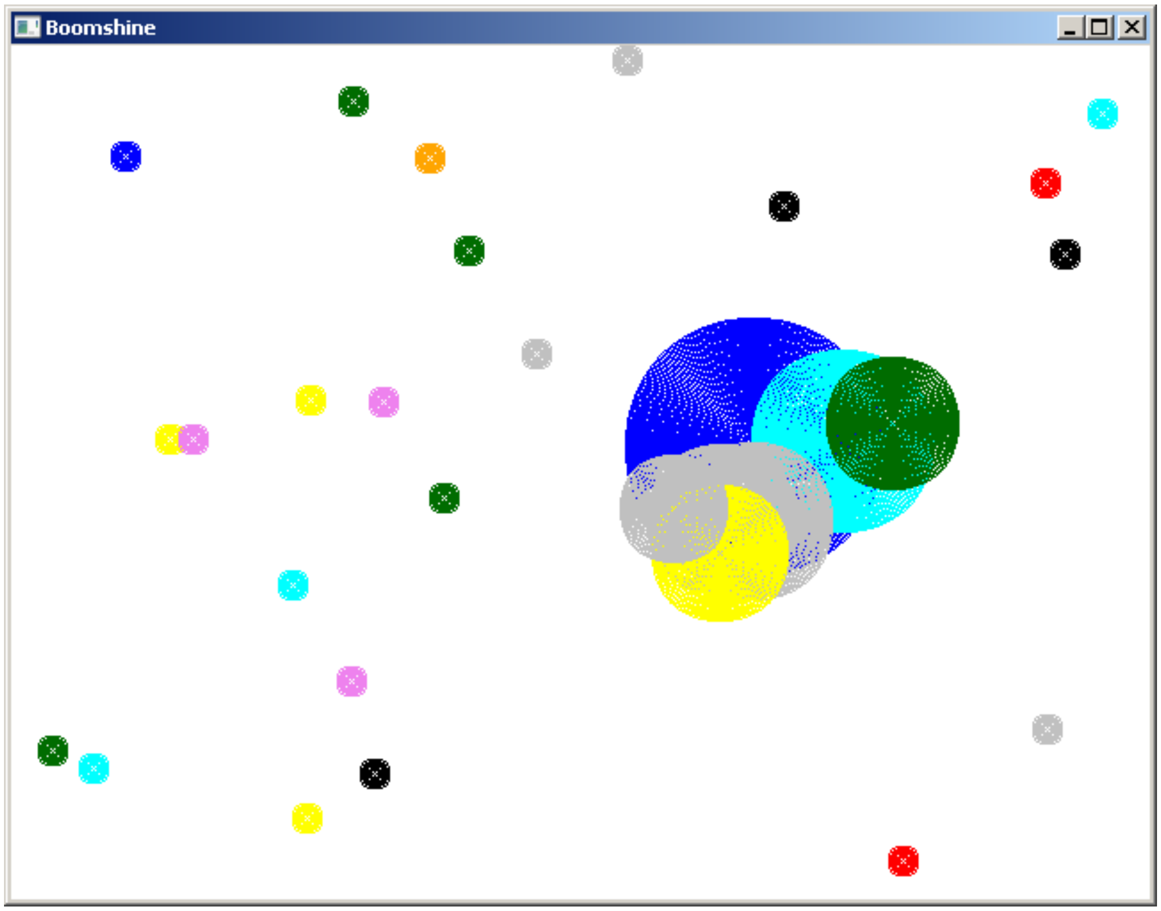
CS250 Assignment 5
Boomshine

Date assigned: Wednesday, April 1, 2015
Part 1 due: Wednesday, April 8, 2015
Part 2 due: Friday, April 17, 2015
Points: 60 (25pts for part1, 35 pts for part 2)

Boomshine (http://www.addictinggames.com/strategy-games/boomshine.jsp) is a single-user game where a player tries to place a circle on the screen such that the circle causes the intersection of as many moving circles as possible. Once the initial expanding circle is placed on the screen, if a moving circle intersects the expanding circle on the screen, the moving circle becomes an expanding circle that doesn't move and begins to intersect any moving circles.

The first fixed circle is placed on the screen for a fixed period of time and expands during a given time period. After the time period expires, the circle disappears and can no longer intersect any of the moving circles. If all fixed circles use up their allotted period of time, the game stops and reports on the number of moving circles that were intersected.

You are to write a project called **Boomshine** that implements the game just described. A video of the game in action is on the class web page. Below is a screen shot of the game in action.



Here are the steps you need to go through to successfully complete this assignment.

