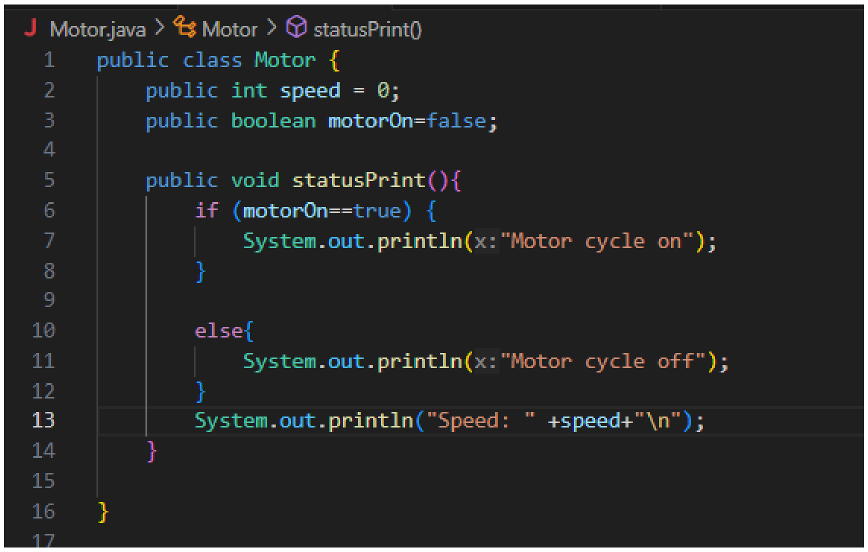
ENCAPSULATION

Erwan Majid/08/2i

Link github: https://github.com/Majid5654/Semester-3/tree/Main/JAVA%20OOP/Week3

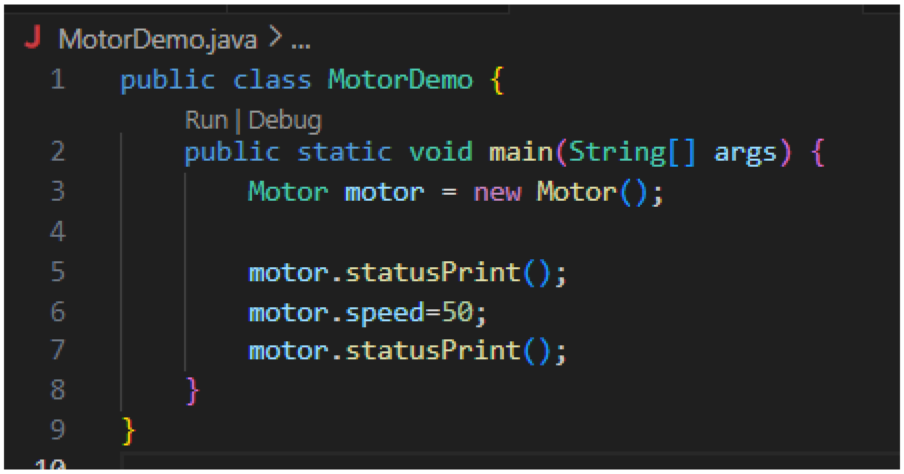
- **3.1 Experiment 1 – Encapsulation**

Class Motor:


Motor.java > Motor > statusPrint()

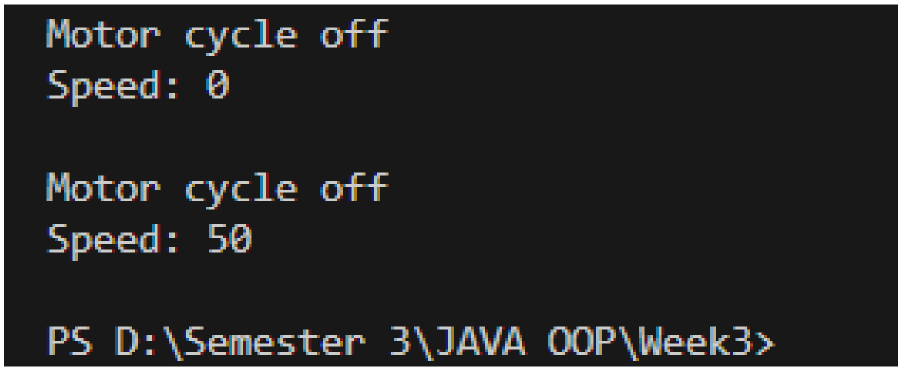
```java
public class Motor {
    public int speed = 0;
    public boolean motorOn=false;

    public void statusPrint(){
        if (motorOn==true) {
            System.out.println(x:"Motor cycle on");
        }

        else{
            System.out.println(x:"Motor cycle off");
        }
        System.out.println("Speed: " +speed+"\n");
    }

}
```

Class Motordemo:


MotorDemo.java > ...

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

        motor.statusPrint();
        motor.speed=50;
        motor.statusPrint();
    }
}
```

Result:



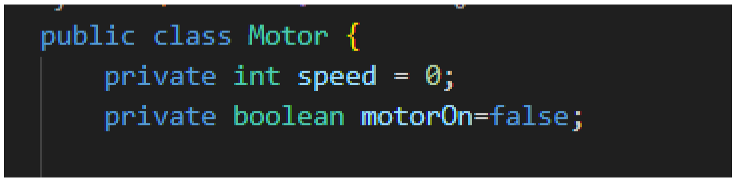
```
Motor cycle off
Speed: 0

Motor cycle off
Speed: 50

PS D:\Semester 3\JAVA OOP\Week3>
```

From experiment 1 - encapsulation, in your opinion, is there anything strange?

That is, the speed of the motor suddenly changes from 0 to 50. Even more awkward, the motor contact position is still in the OFF condition. How is it possible for a motor to be blinked from zero to 50, and even then the ignition is OFF?

Now in this case, access to motor attributes is apparently not controlled. In fact, objects in the real world always have limits and mechanisms for how these objects can be used. Then, how can we improve the Motor class above so that it can be used properly? We can consider the following:

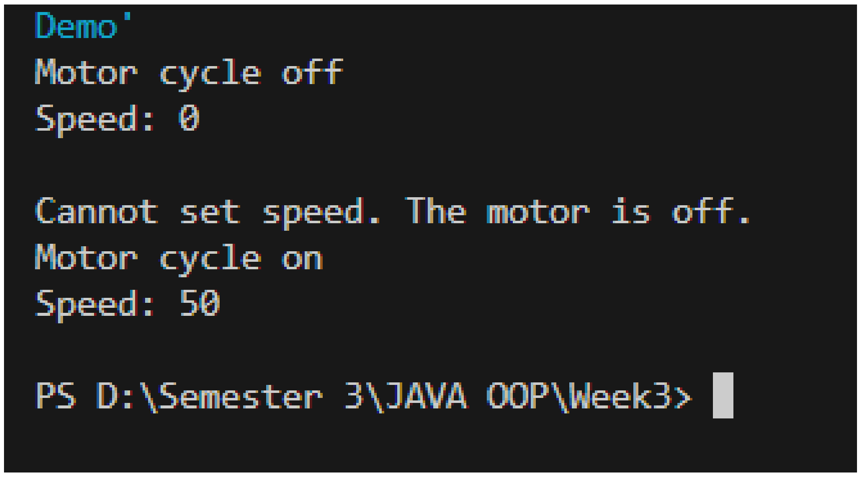
1. Hiding internal attributes (speed, motorOn) from users (other classes)



