Granit Arifi
CS 162
Final Project

Overview:

Design and implement a 1-player, text based game where the player can move through spaces to get items and accomplish goals. This is done using a Space class that has at least 4 pointers as member variables that point to other Space class objects. My game is a text-based dungeon crawler that is linear in fashion, i.e., you proceed from dungeon room to dungeon room, and collect keys to unlock the treasure room by defeating the monsters in each room.

Proposed Solution:

1) main.cpp – In main.cpp, the program creates a Game object and calls the play() method from it.

2) Character.cpp – Uses the Character class from Project 4 to implement the derived Player and Monster classes.

3) Player.cpp – This is derived from the Character class and contains specific methods for setting and getting the number of potions, as well as using potions.

4) Monster.cpp – This class is also derived from the Character class and is used to build out the monster the Player class does battle with throughout the dungeon.

5) Space.cpp – This Class is used to build out our derived Spaces like the Poison Space and the Freeze Space. It is an abstract class with pure virtual functions.

6) PoisonSpace/FreezeSpace/NoAirSpace/BossSpace – Derived classes from the Space class.

7) Game.cpp – Sets up our game by using pointers to our areas derived from the Space class as well as member variables to keep track of the next room, previous room, number of keys collected, and of course, the Player * hero variable which is our hero of the dungeon. A battle() method executes the attack and defense rounds between the hero and the monsters and the play() method runs the loop for the game as well as presenting menu options. The printSpaces() method prints out a visual layout of the dungeon to help the Player visualize what is happening. The spaceAction() method handles the logic for what each space should be doing.

8) validateInput.cpp/Menu.cpp – This is our main input validation function and Menu class that was also used in the previous labs and projects.

Test Plan

Test Case	Input Values	Driver Functions	Expected Outcomes	Observed Outcomes
Input for menu choice is too high/low	Input < 1 or Input > 4 or Input != integer	validateInput()	Asks user to enter a valid input within bounds	Asks user to enter a valid input within bounds or enter an integer
Input for menu choice is not an integer	Input != integer	validateInput()	Asks user to enter a valid input	Asks user to enter a valid input within bounds
Input for option to Play or Quit too low/high	Input < 1 or Input > 2	validateInput()	Asks user to enter a valid input within bounds	Asks user to enter a valid input within bounds
Input for Play or Quit is a non integer	Input =string/char/nothing/float/	validateInput()	Asks user to enter a valid input within bounds	Asks user to enter a valid input within bounds

Reflection:

Being the culmination of the course, this was an excellent project to use the skills we have learned throughout the course. At first I was not sure how to implement an Object that's pointing to 4 other objects because my first thought was a double linked list with 4 pointers instead of just head and tail, and that doesn't make any sense. After clarifications from the helpful TAs, I had the great idea of making a Diablo (my favorite PC game) themed dungeon crawler. With the recent debacle of the Diablo:Immortal announcement, I thought this was only appropriate. Although I am sure I did not do the series any sort of justice, I thought it was a really great exercise to be given the freedom we were given on this project. At first, it seemed daunting and approaching it like the projects was not a good idea as I was quickly overwhelmed. What helped was first sketching out a map of the dungeon and then deciding on the gameplay. I came up with collecting keys by defeating monsters and implementing a potion system to let you heal in between fights. Then I thought, why don't I add a little twist to each area, because why not. So I came up with theme-ing each area with special attributes such as poison damage over time and freezing the player from attacking every other time. Even with all the knowledge from the course, it was still tricky implementing all of this due to the freedom we had. This was a good thing because being able to solve open ended problems is a vital skill to have. Overall, the class and projects and especially this final project, was plenty frustrating but extremely rewarding in the end.