Rocco Piccirillo CMPT220 Project 2 Milestone

Abstract:

My project is one of the first games of our time: Hangman. The word game that you play with your friends when you want to pass some time waiting in a class or even at all. All you need is a canvas to write everything on, and a tool that suits all your wildest dreams. Though for this project, pen and paper cannot be used. This game is being brought to the computer screen. Thousands of words being randomized and selected for a user to guess. The user gets a certain number of guesses till the man, hangs. If wanted, you can continue playing the game once you win and/or lose. Instead of just a classic text-based game, this game will include graphics as well. Usually hangman is played with friends, but this game is a solo ride.

Introduction:

The motivation behind my work lies behind my interest in computer science. My high school did not have any computer science programs at all for any of the students to take. So not many kids were educated in this field. The only reason I ever really heard about it was because my brother majored in it and loved it. When I arrived at Marist College I started taking computer science courses and took a liking to it. The only problem I have been having is that it is much harder than anything I have ever taken in my past, since I am very new to it. I try and push myself, but I constantly have nights when I am googling or watching YouTube videos to help further my knowledge. There were so many projects I could have chosen from but didn't because, down the line it would have ended up with me just taking everything from a YouTube video. I want to be able to learn and prosper so I can have an easier time in the future, but if I start taking the easy way now I will be doing it forever and won't ever learn anything. This led to me to Hangman. Simple yet challenging for someone who is a complete rookie to Java. I really wanted to do steganography, but I found myself just watching video after video and not even being able to understand what it was I was doing. Hangman is something that I know I can do, I have been taking it step by step further researching and learning new bits of information.

Detailed System Description:

The most important and probably the hardest part of the system was importing the words into the file. I could have just pasted them all into java through an array list or an array but that would have taken forever. Also, at first, I had no idea how many words I wanted to place into the game. Right now, there are 1500 words, but I plan on trying to up that. How I did this was by using FileReader and dragged the text file named WordList which includes all of the words on my desktop. All FileReader really does for me is read and store the text file. The BufferedReader is more efficient becauses it buffers for the file. I use it pass the file from FileReader into BufferedReader. I then transfer all the words into a string that reads each line. Then run into a while loop that empties this string into an array list until the specified string is detailed. This is all done in a method, eventually it will be moved into a class but for now it has been sloppily placed in a method due to time restrictions. Originally, I wanted to do it in a linked list but this just didn't seem efficient or I was even looking into an SQL database but what is the point if my project is an application and not a database. Then I made another method but soon to be class for a randomizer. I made an int that is basically picking a random number between zero and 1500. This is then returned as a ArrayList of each value from one to 1500. An arraylist which stores all the words is then brought about and then a string named myWord has one word in it. The word

being one randomized word from the list of 1500. This then brings about an array named secretWord. SecretWord creates an array with the length of the word. This then runs a for loop that takes each letter from the string and separates it into each letter. For example, the word is dog, the for loop separates into an array known that would look like this, [d, o, g]. So, each individual char is split into its own index. This brings us to wordGuess which is the same array as secretWord but it then runs through a for loop and each index is made into an _. Which is what the user views on the screen. The game prints it that secretWord is hidden because it is each letter of the word the user is trying to guess, and wordGuess is printed on the word in _ to the length of the string, myWord. This leads us to the game. The game starts, you get promted with "Your word is "and an array of _'s depending on how long the word actually is. My program runs a while loop until you run out guesses and the youWin Boolean I created was true. The first prompt lets you guess a letter and this is where everything begins to branch. If you guess a letter correct, this takes you into a for loop that depends on the length of secretWord, until the length of this array is reached the for loop runs. So, if your guess is in a string of secretWord, the index of secretWord that matches WordGuess gets replaces with you guess. What that means, is that say your word is dog and you guess d. The game checks if it is in secretWord, it is so now it matches the index of secretWord thanks to the for loop and switches the first _ of _ _ _ and makes its d_ _. The game prints and repeats "Guess a letter:" but above it now reads "d_ _" and below "Guess a Letter:". But, it was not discussed what happens if you are wrong. If you are wrong it takes you to the else branch, so if a letter is not guesses. If you are wrong your number of guesses decreases. And the game prints you have however many guesses left and the same _ _ _ message with guess a letter. Though, if you run out of guesses an if statement, if your guesses left is zero, the game gives you a game over message and break's out of the while loop. If all your guesses are on par and you do get everything correct this leads to the final for loop which tests every time depending on if your secretWord array equals your wordGuess array. This then sets the youWin Boolean to true and it breaks the while loop, thus printing out a you win. And if you didn't win, it breaks out of this for loop and keeps the Boolean at false and breaks by setting the declared I at an incredibly high value.

Hangman.java
+ myList: ArrayList<String> + myWord: String + secretWord: String[] + WordGuess: String[] +convertWord: String +youWin: Boolean +guessesLeft: int
+ static WordBank(wordlist: ArrayList<String>) : ArrayList<String> + static randomizeWord(randomizer: ArrayList<String>): String

Requirements and Literature Survey:

The game isn't really addressing any problems. Which is why I mixed this with literature survey because it is incredibly short. The only real problem that my game is solving is what is the word, and can you guess what it is? The game is a test of mind. Strengthening your ability to guess which in life can be very importance. Say you're taking a test with multiple choice questions, you can use your guessing

skills and deduct the right answer. For the literature survey, there are a ton of different games like this that "solve that problem". Look anywhere, google "hangman game" and you will be able to find the game within seconds.

User Manual:

As of right now, the game is not in it's full potential. All you can really do with it is, receive a blank set of lines to show how many letters are in the word, then guesses for the word given to you, and you can input letters until the game is over as in you win or you lose. As time goes on the game will be upgraded to button clicking but for now you type in 'a' through 'z' until to find the hidden word. Once the game is over I plan to add a try again or exit the game. As well as more graphics. The graphics include a stick man that eventually adds limbs as your guesses begin to slim down. And just to tidy the game up in general. Also, with my main methods in my main class I plan on moving them over into separate classes. One for my file reader, and one for my randomizer.

Conclusion:

The goals accomplished by the system include but are not limited to:

Randomization – randomizes my array list of 1500 words and selects one

Reading a file and importing it to an ArrrayList<String>- done with bufferedReaader and FileReader

Turning a word into _'s depending on the length of a string- done with forloops

Correctly uses a while loop - ends game once guesses are depleted, or winGame boolean is true

References/Bibliography:

I have referred to multiple YouTube videos but there was only one that I took straight code from for this because I had no idea. I will cite that below. Everything else I did I just went to stackflow for a quick assist I will try to link some of my assistances that most heavily improved my code.

BrandonioProductions. "Learning Java: Part 15: Reading Text from Files." *YouTube*, YouTube, 27 Feb. 2012, www.youtube.com/watch?v=VwV1VKvtVtI.

"Retrieving a Random Item from ArrayList." *Java - Retrieving a Random Item from ArrayList - Stack Overflow*, stackoverflow.com/questions/5034370/retrieving-a-random-item-from-arraylist.