1. The first notable obstacle I had to overcome was finding the right file path for my program to read in the text file to fill the valid words array. All in all, the whole set up portion was tough, but once I got the file path correct, everything was smooth sailing from there, but it wasn’t easy to find the file path because macs are weird. Implementing the actual program wasn’t that hard, the main obstacle was making the code base more organized. I created two functions to help me out: a legalProbe function, to check if the probe word is a valid word in the array, and a probing function, to check how many stars and planets are in a probing word. The most notable obstacle in the entire program was correctly implementing the probing function. There was a major issue outputting the correct number of planets. Using two for loops to check each of the arrays, you would sometimes count extra planets or fewer planets if there were more than one of the same planet letters or if planet was already found. I fixed this by using two boolean arrays to track whether or not a letter was a star and which letters were already used as planets.
2. legalProbe(probe array, validwords array, number of valid words)

Check if probe is between 4-6 letters

Print error message

Return false

Loop through probe

Check if probe is contains only letters and lowercase

Print error message

Return false

Loop through validwords

If probe is in validwords

Return true

Print error message

Return false

probing(probe array, target array)

Two bool arrays of length 6, all false, one for stars and one for planets

Two ints set to 0, one for stars and one for planets

Loop through target

Check if probe letter is the same as target letter

Add one to stars

Set stars array and planets array index to true

Loop through probe

If stars is true

Continue

Loop through target

If probe letter is in target and planets is false

Add one to planets

Set planets array to true

Break out of loop

Print stars and planets message

runOneRound(valid words array, number of valid words, word number)

If number of words is negative, or if the word number is negative, or if word number is greater than number of words

Return -1

Create probe array of size 101 and set number of attempts to 0

While the target word isn’t the probe

Prompt user for probe word

If the probe is legal

Run probe function

Add one to attempts

Return attempts

main()

Initialize c string array

Declare int of number of words and fill array

Prompt user for number of rounds to play

Create variables to hold total attempts, number of attempts, and maximum and minimum attempts

Loop through number of rounds

Generate a random number

Run one round and store the returned value into number of attempts

If number of attempts is greater than max, set new max

If number of attempts is less than min, set new min

Print score message