Communication Software Design

Lab 07 Polymorphism

11/24/2019

We're using the same "Shape" base class that we've been using in our lectures. The "Rectangle" class is an example of a subclass of "Shape". If you've been attending lecture, this should look very familiar to you.

Part 1: Create the Trapezoid Class

Create a class called "Trapezoid" that inherits from the class "Shape" (it will be very similar to the existing Rectangle class). A trapezoid is a shape with four sides of varying lengths. You will probably want to have four or five member variables, one for each side and possibly one for the height. You can name them whatever you like (e.g. edgeN, edgeS, edgeW, edgeE and height).

Your definition of the Trapezoid class will go into "trapezoid.h" and your implementation of all the functions in your Trapezoid class will go into "trapezoid.cpp". If you're not sure on some of the implementation details, you can check the implementations in the "rectangle.h" and "rectangle.cpp" files for reference.

You need to implement a constructor and optional getters/setters for your Trapezoid as well. Don't forget to overload the virtual "getType()" function.
At this point, you may want to simply create a Trapezoid object in "main" and test it.

Part2: Create the Perimeter Function

In your Shape class, create a pure virtual function called "getPerimeter()". By making this function pure virtual, you will not be able to create an instance of your Shape class because Shape will become an abstract class. In addition, because your Trapezoid and Rectangle classes inherit from Shape, they will be forced to implement the pure virtual "getPerimeter()" function. As the name implies, the "getPerimeter()" function should return the perimeter of the shape (trapezoid or rectangle).

Part3: Polymorphic Print Function

Finally, in your main file, create a function to print out the type and perimeter of whatever Shape that you pass in. Remember that polymorphism only works with references or pointers in C++, so you'll want to create a print function that takes in a Shape& or a Shape*. Your print function should look like one of these two functions:

```
void print(const Shape &s)
{
}

void print(const Shape* s)
{
}
```

You will need to prove that your print function works for both Rectangles and Trapezoids, so in your main() make sure that you create a Rectangle and a Trapezoid and pass them both to your print function.

```
void print(const Shape* s)
{
   //implement this function to print out its type and perimeter
}

int main()
{
        Rectangle r(8,12,98,100); // (height, width, x, y)
        print(r);
        //This will display the message:
        //1. I am a Rectangle! , perimeter=40

        Trapezoid t(100,100,10,5,20,15,10); //(x, y,e1,e2,e3,e4,height)
        print(t);
        //This will display the message:
        //1. I am a Trapezoid! , perimeter=50
}
```

main.cpp