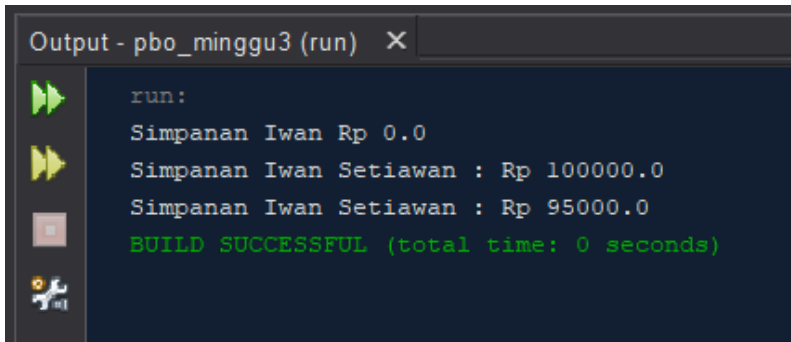
```
1 package KoperasiGetterSetter;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class Anggota {
6     private String nama;
7     private String alamat;
8     private float simpanan;
9
10    Anggota(String nama, String alamat){
11        this.nama = nama;
12        this.alamat = alamat;
13        this.simpanan = 0;
14    }
15
16    public void setNama(String nama){
17        this.nama = nama;
18    }
19    public void setAlamat(String alamat){
20        this.alamat = alamat;
21    }
22    public String getNama(){
23        return nama;
24    }
25    public String getAlamat(){
26        return alamat;
27    }
28    public float getSimpanan(){
29        return simpanan;
30    }
31    public void setor(float uang){
32        simpanan += uang;
33    }
34    public void pinjam(float uang){
35        simpanan -= uang;
36    }
37 }
```

Pada class Anggota dibuat konstruktor dengan access modifier default yang memiliki 2 parameter nama dan alamat. Dan didalam konstruktor tersebut dipastikan nilai simpanan untuk pertama kali adalah Rp. 0

4. selanjut nya ubah class koperasiDemo sebagai berikut :

```
1 package KoperasiGetterSetter;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class KoperasiDemo {
6     public static void main(String[] args) {
7         Anggota anggotat1 = new Anggota("Iwan", "Jalan Mawar");
8         System.out.println("Simpanan "+ anggotat1.getNama() + " Rp "+ anggotat1.getSimpanan());
9         anggotat1.setNama("Iwan Setiawan");
10        anggotat1.setAlamat("Jalan Sukarno Hatta no 10");
11        anggotat1.setor(100000);
12        System.out.println("Simpanan "+ anggotat1.getNama() + " : Rp "+ anggotat1.getSimpanan());
13
14        anggotat1.pinjam(5000);
15        System.out.println("Simpanan "+ anggotat1.getNama() + " : Rp "+ anggotat1.getSimpanan());
16    }
17 }
18 }
```

5. Hasil Running



```
Output - pbo_minggu3 (run) X
run:
Simpanan Iwan Rp 0.0
Simpanan Iwan Setiawan : Rp 100000.0
Simpanan Iwan Setiawan : Rp 95000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Setelah menambah konstruktor pada class Anggota maka atribut nama dan alamat secara otomatis harus diset terlebih dahulu dengan melakukan passing parameter jika melakukan instansiasi class Anggota. Hal ini biasa dilakukan untuk atribut yang membutuhkan nilai yang spesifik. Jika tidak membutuhkan nilai spesifik dalam konstruktor tidak perlu parameter. Contohnya simpanan untuk anggota baru diset 0, maka simpanan tidak perlu untuk dijadikan parameter pada konstruktor.

Pertanyaan - Percobaan 3 dan 4

1. Apa yang dimaksud getter dan setter?

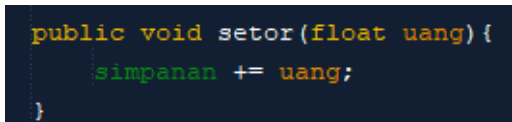
Jawaban : Getter adalah method yang memiliki return statement untuk mengambil nilai dari atribut private, sedangkan setter adalah method yang tidak memiliki return, namun setter berfungsi untuk merubah / manipulasi nilai dari atribut private.

2. Apa kegunaan dari method getSimpanan()?

Jawaban : untuk menyimpan data setor dan data pinjam.

3. Method apa yang digunakan untuk menambah saldo?

Jawaban : method setor :



