**Assignment - 2 ( 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.**

**github link -** <https://github.com/hn2006/rg-assignments/blob/feature-java/Week-1/Assignment-2/Core%20Java/Question_1/TaxUtil.java>

**Explanation -**

**calculateTax()** is not a pure function because, by definition, a pure function must always return the same output for the same input and must not depend on or modify any external state.

In this case, the function relies on the **rate** defined as a class-level variable, which can change over time, making the function impure.

To make the function pure, we should pass **rate** as a parameter instead of using a class-level variable. This way, the function depends only on its inputs and produces consistent results.

**2)**

**What will be the output for 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 link -** <https://github.com/hn2006/rg-assignments/blob/feature-java/Week-1/Assignment-2/Core%20Java/Question_2/Super.java>

**Output-**

super class show method

sub class show method

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 link -** <https://github.com/hn2006/rg-assignments/blob/feature-java/Week-1/Assignment-2/Core%20Java/Question_3/ThisUse.java>

**Note: This code will result in a compilation error because the method Show() is defined with an uppercase 'S', but it is being called as show() (with a lowercase 's'). Java is case-sensitive, so method names must match exactly.**

**If we correct the case by calling Show() instead of show(), the code will compile successfully and produce the expected output.**

**Output-**

display method

display method

10

10

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

**github link -** <https://github.com/hn2006/rg-assignments/tree/feature-java/Week-1/Assignment-2/Core%20Java/Question_4>

**Explanation -**

The Singleton Method Design Pattern ensures a class has only one instance and provides a global access point to it.

**Code -**

**public class Singleton {**

**private static Singleton instance;**

**private Singleton() {**

**// Private constructor prevents instantiation from other classes**

**}**

**public static synchronized Singleton getInstance() {**

**// Synchronized method to make it thread-safe**

**if (instance == null) {**

**instance = new Singleton();**

**}**

**return instance;**

**}**

**}**

* A static variable instance holds the single object. Being static means it’s shared across all usages of the class.
* The constructor is private so no other class can create an object using new Singleton().
* This is a public static synchronized method that returns the Singleton instance.
* The first time it's called, it checks if the instance is null and creates the object.
* On subsequent calls, it returns the same instance.
* The method is synchronized to make it thread-safe, ensuring that in a multi-threaded environment, only one object is created even if multiple threads call getInstance() at the same time.

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

**github link -** <https://github.com/hn2006/rg-assignments/tree/feature-java/Week-1/Assignment-2/Core%20Java/Question_5>

**Explanation -**

A **class is said to be encapsulated** when it **hides its internal data** from outside access and **exposes controlled access** through methods. It is achieved using private fields, public getters and setters for accessing and modifying the data and logic inside the methods.

**Code -**

**public class Car {**

**// Private data members ensure data is hidden from outside access**

**private String model;**

**private int speed;**

**// Public getter for read access**

**public String getModel() {**

**return model;**

**}**

**// Public setter for write access**

**public void setModel(String model) {**

**this.model = model;**

**}**

**public int getSpeed() {**

**return speed;**

**}**

**// Method to increase speed using internal logic controls value**

**public void accelerate(int increment) {**

**if (increment > 0) {**

**speed += increment;**

**}**

**}**

**}**

* Private fields: model and speed cannot be accessed directly from outside.
* Public getters/setters: Controlled access is provided through getModel(), setModel(), and getSpeed().
* Internal logic: accelerate(int increment) applies a rule—only positive values are allowed to increase speed, preventing invalid modifications.

**6)**

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

**class Employee{**

**private int id;**

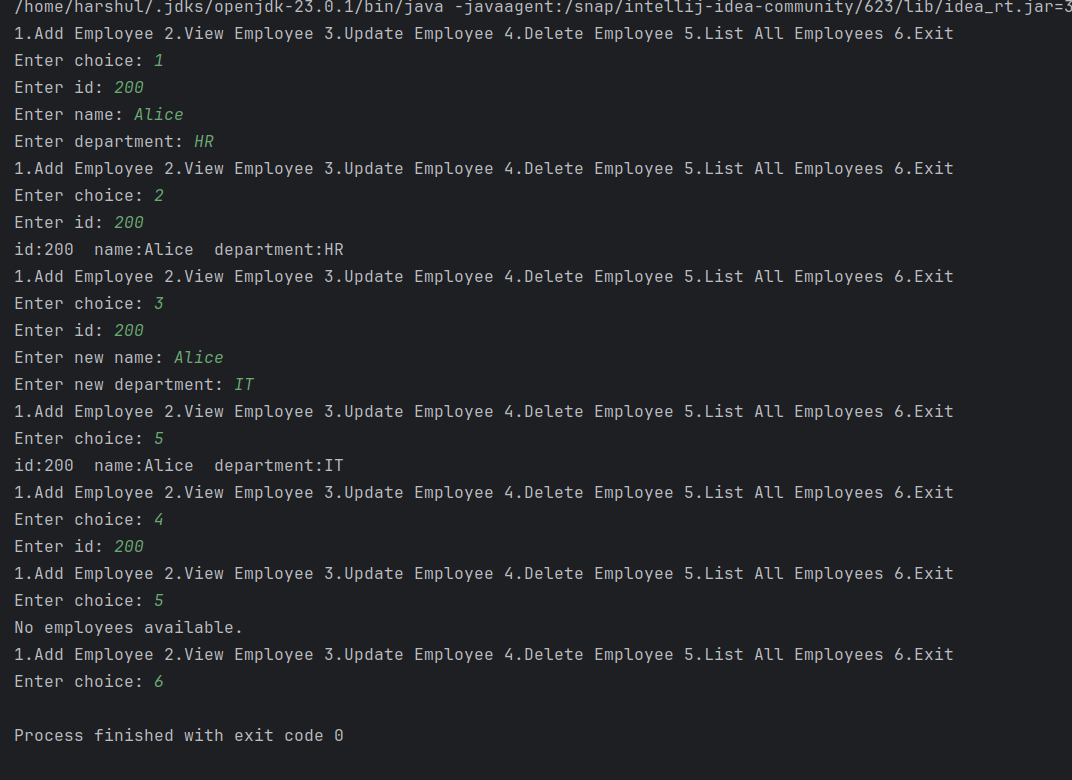
**private String name;**

**private String department;**

**}**

**github link (Code) -** <https://github.com/hn2006/rg-assignments/tree/feature-java/Week-1/Assignment-2/Core%20Java/Question_6>

**Output -**

****

**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;**

**}**

**github link (Code) -** <https://github.com/hn2006/rg-assignments/tree/feature-java/Week-1/Assignment-2/Core%20Java/Question_7>

**Output -**

