OBJECTS ORIENTED PROGRAM

JOBSHEET 3

(Encapsulation on Object-Oriented Programming)

ANANDA AZ HARUDDIN SALIMA

2241720071

2 I

3.1 Percobaan 1: Enkapsulasi

In the encapsulation experiment, create a Motor class that has the speed and kontakOn attributes, and has a printStatus() method to display the motor status. As follows:

1.Open Netbeans, create a MotorEncapsulation project.

2.Create a Motor class. Right-click on package motorencapsulation – New – Java Class.

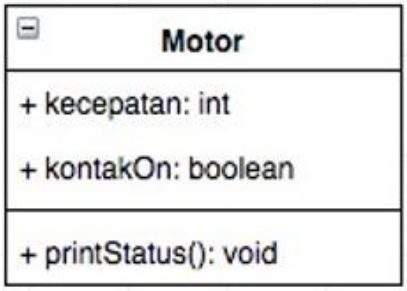
3.Type the Motor class code below.

```java
package motorencapsulation;

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

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

uml.java

```java

public class motor {
    public int kecepatan = 0;
    public boolean kontakOn = false;
    public void printStatus() {
    if (kontakOn == true) {
    System.out.println("Kontak On");
    } else {
    System.out.println("Kontak Off");
    }
    System.out.println("Kecepatan " + kecepatan + "\n");
    }

    }

```

The shape of UML in the motor class is as follows:



4. Then create a MotorDemo class, type the following code.

```java
package motorencapsulation;

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

motor.java > motor > Main

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

    }
}
```

5. The result is as follows:

```
Kontak Off
Kecepatan 0

Kontak Off
Kecepatan 50


Process finished with exit code 0
```

From experiment 1 - encapsulation, do you think there is anything odd?

That is, the speed of the motor suddenly changes from 0 to 50. Even more odd, the contact position of the motor is still in the OFF condition. How can a motor go from zero to 50, and even then the ignition is OFF?

Well in this case, access to the attributes of the motor turned out to be uncontrolled. In fact, objects in the real world always have limits and mechanisms for how they can be used. Then, how can we improve the Motor class above so that it can be used properly? We can consider the following:

1.hide internal attributes (speed,contact on) from users (other classes)

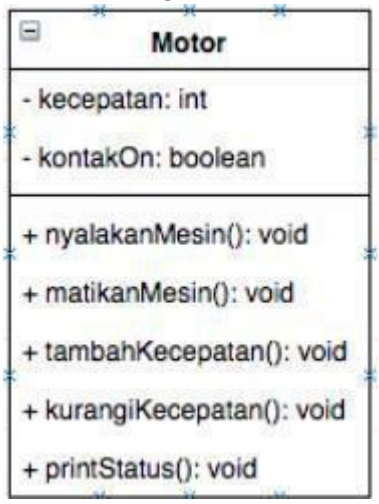
2. Provide custom methids to access attributes

For that, let's continue the following experiment about Access Modifier

3.2 Percobaan 2 – Access Modifier

In this experiment, an access modifier will be used to improve how the Motor class works on the 1st try.

1. Change how the motor class works according to the following UML class diagram.



1. based on the UML class diagram, the Motor class has changes, namely:

    1. Change access modifier speed and kontakOn to private

    2. Add the Turn on Machine method, turn off Engine, increase Speed, decrease Speed.

The implementation of the Motor class is as follows:

```
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");

    }

}

public void kurangiKecepatan() {

    if (kontakOn == true) {

        kecepatan -= 5;
```

```
        } else {

            System.out.println("Kecepatan tidak bisa berkurang karena Mesin ");

        }

    }

    public void printStatus() {

        if (kontakOn == true) {

            System.out.println("Kontak On");

        } else {

            System.out.println("Kontak Off");

        }


        System.out.println("Kecepatan " + kecepatan + "\n");