```
public void setor(float uang){
    simpanan += uang;
}
```

4. Apa yang dimaksud konstruktor?

Jawaban : Konstruktor merupakan suatu method yang tidak memiliki return statement, Konstruktor akan memberikan nilai awal pada saat suatu objek pada saat perintah new / instansiasi dijalankan untuk membuat suatu objek.

5. Sebutkan aturan dalam membuat konstruktor?

Jawaban : Aturan dalam membuat konstruktor adalah : Nama konstruktor harus sama dengan class, konstruktor tidak boleh menggunakan modifier abstract, static, final dan synchronized

6. Apakah boleh konstruktor bertipe private?

Jawaban : boleh jika class main dan class yang akan di konstruktor kan berada dalam satu package yang sama.

7. Kapan menggunakan parameter dengan passing parameter?

Jawaban : passing parameter digunakan untuk memanggil sebuah data Ketika menggunakan sebuah konstruktor berparameter.

8. Apa perbedaan atribut class dan instansiasi atribut?

Jawaban : atribut class merupakan variable pada suatu class, sedangkan instansiasi atribut merupakan nilai yang di inputkan pada suatu input.

9. Apa perbedaan class method dan instansiasi method?

Class method adalah sebuah method yang berada di dalam sebuah class, sedangkan instansiasi method adalah pemanggilan dari method yang berada di luar kelas yang memanggil method. Contoh nya instansiasi method print() yang dilakukan pada method main, namun method print ini di panggil dari class selain main.

Kesimpulan

Dari percobaan diatas, telah dipelajari kosep dari enkapsulasi, kontruktor, access modifier yang terdiri dari 4 jenis yaitu public, protected, default dan private. Konsep atribut atau method class yang ada di dalam blok code class dan konsep instansiasi atribut atau method. Cara penggunaan getter dan setter beserta fungsi dari getter dan setter. Dan juga telah dipelajari atau memahami notasi UML

Tugas

1. Cobalah program dibawah ini dan tuliskan hasil outputnya

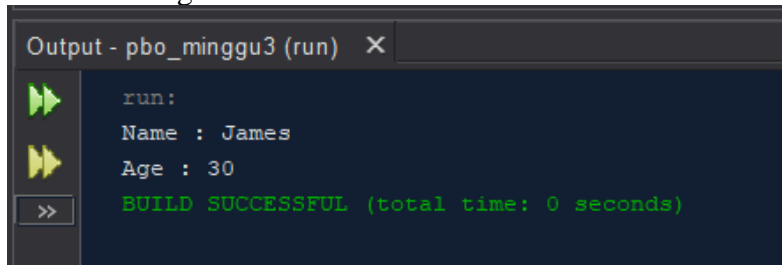
A. Class EncapDemo

```
2 package tugas_minggu3;
3 /**
4  * Raden Dimas Erlangga
5  */
6 public class EncapDemo {
7     private String name;
8     private int age;
9
10    public String getName() {
11        return name;
12    }
13
14    public void setName(String newName) {
15        name = newName;
16    }
17
18    public int getAge() {
19        return age;
20    }
21
22    public void setAge(int newAge) {
23        if (newAge > 30) {
24            age = 30;
25        } else {
26            age = newAge;
27        }
28    }
29 }
```

B. Class EncapTest

