

1) Given:

```
public class TaxUtil {  
    double rate = 0.15;  
  
    public double calculateTax(double amount) {  
        return amount * rate;  
    }  
}
```

Would you consider the method calculateTax() a 'pure function'? Why or why not?

If you claim the method is NOT a pure function, please suggest a way to make it pure.

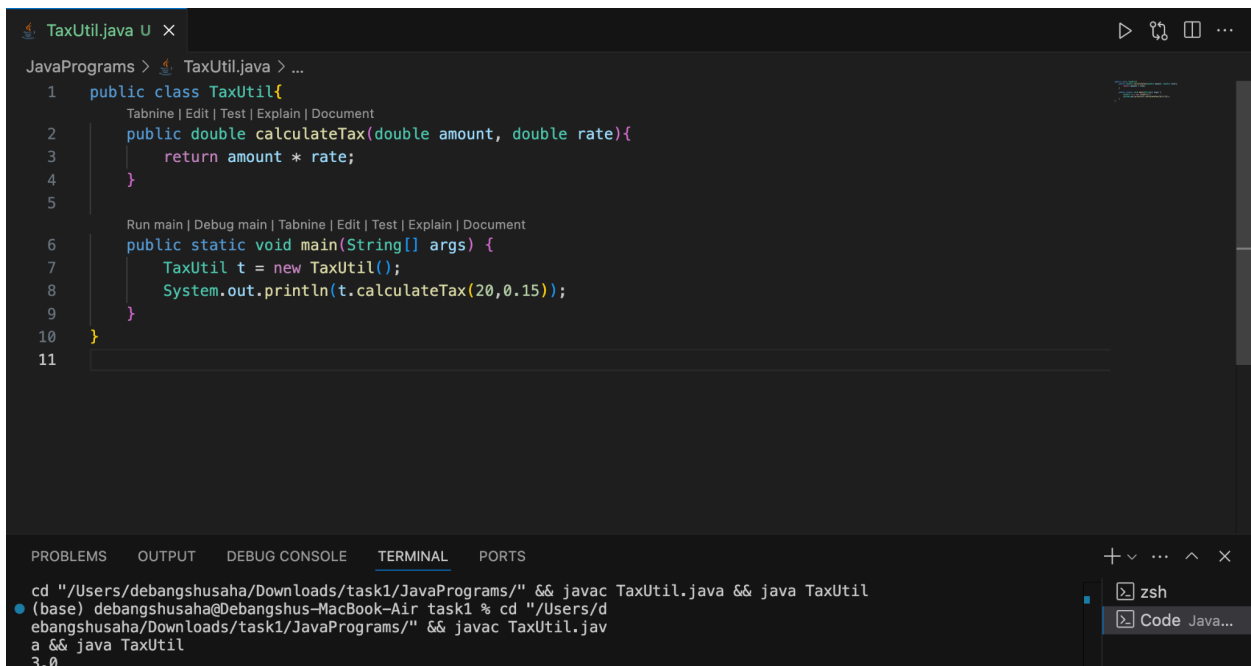
-> No, the method calculateTax() in its current form **is not a pure function**. Because,

- A **pure function** should rely **only on its input parameters** and **should not depend on or modify any external state**.
- calculateTax() depends on the **instance variable rate**, which is external to the method's parameters.
 - If someone changes rate, the output of calculateTax() will change, **even though the input (amount) stays the same**.
 - This violates the principle of referential transparency.

We can accept the rate as a parameter along with the amount, to make it a pure function

Github - <https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/TaxUtil.java>

Screenshot:



The screenshot shows an IDE window with the file 'TaxUtil.java' open. The code in the editor is as follows:

```
1 public class TaxUtil{  
2     Tabnine | Edit | Test | Explain | Document  
3     public double calculateTax(double amount, double rate){  
4         return amount * rate;  
5     }  
6  
7     Run main | Debug main | Tabnine | Edit | Test | Explain | Document  
8     public static void main(String[] args) {  
9         TaxUtil t = new TaxUtil();  
10        System.out.println(t.calculateTax(20,0.15));  
11    }  
12 }
```

Below the editor, the 'TERMINAL' tab is active, showing the following commands and output:

```
cd "/Users/debangshusaha/Downloads/task1/JavaPrograms/" && javac TaxUtil.java && java TaxUtil  
(base) debangshusaha@Debangshus-MacBook-Air task1 % cd "/Users/d  
ebangshusaha/Downloads/task1/JavaPrograms/" && javac TaxUtil.jav  
a && java TaxUtil  
3.0
```

On the right side of the terminal, there are tabs for 'zsh' and 'Code Java...'.

