CS162
Programming Assignment 2 - Ship Class with SFML

The purpose of this assignment is to give you an introduction into using SFML to create an interactive graphics-based program.

Overview

This program is part two of many that will implement an Asteroids-like game. Note: Asteroids is a registered trademark of Atari, Inc.

In this program you'll add code to your ship class to draw it, and use this class within a simple SFML game framework. The result will be a ship that you can "fly" around the screen in similar form to that in the game Asteroids.

Details

Step 1 - Create an SFML-based project that contains four source file (three of which are from your last project):

- vector.h - definition of the Vector struct (which contains an x and y coordinates, as floats)
- ship.h - class definition (interface)
- ship.cpp - class implementation
- SpaceRocks.cpp - NEW! (name it what you want)

Step 2 - Add the code given below to the SpaceRocks.cpp file. This is a simple game processing loop that gets user input (events and keyboard key presses), updates your ships status accordingly, and draws it on the screen. 120 times a seconds as written (updates the properties that fast anyway)….

Step 3 - Add a function to the ship class to draw it on the screen. This will need to accept a reference to a RenderWindow object a parameter. A simple version of what this function could look like is given below. Feel free to modify or improve it. (see step 6)

Step 4 - Tweak the code to make it work well and clean it up.

- There are a number of constant values we unwisely embedded into the ship class and main routine that it would probably be wise to pull out and define as actual constants at the top of their respective source files. These values, along with the framerate,

regulate how slow/fast the ship responds to input and will probably need some adjusting.

- We also used floats in the ship class and the SFML library uses doubles for everything so you'll get a bunch of conversion warnings. I recommend doing a project-wide search and replace to change all occurrences of "float" to "double". Visual Studio provides this ability, because things like this happen...

Step 5 - Thoroughly test the code. Make sure when the ship flies off one side it immediately reappears on the other side. Ditto for going off the top and bottom. Also verify you can spin continuous 360 rotations without any unusual movement when you pass through the 360/0 boundary (test both directions of rotation).

Step 6 - Make it your own. Some possible modifications and/or improvements to the ship are the addition of a visible "flame" or sound effects when the engine is firing (SFML provides support for sound)

---

main() routine that implements a simple game loop

```
#include <SFML/Graphics.hpp>

//========================================================================
==
int main()
{
	sf::RenderWindow window( sf::VideoMode(WINDOW_WIDTH, WINDOW_HEIGHT),
		"Delta Quadrant", sf::Style::Titlebar | sf::Style::Close);

	window.setFramerateLimit(120);
	// this causes loop to execute 120 times a second at most.
	// (a delay is automatically added after screen is drawn)

	Ship ship;
	//ADD Code to set limits on ships location (call setMaxLocation);
	//ADD CODE to position ship in middle of screen (call setLocation)

	while (window.isOpen())
	{
		//-------------------------------------------------------
		//handle user input (events and keyboard keys being pressed)
		sf::Event event;
		while (window.pollEvent(event))	{
			if (event.type == sf::Event::Closed)
```

```
                  window.close();
            }

            if (sf::Keyboard::isKeyPressed(sf::Keyboard::Left))
                  ship.rotateLeft();

            //ADD CODE HERE to rotate ship right when right arrow is pressed.

            //ADD CODE HERE to applyThrust when up arrow is pressed.


            //------------------------------------------------------------
            //draw new frame
            window.clear();

            //draw ship
            //UNCOMMENT THE FOLLOWING LINES after ship.draw() is implemented
            //ship.updateLocation();
            //ship.draw(window);

            //redisplay window
            window.display();
      }

      return 0;
}
```

Simple ship.draw() implementation

```
//-------------------------------------------------------------------------
// Function: draw
// Description: draws the ship on the given window
// Parameters:
//    win - the window on which we'll draw the ship
//-------------------------------------------------------------------------
void Ship::draw(sf::RenderWindow& win) {
      // draw ship
      sf::ConvexShape ship;
      ship.setPointCount(3);
      ship.setPoint(0, sf::Vector2f(10, 0));
      ship.setPoint(1, sf::Vector2f(0, 25));
      ship.setPoint(2, sf::Vector2f(20, 25));

      sf::Vector2f midpoint(10,15);
      ship.setOrigin(midpoint);

      ship.setFillColor(sf::Color(0, 0, 0));
      ship.setOutlineThickness(1);
```

```
      ship.setOutlineColor(sf::Color(255, 255, 255));

      ship.setPosition(location.x, location.y);
      ship.setRotation(angle);
      win.draw(ship);
}
```

Very Important Stuff

All programs should follow the class's coding conventions.
Submit the following:

- A zip file containing the four source files you created and your executable