Patrick Dougan
Final
12/10/2019

Main class: will control the flow of the game.

Menu: will handle user input and output information about the game

Map: Will handle most of the program functionality. It will create the spaces and link them properly. It will store data about different positions on the map. It will handle the fight information when character steps on an enemy space. It will keep track of the step count before player loses. It will display the map at the start of every player turn.

space/empty/enemy/trap/treasure:empty, enemy, trap and treasure spaces will inherit from the space class. When inspected they will each have a different action. Traps will damage the player, empty spaces may contain health potions, traps will damage the the player and treasure space will check inventory to see if the player has won.

Character: Will be used to hold player details including inventory, health, and combat stats. Will also include players position.

Inventory: Inventory will be used to hold keys that the player gets from defeating pirates and inspecting the tiles

Key: Declares key object that will be held in inventory and be checked by the treasure space to see if the player can open the chest and win the game.

Die class will be used for the player fights and for randomizing starting tiles

Function Tested	Input Value	Expected Output	results
checkInt	Valid (1-7)	Moves to appropriate space, displays output, interacts, or exits game	Moves to appropriate space, displays output, interacts, or exits game
checkInt	-10, 20, 200000	Resets for next input	Waits for next input
checkInt	“1 1”	Moves left once	Moves left

Reflection: The most difficult part of this assignment was getting the map setup. It took me a while to figure out how to get the loops to work setting up. I kept running into a problem with the left pointer and up pointer not aligning when moving to a new row eventually I was able to work

out how to reset the pointers after each completed row although this was more verbose than I would have liked. Using the locate function with set integers instead of using roll would have been a good addition. Another challenge I faced was how to setup the map with 4 different spaces that I have created. I decided that the easiest way to do this would be to initialize one type and then randomly swap out smaller number of spaces after the map was created. I think this was a much better solution than trying to have the map::map function randomly add spaces while creating the map. I still do not know how I would have implemented this. The last issue I had was trying to decide what I wanted to implement the most. I ended up including things that I thought I could finish and left out features which would have been interesting but did not think I would have time to implement such as a difficulty setting.