CSCE 240 – Programming Assignment Six

Due: 11:59pm on Friday, December 8

Purpose – Implement three classes to output geometric shapes in the *CSCE240_Program6* namespace

PixelShape

Create a *PixelShape* class that has two non-static private data members: a string for the name of the shape, and a char for the pixel character used to draw the shape. The name can hold any string of positive length. The pixel can be any character with an ASCII value between 33 and 126, inclusive.

The *PixelShape* class will contain the following public member functions:

A constructor with parameters for the name and the pixel character. The constructor's should have default arguments of "?" and '*'.

A virtual destructor.

A *SetName* mutator function that takes a string argument and updates the name data member to the value of the argument if the argument is a positive length string. It should leave the object unchanged if the argument is an empty string.

A *SetPixel* mutator function that takes a char argument and updates the pixel character data member to the value of the argument if the argument is a valid pixel character. It should leave the object unchanged if the argument is not a valid pixel character.

A *GetName* accessor function.

A *GetPixel* accessor function.

A virtual *Print* function that takes a bool parameter that specifies whether or not to fill the interior of the shape with pixel characters. The bool parameter should have a default argument of true. The Print function should output the name of the shape only to the standard output device.

A pure virtual overloaded *= operator that takes a double as its right operand and returns a *PixelShape* reference. Do not add this operator until you've fully tested all of the functions described above.

Initial tests for are provided in *testPixelShapeName.cc* and *testPixelShapePixel.cc*. These tests should be run before the addition of the overloaded *= operator. If you place the provided *makefile*, *testPixelShapeName.cc*, *testPixelShapePixel.cc*, and your *pixel.h* and *pixel.cc* files in the same directory, you can run the initial tests by typing the following commands at the command prompt

make testPixelShapeName

make testPixelShapePixel

You are encouraged to create more rigorous tests.

RightIsosceles

Create a *RightIsosceles* class as a child of the *PixelShape* class. The *RightIsosceles* class should have an integer for the lengths of the legs of the right isosceles triangle. The value of the data member must be at least 2 (so that it will look like a right isosceles triangle when printed). Note: the name (data member inherited from *PixelShape*) of a *RightIsosceles* object is "right isosceles triangle"

The *RightIsosceles* class will contain the following public member functions:

A constructor with an integer parameter for the length of the triangle's legs, and a parameter for the pixel character. The constructor should have default arguments of 2 and '*'.

A virtual destructor.

A *SetLeg* mutator function that takes an integer argument and updates the leg data member to the value of the argument, if the argument is at least 2. It should leave the object unchanged if the argument is less than 2.

A *GetLeg* accessor function.

A virtual overloaded *= operator that takes a double as its right operand and returns a *RightIsosceles* reference (a reference to the left operand). The operator should multiply the value of the leg data member by the right operand as long as the resulting value is at least 2. If multiplying the leg by the right operand would result in a leg that is smaller that 2, the operation should leave the object unchanged.

A virtual *Print* function that takes a bool parameter that specifies whether or not to fill the interior of the shape with pixel characters. The bool parameter should have a default argument of true. The Print function should output the name of the shape on one line, followed by a pixel drawing of the triangle with the right angle of the triangle in the bottom left corner. Each pixel character should be followed by a space. For example, if the leg of object t is 4, the function call t.Print(true) should output the following to the standard output device (using cout)

```
right isosceles triangle
*
* *
* * *
* * * *
```

if the leg of object t is 5, the function call t.Print(false) should output

```
right isosceles triangle
*
* *
*   *
*     *
* * * * *
```

Initial tests for all members except the Print function are provided in *testRightIsoscelesName.cc* and *testRightIsoscelesLeg.cc*, and *testRightIsoscelesTimesEquals.cc*. If you place the provided *makefile*, *testRightIsoscelesName.cc* and *testRightIsoscelesLeg.cc*, and *testRightIsoscelesTimesEquals.cc* and your *pixel.h*, *pixel.cc*, *rightisosceles.h*,

and *rightisosceles.cc* files in the same directory, you can run the initial tests by typing the following commands at the command prompt

make testRightIsoscelesName

make testRightIsoscelesLeg

make testRightIsoscelesTimesEquals

You are encouraged to create more rigorous tests.

Rectangle

Create a *Rectangle* class as a child of the *PixelShape* class. The *Rectangle* class should have integers for the length and width of the rectangle. The length and width must be positive integers. Note: the name (data member inherited from *PixelShape*) of a *Rectangle* object is "rectangle"

The *Rectangle* class will contain the following public member functions:

A constructor with integer parameters for the rectangle's length and width, and a parameter for the pixel character. The constructor should have default arguments of 2, 1, and '*', respectively.

A virtual destructor.

A *SetLength* mutator function that takes an integer argument and updates the length data member to the value of the argument if the argument is positive, and leaves the object unchanged if the argument is not positive.

A *GetLength* accessor function.

A *SetWidth* mutator function that takes an integer argument and updates the width data member to the value of the argument if the argument is positive, and leaves the object unchanged if the argument is not positive.

A *GetWidth* accessor function.