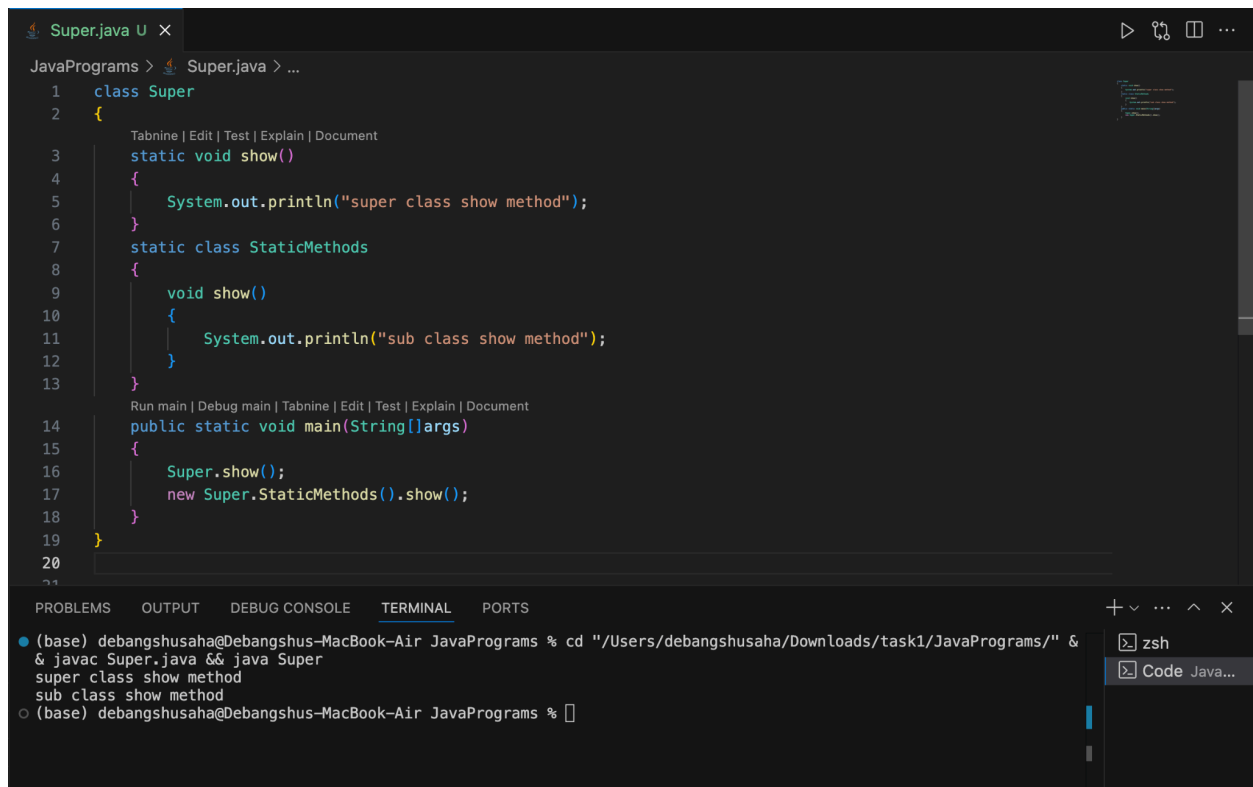
2) What will be the output for the following code?

class Super

```
{
    static void show()
    {
        System.out.println("super class show method");
    }
    static class StaticMethods
    {
        void show()
        {
            System.out.println("sub class show method");
        }
    }
    public static void main(String[]args)
    {
        Super.show();
        new Super.StaticMethods().show();
    }
}
```

Github : <https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/Super.java>

Screenshot:

A screenshot of an IDE window titled 'Super.java'. The code is as follows:

```
1 class Super
2 {
3     static void show()
4     {
5         System.out.println("super class show method");
6     }
7     static class StaticMethods
8     {
9         void show()
10        {
11            System.out.println("sub class show method");
12        }
13    }
14    public static void main(String[]args)
15    {
16        Super.show();
17        new Super.StaticMethods().show();
18    }
19 }
20
```

The IDE has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is active, showing the command:

```
(base) debangshusaha@Debangshus-MacBook-Air JavaPrograms % cd "/Users/debangshusaha/Downloads/task1/JavaPrograms/" &
& javac Super.java && java Super
super class show method
sub class show method
(base) debangshusaha@Debangshus-MacBook-Air JavaPrograms %
```

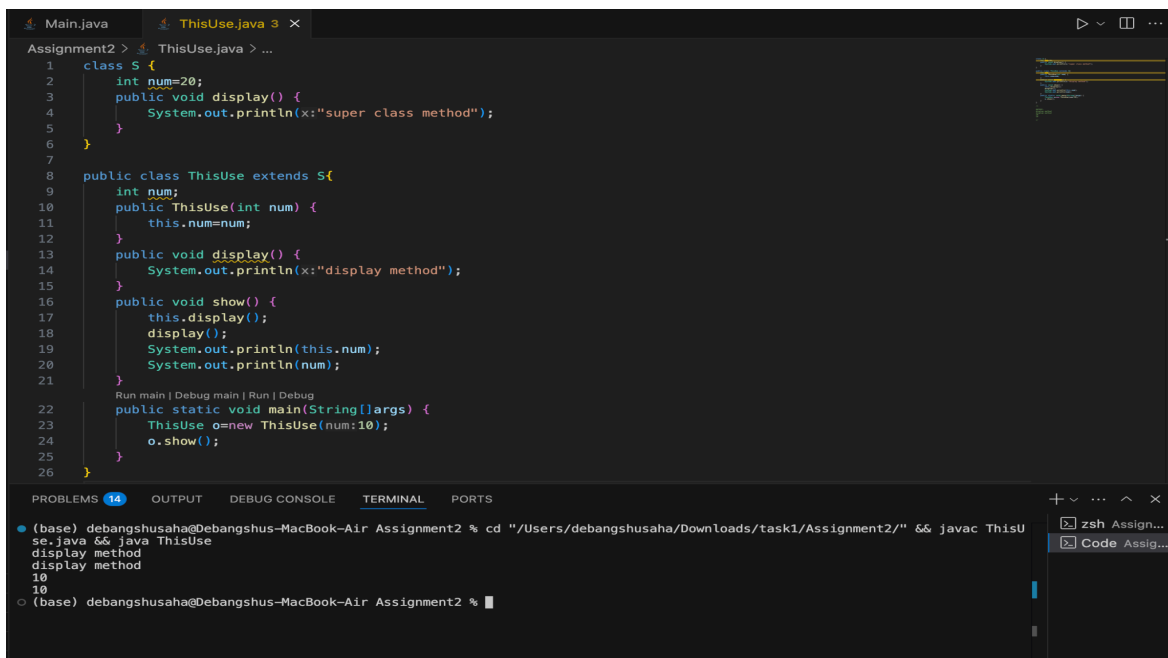
3) What will be the output for the following code?

```
class Super {
    int num=20;
    public void display() {
        System.out.println("super class method");
    }
}

public class ThisUse extends Super {
    int num;
    public ThisUse(int num) {
        this.num=num;
    }
    public void display() {
        System.out.println("display method");
    }
    public void show() {
        this.display();
        display();
        System.out.println(this.num);
        System.out.println(num);
    }
    public static void main(String[]args) {
        ThisUse o=new ThisUse(10);
        o.show();
    }
}
```

-> Github : <https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/ThisUse.java>

Screenshot:



The screenshot shows an IDE with a dark theme. The editor displays the Java code from the previous block, with line numbers 1 through 26. The code defines a `Super` class with a static `num` field and a `display` method. `ThisUse` extends `Super`, has its own `num` field, and implements `display` and `show` methods. The `main` method creates a `ThisUse` object with `num=10` and calls `show`. The `show` method calls `display` twice, prints `this.num`, and prints `num`. The bottom panel shows the `TERMINAL` tab with the command `cd "/Users/debangshusaha/Downloads/task1/Assignment2/" && javac ThisUse.java && java ThisUse` and the output: `display method`, `display method`, `10`, and `10`.

```
1  class S {
2      int num=20;
3      public void display() {
4          System.out.println(x:"super class method");
5      }
6  }
7
8  public class ThisUse extends S{
9      int num;
10     public ThisUse(int num) {
11         this.num=num;
12     }
13     public void display() {
14         System.out.println(x:"display method");
15     }
16     public void show() {
17         this.display();
18         display();
19         System.out.println(this.num);
20         System.out.println(num);
21     }
22     public static void main(String[]args) {
23         ThisUse o=new ThisUse(num:10);
24         o.show();
25     }
26 }
```

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS

(base) debangshusaha@Debangshus-MacBook-Air Assignment2 % cd "/Users/debangshusaha/Downloads/task1/Assignment2/" && javac ThisUse.java && java ThisUse

display method

display method

10

10

(base) debangshusaha@Debangshus-MacBook-Air Assignment2 %

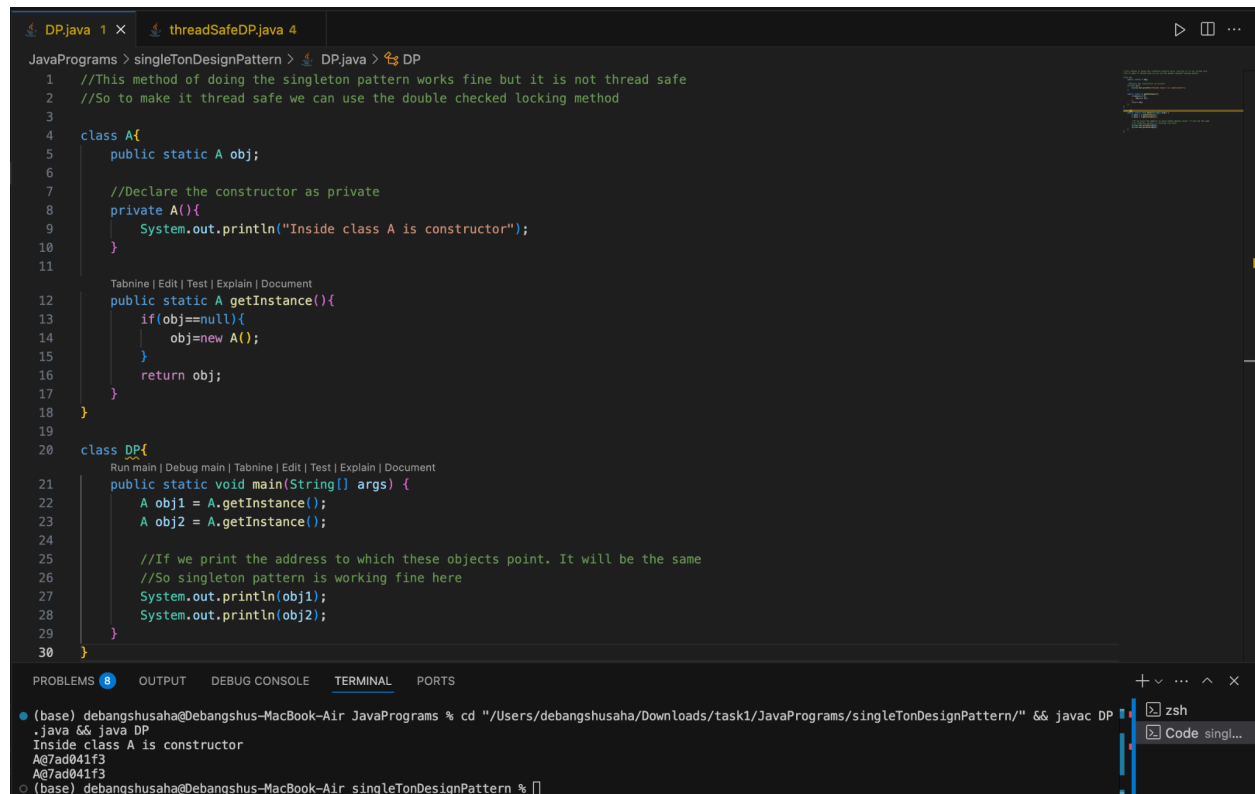
4) What is the singleton design pattern? Explain with a coding example.

