CS250 Assignment 7
Boomshine

http://www.k2xl.com/games/boomshine/

Date assigned:	Friday, April 13, 2018
Date due:	Friday, April 27, 2018
Points:	50

Goals for this assignment

1. Read and use existing code
2 Write an object-oriented program using multiple classes.
3. Use composition, inheritance, friends, and operator overloading.
4. Implement the Boomshine class and ExpandingCircle
5. Practice with basic 2D graphics in SDL.

Create a new project in punetid-Graphics named 07_Boomshine. You must link this project with the SDL libraries (see your notes from class) as well as with 05_Graphics2D.

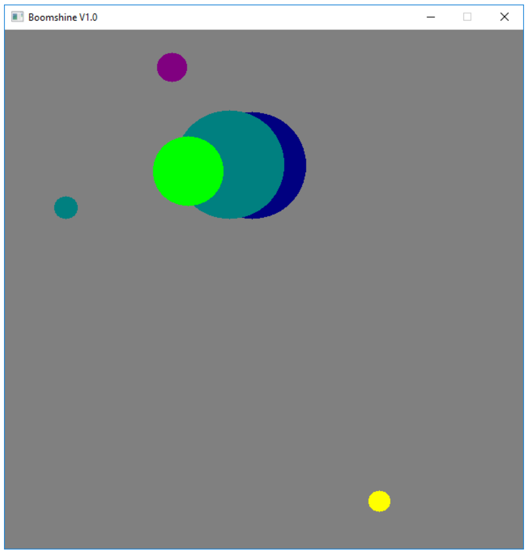
05_Graphics2D must contain ExpandingCircle.h and ExpandingCircle.cpp
07_Boomshine must contain Boomshine.h, Boomshine.cpp, and BoomshineDriver.cpp

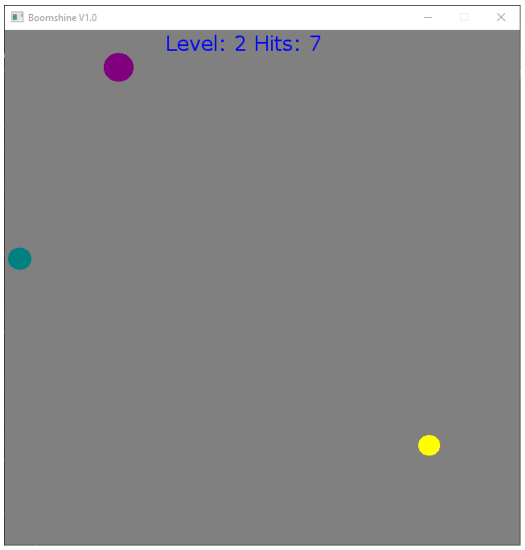
Boomshine

Boomshine (http://www.k2xl.com/games/boomshine) 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 **07_Boomshine** that implements the game just described. Below is a screen shot of the game in action. The window name is Boomshine V1.0.





Write the interface and implementation for a new class **ExpandingCircle** in the project **05_Graphics2D** that is a subclass of Circle. 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.

Write the interface and implementation in 07_Boomshine 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 (level can be 1 to 10 and the default level is 1).

(b) All moving circles start out at a random position, with a random radius of 10 to 19, a random direction, a random speed of 1 to 3, and a random color between SILVER and FUCHSIA.

(c) Wait for the user to place a single expanding circle anywhere on the screen. The initial expanding circle has a radius of 15 pixels and expands by 1 pixel every other 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 other 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. The text is displayed using 25-pt verdana.

(g) You will need to use functions to_string() to display integer values on the screen. Look this function up.

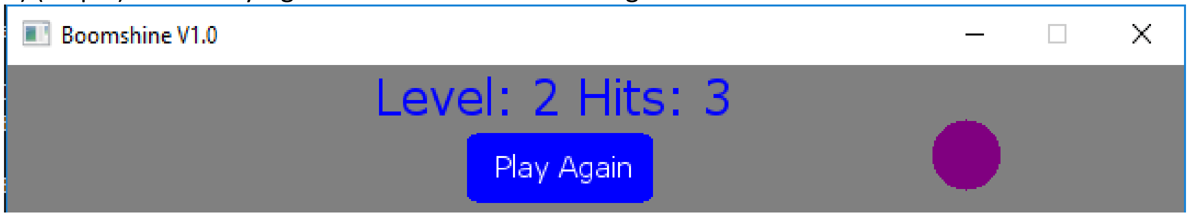
Write the driver for Boomshine that creates a Boomshine object and uses the Boomshine functions to play the game of Boomshine.

1. Allow the user to close the window using the x icon in the top right of the window.
2. Allow the user to restart the game by pressing the space bar.

Bonus

a) (+ 2 pts) Add a blast each time a moving circle collides with an expanding circle. Only one blast is to sound for each collision.

b) (+3 pts) Add a Play Again button that restarts the game when the user clicks inside the button.



Your project must be on time (both the electronic and hard copy) to receive any bonus points.

To complete this assignment you must submit the following:

1. An electronic copy of your program on Grace

a) Add a project named **07_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) Pay attention to the example output. Your program's output must look **exactly** like the sample output.
d) 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.
e) 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.
f) 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

The hard copy must be placed on the instructor's desk by the time class starts on the day that it is due. Print BoomshineDriver.cpp, Boomshine.h, Boomshine.cpp, ExpandingCircle.h and ExpandingCircle.cpp in that exact order.

a) **The hard copy must be printed in color, double-sided, and stapled in the upper left corner if your solution contains multiple pages**. Failure to print properly will result in loss of 4 points (10%)
b) Your tab size must be set to 2 and you must not go past column 80 in your output.

3. Modifications from the previous assignment can be found in Assign7Files

a) Implement the changes in MovingCircle.h which is breaking up move into 3 functions:
move – a simple move based on direction and speed
isEdgeIntersect – returns true if the object intersects any bounding box edge; otherwise, false
reflect – reflects properly off the intersecting edge
b) Use SDLManager 1.2 which adds sound and simple event-handling

Remember, if you have any problems, come to me straight away

with your project on Grace. Good Luck!!!! :)