    }

}
```

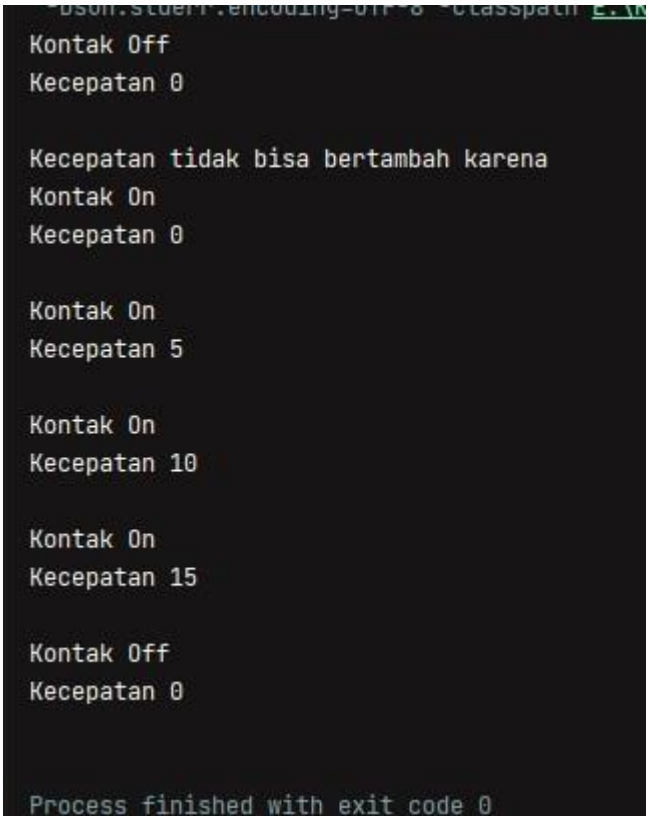
1. Then in the MotorDemo class, change the code to something like this:

```
public class Main {
    public static void main(String[] args) {
        motor motor = new motor();
```

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

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

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

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



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

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



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



        motor.matikanMesin();
        motor.printStatus();




    }
}
```

1. The result of the MotorDemo class is as follows:



From the experiment above, we can observe that now the speed attribute cannot be accessed by the user and is changed arbitrarily. Even when trying to increase speed when the contact position is still OFF, a notification will appear that the machine is OFF. To get the desired speed, it must be done gradually, namely by calling the addSpeed() method several times. This is similar to when we ride a motorcycle.

3.3 Pertanyaan

1. In the TestMobil class, when we pick up speed for the first time, why does the warning appear "Speed cannot increase due to Engine Off!"?

This is because the value of the contactOn attribute is *false*, so that when branching *if else* with the indicator kontakOn == *true* cannot enter into speed increments and will enter into else branching.

2. Why are the speed and kontakOn attributes set to private?

This is to make these attributes only accessible from that class and cannot be accessed from outside the Motor class *.java*

1. Change the Motor class so that the maximum speed is 100!

Add the maxSpeed attribute:

```
private int maxSpeed = 100;
```

Tambahkan kondisi pada method tambahKecepatan:

```
public void tambahKecepatan() {

   if (kontakOn == true && kecepatan < maxSpeed) { kecepatan += 5;

   } else if (kontakOn == false) {

       System.out.println("Kecepatan tidak bisa bertambah karena Mesin

               Off! \n");

   } else {

       System.out.println("Kecepatan melebihi batas! \n");

   }

}
```

Melakukan looping untuk mengetahui hasil akhir:

```
for (int i = 0; i < 22; i++) {

   motor.tambahKecepatan();

   if (i == 20) {

       motor.printStatus();

   }
```

```
}
```

