CS250 Assignment 5: Moving Circles

Date assigned: Friday, April 5, 2019
Date due: Friday, April 12, 2019
Points: 45

Goals for Assignment 5

1. Read and use existing code.
2. Write an object oriented program using multiple classes.
3. Practice with basic 2D graphics in SDL.

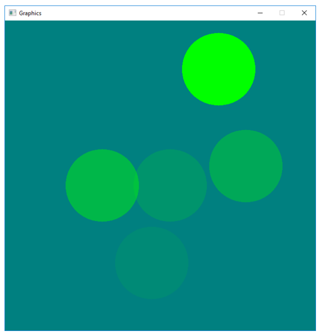
For this assignment, you will be writing a C++ / SDL program to display multiple moving circles to the screen.

Input:

The input to your program will be a text file consisting of one or more lines. Each line contains information about a circle (x, y, radius, R, G, B, A, direction, speed). Note that RGBA represent colors (R-Red, G-Green, B-Blue, A-alpha or transparency). Each RGBA value is a number between 0-255. You can use an online color picker to make colors (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Colors/Color_picker_tool).

Sample input file:

```
100 100 75 0 255 0 255 E 10
200 200 75 0 255 0 110 S 1
300 300 75 0 255 0 80 E 2
400 400 75 0 255 0 40 NW 1
500 500 75 0 255 0 20 E 3
```



Output:

See video on Grace and screenshots on right.

Setting up your solution:

1. Your starting point for this assignment is (05MovingCircles), which is in the public folder on Grace. Copy this folder to your desktop and rename it to (05MovingCircles-PUNetID). The solution contains two projects:
   a. SDLManager: contains the code that handles drawing to the screen. YOU MUST NOT MODIFY/CHANGE ANYTHING IN THIS PROJECT.
   b. Graphics2D: is where you will write the code necessary to display the moving circles as described above. This project contains the following:
      i. **Circle.h:** class that represents a single circle with x and y coordinates and a radius. DO NOT MODIFY.
      ii. **MovingCircle.h:** class that is derived from Circle and represents a moving circle with a Direction, speed, and the bounding box of movements. DO NOT MODIFY.
      iii. **Direction2D.h** and **Direction2D.cpp**: class that represents the direction of movement. The directions are N, S, E, W, NE, NW, SE, SW. This class contains functions to get the x and y coordinates after a movement and functions to reflect off of all four sides of the window. DO NOT MODIFY.
      iv. **Graphics2DDriver**: contains main, which sets up the window and handles the game loop. YOU MUST MODIFY.

   v. **circles.txt:** text file containing the specifications for the circles. This file will have a maximum of 50 circles and will not contain any errors. The file will not end with a blank line.

2. Build and run the program. Make sure that there are no warnings or errors before moving on.

Write the code:

- Add a file Circle.cpp to Graphics2D and write the code that implements Circle.h.
- Test the class Circle by writing some test code in Graphics2DDriver that creates an object of type Circle and uses the functions read() and draw().
- Add a file MovingCircle.cpp to Graphics2D and write the code that implements MovingCircle.cpp.
- Test the class MovingCircle by writing some test code in Graphics2DDriver that creates an object of type MovingCircle and uses the functions setMovementBoundingBox(), read(), move(). You need to make sure that your circle bounces off of the sides in all directions.
- Delete all of the test code from Graphics2D.
- Write the code in Graphics2DDriver that will read circles from the file circles.txt and draw them to the screen.

To complete this assignment you must submit the following:

1. An electronic copy of your program on Grace

a) You need to follow the coding standards from the CS250 class site. Please be sure to read the coding standards.
b) Pay attention to the example output. Your program's output must look **exactly** like the sample output.
c) Make sure that your program builds without errors & warnings and runs correctly. If you get any errors or warnings, double check that you typed everything correctly. Be aware that C++ is case-sensitive. You will lose 10% if there are any warnings and 40% if your program does not build successfully.
d) Once you are sure that the program works, it is time to submit your program. You do this by logging on to Grace and placing your complete solution folder in the **CS250-0X Drop** folder.
e) The solution must be in the drop folder by the time class starts on the day the assignment is due. Anything submitted after that will be considered late.