```java
public class Motor {
    private int speed = 0;
    private boolean motorOn=false;
```

2. Provides a special method for accessing attributes.

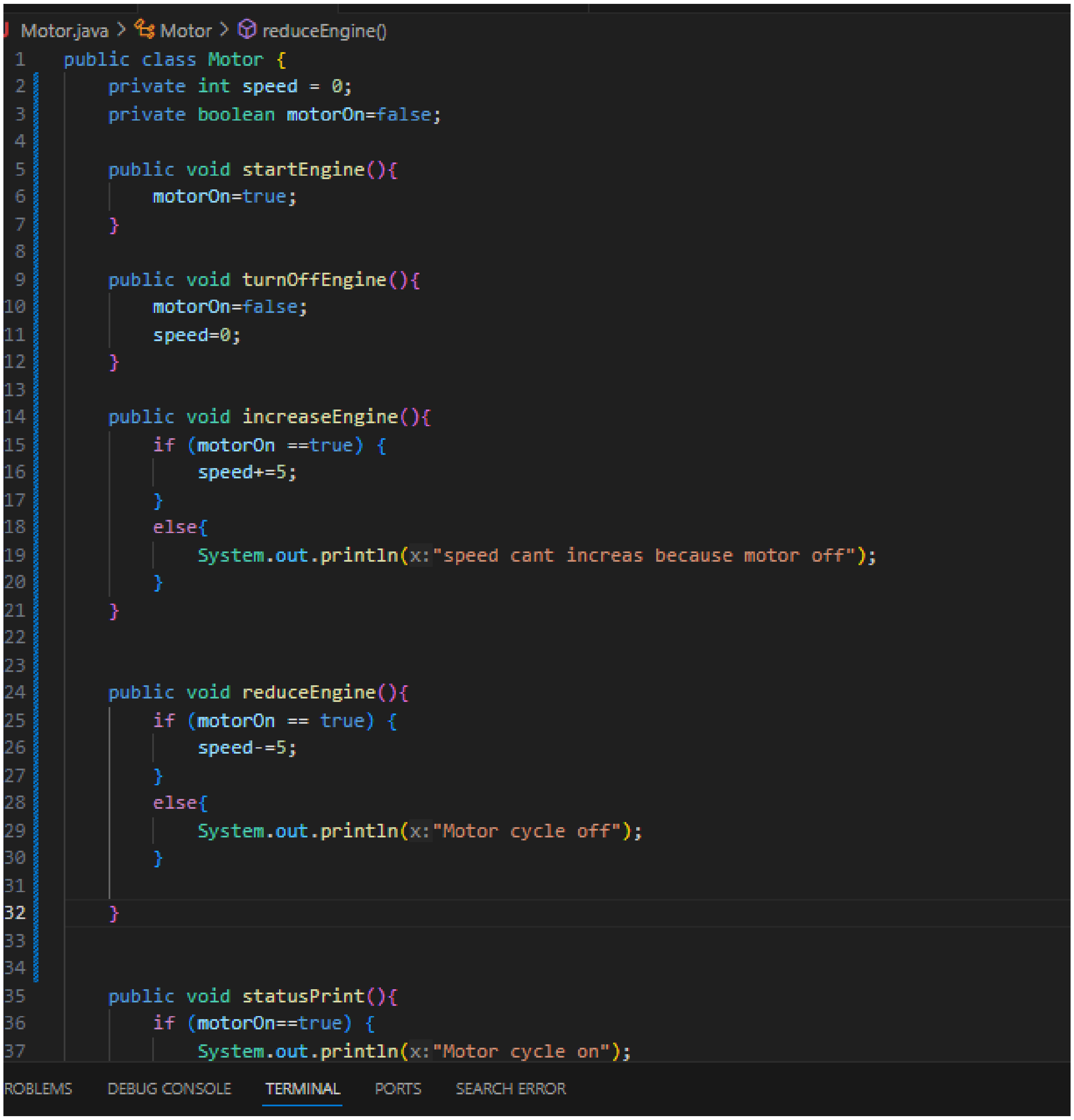
For that, let's continue the next experiment about Access Modifier.



```
Demo'
Motor cycle off
Speed: 0

Cannot set speed. The motor is off.
Motor cycle on
Speed: 50

PS D:\Semester 3\JAVA OOP\Week3>
```

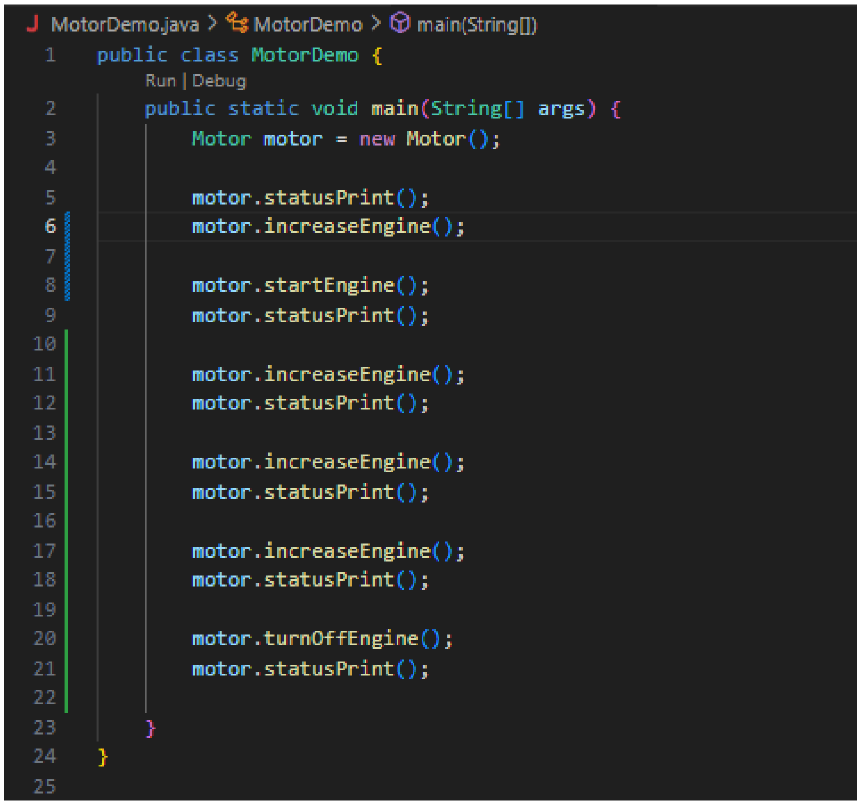
- **Experiment 2**

Class motor:



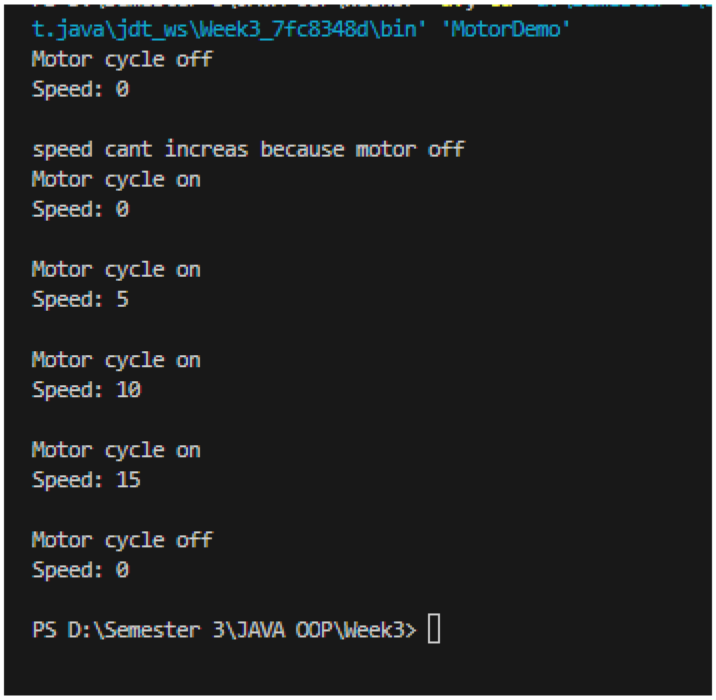
```java
public class Motor {
    private int speed = 0;
    private boolean motorOn=false;

    public void startEngine(){
        motorOn=true;
    }

    public void turnOffEngine(){
        motorOn=false;
        speed=0;
    }

    public void increaseEngine(){
        if (motorOn ==true) {
            speed+=5;
        }
        else{
            System.out.println(x:"speed cant increas because motor off");
        }
    }


    public void reduceEngine(){
        if (motorOn == true) {
            speed-=5;
        }
        else{
            System.out.println(x:"Motor cycle off");
        }

    }


    public void statusPrint(){
        if (motorOn==true) {
            System.out.println(x:"Motor cycle on");
```

Class motordemo:



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

        motor.statusPrint();
        motor.increaseEngine();

        motor.startEngine();
        motor.statusPrint();

        motor.increaseEngine();
        motor.statusPrint();

        motor.increaseEngine();
        motor.statusPrint();

        motor.increaseEngine();
        motor.statusPrint();

        motor.turnOffEngine();
        motor.statusPrint();

    }
}
```

Result:



```
t.java\jdt_ws\Week3_7fc8348d\bin' 'MotorDemo'
Motor cycle off
Speed: 0

speed cant increas because motor off
Motor cycle on
Speed: 0

Motor cycle on
Speed: 5

Motor cycle on
Speed: 10

Motor cycle on
Speed: 15

Motor cycle off
Speed: 0

PS D:\Semester 3\JAVA OOP\Week3>
```

3.3 Questions

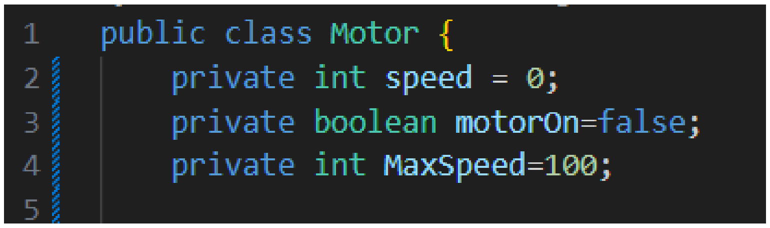
1. In the MotorDemo class, when we add speed for the first time, why does the warning "Speed cannot appear because the engine is off!"