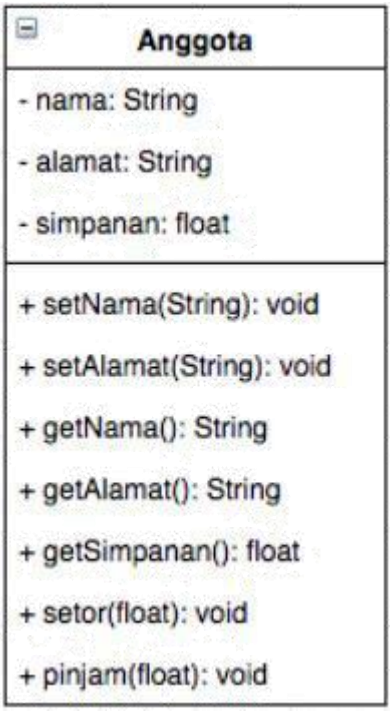
Output di halaman selanjutnya

```
Kontak Off
Kecepatan 0

Kecepatan tidak bisa bertambah karena Mesin
Kontak On
Kecepatan 0

Kontak On
Kecepatan 100

Kontak Off
Kecepatan 0


Process finished with exit code 0
```

3.4 Percobaan 3 - Getter dan Setter

Suppose in a cooperative information system, there is a Member class. Members have name, address and deposit attributes, and setter, getter and deposit and borrow methods. All attributes on members should not be changed arbitrarily, but can only be changed through the setter, getter, deposit and withdraw methods. Especially for the deposit attribute, there is no setter because deposits will increase when making deposit transactions and will decrease when borrowing/withdrawing.

1. The following UML class makes a student class in the program:



1. Same as experiment 1 to create a new project

    1. Open Netbeans, create a **GetterSetter Cooperative** project.
    2. Create a **Member** class. Right-click on the **gettersetter cooperative** package. – New – Java Class
    3. Type the Member class code below

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

setter, getter nama dan alamat

getter simpanan

```
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;

}

}
```

If you look at the Member class, the name and address attributes have 1 getter and setter respectively. While the save attribute only has getSimpanan() only, because like the initial destination, the deposit attribute will change its value if you make a deposit() and borrow/withdraw() transaction.

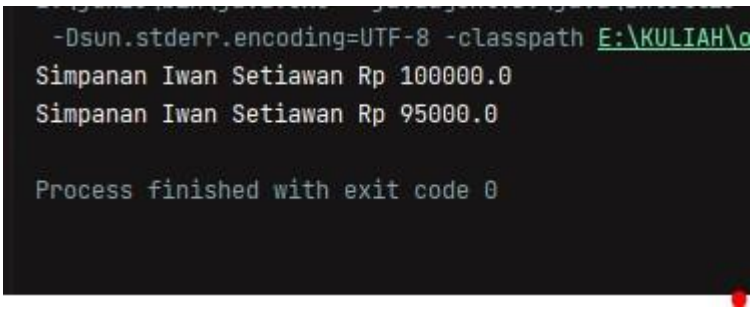
1. Next, create a Demo Cooperative class to try out the Member class.

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

```
public class Main {
    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(5000);


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


                anggota1.getSimpanan());


    }

}
```

1. The result of the main method in step three is:

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



It can be seen in the results of the experiment above, to change deposits is not done directly by changing the attributes of deposits, but through the deposit () and borrow () methods. To display the name must also go through the getName() method, and to display the save via getSave().

3.5 Percobaan 4 - Konstruktur, Instansiasi

1. The first step of experiment 4 is to change the DemoCooperative class as follows

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

```
public class Main {

   public static void main(String[] args) { Anggota anggota1 = new Anggota();



                    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(5000);



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



   }



}
```

1. The results of the program are as follows

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

```
 -Dsun.stderr.encoding=UTF-8 -classpath E:
Simpanan null : Rp 0.0
Simpanan Iwan Setiawan Rp 100000.0
Simpanan Iwan Setiawan Rp 95000.0

Process finished with exit code 0
```

It can be seen the results of running the program, when calling the getName() method the result occurs because the name attribute has not been set to the default value. This can be handled by creating a constructor.

