USE CASES

**Answering riddles**

When the user is prompted to answer a riddle, they will be given a list of four choices. Answering with the right correct answer allows the user to gain stats such as money, intelligence and rarely even time. If the user enters an invalid choice the program prompts the user to enter a valid choice. Answering the riddle wrong however results in lowering of a stat. These stat changes will be from 1-3 and each riddle can only be asked only once per game.

**Main menu**

When the user starts the game, they are asked for their name. This is used to address the player during the game as well for if the user gets a high score. After the user enters their name they are welcome to the game and brought to the main menu where they have 3 choices. They can enter 1 which starts the game. They can enter 2 which views the list of the top 10 current high scores which shows the name and score of the past players. If there are not 10 top scorers then the list is printed followed by “-no more scores to show-“. After this they are brought back to the main menu. Finally, they can quit the game. Entering anything else besides these three options results in the game giving a warning about incorrect choice and returning to the main menu.

**Turns**

During the players turn the user can chose from between moving forward in the hallway, reading technical papers, looking for loose change, view your current stats and finally quitting the game. The first choice allows the user to take a step forward into the depths of the treacherous hallway where all types of crazy things can happen. The next option (reading technical papers) allows the player to boost their intelligence stat by a few points in trade off for some time. The third option (looking for loose changing) allows you to increase your money in exchange for some time. The fourth option allows you to view the current stats of the player. This does not use up any time. The final option allows the user to exit the game at any time. The users current score is not saved because of this. If the user inputs something that is not one of these options, the program gives a warning and allows the user to reinput their selection.

Classes and Functions

openScores

validInt

Struct

Puzzles

Handles the questions, answers, points given and stat changes.

Score

Holds the number of the score

And the name of the player

Classes

player (handles player stats)

changeIntelligence

getIntelligence

changeTime

getTime

changeMoney

getMoney

showStats

scores

readScore

sortScores

editScores

hallway (linked list of puzzles)

setup

events (handles the random events per turn)

roll

riddle

event

game(handle the main portion of the game)

gameOver

play

turn

menu(Handles the starting menu of the game)

mainMenu

showScores

Testing

**Case 1**

**Turns**

1) Move forward(takes time)

2) Read technical papers (boost intelligence, takes time)

3) Search for loose change (boost money, takes time)

4) View character

5) Quit the game

Please choose an action:

For the above situation entering anything (even strings!) will cause the program to output a warning and then ask for the user to reenter their choice. Entering 1-5 will allow the program to function as usual. If the user enters a valid response and then a space or illegal character anything after the valid input should be ignored.(eg entering 1 should give the same result as 1 1 1 1 1).

Entering 1 Should bring up a random event that effects the character. This takes time as well and can modify other stats.

Entering 2 Increases the players intelligence stat at the cost of the time stat.

Entering 3 Increases the players money stat at the cost of the time stat.

Entering 4 Displays the player current stats. This should **not use up any time**.

Entering 5 Exits the game. The high score is **not** saved.

**Case 2 : Reading and Writing Files for High Scores**

**Reading in a file with more than 10 scores.**

The program should read in the file for first 10 scores. Anything else will not be added.

**Reading in an empty file**

The program should not see anything since the file is blank.

**Reading a file that doesn’t exist**

The program should then create the top scores file so that the program can read it in.

**Writing to a file with 10 scores already**

When writing to a file with 10 scores the current score should be compared to the existing scores and the list should be updated to reflect this. The top ten of the eleven scores should be written to the file.

**Case 3: Modifying player stats**

During the course of the game the users, players stats should actually be changed. Thus, the player character created in the in the main menu must be updated each and every time an event happens otherwise the score will not change.

**Lowering stats**

Since the lowest score in the game is zero there must a check so that the users, stats never fall below zero even if they should (e.g if time is 1 but you lose 2 you should still have 0 time and not -1) Failure to do so could two scores to go below zero at the same time, thus resulting in a possible endless game (or until the last score goes below zero) which could result in a unfair high score.

**Case 4: Game Over**

The game should end as soon the player reaches the goal or one of the stats reaches zero. The game should also end if the user selects quit at any point. The game should be able differentiate between the user and system ending the game as well as which stat was lowered to zero.

**Case 5 Answering a riddle**

When encountering a riddle the player will have choice of 4 answers. Entering anything besides 1-4 will result in a character warning and prompting the player with the riddle again. If the user enters the correct answer then the stat associated with the riddle will be increased. Otherwise the stat associated with the riddle will be drastically decreased.