Game Description
Dracula's Dungeon is a top down maze exploration and enemy avoidance game. The goal is to navigate the maze while avoiding the imposing monster. Dracula can sense your presence and he knows these dungeons much better than you. From the moment the game starts he will pesue you. Get moving, find the ladder up, and stay as far away from Dracula as you can.

How To Play
Use the arrow keys to move around, and walk through doors to move to different rooms.

Special features listed in each person's contributions.

Marc Herdman
I started by converting the Scene class into a pure virtual function. Then I made all menus and levels child classes of Scene. That allowed for a stack to be created in main and all calls in main to Scene call the top of the stack instead. Now when a menu is pushed to the stack in the game or main menu it will render that scene automatically and when it's done, it pops off the stack and the original scene is automatically rendered again. I made a button system with onMouseOver style highlighting to navigate the various menus of the game. Next I converted the texture class to allow for sprite sheets, scale, and pivot points. In addition, when rendering any sprite, flags can be set for flip and mirror. I used a recursive backtracking algorithm to build a random maze for each level and incorporated Dijkstra's algorithm to generate a path and manhattan distance to the player from every cell of the maze. That allowed the monster and exit to always start a reasonable distance from the player. It also allows the monster to hunt down the player.

Christian Vaughn
I programmed the visual effect used in our game. I started with a Stencil mask around the player, I enabled the stencil buffer in main.cpp and drew a mask which was a triangle fan to make a circle, and a triangle to act as a flashlight beam. Stencil buffers lacking alpha made it look very bad because most of the map was all black. I ended up using a shader that generates a gradient noise instead. This plus a timer in the code makes a semi-realistic fog effect that moves through the play screen. I had to go online and read many articles to figure out how to generate noise with an algorithm because my first approach of using a noise map texture for fog didn't look appealing. my quad with the shader somehow ended up being slightly larger than our game area, so I used the stencil mask I had set up to hide the fog that would float outside the game screen. I also programmed the sound and added button click, and footstep sound effects and

level music. I had to add a function to properly pause and unpause the music in the game so it would start and stop at the same spot. The last thing I did was make a death screen and win screen by modifying the pause menu popups.

Kyle Schuck

I created the level transition for the game. I used a sort of algorithm to randomly generate an exit staircase that would move the player to a new level and would never start nearby since the players location would be generated as well. I then updated the game to generate new levels each time an exit was reached. I worked a bit the player's starting position, and I made some additions to the printed text map in the console that would show where the exit spawns at with a 'e' representing the exit in an empty room, and an 'E' representing an exit where there was a southern wall, since that was the way our map printed. This was mostly done for testing and demonstration purposes to let us know the quickest way to the end. I worked a little bit between the player, maze and level files updating them as needed to accommodate.

Shaunt Der Haroutunian

I created the enemy (vampire) character that chases the player in the maze. I created the monster class that extends Marc's already created entity class. This allowed me to initiate the monster with the texture from a sprite sheet and add action animations to the vampire. I added a simple algorithm that would calculate the shortest path between the vampire and player (when they are both in the same room) so the vampire can chase down the player. I added collision detection for player death as well, so when the vampire gets close enough to the player the player dies and the death menu pops up.