1. Change the Member class to as follows

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

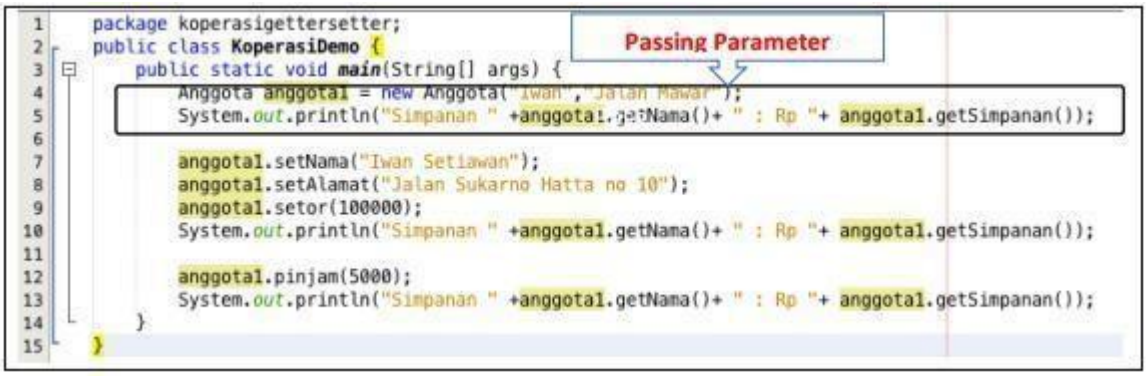
```
public class Anggota {private String nama;



   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 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;

    }

}
```

In the Member class, a constructor is created with a default access modifier that has 2 name and address parameters. And in the constructor it is confirmed that the value of deposits for the first time is Rp. 0

1. Next, change the DemoCooperative class as follows:




```
public class Main {
    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(5000);



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


    }


}
```

1. Hasil dari program tersebut adalah sebagai berikut

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

```
Simpanan Iwan : Rp 0.0
Simpanan Iwan Setiawan Rp 100000.0
Simpanan Iwan Setiawan Rp 95000.0

Process finished with exit code 0
```

After adding the constructor to the Anggoata class, the name and address attributes must automatically be set first by passing parameters if instantiating the Member class. This is usually done for attributes that require specific values. If it does not require a specific value in the constructor there is no need for parameters. For example, deposits for new members are set to 0, so deposits do not need to be used as parameters in the constructor.

3.6 Pertanyaan – Percobaan 3 dan 4

1. What is a getter and Setter?

Getter is an action when we take a value from a variable / object while setter is an action when we enter a value / value into a variable / object.

2. What is the use of the method

getSaves()? Answer:

getSave() is used to enter the save value (setting)

3. What method is used to increase the balance?

deposit()

4. What does yand mean Constructor?

A method that will provide an initial value at the time an object is created.

5. Name the rules for creating constructors?

The constructor name must be the same as the class name, in a class there is only one constructor, the constructor must be public.

6.Can constructors be private? Answer:

Not allowed, because the constructor cannot be accessed from outside the class if it is made private.

7.When to use parameters with parameter passsing?

Passing parameters are used after adding a constructor to add specific values and when a method requires specific values.

8.What is the difference between class attributes and

instantiation attributes:

Class attributes are attributes that are in class attributes, while instantiation attributes are attributes owned by an object in instantiating it.

9.What is the difference between a class method and an

instantiation method?

Class method is a method that is in a class and has not been done while instantiation method is the process of calling a method after instantiating the object.

6. Conclusion

From the experiment above, the concept of encapsulation, constructor, access modifier has been studied which consists of 4 types, namely public, protected, default and private. The concept of attributes or class methods contained in code class blocks and the concept of instantiation of attributes or methods. How to use getters and setters along with the functions of getters and setters. And also have learned or understood UML notation.

