

Object Oriented Programming Encapsulation Week 3



From:

AL AZHAR RIZQI RIFA'I FIRDAUS

Class:

2 I

Absence:

01

Student Number Identity:

2241720263

Department:

Information Technology

Study Program:

Informatics Engineering

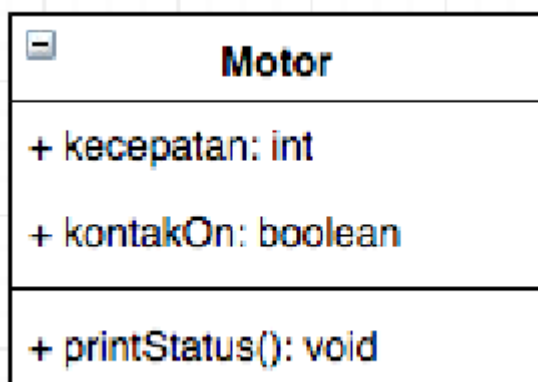
Experiment 1 - Encapsulation

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

1. Open Netbeans, create the MotorEncapsulation project.
2. Create the Motor class. Right-click on the package motorencapsulation - New - Java Class.
3. Type the Motor class code below.

```
Motorcycle.java ×
src > Exp1 > Motorcycle.java > Motorcycle > printStatus()
1  package Exp1;
2
3  public class Motorcycle {
4      public int speed = 0;
5      public boolean contactOn = false;
6
7      Codeium: Refactor | Explain | Generate Javadoc
8      public void printStatus() {
9          if (contactOn == true) {
10             System.out.println("Contact on");
11         } else {
12             System.out.println("Contact off");
13         }
14         System.out.println("Speed: " + speed);
15     }
16 }
```

The UML form of the Motor class diagram class is as follows:



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

```
Motorcycle.java  MotorcycleDemo.java x
src > Exp1 > MotorcycleDemo.java > MotorcycleDemo > main(String[])
1  package Exp1;
2
3  public class MotorcycleDemo {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
4      public static void main(String[] args) {
5          Motorcycle motorcycle1 = new Motorcycle();
6          motorcycle1.printStatus();
7          motorcycle1.speed = 50;
8          motorcycle1.printStatus();
9      }
10 }
11
```

5. The result is as follows:

```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
InExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/c
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Contact off
Speed: 0
Contact off
Speed: 50
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```

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

Namely, the motor speed suddenly changed from 0 to 50. Even more oddly, the motor contact position is still in the OFF condition. How is it possible for a motor to go from zero to 50 in an instant? 50, and even then the ignition is OFF?

In this case, access to the motorcycle attributes was not controlled. In fact, objects in the real world always have restrictions and mechanisms on 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 things:

1. Hide the internal attributes (speed, contactOn) from the user (other classes)
2. Provide special methods to access the attributes.

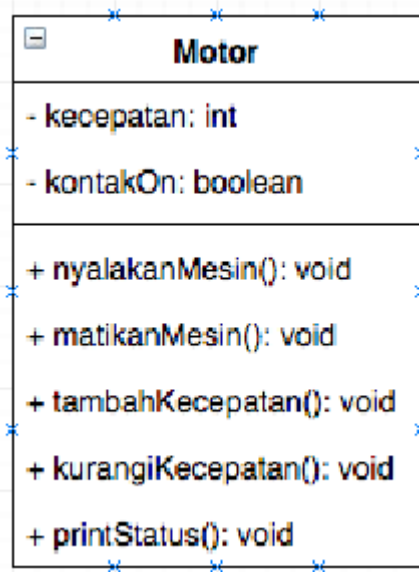
For that, let's continue with the next experiment about Access Modifier

Experiment 2 - Access Modifier

In this experiment, the access modifier will be used to improve the way the Motor class works.

experiment 1.

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



2. Based on the UML class diagram, the Motor class has changes, namely:

- a. Change the access modifiers speed and kontakOn to private
- b. Add methods turn onMachine, turn offMachine, increaseSpeed, decreaseSpeed.

The implementation of the Motor class is as follows:

```
Motorcycle.java x MotorcycleDemo.java
src > Exp1 > Motorcycle.java > Motorcycle > subtractSpeed()

3 public class Motorcycle {
4     private int speed = 0;
5     private boolean contactOn = false;
6
7     Codeium: Refactor | Explain | Generate Javadoc
8     public void engineOn() {
9         contactOn = true;
10    }
11
12    Codeium: Refactor | Explain | Generate Javadoc
13    public void engineOff() {
14        contactOn = false;
15        speed = 0;
16    }
17
18    Codeium: Refactor | Explain | Generate Javadoc
19    public void addSpeed() {
20        if (contactOn == true) {
21            speed += 5;
22        } else {
23            System.out.println("Speed can't increase because the engine is off!");
24        }
25    }
26
27    Codeium: Refactor | Explain | Generate Javadoc
28    public void subtractSpeed() {
29        if (contactOn == true) {
30            speed -= 5;
31        } else {
32            System.out.println("Speed can't decrease because the engine is off!");
33        }
34    }
35
36    Codeium: Refactor | Explain | Generate Javadoc
37    public void printStatus() {
38        if (contactOn == true) {
39            System.out.println("Contact on");
40        } else {
41            System.out.println("Contact off");
42        }
43        System.out.println("Speed: " + speed);
44    }
45 }
```

3. Then in the MotorDemo class, change the code to be as follows:

```
Motorcycle.java  MotorcycleDemo.java ×
src > Exp1 > MotorcycleDemo.java > MotorcycleDemo > main(String[])
1  package Exp1;
2
3  public class MotorcycleDemo {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
4      public static void main(String[] args) {
5          Motorcycle motorcycle1 = new Motorcycle();
6          motorcycle1.printStatus();
7          motorcycle1.addSpeed();
8
9          motorcycle1.engineOn();
10         motorcycle1.printStatus();
11
12         motorcycle1.addSpeed();
13         motorcycle1.printStatus();
14
15         motorcycle1.addSpeed();
16         motorcycle1.printStatus();
17
18         motorcycle1.addSpeed();
19         motorcycle1.printStatus();
20
21         motorcycle1.engineOff();
22         motorcycle1.printStatus();
23     }
24 }
25
```

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

```

→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Contact off
Speed: 0
Speed can't increase because the engine is off!
Contact on
Speed: 0
Contact on
Speed: 5
Contact on
Speed: 10
Contact on
Speed: 15
Contact off
Speed: 0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x

```

From the above experiment, we can observe that now the speed attribute cannot be accessed by the user and changed its value arbitrarily. Even when trying to increase the speed when the contact position is still OFF, a notification will appear that the engine 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.

Questions

1. In the `TestMobil` class, when we increase the speed for the first time, why is the "Speed cannot be increased because the engine is off!"?

- In the `MotorcycleDemo` class, when you increase the speed for the first time with `motorcycle1.addSpeed()`, the warning "Speed cannot be increased because the Engine is Off!" appears because the `contactOn` attribute is initially set to false when you create a new `Motorcycle` object (`motorcycle1`). Since the engine is off (`contactOn` is false), you cannot increase the speed until you turn the engine on with `motorcycle1.engineOn()`.

2. Can the speed and `contactOn` attributes be set private?

- The speed and `contactOn` attributes are set to private to encapsulate the internal state of the `Motorcycle` class. This means that these attributes can only be accessed and modified within the class itself, and not directly from outside code. This encapsulation helps maintain control over the state of the object and ensures that it can be modified only through controlled methods like `engineOn()`, `engineOff()`, `addSpeed()`, and `subtractSpeed()`.

3. Modify the `Motor` class so that the maximum speed is 100!