```
1 package tugas_minggu3;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class EncapTest {
6     public static void main(String[] args) {
7         EncapDemo encap = new EncapDemo();
8         encap.setName("James");
9         encap.setAge(35);
10        System.out.println("Name : " + encap.getName());
11        System.out.println("Age : " + encap.getAge());
12    }
13 }
14
```

Hasil Running :



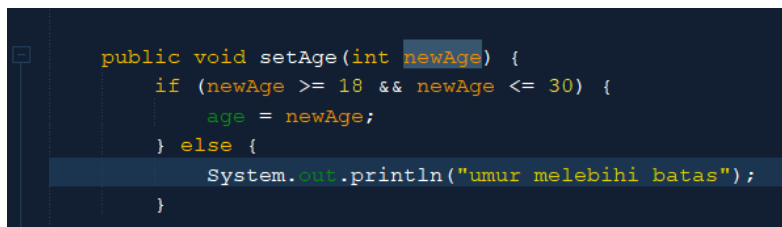
```
Output - pbo_minggu3 (run) X
run:
Name : James
Age : 30
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Pada program diatas, pada class EncapTest kita mengeset age dengan nilai 35, namun pada saat ditampilkan ke layar nilainya 30, jelaskan mengapa.

Jawaban : karena pada method setAge jika nilai yang di inputkan lebih dari 30 maka akan mendapatkan nilai age = 30.

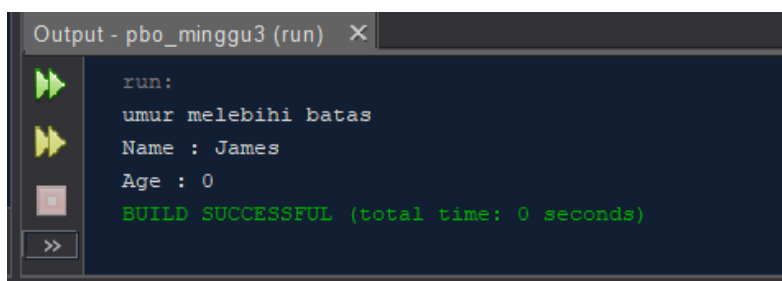
3. Ubah program diatas agar atribut age dapat diberi nilai maksimal 30 dan minimal 18.

Jawaban :



```
public void setAge(int newAge) {
    if (newAge >= 18 && newAge <= 30) {
        age = newAge;
    } else {
        System.out.println("umur melebihi batas");
    }
}
```