4. Tugas

1. Cobalah program dibawah ini dan tuliskan hasil outputnya

```java
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;
      }

   }
}
```

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

jawab

EncapDemo.java

```java
public class EncapDemo {
```

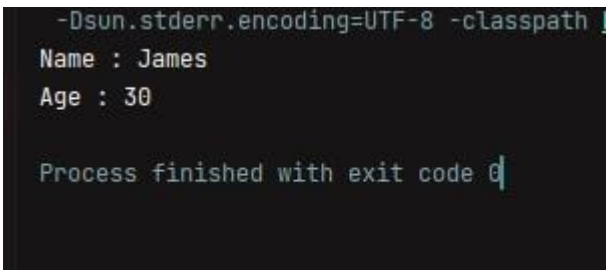
```
    private String name;


    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;


        }
```

```
    }
}
```

EncapTest.java

```
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());

    }

}
```

Output



1. In the program above, in the EncapTest class we set age with a value of 35, but when displayed to the screen the value is 30. Explain!

This is because there is an if else at the time of setAge which has an argument if newAge (input age) > 30 then the age value will be 30. Because the input is 35, the age used by the system is 30, the same as that condition.

1. Change the program above so that the age attribute can be given a maximum value of 30 and a minimum of 18.

EncapDemo.java

```
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) {

            System.out.println("Umur melebihi batas maksimal. Batas maksimal Tahun");
```

```
        } else if (newAge < 18) {


             System.out.println("Umur kurang dari batas minimal. Batas minimal Tahun");


        } else {


            age = newAge;


        }


    }

}
```

EncapTest.java

```
public class EncapTest {



   public static void main(String[] args) { EncapDemo encap = new EncapDemo();
EncapDemo encap2 = new EncapDemo(); EncapDemo encap3 = new EncapDemo();


        encap.setName("James");


        encap.setAge(17);


        System.out.println("Name : " + encap.getName());


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


        encap2.setName("Dhayu");


        encap2.setAge(33);


        System.out.println("Name : " + encap2.getName());


        System.out.println("Age : " + encap2.getAge());
```

```
        encap3.setName("uzi");


        encap3.setAge(27);



        System.out.println("Name : " + encap3.getName());


        System.out.println("Age : " + encap3.getAge());


    }

}
```

Output

```
Umur kurang dari batas minimal. Batas minimal Tahun
Name : James
Age : 0
Umur melebihi batas maksimal. Batas maksimal Tahun
Name : Dhayu
Age : 0
Name : Wildan
Age : 27

Process finished with exit code 0
```

1. In a savings and loan cooperative information system, there is a member class that has attributes including ID card number, name, loan limit, and loan amount. Members can borrow money with a specified lending limit. Members can also repay loans. When the Member installs the loan, the loan amount will be reduced according to the nominal installment. Create the Member class, pass attributes, methods and constructors as needed. Test with the following Cooperative Test to check if the Member class you created is as expected

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

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

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

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

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

        System.out.println("\nMembayar angsuran 3.000.000");
        donny.angsur(3000000);
        System.out.println("Jumlah pinjaman saat ini: " + donny.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
```

Anggota.java

```
public class Anggota {


   private String noKTP, nama;


   private int limitPinjaman, jumlahPinjaman;
```

```
Anggota(String noKTP, String nama, int limitPinjaman) { this.noKTP = noKTP;

    this.nama = nama;

    this.limitPinjaman = limitPinjaman;

}

public String getNama() {

    return nama;

}

public int getLimitPinjaman() {
```

```
        return limitPinjaman;

    }

    public void pinjam(int pinjaman) {

        if (jumlahPinjaman + pinjaman < limitPinjaman) { jumlahPinjaman += pinjaman;


        } else {

            System.out.println("Maaf, jumlah pinjaman melebihi limit");

        }

    }

    public int getJumlahPinjaman() {

        return jumlahPinjaman;

    }

    public void angsur(int angsuran) {

        jumlahPinjaman -= angsuran;