```

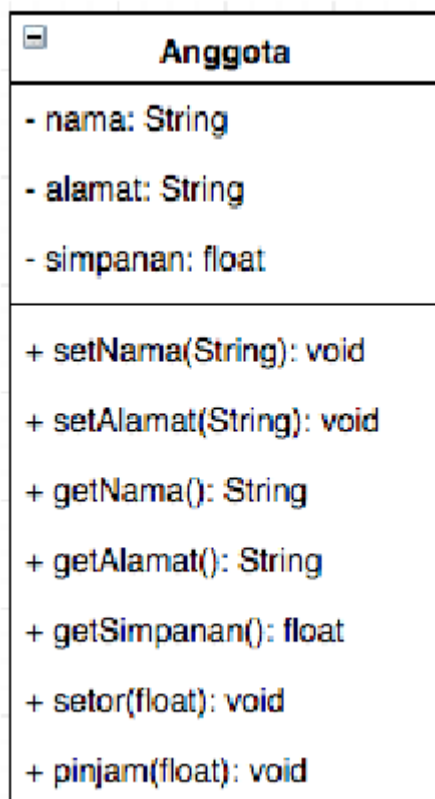
16     public void addSpeed() {
17         if (contactOn == true) {
18             if (speed > 100) {
19                 System.out.println("Speed can't increase!");
20             } else {
21                 speed += 5;
22             }
23         } else {
24             System.out.println("Speed can't increase because the engine is off!");
25         }
26     }

```

Experiment 3 - Getter and Setter

Suppose in a cooperative information system, there is a Member class. Member has attributes name, address and deposit, and methods setter, getter and deposit and borrow. All attributes of members cannot be changed arbitrarily, but can only be changed through the setter method, getter, deposit and withdraw methods. Specifically for the savings attribute, there is no setter because the savings will be increased when making a deposit transaction and will decrease when making a loan / withdrawal.

1. The following UML class makes the Student class in the program:



2. Same as experiment 1 to create a new project

a. Open Netbeans, create project KoperasiGetterSetter.

b. Create the Member class. Right click on the package `koperasigettersetter` - New - Java Class.

c. Type the Member class code below.

Member.java ×

CooperativeDemo.java

src > Exp3 > Member.java > Member > loan(float)

```
1  package Exp3;
2
3  public class Member {
4      private String name;
5      private String address;
6      private float saving;
7
8      Codeium: Refactor | Explain | Generate Javadoc
9      public void setName(String name) {
10         this.name = name;
11     }
12
13     Codeium: Refactor | Explain | Generate Javadoc
14     public void setAddress(String address) {
15         this.address = address;
16     }
17
18     Codeium: Refactor | Explain | Generate Javadoc
19     public String getName() {
20         return name;
21     }
22
23     Codeium: Refactor | Explain | Generate Javadoc
24     public String getAddress() {
25         return address;
26     }
27
28     Codeium: Refactor | Explain | Generate Javadoc
29     public float getSaving() {
30         return saving;
31     }
32
33     Codeium: Refactor | Explain | Generate Javadoc
34     public void deposit(float money) {
35         saving += money;
36     }
37
38     Codeium: Refactor | Explain | Generate Javadoc
39     public void loan(float money) {
40         saving -= money;
41     }
42 }
```

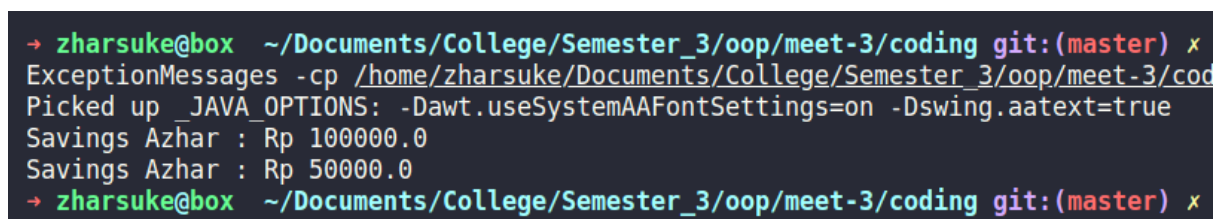
If you look at the Member class, the name and address attributes have one getter and setter each. While the deposit attribute only has getSimpanan() only, because attribute will change its value if it performs a deposit() and borrow/withdraw() transactions.

3. Next, create a CooperativeDemo class to test the Member class.



