Notable Obstacles:

Figuring out how to use multiple files all together.

Had to figure out how to make sure golds and silvers did not count twice, or if there is a single “a” in the target letter and 2 “a” in the probe word, how to make sure only one of those is counted as a silver.

For this, I removed the letters that already had a gold or silver attributed to them, that way they would be ignored regardless.

I made a function that removed characters and found the location of characters in a word to help me with this.

main(){

* declare an array of c strings
* declare word file name with path
* calls getWords()
* ask for number of rounds
* check if rounds are positive
* if rounds are 0, code will automatically play 0 rounds but
* will not release an error
* declares score, minimum, and maximum variables
* for loop for each rounds begins, plays for number of rounds input
  + outputs round number
  + generates random number and uses that to output how many letters the word is
  + starts round with word, using playOneRound()
  + gets round score
    - if is less than 0 ends function
  + checks and sets if new max or minimum value for round score
    - sets initial values if it is the first round
  + says how many tries it took
  + outputs average, minimum, and maximum

**int** playOneRound(**const** **char** words[][MAXWORDLEN + 1], **int** nWords, **int** wordnum){

* checks if wordnum is less than 0 or greater than the number of words taken from getwords
* declares amount of word tries to 0
* starts infinite loop
  + takes input for a probe word

compares it to solution word, if successful outputs the number of tries + 1 to get the correct result for number of tries

checks for lowercase, word length, and if it is in known word list

if not, returns to start of for loop

increments number of tries

gold and silver vars declared equal to 0

counts gold by checking each letter of the probe word with the same position of solution word

removes any gold letters from both words

counts silvers by checking each letter of the probe word with every remaining letter of the solution word

removes any silver letters from both words to avoid duplicates

outputs number of golds and silver

**void** removeChar(**char** string1[], **int** pos){

* iterates through for loop starting at position of character being removed and moves each character left
* sets last value to null character
* c string is one character shorter with desired letter removed

**int** lookupChar(**const** **char** string1[], **char** search){

* iterates through c string and returns position of letter that is being searched for
* if not found returns -1