    }

}
```

Testkoperasi.java

```
public class TestKoperasi {


    public static void main(String[] args) {

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

        System.out.println("Nama Anggota : " + donny.getNama());
```

```
        System.out.println("Limit Pinjaman : " + donny.getLimitPinjaman());
System.out.println("\nMeminjam uang 10000000"); donny.pinjam(10000000);



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



        System.out.println("\nMeminjam uang 4000000");


        donny.pinjam(4000000);


        System.out.println("Jumlah Pinjaman saat ini : " +


                donny.getJumlahPinjaman());



        System.out.println("Membayar angsuran 10000000");


        donny.angsur(1000000);


        System.out.println("Jumlah Pinjaman saat ini : " +


                donny.getJumlahPinjaman());


        System.out.println("Membayar angsuran 3000000");


        donny.angsur(3000000);



        System.out.println("Jumlah Pinjaman saat ini : " +


                donny.getJumlahPinjaman());


    }


}
```

Output

```
  -Dsun.stderr.encoding=UTF-8 -classpath E:\KULIAH\oop\koperasigettersetter\c
Nama Anggota : uzi
Limit Pinjaman : 5000000

Meminjam uang 10000000
Maaf, jumlah pinjaman melebihi limit
Jumlah Pinjaman saat ini : 0

Meminjam uang 4000000
Jumlah Pinjaman saat ini : 4000000
Membayar angsuran 10000000
Jumlah Pinjaman saat ini : 3000000
Membayar angsuran 3000000
Jumlah Pinjaman saat ini : 0

Process finished with exit code 0
```

1. Modify question no. 4 so that the nominal that can be paid in installments is at least 10% of the current loan amount. If the installments are less than that, then a warning appears "Sorry, installments must be 10% of the loan amount".

```
public class Anggota {



    private String noKTP, nama;

    private int limitPinjaman, jumlahPinjaman;



    Anggota(String noKTP, String nama, int limitPinjaman) { this.noKTP = noKTP;

        this.nama = nama;

        this.limitPinjaman = limitPinjaman;

    }

    public String getNama() {
```

```
        return nama;
```

```
}

public int getLimitPinjaman() {

    return limitPinjaman;

}

public void pinjam(int pinjaman) {



    if (jumlahPinjaman + pinjaman < limitPinjaman) { jumlahPinjaman += pinjaman;

    } else {

        System.out.println("Maaf, jumlah pinjaman melebihi limit");

    }

}

public int getJumlahPinjaman() {

    return jumlahPinjaman;

}

public void angsur(int angsuran) {

     int minAngsuran = jumlahPinjaman * 10 / 100; if (angsuran > minAngsuran) {

        jumlahPinjaman -= angsuran;
```

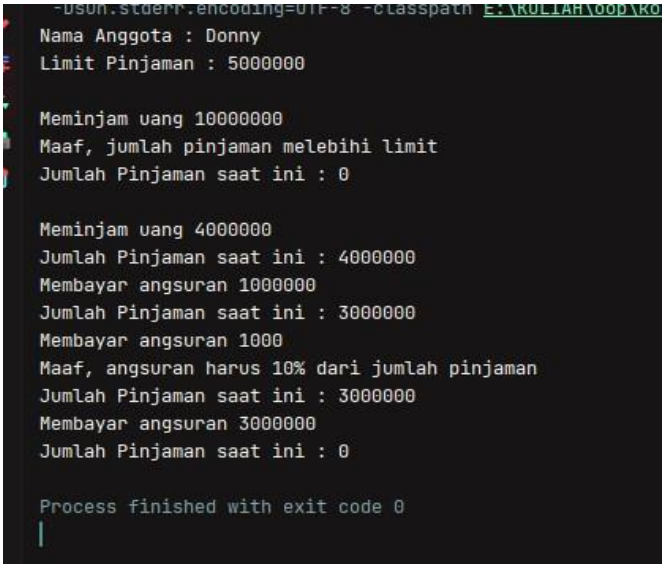
```
    } else {

        System.out.println("Maaf, angsuran harus 10% dari jumlah pinjaman");