```
src > Exp3 > CooperativeDemo.java > CooperativeDemo > main(String[])
1 package Exp3;
2
3 public class CooperativeDemo {
4     public static void main(String[] args) {
5         Member member1 = new Member();
6         member1.setName(name:"Azhar");
7         member1.setAddress(address:"Sukarno Hatta Street No. 1");
8         member1.deposit(money:100_000);
9         System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
10
11         member1.loan(money:50_000);
12         System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
13     }
14 }
15
```

4. The result of the main method in the third step is



```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Savings Azhar : Rp 100000.0
Savings Azhar : Rp 50000.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```

It can be seen in the results of the above experiments, to change the savings is not done directly by changing the savings attribute, but through the deposit() and borrow() methods.

directly by changing the deposit attribute, but through the deposit() and borrow() methods.

To display the name must also go through the getName() method, and to display the savings through getStorage().

Experiment 4 - Constructor, Instantiation

1. The first step of experiment 4 is to modify the KoperasiDemo class as follows

```
CooperativeDemo.java x Member.java
src > Exp3 > CooperativeDemo.java > CooperativeDemo > main(String[])
1 package Exp3;
2
3 public class CooperativeDemo {
4     Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
5     public static void main(String[] args) {
6         Member member1 = new Member();
7         System.out.println("Savings : " + member1.getName() + " : Rp " + member1.getSaving());
8         member1.setName(name:"Azhar");
9         member1.setAddress(address:"Sukarno Hatta Street No. 1");
10        member1.deposit(money:100_000);
11        System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
12
13        member1.loan(money:50_000);
14        System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
15    }
16 }
```

2. The result of the program is as follows

```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Savings : null : Rp 0.0
Savings Azhar : Rp 100000.0
Savings Azhar : Rp 50000.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```

It can be seen from the results of the running program, when the getName() method is called.

The result occurs because the name attribute has not been set to its default value. This can be handled by creating a constructor.

3. Change the Member class to be as follows

```
CooperativeDemo.java Member.java X
src > Exp3 > Member.java > Member > Member(String, String)
1 package Exp3;
2
3 public class Member {
4     private String name;
5     private String address;
6     private float saving;
7
8     Member (String name, String address) {
9         this.name = name;
10        this.address = address;
11        this.saving = 0;
12    }
13
```

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

4. Next change the KoperasiDemo class as follows

```
CooperativeDemo.java X Member.java
src > Exp3 > CooperativeDemo.java > CooperativeDemo > main(String[])
1 package Exp3;
2
3 public class CooperativeDemo {
4     public static void main(String[] args) {
5         Member member1 = new Member(name:"Azhar", address:"Mawar Street");
6         System.out.println("Savings : " + member1.getName() + " : Rp " + member1.getSaving());
7         member1.setName(name:"AL Azhar");
8         member1.setAddress(address:"Sukarno Hatta Street No. 1");
9         member1.deposit(money:100_000);
10        System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
11
12        member1.loan(money:50_000);
13        System.out.println("Savings " + member1.getName() + " : Rp " + member1.getSaving());
14    }
15 }
16
```

5. The result of the program is as follows

