a. Obstacles

* The biggest obstacle was creating a function that would process the pebbles
  + The rock function was easy, but creating a function that would only process pebbles that weren't rocks or other pebbles was difficult
  + I was able to do so, by creating two c-strings that would be used to track pebbles and rocks, and check if a letter was already a pre-existing pebble or rock
* Also, creating two functions of type bool that would process to see if a word was a word in the word list, or if it was a proper word
  + It was challenging to work with this new topic, and using the various c-string functions that would process the words
  + It was different from arrays, and just getting used to the new methods took time
  + I was able to work through the bool functions by using c-string functions to check length and compare c-strings

b. Psuedocode

Prototypes : manageOneRound, isAllLower, isKnown, countRocks, countPebbles

main function

Declare array of C strings to hold list of words

set number of words

if number of words is out of range

cout no words loaded

prompt user for the number of rounds

if rounds is not positive

cout enter positive number of rounds

for loop (to iterate through each round)

call randInt to get random position

tell user the length of secret word

call manageOneRound to play a given round

Process the statistics for each round

process the min tries, max tries, and the average

cout the statistics

manageOneRound : Function to play each round

set conditions to play a valid round

initialize the secret word

while loop : loop through until the word is guessed

prompt user for probe word

if statement : set conditions so length is 4 to 6 and is lower case letter

if not valid word : cout word isn't valid

continue: start loop again

if word is known: check to see if word is present in word list

if not known: cout word isn't known

continue: start loop again

count the number of tries

if statement: check if probe word is equal to secret word

if so: break out of loop

create nRocks: call function countRocks

create nPebbles: call function countPebblees

cout rocks and pebbles

return number of tries

bool isAllLower : check to see if word is lower case

for loop

return false if an uppercase letter

return true if all lowercase letters

bool isKnown : check if word exists in word list

for loop

if statement: compare probe word to words in word list

return true if word found

return false if word isn't in word list

int countRocks: to count rocks

create a rock counter

for loop

if element at a given position in probe word is equal to element at a tiven position in secret word then it is a rock

return count of rocks

int countPebbles: to count pebbles

create a pebble counter

create two C-strings to track pre-existing rocks and pebbles

for loop

to check for rocks

if rock, then track the position number as 'r'

for loop : to iterate through probe word

to check for pebbles

for loop: to iterate through secret word

if statement: if not a rock or pebble

increment pebble counter

track position number as 'p'

return pebble counter