-because the motor still off,and the other reason is not using motor.startEngine(); method first,so in the program Boolean engine is off

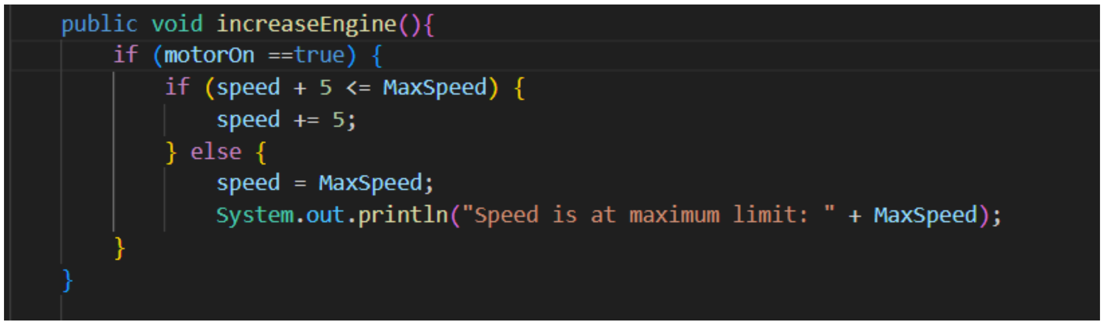
2. Can the speed and contact attributes be set private?

-it can these internal attributes cannot be accessed directly from outside the class, and can only be modified through specific methods (like startEngine, increaseEngine, etc.)

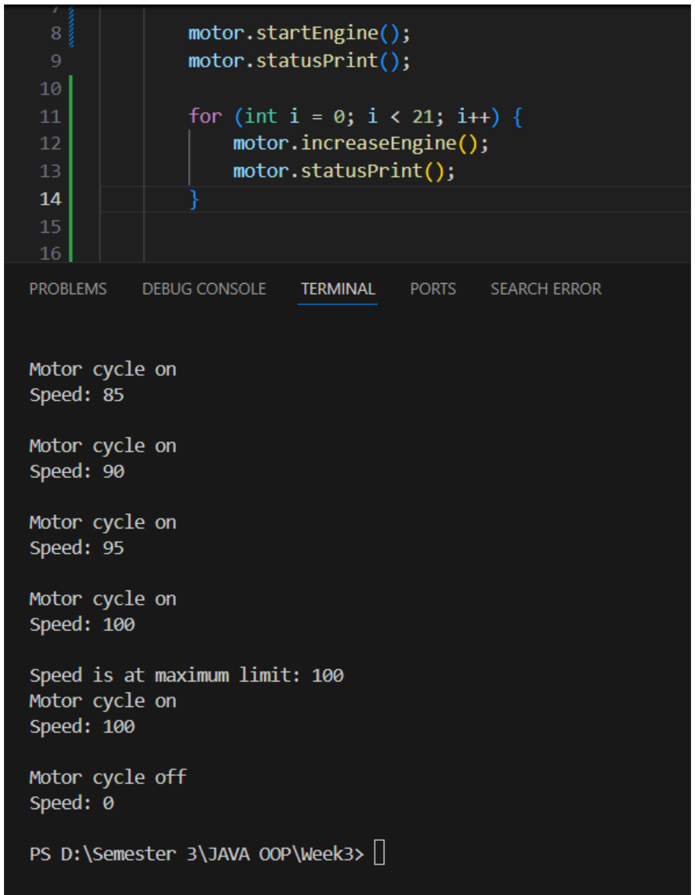
3. Change the Motor class so that the maximum speed is 100!



```java
public class Motor {
    private int speed = 0;
    private boolean motorOn=false;
    private int MaxSpeed=100;

```



```java
public void increaseEngine(){
    if (motorOn ==true) {
        if (speed + 5 <= MaxSpeed) {
            speed += 5;
        } else {
            speed = MaxSpeed;
            System.out.println("Speed is at maximum limit: " + MaxSpeed);
    }
}
```

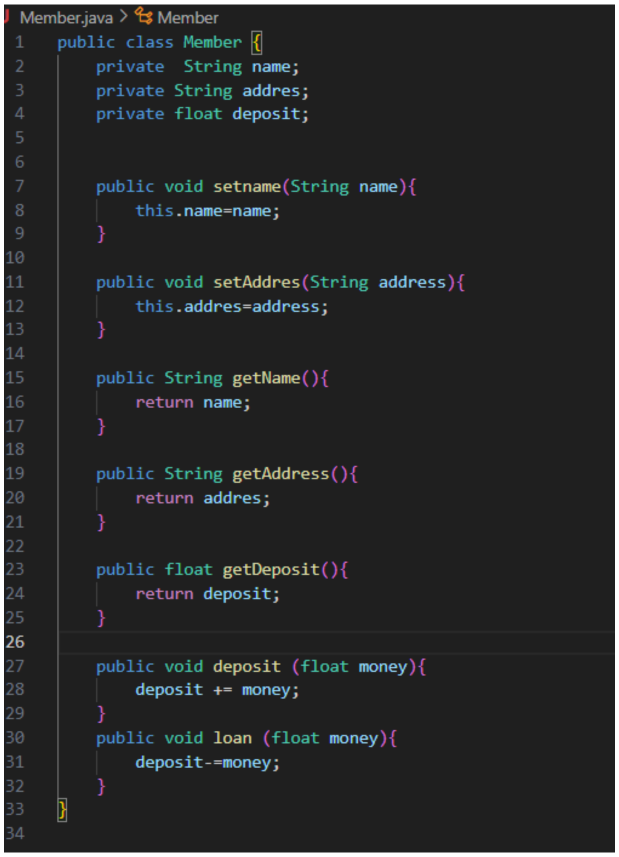


```java
        motor.startEngine();
        motor.statusPrint();

        for (int i = 0; i < 21; i++) {
            motor.increaseEngine();
            motor.statusPrint();
        }


```

```
Motor cycle on
Speed: 85

Motor cycle on
Speed: 90

Motor cycle on
Speed: 95

Motor cycle on
Speed: 100

Speed is at maximum limit: 100
Motor cycle on
Speed: 100

Motor cycle off
Speed: 0

PS D:\Semester 3\JAVA OOP\Week3> 
```

- **3.4 Experiment 3 - Getter and Setter**

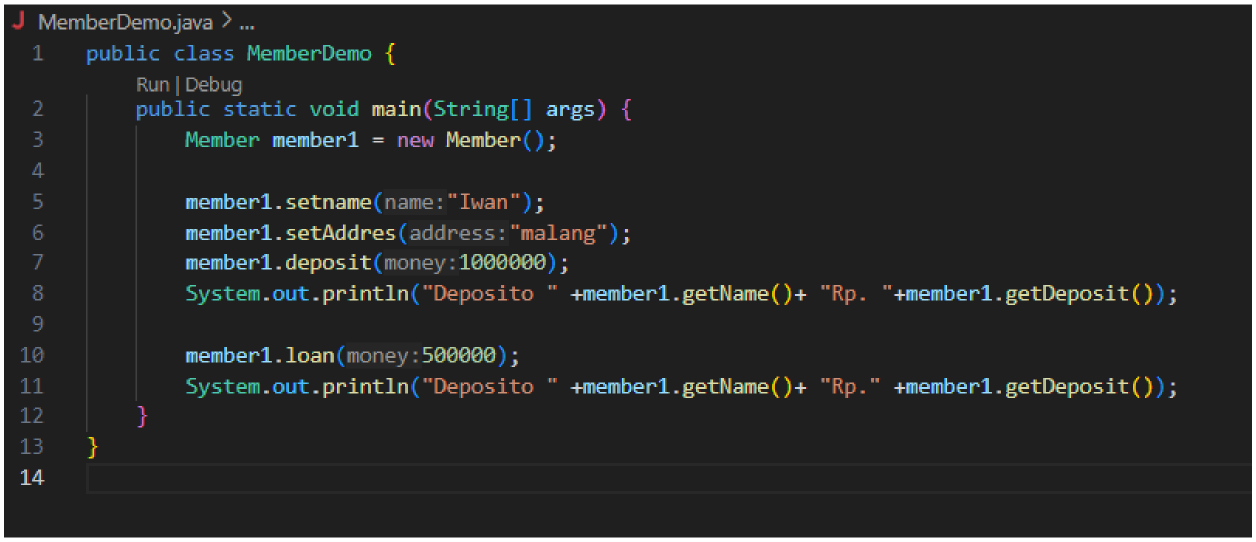
Class member:



Member.java > Member

```java
public class Member {
    private  String name;
    private String addres;
    private float deposit;


    public void setname(String name){
        this.name=name;
    }

    public void setAddres(String address){
        this.addres=address;
    }

    public String getName(){
        return name;
    }

    public String getAddress(){
        return addres;
    }

    public float getDeposit(){
        return deposit;
    }

    public void deposit (float money){
        deposit += money;
    }
    public void loan (float money){
        deposit-=money;
    }
}
```

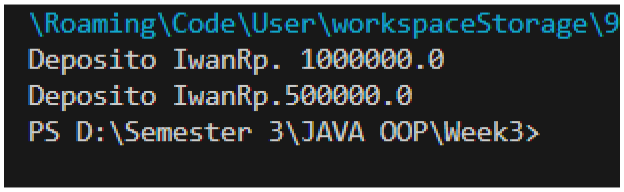
Class MemberDemo:



MemberDemo.java > ...

```java
public class MemberDemo {
    Run | Debug
    public static void main(String[] args) {
        Member member1 = new Member();

        member1.setname(name:"Iwan");
        member1.setAddres(address:"malang");
        member1.deposit(money:1000000);
        System.out.println("Deposito " +member1.getName()+ "Rp. "+member1.getDeposit());

        member1.loan(money:500000);
        System.out.println("Deposito " +member1.getName()+ "Rp." +member1.getDeposit());
    }
}
```

Result:

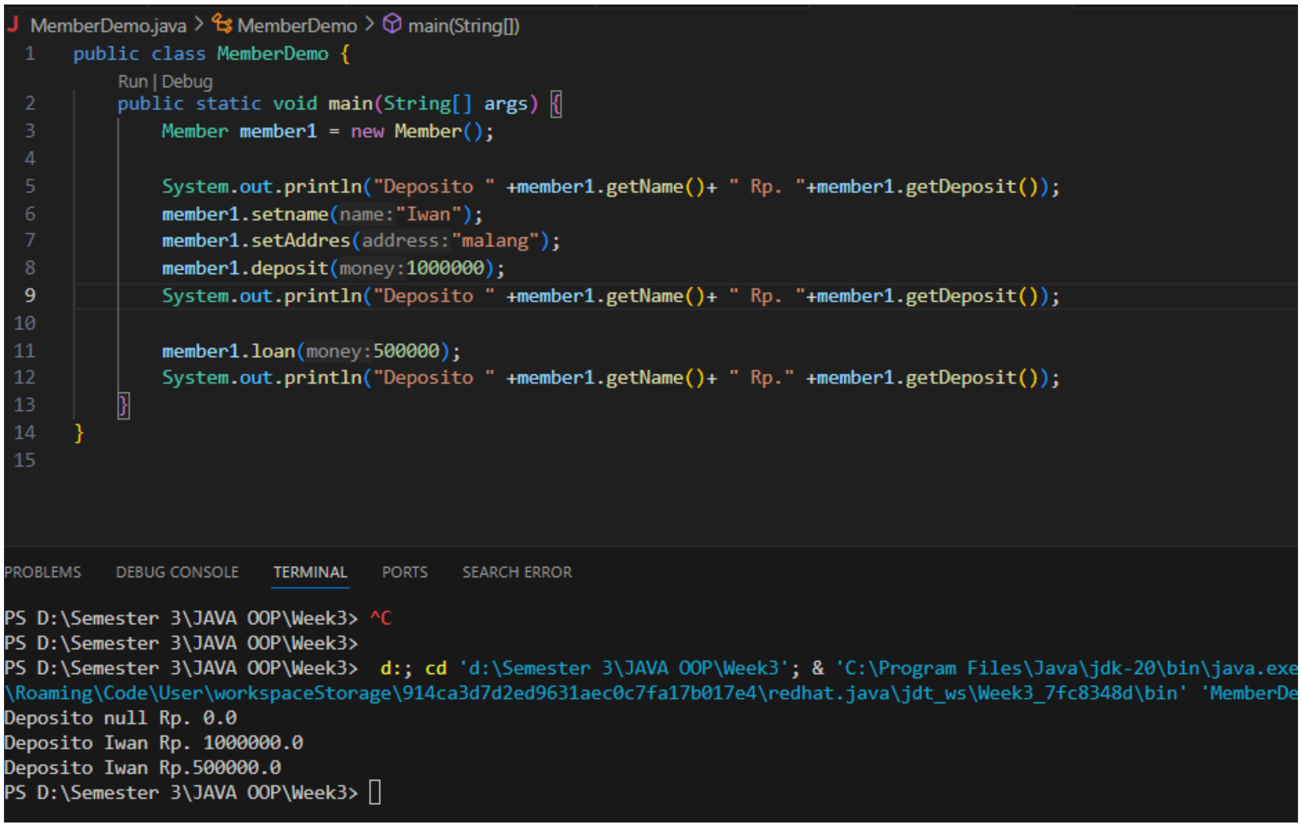


```
\Roaming\Code\User\workspaceStorage\9
Deposito IwanRp. 1000000.0
Deposito IwanRp.500000.0
PS D:\Semester 3\JAVA OOP\Week3>
```

- **Percobaan 4 - Construktor, Instantiation**

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

Class demo:

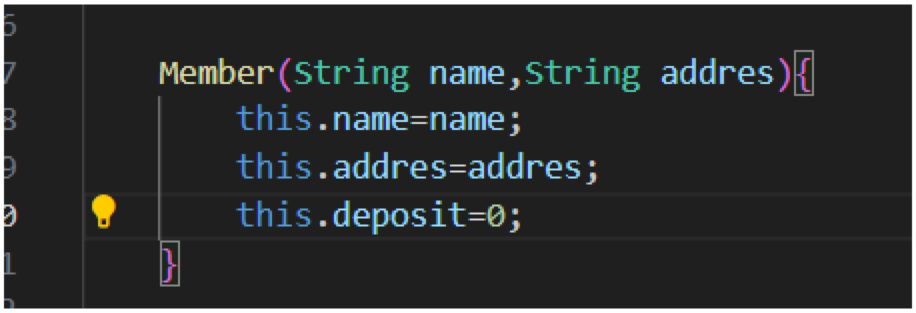


MemberDemo.java > MemberDemo > main(String[])
Run | Debug
PROBLEMS DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

```java
public class MemberDemo {
    public static void main(String[] args) {
        Member member1 = new Member();

        System.out.println("Deposito " +member1.getName()+ " Rp. "+member1.getDeposit());
        member1.setname(name:"Iwan");
        member1.setAddres(address:"malang");
        member1.deposit(money:1000000);
        System.out.println("Deposito " +member1.getName()+ " Rp. "+member1.getDeposit());

        member1.loan(money:500000);
        System.out.println("Deposito " +member1.getName()+ " Rp." +member1.getDeposit());
    }
}
```