```

→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Savings : Azhar : Rp 0.0
Savings AL Azhar : Rp 100000.0
Savings AL Azhar : Rp 50000.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x

```

After adding a constructor to the Member class, the name and address attributes must automatically be set first by passing parameters when instantiating the Member class. instantiation of 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, if the deposit for a new member is set to 0, then the deposit does not need to be parameterized in the constructor.

Questions - Experiments 3 and 4

1. What are getters and setters?

- **A getter is a method used to retrieve (read) the value of an attribute or property within an object. And a setter is a method used to set (write/modify) the value of an attribute or property within an object.**

2. What is the use of getSaving() method?

- **The getSaving() method is used to retrieve the current savings balance of a Member object.**

3. What method is used to increase the balance?

- **The deposit(float money) method is used to increase the savings balance of a Member object by a specified amount of money.**

4. What is a constructor?

- **A constructor is a special method in a class that is automatically called when an object of that class is created. It is used to initialize the object's attributes and perform any necessary setup.**

5. Name the rules for creating a constructor?

- **The constructor must have the same name as the class.**
- **It cannot have a return type, not even void.**

6. Can a constructor be of private type?

- **Yes, a constructor can be of private type. A private constructor is typically used to prevent the instantiation of a class from outside the class itself. It's commonly used in Singleton design patterns.**

7. When to use parameters with parameter passing?

- **Parameters are used with parameter passing when you need to initialize an object's attributes with specific values during its creation. They allow you to customize the object's state at the time of instantiation.**

8. What is the difference between class attributes and attribute instantiation?

- **Class attributes (fields) are variables that belong to the class and are shared among all instances (objects) of that class. Attribute instantiation refers to assigning initial values to these attributes within the class.**

9. What is the difference between class method and method instantiation?

- **Class methods are functions that belong to the class itself and are shared among all instances of the class. Method instantiation refers to defining and implementing these methods within the class. Class methods can operate on class attributes and instance attributes, but they are not tied to a specific instance of the class. Instance methods, on the other hand, are methods that operate on instance-specific data and are called on a particular object of the class.**

Conclusion

From the above experiments, we have learned the concept of encapsulation, constructor, access modifier which consists of 4 types namely public, protected, default and private. The concept of class attributes or methods in the block code class and the concept of instantiation of attributes or methods. How to use getter and setter and the function of getter and setter. And also have learned or understand UML notation.

Assignment

1. Try the program below and write down the output results.

Code :

EncapDemo.java × EncapMain.java

src > AssNum1 > EncapDemo.java > EncapDemo > setAge(i

```
1  package AssNum1;
2
3  public class EncapDemo {
4      private String name;
5      private int age;
6
7      Codeium: Refactor | Explain | Generate Javadoc
8      public String getName() {
9          return name;
10     }
11
12     Codeium: Refactor | Explain | Generate Javadoc
13     public void setName(String newName) {
14         this.name = newName;
15     }
16
17     Codeium: Refactor | Explain | Generate Javadoc
18     public int getAge() {
19         return age;
20     }
21
22     Codeium: Refactor | Explain | Generate Javadoc
23     public void setAge(int newAge) {
24         if (newAge > 30) {
25             age = 30;
26         } else {
27             this.age = newAge;
28         }
29     }
30 }
```

```
EncapDemo.java EncapMain.java x
src > AssNum1 > EncapMain.java > EncapMain > main(String[])
1 package AssNum1;
2
3 public class EncapMain {
4     public static void main(String[] args) {
5         EncapDemo encap = new EncapDemo();
6         encap.setName(newName: "Azhar");
7         encap.setAge(newAge: 35);
8
9         System.out.println("Name : " + encap.getName());
10        System.out.println("Age : " + encap.getAge());
11    }
12 }
13
```

Result :

```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
InExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/c
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Name : Azhar
Age : 30
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```

2. In the program above, in the EncapTest class we set the age with a value of 35, but when it is displayed on the screen it is 30, explain why.

Answer: **Because there is a condition that if newAge is bigger than 30, it'll be set to 30.**

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

Code:

```
19     public void setAge(int newAge) {
20         if (newAge > 30) {
21             age = 30;
22         } else if (newAge < 18) {
23             age = 18;
24         } else {
25             this.age = newAge;
26         }
27     }
```



```
EncapDemo.java  EncapMain.java X
src > AssNum1 > EncapMain.java > EncapMain > main(String[])
1  package AssNum1;
2
3  public class EncapMain {
4      Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
5      public static void main(String[] args) {
6          EncapDemo encap1 = new EncapDemo();
7          encap1.setName(newName: "Azhar");
8          encap1.setAge(newAge: 35);
9          EncapDemo encap2 = new EncapDemo();
10         encap2.setName(newName: "Rizqi");
11         encap2.setAge(newAge: 15);
12
13         System.out.println("Name : " + encap1.getName());
14         System.out.println("Age : " + encap1.getAge());
15         System.out.println("Name : " + encap2.getName());
16         System.out.println("Age : " + encap2.getAge());
17     }
18 }
```