Hasil Running :



```
Output - pbo_minggu3 (run) X
run:
umur melebihi batas
Name : James
Age : 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

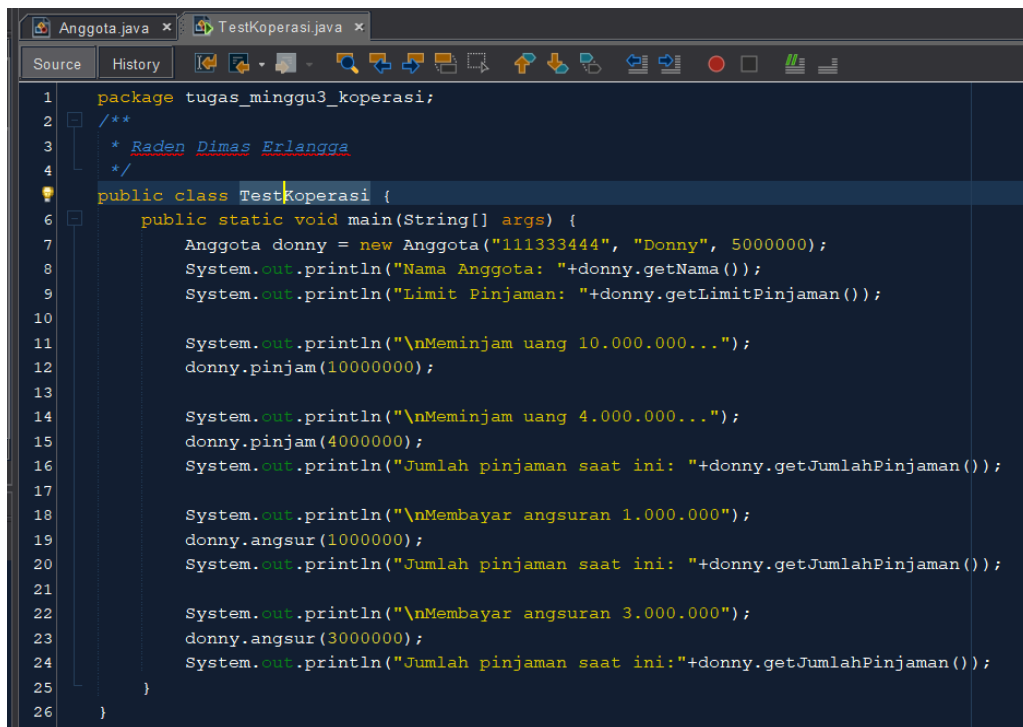
4. Pada sebuah sistem informasi koperasi simpan pinjam, terdapat class Anggota yang memiliki atribut antara lain nomor KTP, nama, limit peminjaman, dan jumlah pinjaman. Anggota dapat meminjam uang dengan batas limit peminjaman yang ditentukan. Anggota juga dapat mengangsur pinjaman. Ketika Anggota tersebut mengangsur pinjaman, maka jumlah pinjaman akan berkurang sesuai dengan nominal yang diangsur. Buatlah class Anggota tersebut, berikan atribut, method dan konstruktor sesuai dengan kebutuhan. Uji dengan TestKoperasi berikut ini untuk memeriksa apakah class Anggota yang anda buat telah sesuai dengan yang diharapkan.

Jawaban :

A. Class Anggota :

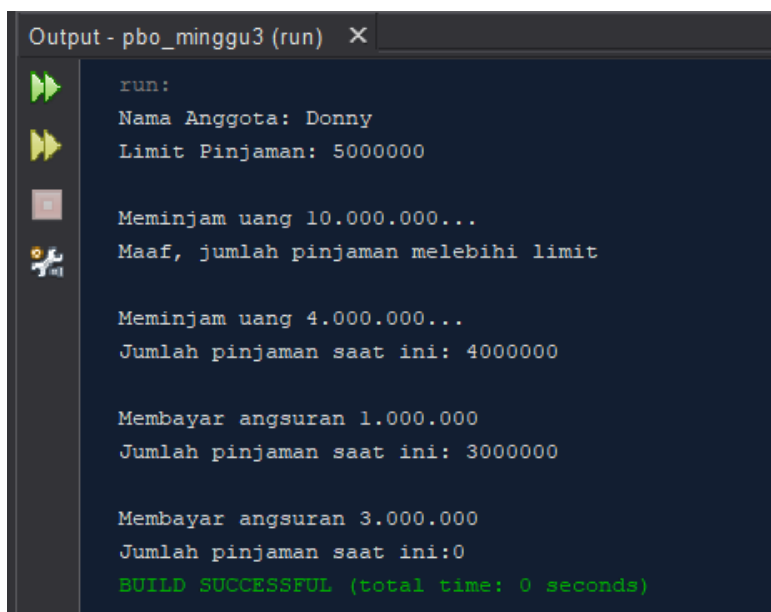
```
1 package tugas_minggu3_koperasi;
2 /**
3  * Raden Dimas Erlangga
4  */
5 public class Anggota {
6     private String nomorKTP;
7     private String nama;
8     private int limitPeminjaman;
9     private int jumlahPeminjaman;
10
11     public Anggota(String nomorKTP, String nama, int limitPinjaman) {
12         this.nomorKTP = nomorKTP;
13         this.nama = nama;
14         this.limitPeminjaman = limitPinjaman;
15     }
16
17     public void setNama(String newName) {
18         nama = newName;
19     }
20
21     public void setLimitPeminjaman(int newLimit) {
22         limitPeminjaman = newLimit;
23     }
24
25     public String getNama() {
26         return nama;
27     }
28
29     public int getLimitPinjaman() {
30         return limitPeminjaman;
31     }
32
33     public int getJumlahPinjaman() {
34         return jumlahPeminjaman;
35     }
36
37     public void pinjam(int Pinjam) {
38         if (Pinjam > limitPeminjaman) {
39             System.out.println("Maaf, jumlah pinjaman melebihi limit");
40         } else {
41             jumlahPeminjaman = jumlahPeminjaman + Pinjam;
42         }
43     }
44
45     public void angsur(int newAngsur) {
46         if (jumlahPeminjaman == 0) {
47             System.out.println("Anda belum pernah melakukan peminjaman");
48         } else {
49             jumlahPeminjaman = jumlahPeminjaman - newAngsur;
50         }
51     }
52 }
```

B. Class Test koperasi :



