Roxane Martin

CS 31

Professor Smallberg

14 November, 2019

Project 5

1. One of the obstacles I overcame was finding a way to cleanly check all the conditions that the trial word must meet. My solution for doing this as efficiently as I could was to create two Booleans: one that held whether the trial word was the correct length and one for whether it is in the wordlist. Having these two Booleans allowed me to do a cleaner check and only write the code to print the error message once. Another issue I had was figuring out how to not double count characters that appeared once in the mystery word and twice in the trial word. My solution was to make a temporary string that held the mystery word for each trial and change the character to ‘\*’ once it has been counted such that it cannot be counted again.
2. Below is the pseudo code for my program in the progression in which it executes:

main function:

set maxWords to number of words in words.txt

create wordList array

fill it using getWords and file path to words.txt, and set nWords to the number of words read into the array

if nWords is less than 1

print "No words were loaded, so I can't play the game."

if nWords is greater than 1 and less than maxWords

read in number of rounds

set up variables to hold stats: minimum, maximum, sum, current score

if number of rounds is negative

print error message and terminate program

if number of rounds is positive

loop through number of rounds

print round number

generate random int

print length of word at index of random in the word list, this word is the mystery word.

set current score to integer returned by calling playOneRound with word list, nWords, and the randomly generated number

when main function calls playOneRound, playOneRound runs like so:

if nWords is not positive or wordnum is less than 0 or greater than or equal to nWords

return −1

initialize score to 0

for loop of getting trial words and finding flowers and bees

get trial word from user

intialize boolean that will hold whether trial word meets length/case conditions

if trial word contains any uppercase characters

set trialCheck boolean to false

if trial word is not between 4 and 6 characters

set trialCheck boolean to false

intialize boolean that will hold whether trial word is in words array

if the trial word is in the words array

set contains boolean to true

if does not pass lower/length check

print error message

else if is not contained in array

print error message

else if trial conditions are passed, continue to counting flowers and bees

copy randomly chosen word into 'mystery' c string so that it can be edited

if trial word matches mystery word

break loop and exit

initialize flowers and bees to 0

loop through characters in trial word

if character in trial word matches character in mystery word at same index

increment number of flowers

set character at this index in mystery word to '\*' so it cannot be double counted

if character in trial word does not match character in mystery word at same index

loop through characters in mystery word

if character in trial word matches any character in mystery word

increment bees

set character at this index in mystery word to '\*' so it cannot be double counted

break out of while loop of characters in mystery word

print out total number of flowers and bees

increment score

return score to main function

returning to main function, the end of the program executes like this:

if current score is 1

print out “You got it in 1 try.”

else

print out “You got it in currentScore tries.”

update stats:

if current score is less than minimum

update minimum

if current score is greater than maximum, update maximum

update maximum

add current score to sum

set decimal precision to two decimal places

print out calculated average, minimum, and maximum scores

and now we’re all done!