Result:

```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Name : Azhar
Age : 30
Name : Rizqi
Age : 18
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```

4. In a savings and loan cooperative information system, there is a Member class that has attributes such as ID number, name, loan limit, and loan amount. Member can borrow money with a specified loan limit. Members can also install the loan. When the member installs the loan, the loan amount will be reduced according to the installment amount. amount will be reduced according to the installment amount. Create the Member class class, provide attributes, methods and constructors as needed. Test with TestKoperasi below to check whether the Member class you created is as expected.

Main:

```

Member.java 1 MemberMain.java x
src > AssNum4 > MemberMain.java > MemberMain > main(String[])
1 package AssNum4;
2
3 public class MemberMain {
4     public static void main(String[] args) {
5         Member donny = new Member(numberIdentity:111333444, name:"Donny", limitLoan:5_000_000);
6
7         System.out.println("Member Name : " + donny.getName());
8         System.out.println("Limit Loan : " + donny.getLimitLoan());
9
10        System.out.println("\nLoan Money for 10.000.000");
11        donny.loan(money:10_000_000);
12        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
13
14        System.out.println("\nLoan Money for 4.000.000");
15        donny.loan(money:4_000_000);
16        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
17
18        System.out.println("\npay the installment 1.000.000");
19        donny.installment(money:1_000_000);
20        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
21
22        System.out.println("\npay the installment 3.000.000");
23        donny.installment(money:3_000_000);
24        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
25    }
26 }
27

```

Expected results:

```

→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Member Name : Donny
Limit Loan : 5000000.0

Loan Money for 10.000.000
Sorry, amount loan is more than limit loan
Amount of loan currently : 0.0

Loan Money for 4.000.000
Amount of loan currently : 4000000.0

pay the installment 1.000.000
Amount of loan currently : 3000000.0

pay the installment 3.000.000
Amount of loan currently : 0.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x

```

Member:

Member.java 1 ×

MemberMain.java

src > AssNum4 > Member.java > Member > Member(int, String, double)

```
1  package AssNum4;
2
3  public class Member {
4      private int numberIdentity;
5      private String name;
6      private double limitLoan;
7      private double amountLoan;
8
9      Member(int numberIdentity, String name, double limitLoan){
10         this.numberIdentity = numberIdentity;
11         this.name = name;
12         this.limitLoan = limitLoan;
13         this.amountLoan = 0;
14     }
15
16     Codeium: Refactor | Explain | Generate Javadoc
17     public String getName() {
18         return name;
19     }
20
21     Codeium: Refactor | Explain | Generate Javadoc
22     public double getLimitLoan() {
23         return limitLoan;
24     }
25
26     Codeium: Refactor | Explain | Generate Javadoc
27     public double getAmountLoan() {
28         return amountLoan;
29     }
30
31     Codeium: Refactor | Explain | Generate Javadoc
32     public void installment(double money) {
33         if (amountLoan < 0) {
34             System.out.println("You don't have loan! great!");
35         } else {
36             amountLoan -= money;
37         }
38     }
39
40     Codeium: Refactor | Explain | Generate Javadoc
41     public void loan(double money) {
42         if ((money + amountLoan) > limitLoan) {
```

```

38         System.out.println("Sorry, amount loan is more than limit loan");
39     } else {
40         amountLoan += money;
41     }
42 }
43 }
44

```

5. Modify question no. 4 so that the minimum installment amount is 10% of the current loan amount. If the installment is less than that, the warning "Sorry, installments must be 10% of the loan amount".

Member:

```

28     public double minInstallment() {
29         return amountLoan * 0.1;
30     }
31
32     Codeium: Refactor | Explain | Generate Javadoc
33     public void installment(double money) {
34         if (amountLoan < 0) {
35             System.out.println("You don't have loan! great!");
36         } else {
37             if (money > minInstallment()) {
38                 amountLoan -= money;
39             } else {
40                 System.out.println("Sorry, the installment must be more than 10% of the loan");
41             }
42         }
43     }

```

Main:

```

Member.java 1  MemberMain.java x
src > AssNum4 > MemberMain.java > MemberMain > main(String[])
1  package AssNum4;
2
3  public class MemberMain {
4      Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
5      public static void main(String[] args) {
6          Member donny = new Member(numberIdentity:111333444, name:"Donny", limitLoan:5_000_000);
7
8          System.out.println("Member Name : " + donny.getName());
9          System.out.println("Limit Loan : " + donny.getLimitLoan());
10
11         System.out.println("\nLoan Money for 10.000.000");
12         donny.loan(money:10_000_000);
13         System.out.println("Amount of loan currently : " + donny.getAmountLoan());
14
15         System.out.println("\nLoan Money for 4.000.000");
16         donny.loan(money:4_000_000);
17         System.out.println("Amount of loan currently : " + donny.getAmountLoan());
18
19         System.out.println("\npay the installment 1.000.000");
20         donny.installment(money:1_000_000);
21         System.out.println("Amount of loan currently : " + donny.getAmountLoan());
22
23         System.out.println("\npay the installment 3.000.000");
24         donny.installment(money:1_000);
25         System.out.println("Amount of loan currently : " + donny.getAmountLoan());
26     }
27

```

Result:

```

→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Member Name : Donny
Limit Loan : 5000000.0

Loan Money for 10.000.000
Sorry, amount loan is more than limit loan
Amount of loan currently : 0.0

Loan Money for 4.000.000
Amount of loan currently : 4000000.0

pay the installment 1.000.000
Amount of loan currently : 3000000.0

pay the installment 3.000.000
Sorry, the installment must be more than 10% of the loan
Amount of loan currently : 3000000.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x

```

6. Modify the TestKoperasi class, so that the loan amount and installment can receive input from the console.

Main:

```
Member.java 1 MemberMain.java X
src > AssNum4 > MemberMain.java > MemberMain > main(String[])
1 package AssNum4;
2
3 import java.util.Scanner;
4
5 public class MemberMain {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         Member donny = new Member(numberIdentity:111333444, name:"Donny", limitLoan:5_000_000);
9
10        System.out.println("Member Name : " + donny.getName());
11        System.out.println("Limit Loan : " + donny.getLimitLoan());
12
13        System.out.print("\nInsert amount of loan : ");
14        double loan = scanner.nextDouble();
15        donny.loan(loan);
16        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
17
18        System.out.print("\nInsert amount of loan : ");
19        loan = scanner.nextDouble();
20        donny.loan(loan);
21        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
22
23        System.out.print("\nInsert amount of installment : ");
24        double installment = scanner.nextDouble();
25        donny.installment(installment);
26        System.out.print("Amount of loan currently : " + donny.getAmountLoan());
27
28        System.out.print("\nInsert amount of installment : ");
29        installment = scanner.nextDouble();
30        donny.installment(installment);
31        System.out.println("Amount of loan currently : " + donny.getAmountLoan());
32
33        scanner.close();
34    }
35 }
36
```

Result:

```
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-3/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Member Name : Donny
Limit Loan : 5000000.0

Insert amount of loan : 4000000
Amount of loan currently : 4000000.0

Insert amount of loan : 0
Amount of loan currently : 4000000.0

Insert amount of installment : 2500000
Amount of loan currently : 1500000.0
Insert amount of installment : 0
Sorry, the installment must be more than 10% of the loan
Amount of loan currently : 1500000.0
→ zharsuke@box ~/Documents/College/Semester_3/oop/meet-3/coding git:(master) x
```