Project 1 Design

Analysis

**Use Case 1: Name Input**

Precondition: Program has begun

Description:

* The user will be asked to enter their name: “What’s your name?”
* The user may enter anything

Postcondition: The user will be given a welcome message with his name and then be outputted an options menu

**Use Case 2: Options Menu**

Precondition: The user has been prompted to enter their name and has entered it

Description:

* The user will be prompted multiple options
* The three options are (1) start a new game, (2) view high scores, (3) Quit
* If the user enters (1), then they begin a new game
* If the user enters (2), then a list of the 10 high scores will appear
* If the user enters (3), then they will quit the game
* If the user enters any other value, then they will be told that their answer is invalid and they need to enter another value

Postcondition:

* If the user enters (1), a new game will have begun
* If the user enters (4), the program is terminated
* If any other value is entered the user will return to the same options menu

**Use Case 3: In-Game Menu**

Precondition:

* The user has begun a new game

Description:

* At the beginning of each turn the user will be given several options
* The five options are (1) move, (2) read technical papers, (3) search for loose change, (4) view character, (5) quit the game
* If the user enters (1), the player moves one step in the grid, but risks an Encounter or a Puzzle.
* If the user enters (2), the player loses a fixed amount of time, but increases intelligence by a random amount.
* If the user enters (3), the player loses a fixed amount of time, but increases money by a random amount.
* If the user enters (4), A simple display shows the character attributes and current position in the hall.
* If the user enters (5), the game ends and a “You Lose” screen appears
* If the user enters any other value, then they will be told that their answer is invalid, and they need to enter another value

Postcondition:

* If (1) was chosen, the user will move one space and may face an encounter
* If (2) or (3) was chosen, the user will be brought back to the In-Game menu
* If (4) was chosen, the user will be brought back to the In-Game menu and their stats will be displayed
* If (5) was chosen, the user will be told that they lose, and the game will end
* If any other value is chosen, the user will return to the In-Game menu

Design

**Class 1: Player**

* The player class keeps track of all the attributes that relate to the player’s character including name, point total, and remaining steps. The member variables of this class are name, remainingSteps, time, money, and intelligence.

**Class 2: High Score**

* The high score class deals with the list of the top 10 high scores. It changes the list, prints the list, and writes the list to a separate file. The functions of this class are printList(), changeList(), writeList(), and viewList(). This class has two member variables, an array that contains the names of the players and array that contains the high scores of those players. This class uses elements of the Player class.

**Class 3: Encounter**

* The encounter class deals with the random encounters the player may experience when he/she chooses to move. It randomly decides which encounter occurs. The functions of this class are chooseEncounter(), choosePuzzle(), puzzleAward(), puzzle(), professor(), student(), move(), gruntWork(), gradePapers(), raise(), puzzlePunishent(). The only member variable is a string representing the players answer. This class uses elements of the Player class.

**Class 4: Menu**

* The menu class deals with all the menus that the player encounters. It contains functions for the welcome message, options menu, and in-game options menu. The functions of this class are welcomeMenu(), optionsMenu(), gameMenu(), move(), readPaper(), searchforchange(), viewCharacter(), Lose(), Win(), and startGame(). The only member variable is a string representing the players name. This class use elements from the Encounter, High Score, and Player classes

Testing

**Case 1: Name Entry Test**

* This function tests the users input when the program begins. The two conditions that are tested are the user entering a one word name and a two word name

**Case 2: Start Menu Test**

* This function tests the Menu that appears after the user enters their name. There is a test for when the user begins a new game, a test for when the user chooses to view the high scores, a test for when the user quits the game, and a test for when the user enters a value that does not have a function.

**Case 3: In-Game Menu Test**

* This function tests the Menu that appears after the user has begun a new game. There is a test for when the character moves, a test for when the character reads technical papers, a test for when the character searches for loose change, a test for when the character views the character, and a test for an invalid input

**Case 5: Puzzle Test**

* This function tests the puzzle that randomly occurs when the user chooses to move in the In-Game Menu. There is a test for a incorrect answer and for a correct answer

**Case 4: High Score Test**

* This function tests the method that updates the high score list. There is a test for when the game ends and the user has a score that isn’t high enough and a test for when the user has a score that is high enough and the list needs to be updated