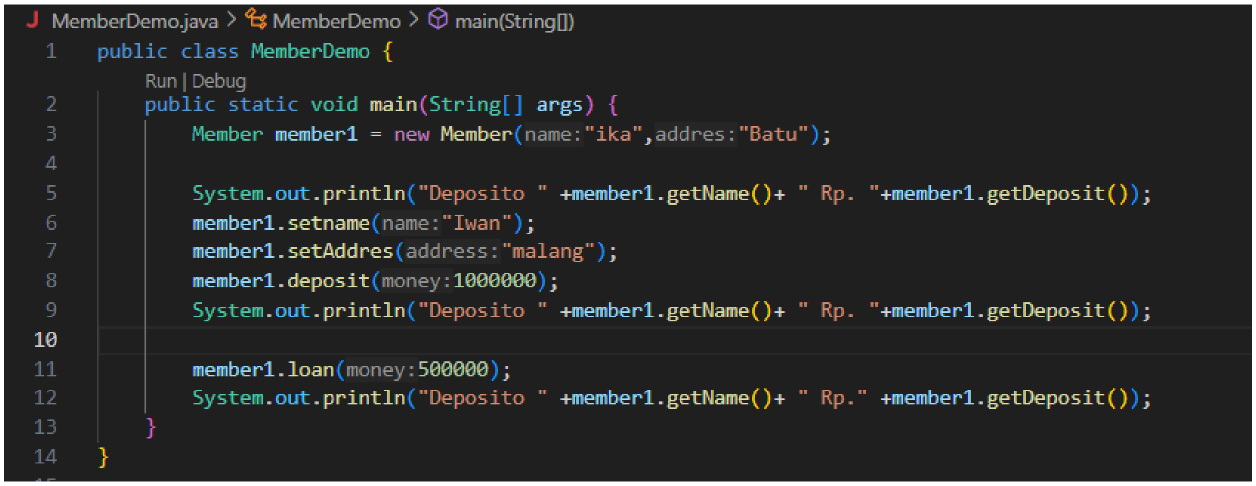
```
PS D:\Semester 3\JAVA OOP\Week3> ^C
PS D:\Semester 3\JAVA OOP\Week3>
PS D:\Semester 3\JAVA OOP\Week3>  d:; cd 'd:\Semester 3\JAVA OOP\Week3'; & 'C:\Program Files\Java\jdk-20\bin\java.exe
\Roaming\Code\User\workspaceStorage\914ca3d7d2ed9631aec0c7fa17b017e4\redhat.java\jdt_ws\Week3_7fc8348d\bin'  'MemberDe
Deposito null Rp. 0.0
Deposito Iwan Rp. 1000000.0
Deposito Iwan Rp.500000.0
PS D:\Semester 3\JAVA OOP\Week3>
```

Class member:


```java
Member(String name,String addres){
    this.name=name;
    this.addres=addres;
    this.deposit=0;
}
```

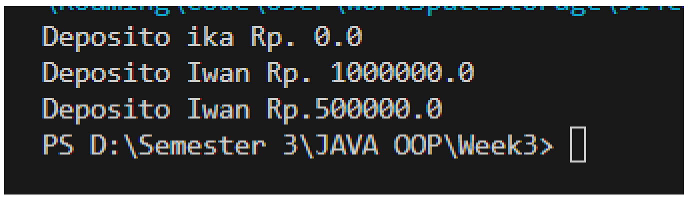
In the Anggota class, a constructor is created with a default access modifier that has 2 nama and alamat parameters. And in the constructor, the pinjam value for the first time is Rp. 0

5. Change class Demo:



```java
public class MemberDemo {
    public static void main(String[] args) {
        Member member1 = new Member(name:"ika",addres:"Batu");

        System.out.println("Deposito " +member1.getName()+ " Rp. "+member1.getDeposit());
        member1.setname(name:"Iwan");
        member1.setAddres(address:"malang");
        member1.deposit(money:1000000);
        System.out.println("Deposito " +member1.getName()+ " Rp. "+member1.getDeposit());

        member1.loan(money:500000);
        System.out.println("Deposito " +member1.getName()+ " Rp." +member1.getDeposit());
    }
}
```

6. The result:


```
Deposito ika Rp. 0.0
Deposito Iwan Rp. 1000000.0
Deposito Iwan Rp.500000.0
PS D:\Semester 3\JAVA OOP\Week3>
```

- **3.6 Question - Experiments 3 and 4**

1. What are getters and setters?

- to access and modify the private attributes of a class. It help to encapsulate the internal representation of an object.

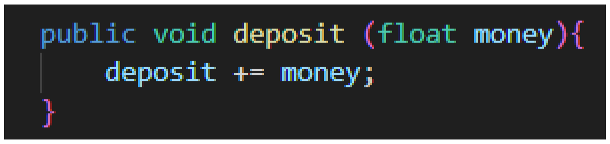
2. What is the use of the getDeposit () method?

- It allows access to the private deposit field in a controlled manner

- getDeposit() is used to retrieve and print the current deposit value

3. What method is used to add balance?

- the deposit() method.



```
public void deposit (float money){
    deposit += money;
}
```

4. What does the constructor mean?

- special method in a class that is called when an instance (object) of the class is created

5. Mention the rules in making a constructor?

- Name Matching: The constructor's name must exactly match the name of the class. For example, if the class is named Member, the constructor must also be named Member

- No Return Type: Constructors do not have a return type, not even void. They are used solely to initialize the object.

- Constructor Chaining: Constructors can call other constructors in the same class using the this() keyword

6. Can the constructor be private?

- Yes, a constructor can be private. Using a private constructor restricts the instantiation of the class from outside its own class

7. When to use parameters with passsing parameters?

- To Provide Input Data: Parameters allow to pass data into a method or constructor so that it can perform operations based on that input.

- To Customize Behavior:Parameters allow to customize the behavior of methods and constructors, making them more flexible

- To Pass Data Between Methods: Parameters can be used to pass data between methods within the same class or across different classes.

8. What is the difference between class attribute and instance attribute?

- Class Attributes:

Scope: Shared among all instances of the class.

Defined: At the class level (outside methods).

Access: Can be accessed via the class name or any instance.

Modification: Changes affect all instances.

Instance Attributes:

Scope: Unique to each instance of the class.

Defined: Inside constructors or methods.

Access: Accessed only through the instance.

Modification: Changes affect only that specific instance.

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

- Class Methods:

Bound to: The class itself.

Access: Class attributes.

Defined with: static keyword (Java) or @classmethod (Python).

Called by: Class name.

Instance Methods:

Bound to: Instances of the class.

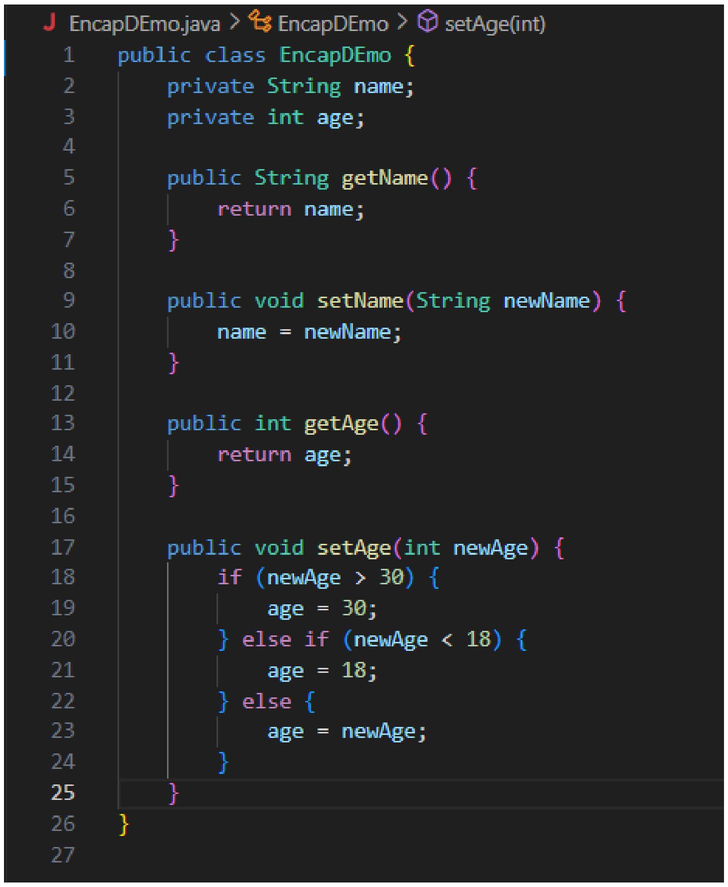
Access: Both instance and class attributes.

Defined without: static keyword or @classmethod.