-> Singleton Design Pattern is one of the simplest design patterns. It ensures that a class is having only one instance of it, and it provides a global point of access to it.

To implement a singleton design pattern, we need to make sure that we use a private constructor.

Singleton design pattern code -

<https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/singleTonDesignPattern/DP.java>



```
1 //This method of doing the singleton pattern works fine but it is not thread safe
2 //So to make it thread safe we can use the double checked locking method
3
4 class A{
5     public static A obj;
6
7     //Declare the constructor as private
8     private A(){
9         System.out.println("Inside class A is constructor");
10    }
11
12    public static A getInstance(){
13        if(obj==null){
14            obj=new A();
15        }
16        return obj;
17    }
18 }
19
20 class DP{
21     public static void main(String[] args) {
22         A obj1 = A.getInstance();
23         A obj2 = A.getInstance();
24
25         //If we print the address to which these objects point. It will be the same
26         //So singleton pattern is working fine here
27         System.out.println(obj1);
28         System.out.println(obj2);
29     }
30 }
```

PROBLEMS 0 OUTPUT DEBUG CONSOLE TERMINAL PORTS

(base) debangshusaha@Debangshus-MacBook-Air JavaPrograms % cd "/Users/debangshusaha/Downloads/task1/JavaPrograms/singleTonDesignPattern/" && javac DP.java && java DP