2. A hard copy of your program

a) The hard copy must be placed on the instructor's desk by the time class starts on the day that it is due.
b) The hard copy must be printed in color, double-sided, and stapled in the upper left corner if your solution contains multiple pages.
c) Your tab size must be set to 2 and you must not go past column 80 in your output.

Remember, if you have any problems, come to me straight away
with your project on Grace. Good Luck!!!! ☺

```
//****************************************************************************
// File name:  Direction2D.h
// Author:     Computer Science, Pacific University
// Date:       4/6/2018
// Class:      CS 250
// Assignment: SDL_MovingCircle
// Purpose:    Represent a two-dimensional direction
//*****************************************************************************

#pragma once

#include <iostream>

class Direction2D
{
public:

  enum CompassDirection {
    N, NE, E, SE, S, NW, W, SW, X
  };

  Direction2D (CompassDirection = N);

  Direction2D reflectOnLeftSide () const;
  Direction2D reflectOnRightSide () const;
  Direction2D reflectOnTop () const;
  Direction2D reflectOnBottom () const;

  int getXMove () const;
  int getYMove () const;

  CompassDirection getCompassDirection () const;

  friend std::istream& operator>> (std::istream& rcInStream, Direction2D
&rcDirection);

private:
  CompassDirection meCompassDirection;
  int mXMove;
  int mYMove;

  void setCompassDirection (std::string compassDirection);
  void setMoveByCompassDirection ();

  static const int NUMBER_OF_DIRECTIONS = 8;
  static const int NUMBER_OF_SIDES = 4;
  static const int NUMBER_OF_COORDINATES = 2;


  enum Side {
    TOP_SIDE, RIGHT_SIDE, BOTTOM_SIDE, LEFT_SIDE
  };

  static const std::string CompassDirectionNames[];
  static const int MoveLookup[][NUMBER_OF_COORDINATES];
  static const int ReflectLookup[][NUMBER_OF_SIDES];
};

```

```
//*************************************************************************
// File name:  MovingCircle.h
// Author:     Computer Science, Pacific University
// Date:       4/3/2019
// Class:      CS 250
// Assignment: MovingCircles
// Purpose:    Build a class to represent a moving circle
//*************************************************************************
#ifndef MOVINGCIRCLE_H
#define MOVINGCIRCLE_H

#include "SDLManager.h"
#include "Circle.h"
#include "Direction2D.h"

using namespace std;

class MovingCircle : public Circle {

public:
  MovingCircle (int xCenter = 100, int yCenter = 100, int radius = 10,
                const Color &rcColor = Color::BLACK,
                const Direction2D &rcDirection = Direction2D::N,
                int speed = 1);

  bool move ();

  void setDirection (const Direction2D &rcDirection);
  Direction2D getDirection () const;

  void setSpeed (int speed);
  int getSpeed () const;

  void setMovementBoundingBox (int xMin, int yMin,
                               int xMax, int yMax);

  bool read(istream &rcInStream);

private:
  // bounding box of movement
  int mXMin = 0;
  int mYMin = 0;
  int mXMax = 640;
  int mYMax = 640;

  Direction2D mcDirection;
  int mSpeed;
};

#endif
```

```cpp
//***************************************************************************
// File name:  Circle.h
// Author:     Computer Science, Pacific University
// Date:       4/4/19
// Class:      CS250
// Assignment: SDL Simple Circle
// Purpose:    Specify the Circle interface for displaying a fixed circle on
//             the screen.
//***************************************************************************
#ifndef CIRCLE_H
#define CIRCLE_H

#include "SDLManager.h"
#include "Color.h"
#include <iostream>

using namespace std;

class Circle
{
  public:
  Circle(int xCenter = 10, int yCenter = 10, int radius = 10,
         const Color &rcColor = Color::getColor(Color::BLACK));
  void draw(SDLManager &rcManager) const;
  bool read(istream &rcInStream);

  protected:
  int mXCenter, mYCenter, mRadius;
  Color mcColor;
};

#endif
```