```
1 package tugas_minggu3_koperasi;
2 /**
3  * Raden Dimas Erlangga
4  */
5
6 public class TestKoperasi {
7     public static void main(String[] args) {
8         Anggota donny = new Anggota("111333444", "Donny", 5000000);
9         System.out.println("Nama Anggota: "+donny.getNama());
10        System.out.println("Limit Pinjaman: "+donny.getLimitPinjaman());
11
12        System.out.println("\nMeminjam uang 10.000.000...");
13        donny.pinjam(10000000);
14
15        System.out.println("\nMeminjam uang 4.000.000...");
16        donny.pinjam(4000000);
17        System.out.println("Jumlah pinjaman saat ini: "+donny.getJumlahPinjaman());
18
19        System.out.println("\nMembayar angsuran 1.000.000");
20        donny.angsur(1000000);
21        System.out.println("Jumlah pinjaman saat ini: "+donny.getJumlahPinjaman());
22
23        System.out.println("\nMembayar angsuran 3.000.000");
24        donny.angsur(3000000);
25        System.out.println("Jumlah pinjaman saat ini: "+donny.getJumlahPinjaman());
26    }
27 }
```

C. Hasil running :



```
Output - pbo_minggu3 (run) X
run:
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
BUILD SUCCESSFUL (total time: 0 seconds)
```


5. Modifikasi soal no. 4 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”.

```

45     public void angsur(int newAngsur) {
46         if (newAngsur < 0.1 * jumlahPeminjaman) {
47             System.out.println("Maaf Angsuran harus 10% dari jumlah peminjaman");
48         } else {
49             jumlahPeminjaman = jumlahPeminjaman - newAngsur;
50         }

```

6. Modifikasi class TestKoperasi, agar jumlah pinjaman dan angsuran dapat menerima input dari console.

```

1  package tugas_minggu3_koperasi;
2  /**
3   * Raden Dimas Erlangga
4   */
5  import java.util.Scanner;
6
7  public class TestKoperasi {
8      public static void main(String[] args) {
9          Scanner sc = new Scanner (System.in);
10         int pinjam, angsur;
11
12         Anggota donny = new Anggota("111333444", "Donny", 5000000);
13         System.out.println("Nama Anggota: "+donny.getNama());
14         System.out.println("Limit Pinjaman: "+donny.getLimitPinjaman());
15
16         System.out.println("Masukkan jumlah uang yang akan dipinjam : ");
17         pinjam = sc.nextInt();
18         donny.pinjam(pinjam);
19
20         System.out.println("\nMeminjam uang Rp."+pinjam+".....");
21         System.out.println("Jumlah pinjaman saat ini: Rp."+donny.getJumlahPinjaman());
22
23         System.out.println("Masukkan jumlah angsuran : ");
24         angsur = sc.nextInt();
25         donny.angsur(angsur);
26         System.out.println("\nMembayar angsuran sejumlah Rp."+angsur+".....");
27         System.out.println("Jumlah pinjaman saat ini: "+donny.getJumlahPinjaman());
28     }
29 }
30

```

Hasil running :

```

Output - pbo_minggu3 (run) X
run:
Nama Anggota: Donny
Limit Pinjaman: 5000000
Masukkan jumlah uang yang akan dipinjam :
3000000

Meminjam uang Rp.3000000.....
Jumlah pinjaman saat ini: Rp.3000000
Masukkan jumlah angsuran :
3000000

Membayar angsuran sejumlah Rp.3000000.....
Jumlah pinjaman saat ini: 0
BUILD SUCCESSFUL (total time: 16 seconds)

```