Inside class A is constructor

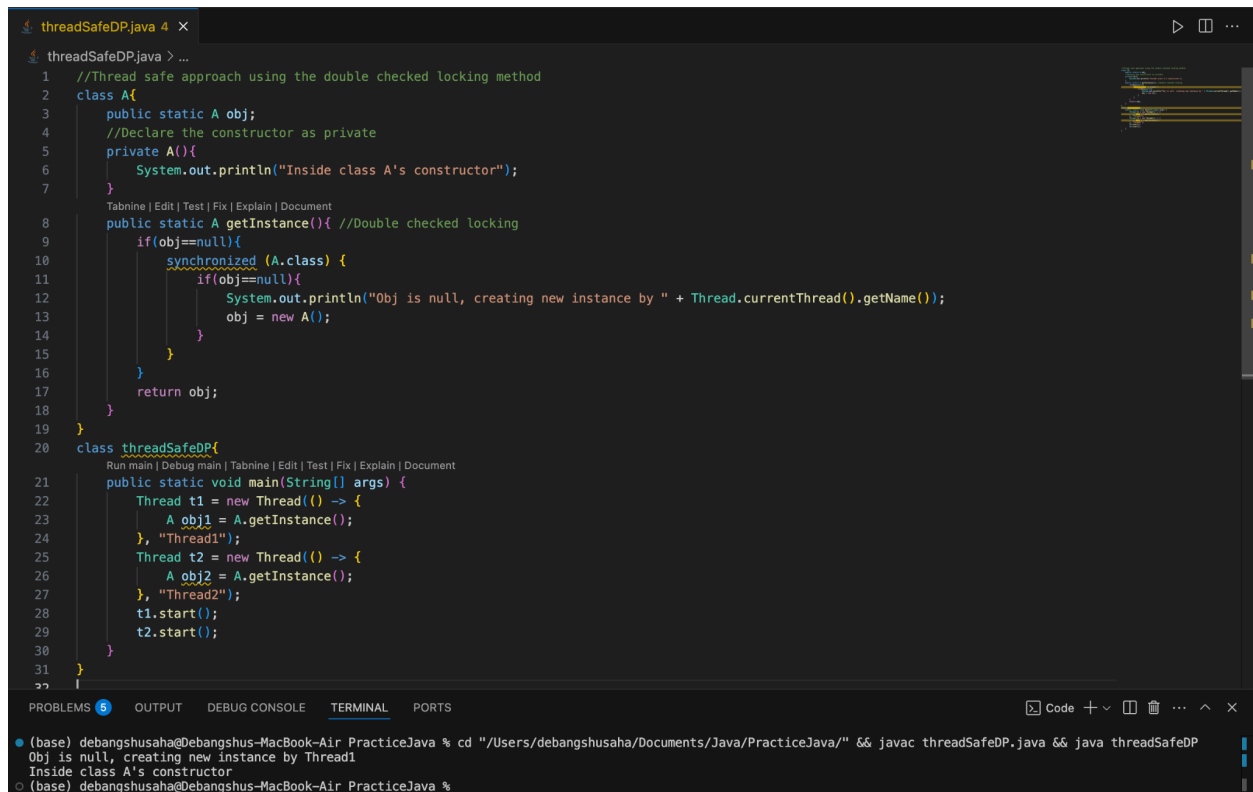
A@7ad041f3

A@7ad041f3

(base) debangshusaha@Debangshus-MacBook-Air singleTonDesignPattern %

-> Singleton Design Pattern using the double checked locking to make it thread safe as well -

<https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/singletonDesignPattern/threadSafeDP.java>



```
1 //Thread safe approach using the double checked locking method
2 class A{
3     public static A obj;
4     //Declare the constructor as private
5     private A(){
6         System.out.println("Inside class A's constructor");
7     }
8     public static A getInstance(){ //Double checked locking
9         if(obj==null){
10             synchronized (A.class) {
11                 if(obj==null){
12                     System.out.println("Obj is null, creating new instance by " + Thread.currentThread().getName());
13                     obj = new A();
14                 }
15             }
16         }
17         return obj;
18     }
19 }
20 class threadSafeDP{
21     public static void main(String[] args) {
22         Thread t1 = new Thread() -> {
23             A obj1 = A.getInstance();
24             }, "Thread1";
25         Thread t2 = new Thread() -> {
26             A obj2 = A.getInstance();
27             }, "Thread2";
28         t1.start();
29         t2.start();
30     }
31 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(base) debangshusaha@Debangshus-MacBook-Air PracticeJava % cd "/Users/debangshusaha/Documents/Java/PracticeJava/" && javac threadSafeDP.java && java threadSafeDP
Obj is null, creating new instance by Thread1
Inside class A's constructor
(base) debangshusaha@Debangshus-MacBook-Air PracticeJava %

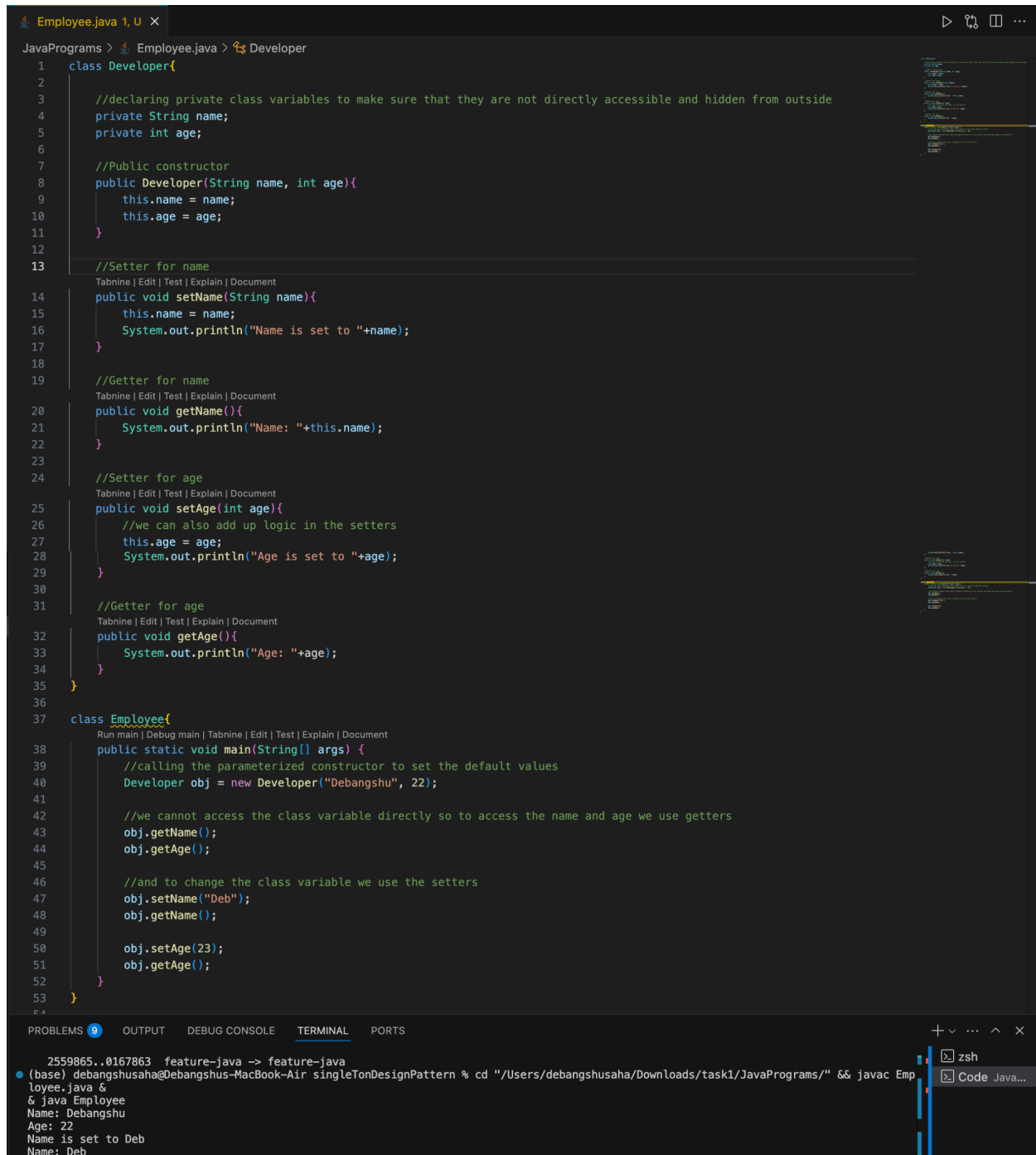
5) How do we make sure a class is encapsulated? Explain with a coding example.

-> We can make sure a class is encapsulated by-

- Declaring the class variables as **private**.
- Providing **public getter** and **setter methods** to access and modify these variables.
- Keeping the business logic and data safe from unintended modification.

Github link -

<https://github.com/DebangshuSaha2002/rg-assignments/blob/feature-java/Assignment2/Employee.java>