A virtual overloaded *= operator that takes a double as its right operand and returns a *Rectangle* reference (a reference to the left operand). The operator should multiply the values of the length and width data members by the right operand as long as the resulting lengths and widths are at least 1. If multiplying the length and width by the right operand would result in a length or width that is less than 1, the operation should leave the object unchanged.

A virtual *Print* function that takes a bool parameter that specifies whether or not to fill the interior of the shape with pixel characters. The bool parameter should have a default argument of true. The Print function should output the name of the shape on one line, followed by a pixel drawing of the rectangle using the length as the vertical distance (number of lines) and the width as the horizontal distance. Each pixel character should be followed by a space. For example, if the length and width of object r are 4 and 3, the function call r.Print(true) should output the following to the standard output device (using cout)

```
rectangle
* * *
* * *
* * *
* * *
```

if the length and width of object r are 3 and 5, the function call r.Print(false) should output the following to the standard output device (using cout)

```
rectangle
* * * * *
*       *
* * * * *
```

Initial tests for all members except the Print function are provided in *testRectangleName.cc*, *testRectangleSides.cc*, and *testRectangleTimesEquals.cc*. If you place the provided *makefile*, *testRectangleName.cc*, *testRectangleSides.cc*, and *testRectangleTimesEquals.cc*, and your your *pixel.h*, *pixel.cc*, *rectangle.h*, and *rectangle.cc* files in the same directory, you can run the initial tests by typing the following commands at the command prompt

make testRectangleName

make testRectangleSides

make testRectangleTimesEquals

You are encouraged to create more rigorous tests.

Initial Testing for the Print Functions

Initial tests for the virtual print functions are provided in the attached *testVirtualPrints.cc* source file with the expected output the for various tests provided in *expectedOutput1.txt*, *expectedOutput2.txt*, *expectedOutput3.txt*, and *expectedOutput4.txt*. Your output is compared to the expected output by *checkit.cc*.

If you place the provided *makefile*, *testVirtualPrints.cc*, *checkit.cc*, *sampleinput.txt*, *expectedOutput1.txt*, *expectedOutput2.txt*, *expectedOutput3.txt*, and *expectedOutput4.txt* and your *pixel.h*, *pixel.cc*, *rightisosceles.h*, *rightisosceles.cc*, *rectangle.h*, and *rectangle.cc* files in the same directory, you can run the initial tests by typing the following commands at the command prompt

make testPrint1

make testPrint2

make testPrint3

make testPrint4

You are encouraged to create more rigorous tests.

Specifications

- Add all code for the definition of the *PixelShape* class in a header file named *pixelshape.h*.
- Include all of the necessary code for the *PixelShape* class function implementations in a source file name *pixelshape.cc*
- Add all code for the definition of the *RightIsosceles* class in a header file named *rightisosceles.h*.
- Include all of the necessary code for the *RightIsosceles* class function implementations in a source file name *rightisosceles.cc*
- Add all code for the definition of the *Rectangle* class in a header file named *rectangle.h*.

- Include all of the necessary code for the *Rectangle* class function implementations in a source file name *rectangle.cc*
- All three classes (*PixelShape*, *RightIsosceles*, and *Rectangle*) must be added to the *CSCE240_Program6* namespace.
- You will submit a zip file (only a zip file will be accepted) containing *pixelshape.h, pixelshape.cc, rightisosceles.h, rightisosceles.cc, rectangle.h,* and *rectangle.cc* to the assignment in Blackboard.
- Source files must compile and run on a computer of the instructor's choosing in the Linux lab (see your course syllabus for additional details).

Grade Breakdown

Style *pixelshape.h*: 0.2 points

Style *pixelshape.cc*: 0.2 points

Style *rightisosceles.h*: 0.15 points

Style *rightisosceles.cc*: 0.15 points

Style *rectangle.h*: 0.15 points

Style *rectangle.cc*: 0.15 points

Documentation: 1 point

Clean compilation of *pixelshape.cc*: 0.4 points

Clean compilation of *rightisosceles.cc*: 0.3 points

Clean compilation of *rectangle.cc*: 0.3 points

PixelShape contains a virtual destructor: 0.4 points

Passes instructor's modified *testPixelShapeName.cc* tests: 0.5 points

Passes instructor's modified *testPixelShapePixel.cc* tests: 0.5 points

Passes instructor's modified *testRightIsoscelesName.cc* tests: 0.5 points

Passes instructor's modified *testRightIsoscelesLeg.cc* tests: 0.5 points

Passes instructor's modified *testRightIsoscelesTimesEquals.cc* tests: 0.6 points

Passes instructor's modified *testRectangleName.cc* tests: 0.5 points

Passes instructor's modified *testRectangleSides.cc* tests: 0.5 points

Passes instructor's modified *testRectangleTimesEquals.cc* tests: 0.6 points

Passes instructor's modified virtual print test 1: 0.6 points

Passes instructor's modified virtual print test 2: 0.6 points

Passes instructor's modified virtual print test 3: 0.6 points

Passes instructor's modified virtual print test 4: 0.6 points

The penalty for late program submissions is 10% per day, with no submission accepted after 3 days.