Report for Project 5

2. (a.)

There were a few obstacles that I faced in this project. First, I learned that it is best to try to avoid nesting multiple loops because keeping track of many counters and increments becomes difficult with an increasing number of loops. Also, for this project I had to learn how to use the debugger at specific points in my code to trace the values of specific variables. In addition, I also found it very difficult to trace my own code for manageOneRound, so I decided to include as many comments I could to detail every step of my implementation. Finally, I also encountered problems with testing my program through SEASNET so I had to fix my program according to the other compiler as well.

(b.)

The function isValidProbe verifies that the given probe word is 4 to 6 letters and lowercase

bool isValidProbe (cosnt char given[])

{

if the length of given is not between 4 and 6, inclusive

return false

for each character of given

if the character is not a letter or is not lowercase

return false

return true

}

The function manageOneRound plays one round of the game

int manageOneRound (const char words[][7], int nWords, int wordnum)

{

set numberOfTries to 1

infinite loop

if either nWords or wordnum are invalid

return -1

read in the probe word

validate the probe word and make sure it exists in dictionary

set rocks to 0

set pebbles to 0

for each character of the probe word

if the character matches in the secret word

increment rocks

fill two usedPosition arrays with the positions of where rocks exist

for each character of the probe word

if the character is already part of a rock, skip to the next character of the probe

word

compare the probe character against each character of the secret word

if the character in secret word is already part of a rock or pebble, skip to the next character in the secret word

if the probe character matches in the secret word

increment pebbles

increment the numTries at the end

print out the number of rocks and pebbles to the screen for the user

}

The main function plays the full game (playing multiple rounds and keeping track of statistics)

int main

{

create an array of C strings to hold the “dictionary”

load words into the array

validate if the words were loaded into the array

if words were not loaded

print as such to the screen for the user

prompt and receive number of rounds to play according to user

validate number of rounds

set the minimum, maximum, average, and number of tries to 0

for each round of the given number of rounds to play

print round number and give the user the number of letters

play the game using manageOneRound

update minimum, maximum, average, and number of tries at the end of reach run

clean up the output for the average to have 2 decimal places

print out the user’s statistics at the end of the round