# **Assignment: Exploring Java final Keyword and Real-World Scenarios**

# **Question 1: Immutable Class Design**

# Design an immutable class called **Contact** that represents a contact's name and phone number. The class should have final instance variables for name and number, and no setter methods. Implement a constructor to initialize these variables and provide methods for accessing the information.

# **Question 2: Using final Constants**

# Create a class **CircleCalculator** that calculates the area and circumference of a circle. Use a **final** variable for the value of PI. Implement methods for the calculations and demonstrate their use.

# **Question 3: Non-Overridable Methods**

# Create a base class **Animal** with a **final** method **makeSound()** that prints "Animal makes a sound." Design a subclass **Dog** that attempts to override the **makeSound()** method. Explain why this override is not allowed.

# **Question 4: Creating a Final Class**

# Design a final class named **Shape** that represents geometric shapes. Implement a method **calculateArea()** that returns the area of a shape. Then, try to create a subclass **Triangle** that extends **Shape** and explain the resulting error.

# **Question 5: String Pooling and Final Strings**

# Declare a final String named **COUNTRY** with a value "USA". Then, create another String variable with the same value. Compare the references of the two String objects and explain the result. Provide insight into why string pooling is relevant here.

# **Question 6: Final Parameters and Method Arguments**

# Write a method called **sumArray** that takes a final array of integers as an argument. Calculate and return the sum of the integers in the array. Explain how using **final** prevents the method from modifying the array's content.

# **Question 7: Final Methods and Inheritance**

# Create a class **Vehicle** with a final method **startEngine()**. Design a subclass **Car** that attempts to override this method. Describe the outcome and why Java prevents the overriding of final methods.

# **Question 8: Finalizing Class Behavior**

# Design a final class **Logger** that simulates logging behavior. Include final methods for opening, writing, and closing log files. Explain why a class like **Logger** might be made final to control its behavior.

# **Question 9: Final Instance Variables**

# Create a class **Employee** with final instance variables for employee ID and salary. Implement a constructor to initialize these variables. Show how these variables can be set during object creation but cannot be modified afterwards.

# **Question 10: Using final with Enums**

# Design a final enum **Colors** with constants for primary colors. Explain why making enums **final** could be advantageous. Implement a method that takes a **Colors** enum constant and prints its name.