Called by: Instance of the class

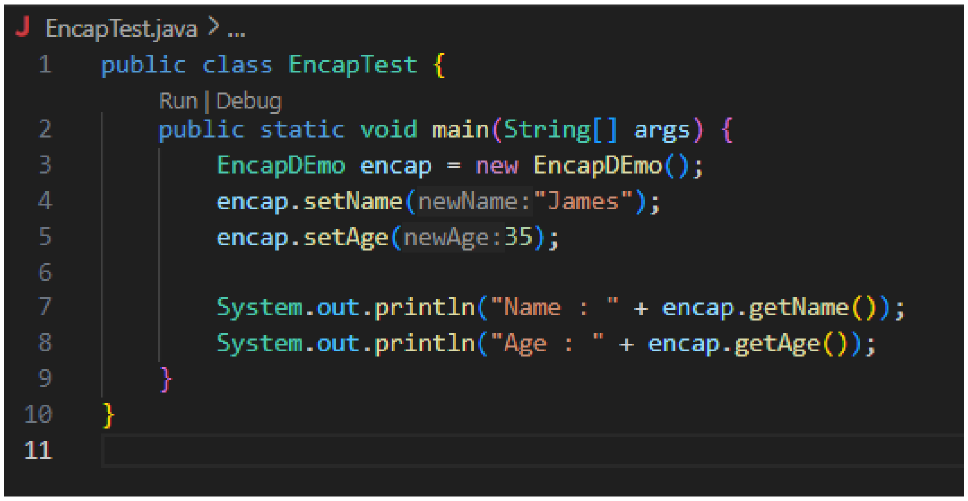
- **5. Task**

1. Try the program below and write the output results



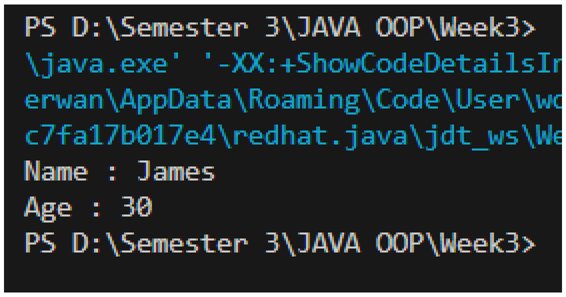
EncapDEmo.java > EncapDEmo > setAge(int)

```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 if (newAge < 18) {
            age = 18;
        } else {
            age = newAge;
        }
    }
}
```

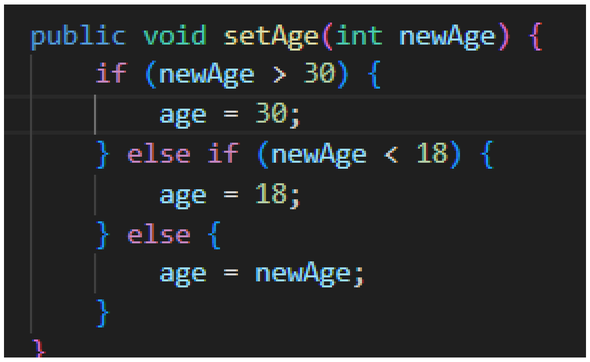


EncapTest.java > ...

```java
public class EncapTest {
    Run | Debug
    public static void main(String[] args) {
        EncapDEmo encap = new EncapDEmo();
        encap.setName(newName:"James");
        encap.setAge(newAge:35);

        System.out.println("Name : " + encap.getName());
        System.out.println("Age : " + encap.getAge());
    }
}
```


```
PS D:\Semester 3\JAVA OOP\Week3>
\java.exe' '-XX:+ShowCodeDetailsIn
erwan\AppData\Roaming\Code\User\wo
c7fa17b017e4\redhat.java\jdt_ws\We
Name : James
Age : 30
PS D:\Semester 3\JAVA OOP\Week3>
```

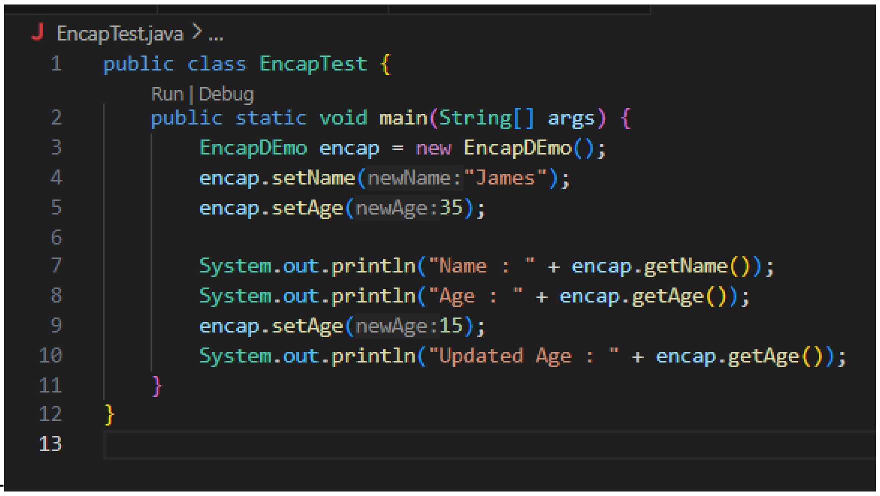
2. In the above program, in the EncapTest class we set age with a value of 35, but when displayed on the screen the value is 30, explain why.

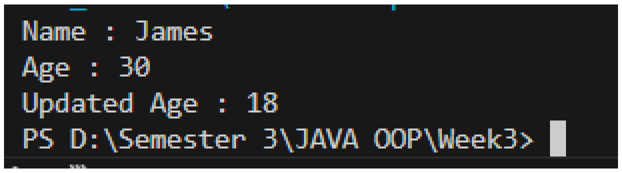

```java
public void setAge(int newAge) {
    if (newAge > 30) {
        age = 30;
    } else if (newAge < 18) {
        age = 18;
    } else {
        age = newAge;
    }
}
```
-

because the setAge() method limited the age to 30.

and when we type 35 it will run in first if newage greater than 30 ,and declare age is 30

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


```java
EncapTest.java > ...
public class EncapTest {
    Run | Debug
    public static void main(String[] args) {
        EncapDEmo encap = new EncapDEmo();
        encap.setName(newName:"James");
        encap.setAge(newAge:35);

        System.out.println("Name : " + encap.getName());
        System.out.println("Age : " + encap.getAge());
        encap.setAge(newAge:15);
        System.out.println("Updated Age : " + encap.getAge());
    }
}
```
-

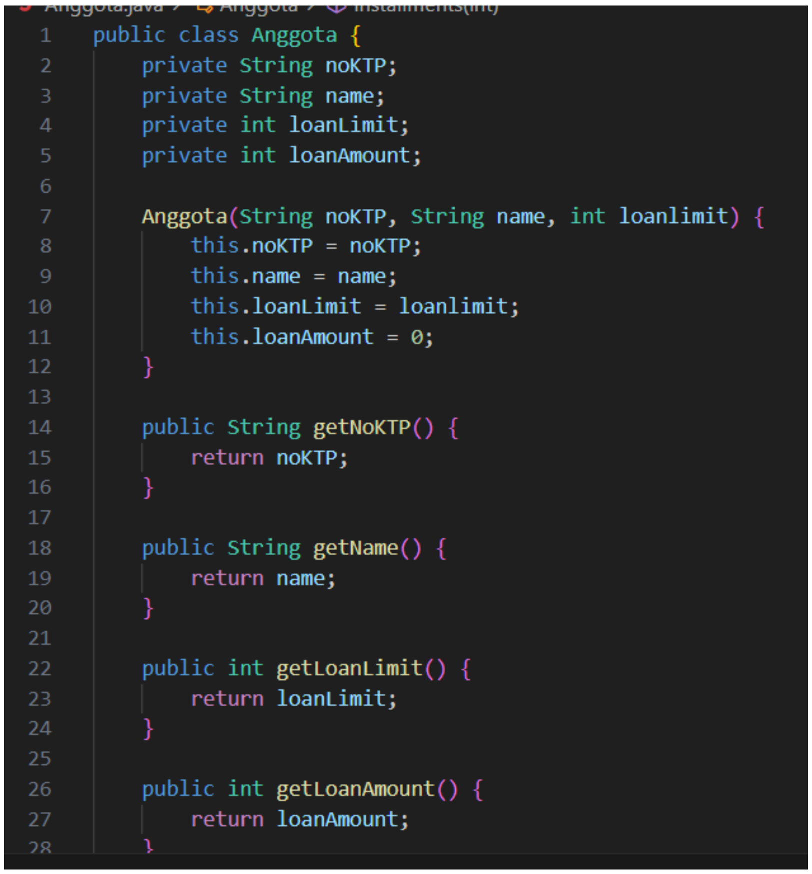


```
Name : James
Age : 30
Updated Age : 18
PS D:\Semester 3\JAVA OOP\Week3>
```

