CS 200 – Intro to Programming

Assignment 4 [Fall 2018]

Release Date: Sunday 4th November 2018, 2:00 PM

Due Date: ***Friday*** 16th November 2018, 11:55 PM

Please keep in mind the following guidelines:

- Do not share your program code with anyone.
- Do not copy code from the internet.
- If you receive any assistance, mention the part of code in which you received assistance.
- You must be able to explain any part of your submitted code.
- All submissions are subject to automated plagiarism detection.

Submission:

You have to submit all the .cpp files containing source code. Zip all .cpp files into one file named as <your8DigitRollNumber> .zip and submit the zip file.

Direct your questions only to *Talal Bin Yousaf* and *Wasiq Noor* for this assignment. A sheet named "Asgn_4_Queries" has already been made on the seating plan link which we'll keep updated with your latest questions. This sheet should be the first thing you refer when you have a question, email one of the TAs of your question is not answered there.

Role Playing Game (RPG)

(While reading this also refer to the file RPG.h)

Your goal in this assignment is to implement RPG game using the following description:

Description:

You are a hero who is lost in a world full of minions. Your objective is to explore the map and find the missing pieces of the divine **sword**. Once all pieces have been gathered, you can use them to craft the divine sword and defeat the **boss**.

You'll have to implement the following structures and classes:

- struct Character
- struct Item
- struct Node
- class GameManager

Character:

Your game has three characters type.

- Player
- Minion
- Boss

Each character has HP (hit points) which represent the health of a character. A character dies when his HP drops to 0. A character can have at max 100 HP. Each character has an array of Item structure in which the character can keep his items. Minions and Boss characters have fixed items in their inventory. Minions have a dagger (15 points) while Boss has a Scythe (80 points). Player starts his game with a Sword (20 points) and a Health Portion (20 points) in his inventory. Player's inventory can hold at max 6 items at a time. All characters start at 100HP

Item:

Items are divided into two categories

- Weapons
- Health Portions

Weapons can have the category type 1 while Health Portions have the category type 2. Health Portions are always self-casted (Player character can only use it on itself) while Weapons are always used for enemy characters. A Health Portion can only be used once. After use, it has to be removed from Player's inventory. Each item has its points which determine the effectiveness of the item. Weapons decrease the HP of the opposing character by the number of points that Weapon has. Health Portions increase HP. (If a Health Portion having points 30 is used by a Player with HP 60 then its HP should increase by 30 i.e new HP is 90). You can make items of your own. The pieces of divine sword have 0 points and you have to combine all three piece to craft The Divine Sword with 85 points.

Node: A node is a block of world which holds one character and one item at a time. Each node has a connection to 4 other nodes of the map. You can imagine your map as a layer (n x n) of such nodes where every node is connected to adjacent nodes. Each node has a position assigned to it on the map and has to be connected to other nodes based on the position. For example, node with (0,0) is connected to two other nodes (0,1) and (1,0). Similarly a node with position (4,4) is connected to nodes (4,3) (4,5) (3,4) (5,4). Keep in mind that you won't be generating all the nodes at the start of the game. A node will be created and connected once it has been explored by the player. (You'll have to implement the link function which links the two nodes)

Game Manager: As the name suggests, the game manager will be handling all your game functions. A game manager has a map which keeps track of the all the nodes explored by the player (which is basically what nodes have been created in the game), a head node which can be used to access any node of the game (you'll have to write a recursive function for that), a character node which is to reference the node which contains the player and an array of nodes which contains pointers to nodes in which minions are present. Game Manager has following functions you need to implement:

- **Constructor:** Constructor initializes the game. An n x n (7x7) array of strings has to be generated which arbitrarily determines where all the characters (Boss, Minions and Player) and items of the game are placed. Your game will have 4 minions, 1 Boss, 1 Player, 2 Health Portions (30 points), 1 Axe (40 points) and 3 piece of the divine sword (0 points). All positions are unexplored (i.e. initialized to 0) except for one where the player character is placed. A node is created for the player (playerNode) and is given that position which was marked explored. The head node also points to this node. Head node will not change during the game while playerNode and map will change as other nodes are created.
- **searchNode:** Given a position, this function should go through all the nodes which are created (use the head pointer just like a linked list to access nodes) and if a node of the given position exists, the pointer passed in the function arguments (pointing to NULL) should start pointing towards it. If the node is not found the pointer remain NULL. You are allowed to change this function but this function should be able to access any node of the map. Keep one thing in mind that you have to write a recursive function.
- **playerMove:** A player is able to move in four directions (except for the edges of the map). If a player is trying to move to a direction where no node exists a new nodes is created and is linked with the respective nodes (You have to use your searchNode to find the nodes which are to be linked with the new node and place the characters and items in the new node using your map array. Also

mark that new node as explored). If player is trying to move to a node which already has a character in it then player can either fight the character or cancel his move.

- **fight:** A player can only fight the other characters when it is trying to acquire their node (i.e. moving to their node). Both characters have three options.
    - Attack: Choose one of your weapons from inventory and attack the other player. Player has the option to use a health portion instead of using a weapon
    - Dodge: Each player has a 50% chance of dodging the other players attack
    - Defend: Your chance of dodging the other players attack falls to 0% but you get half the damage of other players attack

Each character has an attack phase and a defense phase. If one character is in attack phase then the other character is in defense phase. In the attack phase the character decides which item to choose while in defense phase the character decides whether to dodge the attack or to defend it. Each attack phase uses the function **useItem**. Boss and Minion characters arbitrarily choose to the items from inventory and arbitrarily choose whether to dodge or defend.

At the start of each phase, two functions **showEnemy** and **showPlayerStatus** are called which show the current health and items of each character. If the enemy dies the player is able to move to its node. However, if the player dies the game ends.

- **minionMove:** At the end of every playerMove, minions who have appeared on the map can move to the adjacent locations if they are explored by the player (minionMove should not create any new nodes) and there is no character in that node. If there is no possible locations for the minion to move it stays at its position. Boss character can not move.

- **dropItem:** You can drop an item in a node which has no item in it. (Change your inventory and its size accordingly)

- **<u>pickItem:</u>** You can pick up the items that are in a node. (Change your inventory and its size accordingly)

- **<u>craft:</u>** You can craft the divine sword once you have collected all the pieces. (Since there is no restriction on items you make your own item with different points and can make your own crafting system).

- **<u>displayMap:</u>** displayMap should display the nodes and items explored by the player (Should not display any character other than the player). For example, if has player has explored following positions on map:

	1					
	1	1		1		
		1	1	1		
				1		
				1		

Output should be:

```
1 _ _ _
1 1 _ 1
_ 1 1 1
_ _ _ 1
_ _ _ 1
```

(You'll also have to show the items and Player character, the example only includes the explored map)

- **<u>saveGame/loadGame:</u>** You'll have to implement the functions for saving and loading the game from a file. If a player loads the game, all his progress should be lost. You'll have to come up with your own way of writing to the file to save the game and loading from it.

- **<u>isPlayerAlive:</u>** Should return if the player is alive or not. (Use it when writing the main file)

<u>Main File:</u> You'll be writing a main file which will have a single instance of GameManager and User should be able to use the functions of GameManager to play the game. At any time, user should be able to move, use portions, if user has encountered an enemy character during move then the user should be able to give input for the fight, drop and pick items, save and load the game, craft items, check the current status of player and quit the game.