```
Employee.java 1, U X
JavaPrograms > Employee.java > Developer
1 class Developer{
2
3     //declaring private class variables to make sure that they are not directly accessible and hidden from outside
4     private String name;
5     private int age;
6
7     //Public constructor
8     public Developer(String name, int age){
9         this.name = name;
10        this.age = age;
11    }
12
13    //Setter for name
14    public void setName(String name){
15        this.name = name;
16        System.out.println("Name is set to "+name);
17    }
18
19    //Getter for name
20    public void getName(){
21        System.out.println("Name: "+this.name);
22    }
23
24    //Setter for age
25    public void setAge(int age){
26        //we can also add up logic in the setters
27        this.age = age;
28        System.out.println("Age is set to "+age);
29    }
30
31    //Getter for age
32    public void getAge(){
33        System.out.println("Age: "+age);
34    }
35 }
36
37 class Employee{
38     public static void main(String[] args) {
39         //calling the parameterized constructor to set the default values
40         Developer obj = new Developer("Debangshu", 22);
41
42         //we cannot access the class variable directly so to access the name and age we use getters
43         obj.getName();
44         obj.getAge();
45
46         //and to change the class variable we use the setters
47         obj.setName("Deb");
48         obj.getName();
49
50         obj.setAge(23);
51         obj.getAge();
52     }
53 }
54
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS
2559865..0167863 feature-java -> feature-java
(base) debangshusaha@Debangshus-MacBook-Air singleTonDesignPattern % cd "/Users/debangshusaha/Downloads/task1/JavaPrograms/" && javac Employee.java & java Employee
Name: Debangshu
Age: 22
Name is set to Deb
Name: Deb
```

6) Perform CRUD operation using ArrayList collection in an EmployeeCRUD class for the below Employee

```
class Employee{  
    private int id;  
    private String name;  
    private String department;  
}
```

-> Github link -

https://github.com/DebangshuSaha2002/rg-assignments/tree/feature-java/Assignment2/CRUD_JDBC/src