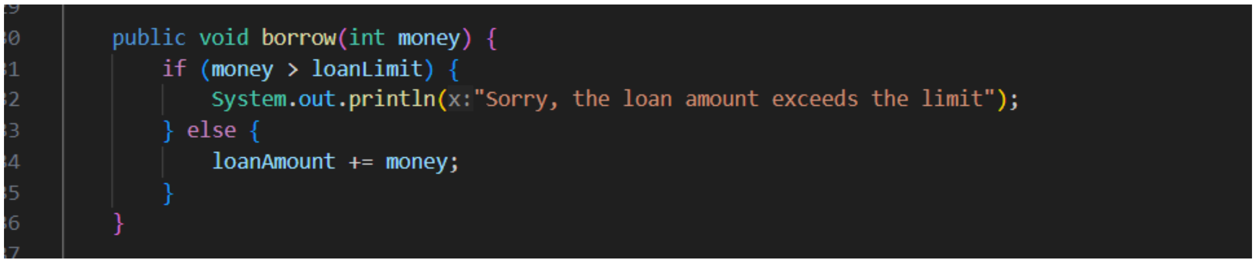
Indeed ,in first program run it set maximum age value 30 and a minimum 18

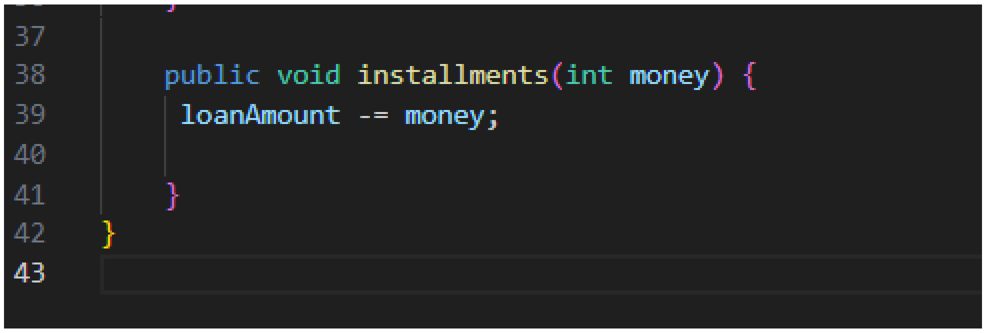
4. In a savings and loan information system, there are class Members who have attributes including KTP number, name, loan limit, and loan amount. Members can borrow money within the specified lending limits. Members can also repay loans. When the Member repays the loan, the loan amount will decrease according to the nominal installment. Create a Member class, provide attributes, methods and constructors as needed. Test with the following TestKoperasi to check whether the class of the Member you created is as expected

Class Anggota:

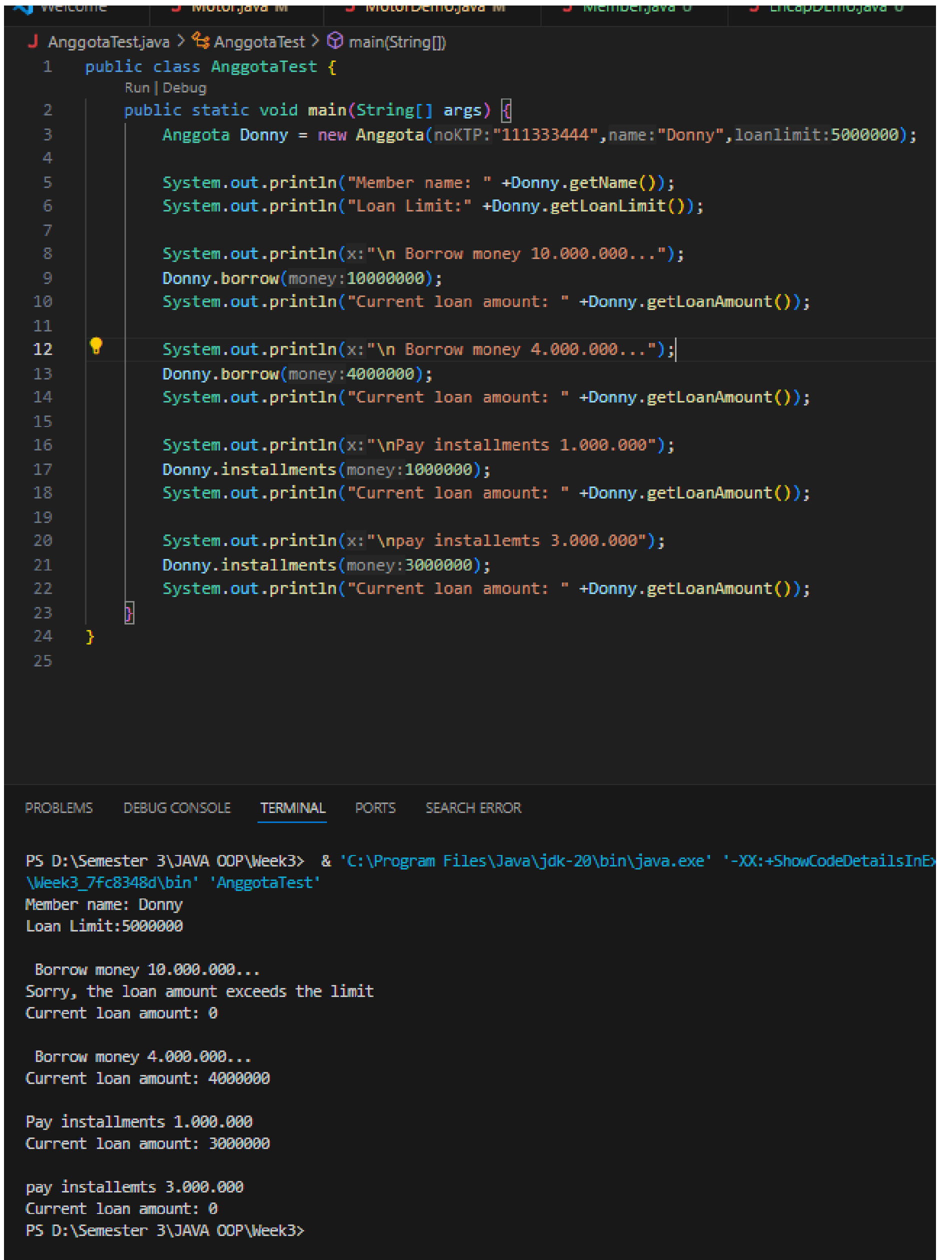


```java
public class Anggota {
    private String noKTP;
    private String name;
    private int loanLimit;
    private int loanAmount;

    Anggota(String noKTP, String name, int loanlimit) {
        this.noKTP = noKTP;
        this.name = name;
        this.loanLimit = loanlimit;
        this.loanAmount = 0;
    }

    public String getNoKTP() {
        return noKTP;
    }

    public String getName() {
        return name;
    }

    public int getLoanLimit() {
        return loanLimit;
    }

    public int getLoanAmount() {
        return loanAmount;
```


```
    public void borrow(int money) {
        if (money > loanLimit) {
            System.out.println(x:"Sorry, the loan amount exceeds the limit");
        } else {
            loanAmount += money;
        }
    }
```


```

    public void installments(int money) {
        loanAmount -= money;

    }
}