Part I:

1. In your CircleAnimation project, you are to implement the following interface for a new Circle class. I don't want you to mess with your existing Circle.h interface, which is why the class name is **ACircle**.

```
#ifndef ACIRCLE_H
#define ACIRCLE_H

#include "Color.h"

class ACircle
{

public:
  // x, y, radius, color
  ACircle (int = 10, int = 10, int = 10, Color = Color::AQUA);
  void setRadius (int);
  int getRadius () const;
  void setXCenter (int);
  int getXCenter () const;
  void setYCenter (int);
  int getYCenter () const;
  void setColor (const Color &);
  Color getColor () const;
  void draw () const;
  void drawFilled () const;
  bool intersectsWith (const ACircle &) const;

private:
  int mXCenter, mYCenter, mRadius;
  Color mColor;
};

#endif
```

2. Write the interface and implementation for a new class **MovingCircle** in the project CircleAnimation, which is a subclass of ACircle. A moving circle includes a speed and direction as additional attributes and behaviors that allow the setting/getting of the additional attributes as well as determining if a screen edge has been hit. If a screen edge has been hit, add a behavior that allows a circle to bounce off the screen edge.

3. Write a driver called **CircleAnimationDriver.cpp** that places 25 moving circles with a random: a) location completely on the screen, b) color, c) direction, d) radius (5 to 25 inclusive), and e) speed of 1 on the screen. All circles are to bounce off the edge of the screen when the edge of the circle encounters the edge of the screen. The bouncing is to be natural, so for instance, a circle moving northeast and intersecting the right edge of the screen will change direction to northwest.

Part II:

4. Write the interface and implementation for a new class **ExpandingCircle** in the project CircleAnimation that is a subclass of ACircle. An expanding circle expands uniformly to a certain size over a given period of time. Add an additional attribute for the amount of expanding time.

5. Create a project called **Boomshine** in your existing solution.

6. Write the interface and implementation for a class called **Boomshine**, which plays the game of Boomshine as previously described. Here are a few more details that are to be implemented in the game of Boomshine:

(a) The constructor for Boomshine is to accept an integer that specifies the level of the game being played. The number of moving circles initially moving on the screen is five times the level.

(b) All moving circles start out with a radius of 8 pixels.

(c) For the game of Boomshine, circles can only move in a diagonal direction (NE, SE, NW, and SW). This is not true for Part I, only Part II.

(d) Wait for the user to place a single expanding circle anywhere on the screen. The initial expanding circle has a radius of 8 pixels and expands by 1 pixel every iteration through the game loop for exactly 2 seconds (120 frames at 60 fps).

(e) If a moving circle intersects with an expanding circle, the moving circle becomes an expanding circle and expands by 1 pixel every iteration through the game loop for 2 seconds.

(f) After all expanding circles have exhausted their time, the game stops and the results are displayed as shown above.

(g) You will need to use functions _itoa and strcat to display integer values on the screen. Look these up.

7. Write the driver for Boomshine that creates a Boomshine object and uses the Boomshine functions to play the game of Boomshine.

To complete this assignment you must submit the following:

1. An electronic copy of your program on Grace

a) Add a project named **Boomshine** to your assignment solutions folder. It is vital that you name your solution and your project correctly!
b) Type your program (fully documented/commented) into the project. You need to follow the coding standards from the CS250 Web page. These coding standards have been modified to include additional C++ language features introduced in CS250, so please be sure to read the new coding standards. Make sure that you include the hours you worked on the assignment in your header comments.
c) 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 **CS250-01 Drop** folder.
d) 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.

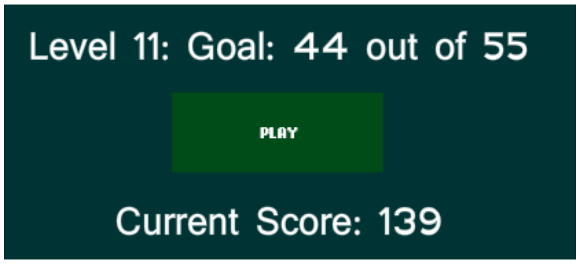
2. A hard copy of your program

a) The hard copy must be placed on the instructor's desk by the time class starts on the day that it is due. The hard copy must be printed in color, double-sided, and stapled in the upper left corner. Printing order is: ACircle.cpp, MovingCircle.h, MovingCircle.cpp, CircleAnimationDriver.cpp for Part I and ACircle.cpp, MovingCircle.h, MovingCircle.cpp, ExpandingCircle.h, ExpandingCircle.cpp, Boomshine.cpp for Part II.
b) 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 a flash drive or on Grace. Good Luck!!!! ☺

Extra Credit

The online game of Boomshine (http://www.addictinggames.com/strategy-games/boomshine.jsp) has multiple levels of play. I will give extra credit for anyone implementing levels of play as in the original game. That is, you are to implement the following from the original game including the PLAY again button.


Level 11: Goal: 44 out of 55

Current Score: 139

Here are the number of circles displayed and the number of circles that must be intersected to move to the next level. Yes, I played the whole thing!

Level	Circles Displayed	Intersected Circles Necessary
1	5	1
2	10	2
3	15	3
4	20	4
5	25	5
6	30	10
7	35	15
8	40	21
9	45	27
10	50	33
11	55	44
12	60	55