CSE 4252: Lab 4 (20 Points)

Due Date: October 24th (11:59 PM)

Overview

This exercise will familiarize you with basic concepts of Classes and Objects in C++.

Exercise 1 Description

Write a complete C++ class to represent a circle in two-dimensional space. Your class should consist of a header (**Circle.h)** and an implementation (**Circle.cpp**) file as described below. The Circle class should have the following (private) attributes and (public) methods:

Attributes

- x coordinate (an int), can be negative
- y coordinate (an int), can be negative
- radius (a double), must be greater than 0.0

Methods

- A default constructor that creates a circle of radius 5 at position 0,0 **[0.5 points]**
- A constructor with parameters for x, y, and radius **[0.5 points]**
- A copy constructor which copies one circle details to the other **[1.5 points]**
- Getters and setters for radius, x and y co-ordinate. Take care of invalid values for radius. **[1.5 points]**
- **void** translate (**int** horizShift, **int** vertShift) - translates the circle along x and y axis **[0.5 points]**
- **double** getArea() - returns the circle's area **[0.5 points]**
- **void** displayCircle() - displays the circle's attributes like this: "Radius: 5.0 at point x = 0, y = 0" **[0.5 points]**
- **bool** intersect (Circle c) - returns true if c intersects the calling circle **[3 points]**
- **bool** operator==(Circle c) - compares two circles and returns true if both circles have same radius and x, y co-ordinate else returns false. **[1.5 points]**
- **Circle** operator+(const Circle &c1, const Circle &c2) - returns a new circle with radius as sum of radius of c1 and c2. Compare the absolute magnitude of the co-ordinates of both the circles and set the x and y values with the greater value. **[2 points]**

- radius (a double), must be greater than 0.0

Methods

- A default constructor that creates a circle of radius 5 at position 0,0 **[0.5 points]**
- A constructor with parameters for x, y, and radius **[0.5 points]**
- A copy constructor which copies one circle details to the other **[1.5 points]**
- Getters and setters for radius, x and y co-ordinate. Take care of invalid values for radius. **[1.5 points]**
- **void** translate (**int** horizShift, **int** vertShift) - translates the circle along x and y axis **[0.5 points]**
- **double** getArea() - returns the circle's area **[0.5 points]**
- **void** displayCircle() - displays the circle's attributes like this: "Radius: 5.0 at point x = 0, y = 0" **[0.5 points]**
- **bool** intersect (Circle c) - returns true if c intersects the calling circle **[3 points]**
- **bool** operator==(Circle c) - compares two circles and returns true if both circles have same radius and x, y co-ordinate else returns false. **[1.5 points]**
- **Circle** operator+(const Circle &c1, const Circle &c2) - returns a new circle with radius as sum of radius of c1 and c2. Compare the absolute magnitude of the co-ordinates of both the circles and set the x and y values with the greater value. **[2 points]**

.h and .cpp files

The header file contains the class interface, and the .cpp file contains the implementation. The header file consists of the class name, and the name (and type) of the member variables and the header for each of the methods. The .cpp file consists of the definition for each of the class methods.

Example of .h and .cpp file

```
// Rectangle.h file
#ifndef RECTANGLE_H
#define RECTANGLE_H

class Rectangle
{
public:
	//Default constructor
	Rectangle(); //the default constructor
	//Constructor with parameters
	Rectangle(int w, int h);
```

```
	//Setters that change the values of the attributes
	void setWidth(int w);
	void setHeight(int h);

	//Getters that return information about the rectangle, note the
const at the end of the method
	int getWidth() const;
	int getHeight() const;
	int getArea() const;

	//Display method that prints the rectangle's height and width
	void displayRectangle() const;

private:
	int width;
	int height;
};
#endif
```

```
//Rectangle.cpp file
#include <iostream>
#include "Rectangle.h"

using namespace std;


/*
 * Now follows each of the method implementations.  The <class>:: that
precedes each method identifies
 * that the function belongs to the class
 */

//Default constructor
Rectangle::Rectangle(){
	width = 1;
	height = 1;
}
// Constructor to create a new rectangle with the given values
Rectangle::Rectangle(int w, int h){
…
```

```
}
//Setters that change the values of the attributes
void Rectangle::setHeight(int h){
	if (h > 0)
		height = h;
	else
		height = 1;
}
//Other setters

// Getters that return information about the rectangle
int Rectangle::getWidth() const{
	return width;
}
…

//Other getters
int Rectangle::getHeight() const{
	return height;
}

void Rectangle::displayRectangle() const {
	cout << endl << "width = " << getWidth();
	cout << ", height = " << getHeight() << endl;
}
```

main function/test program (test.cpp)
Use circle methods to show the functionality. **[3 points]**

Test the code with inputs and record the *script* in **Lab4.txt**.

Submission Instructions

Make sure your programs compile and run correctly before submitting. To submit, zip all the files in **Lab04.zip** and upload on carmen.

Circle.h [3 points]
Circle.cpp [12 points]
test.cpp [3 points]
Lab4.txt [2 points]