CSCI 1250 – Lab 8

Classes & Objects / UML

Introduction

This lab will reinforce topics covered in the Classes & Objects lecture.

NOTE: All of your programs must abide by the documentation standards discussed in the Coding Standards document on D2L.

Create a .NET console application in a directory called Lab8. In the Lab8 project, you will have three files – **Circle.cs** (the *Circle* class), **PersonalInformation.cs** (the *PersonalInformation* class), and **Program.cs** (your driver that will create objects of the classes and execute their methods as specified).

Instructions

Part 1: The *Circle* Class

Creating the *Circle* Class

Create a new Java class called *Circle* based on the provided UML diagram (Circle_UML.png). Make sure the names of your class, fields, methods, and parameters all match the UML as well as any data types / return types.

The methods in the UML should function as described below.

- There will be a single constructor that takes a single argument, *radius*, and sets the value of the *_radius* field to match the parameter value
- *GetRadius* is an accessor (getter) method that returns the value of the *_radius* field
- *SetRadius* is a mutator (setter) method that changes the value of the *_radius* field to be equal to the parameter variable's value
- *GetDiameter* will return the diameter of the circle object based on the value of the *_radius* field
- *GetCircumference* will return the circumference of the circle object based on the value of the *_radius* field
- *GetArea* will return the area of the circle object based on the value of the *_radius* field

Hint: Use the *Math.PI* property in any mathematical operations that require the value of pi (circumference and area of a circle require pi)

Using the *Circle* Class

In *Program.cs*, implement the following criteria.

- Write a *static* method called D*isplayCircleInformation*
    - This method does not return any values
    - This method has a single parameter, a *double* named *radius*

        - Create a new *Circle* object and assign its address to a reference variable
            - Pass the value of the *radius* parameter into the constructor method
        - Use the G*etDiameter*, G*etCircumference*, and G*etArea* methods of the *Circle* object to display the circle's diameter, circumference, and area.
- In your *Main* method
    - Ask the user to enter a radius and assign their input to a *double* variable
    - Pass the radius the user entered to the D*isplayCircleInformation* method

Part 2: The *PersonalInformation* Class

Creating the *PersonalInformation* Class

Using the provided UML for the *PersonalInformation* class, write the necessary code for the class. Make sure your fields, methods, and parameters match what is in the UML exactly. Your constructor should take the arguments shown in the UML and set the corresponding field values accordingly.

Using the *PersonalInformation* Class

In *Program.cs*, implement the following criteria.

- In your Main method
    - Ask the user to enter a name, address, age, and phone number (make sure the data types match the fields of the *PersonalInformation* class)
    - Create a new object of the *PersonalInformation* class using the constructor shown in the UML, passing in the values the user entered to the constructor
    - Display each of the field values to the screen, using the accessor methods

Submission

Submit your code to the Lab8 repository on GitHub Classroom using the Lab 8 GitHub Classroom link on D2L. Then, submit a text file containing the URL of your Lab8 repository to the drop box on D2L by the specified due date.