```

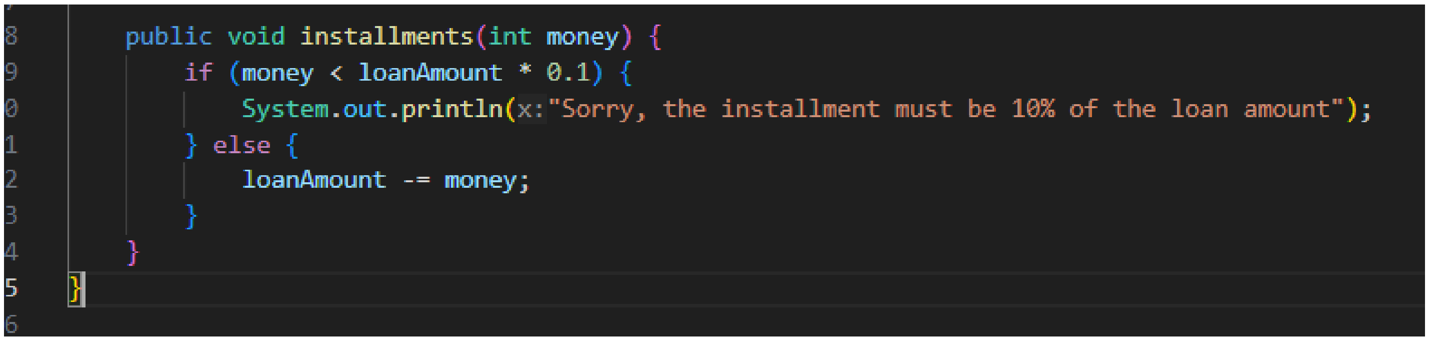
Output:


```
public class AnggotaTest {
    public static void main(String[] args) {
        Anggota Donny = new Anggota(noKTP:"111333444",name:"Donny",loanlimit:5000000);

        System.out.println("Member name: " +Donny.getName());
        System.out.println("Loan Limit:" +Donny.getLoanLimit());

        System.out.println(x:"\n Borrow money 10.000.000...");
        Donny.borrow(money:10000000);
        System.out.println("Current loan amount: " +Donny.getLoanAmount());

        System.out.println(x:"\n Borrow money 4.000.000...");
        Donny.borrow(money:4000000);
        System.out.println("Current loan amount: " +Donny.getLoanAmount());

        System.out.println(x:"\nPay installments 1.000.000");
        Donny.installments(money:1000000);
        System.out.println("Current loan amount: " +Donny.getLoanAmount());

        System.out.println(x:"\npay installemts 3.000.000");
        Donny.installments(money:3000000);
        System.out.println("Current loan amount: " +Donny.getLoanAmount());
    }
}
```

```
PS D:\Semester 3\JAVA OOP\Week3>  & 'C:\Program Files\Java\jdk-20\bin\java.exe' '-XX:+ShowCodeDetailsInEx
\Week3_7fc8348d\bin' 'AnggotaTest'
Member name: Donny
Loan Limit:5000000

 Borrow money 10.000.000...
Sorry, the loan amount exceeds the limit
Current loan amount: 0

 Borrow money 4.000.000...
Current loan amount: 4000000

Pay installments 1.000.000
Current loan amount: 3000000

pay installemts 3.000.000
Current loan amount: 0
PS D:\Semester 3\JAVA OOP\Week3>
```

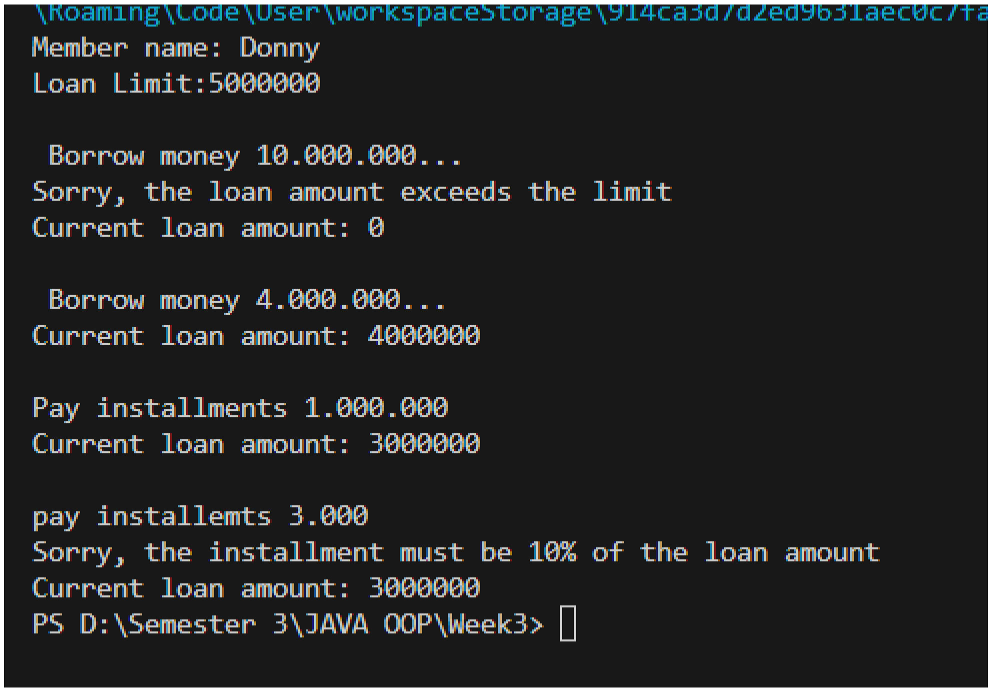
5. Modification of question no. 4 so that the minimum nominal installment is 10% of the current loan amount. If the installments are less than that, then the warning "Sorry, installments must be 10% of the loan amount".

-



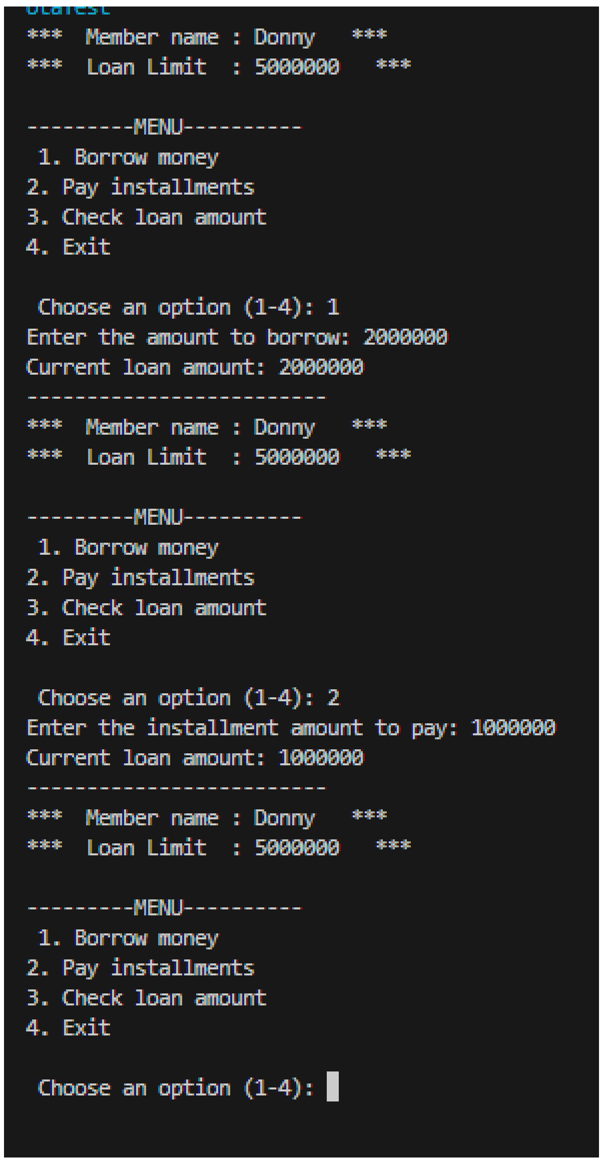
```java
    public void installments(int money) {
        if (money < loanAmount * 0.1) {
            System.out.println(x:"Sorry, the installment must be 10% of the loan amount");
        } else {
            loanAmount -= money;
        }
    }
}
```

Output:



```
\Roaming\Code\User\workspaceStorage\914ca3d7d2ed9631aec0c7fa
Member name: Donny
Loan Limit:5000000

 Borrow money 10.000.000...
Sorry, the loan amount exceeds the limit
Current loan amount: 0

 Borrow money 4.000.000...
Current loan amount: 4000000

Pay installments 1.000.000
Current loan amount: 3000000

pay installemts 3.000
Sorry, the installment must be 10% of the loan amount
Current loan amount: 3000000
PS D:\Semester 3\JAVA OOP\Week3>
```

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



```
***  Member name : Donny   ***
***  Loan Limit  : 5000000   ***

---------MENU----------
 1. Borrow money
2. Pay installments
3. Check loan amount
4. Exit

 Choose an option (1-4): 1
Enter the amount to borrow: 2000000
Current loan amount: 2000000
-------------------------
***  Member name : Donny   ***
***  Loan Limit  : 5000000   ***

---------MENU----------
 1. Borrow money
2. Pay installments
3. Check loan amount
4. Exit

 Choose an option (1-4): 2
Enter the installment amount to pay: 1000000
Current loan amount: 1000000
-------------------------
***  Member name : Donny   ***
***  Loan Limit  : 5000000   ***

---------MENU----------
 1. Borrow money
2. Pay installments
3. Check loan amount
4. Exit

 Choose an option (1-4): 
```

-