    }

   }

}
```

Testkoperasi.java

```
public class TestKoperasi {

   public static void main(String[] args) {

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

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

                   System.out.println("Limit Pinjaman : " + donny.getLimitPinjaman());
System.out.println("\nMeminjam uang 10000000"); donny.pinjam(10000000);

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

       System.out.println("\nMeminjam uang 4000000");

       donny.pinjam(4000000);

       System.out.println("Jumlah Pinjaman saat ini : " +

               donny.getJumlahPinjaman());

       System.out.println("Membayar angsuran 1000000");

       donny.angsur(1000000);

       System.out.println("Jumlah Pinjaman saat ini : " +
```

```
                donny.getJumlahPinjaman());

        System.out.println("Membayar angsuran 1000");

        donny.angsur(1000);

        System.out.println("Jumlah Pinjaman saat ini : " +

                donny.getJumlahPinjaman());

        System.out.println("Membayar angsuran 3000000");

        donny.angsur(3000000);

        System.out.println("Jumlah Pinjaman saat ini : " +

                donny.getJumlahPinjaman());

    }

}
```

Output



1. Modify the TestCooperative class, so that the loan amount and installments can receive input from the console.

Anggota.java

```
public class Anggota {

    private String noKTP, nama;

    private int limitPinjaman, jumlahPinjaman;

    Anggota(String noKTP, String nama, int limitPinjaman) { this.noKTP = noKTP;

        this.nama = nama;

        this.limitPinjaman = limitPinjaman;

    }

    public String getNama() {

        return nama;

    }

    public int getLimitPinjaman() {

        return limitPinjaman;

    }

    public void pinjam(int pinjaman) {
```

```
    if (jumlahPinjaman + pinjaman <= limitPinjaman) { jumlahPinjaman += pinjaman;

    } else {

        System.out.println("Maaf, jumlah pinjaman melebihi limit");

    }

}

public int getJumlahPinjaman() {

    return jumlahPinjaman;

}

public void angsur(int angsuran) {

    int minAngsuran = jumlahPinjaman * 10 / 100; if (angsuran > minAngsuran) {

        jumlahPinjaman -= angsuran;

    } else {

        System.out.println("Maaf, angsuran harus 10% dari jumlah pinjaman");

    }

}
```

```
}
```

Testkoperasi.java

```
import java.util.Scanner;;

public class TestKoperasi {
```

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

      Anggota donny = new Anggota("111333444", "Donny", 5000000); Scanner sc = new
Scanner(System.in);

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

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

      System.out.print("\nInput Pinjaman : "); donny.pinjam(sc.nextInt());

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

      System.out.print("\nInput Pinjaman : ");

      donny.pinjam(sc.nextInt());

      System.out.println("Jumlah Pinjaman saat ini : " +

              donny.getJumlahPinjaman());

      System.out.print("Input Angsuran : ");

      donny.angsur(sc.nextInt());
```

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

        System.out.print("Input Angsuran : ");

        donny.angsur(sc.nextInt());

        System.out.println("Jumlah Pinjaman saat ini : " +

                donny.getJumlahPinjaman());

        System.out.print("Input Angsuran : ");

        donny.angsur(sc.nextInt());

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

    }

}
```

Output

```
Nama Anggota : Donny
Limit Pinjaman : 5000000

Input Pinjaman : 7000000
Maaf, jumlah pinjaman melebihi limit
Jumlah Pinjaman saat ini : 0

Input Pinjaman : 5000000
Jumlah Pinjaman saat ini : 5000000
Input Angsuran : 300000
Maaf, angsuran harus 10% dari jumlah pinjaman
Jumlah Pinjaman saat ini : 5000000
Input Angsuran : 2000
Maaf, angsuran harus 10% dari jumlah pinjaman
Jumlah Pinjaman saat ini : 5000000
Input Angsuran : 2000000
Jumlah Pinjaman saat ini : 3000000

Process finished with exit code 0
```