CORE JAVA

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.

   Ans: No, because it depends on the instance variable `rate`. A pure function should not depend on or modify any state outside its inputs. To make it pure, pass `rate` as a parameter.

   Code:

   public double calculateTax(double amount, double rate) {

   *return amount * rate;*

   }

   https://github.com/Anandswati712/rg-assignments/blob/feature-java/Week2/Core%20Java/question1/TaxUtil.java

2. **What will be the output for following code?**

   Ans: Static methods belong to the class, not instances. 'Super.show()' calls the static method from the outer class and 'new Super.StaticMethods().show()' calls the inner class method.

   Output:

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment> java Super
super class show method
sub class show method
```

   https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question2

3. **What will be the output for the following code?**

   Ans: 'this.display()' explicitly calls the current class's method andt 'his.num' refers to the subclass variable.

error: cannot find symbol

To Fix call to o.Show(); and then output will be:

Output:

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment\question3>javac ThisUse.java
ThisUse.java:30: error: cannot find symbol
        o.show();
         ^
  symbol:   method show()
  location: variable o of type ThisUse
1 error
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment\question3>javac ThisUse.java
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment\question3>java ThisUse
display method
display method
10
10
```

https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question3

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

   Ans: Singleton ensures that a class has only one instance throughout the application and provides a global point of access to it. We use a private static variable, a private constructor, and a public static method.

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment> javac Singleton.java
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment> java Singleton
Singleton instance method called
Both instances are the same.
```

https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question4

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

   Ans: To ensure a class is encapsulated, its attributes (data) should be hidden from outside access and modification, while access is controlled through well-defined methods (getters and setters). This principle restricts direct access to the internal state of the class, promoting data integrity and easier maintenance.

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment> java Encapsulation
Name: swana
Age: 11

Setting new values...
Name: Alice
Age: 25
```

https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question5

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;

}
```

Ans: We perform Create, Read, Update, and Delete operations using an ArrayList.

create() – adds an employee to the list
read() – returns all employees
update() – modifies employee details by ID
delete() – removes employee by ID

Output:

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment\question6> java
Employees:
Employee ID: 1, Name: Alice, Department: Merchant
Employee ID: 2, Name: Bob, Department: Sales
Employee ID: 3, Name: Charlie, Department: Marketing

Updating Employee with ID 2...
Employees:
Employee ID: 1, Name: Alice, Department: Merchant
Employee ID: 2, Name: Bob Smith, Department: Finance
Employee ID: 3, Name: Charlie, Department: Marketing

Deleting Employee with ID 1...
Employees:
Employee ID: 2, Name: Bob Smith, Department: Finance
Employee ID: 3, Name: Charlie, Department: Marketing
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignment\question6>
```

https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question6

7) Perform CRUD operation using JDBC in an EmployeeJDBC class for the below Employee

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

	id	name	department
▶	1	Alice	HR
	2	John	Finance
	4	Harry	Engineering
*	NULL	NULL	NULL

	id	name	department
▶	1	Alice	HR
	2	John	Finance
	3	Harry	Engineering
	4	Bill	Sales
*	NULL	NULL	NULL

	id	name	department
▶	1	Alice	HR
	2	John	Finance
	3	Harry	Engineering
	4	Bill	Marketing
*	NULL	NULL	NULL

```
PS C:\Users\Swana\OneDrive\Documents\JAVA\rg-assignments\Week2\q
.0/mysql-connector-j-9.2.0.jar" EmployeeJDBC
Connection established successfully!
1 row(s) inserted.
ID | Name | Department
1 | Alice | HR
2 | John | Finance
3 | Harry | Engineering
4 | Bill | Sales

1 row(s) updated.
ID | Name | Department
1 | Alice | HR
2 | John | Finance
3 | Harry | Engineering
4 | Bill | Marketing

1 row(s) deleted.
ID | Name | Department
1 | Alice | HR
2 | John | Finance
3 | Harry | Engineering
```

https://github.com/Anandswati712/rg-assignments/tree/feature-java/Week2/Core%20Java/question7