Exam : 20%

Question 1 : 10 Marks

Certainly! Here's another exercise for practicing inheritance in Java:

Create a Java program that models a university's employee system. The system should have the following classes:

1. **`Employee` class:**
   - Properties:
     `id` (int): Represents the employee's ID number.
     `name` (String): Represents the employee's name.
   - Methods:
     Write 3 different constructor
     `calculateSalary()`: This method should be overridden by subclasses to calculate the salary of different types of employees.

2. **`Faculty` class (subclass of `Employee`):**
   - Additional Properties:
     `monthlySalary` (double): Represents the monthly salary of the faculty member.
   - Additional Methods:
     `calculateSalary()`: Overrides the `calculateSalary()` method to calculate the annual salary of the faculty member, which is the monthly salary multiplied by 12.

3. **`Staff` class (subclass of `Employee`):**
   - Additional Properties:
     `hourlyRate` (double): Represents the hourly rate of the staff member.
     `hoursWorked` (int): Represents the number of hours worked by the staff member.
   - Additional Methods:
     `calculateSalary()`: Overrides the `calculateSalary()` method to calculate the monthly salary of the staff member, which is the product of the hourly rate and the number of hours worked.

In your `main` method, create objects of the `Faculty` and `Staff` classes and perform the following actions:

A. Create a `Faculty` object with an ID, name, and a monthly salary of $5000. Calculate and display the annual salary of the faculty member.

B. Create a `Staff` object with an ID, name, an hourly rate of $15, and 160 hours worked. Calculate and display the monthly salary of the staff member.

Ensure that you utilize inheritance, override methods, and apply appropriate access modifiers.

Question 2: 10 Marks

Create a Java program to model different shapes. The program should have the following classes:

1. `Shape` class:
   - Methods:

     `calculateArea()`: This method should be overridden by subclasses to calculate the area of specific shapes.

     `displayInfo()`: This method should be overridden by subclasses to display specific information about each shape.

2. `Rectangle` class (subclass of `Shape`):
   - Properties:

     `length` (double): Represents the length of the rectangle.

     `width` (double): Represents the width of the rectangle.

   - Methods:

     `calculateArea()`: Overrides the `calculateArea()` method to calculate the area of the rectangle as the product of length and width.

     `displayInfo()`: Overrides the `displayInfo()` method to display the length, width, and area of the rectangle.

3. `Circle` class (subclass of `Shape`):
   - Properties:

     `radius` (double): Represents the radius of the circle.

   - Methods:

     `calculateArea()`: Overrides the `calculateArea()` method to calculate the area of the circle as π * radius^2.

     `displayInfo()`: Overrides the `displayInfo()` method to display the radius and area of the circle.

In your `main` method, create an array of `Shape` objects that includes rectangles and circles. Loop through the array and perform the following actions:

1. Create a `Rectangle` object with a length of 5.0 and width of 3.0.

2. Create a `Circle` object with a radius of 2.5.

3. Ask user witch shape he wants to make , base of his need take the values and calculate the area.

For each shape in the array, call the `calculateArea()` method and display the shape's information using the `displayInfo()` method.

Remember to utilize polymorphism by treating each shape as a `Shape` object, even though they are instances of specific subclasses.