NOTE: The red writing are my notes I wrote down initially and the blue writing are the notes I wrote down as I was beginning to code and came across parts of the code that needed to be added still or changed.

Need to include necessary include<>, <iostream> for sure, statements and using namespace std, add throughout the writing of the code if necessary

Potentially need functions listed before main to simplify code, add as potential sections of code could use a function

Begin main function and list out variables. Add variables while progressing through code if necessary, need constant variable for number of wrong guesses so there is an option to lose

1. Greet the user welcoming them to the game

* Print out welcome statement
* Ask for user’s name and receive input use getline, include string lib
* Ask user if ready to start and receive answer getline
* Create loop to contain code that allows user to play game multiple times if desired, do while loop to allow input at end of loop to determine if loop iterates again, create bool variable to be the loop condition—if false, exits loop. If true, keeps looping

1. The program randomly selects a word from a list of 10 words with different lengths, you as the developer choose the words.

* Create a vector to store the chosen words and add words to vector before welcome statement, include vector lib
* Create random statement using the vector to have program select one of vector’s words
* change word to be printed out with \_ instead of letters, use for loop, create new variable to equal the selected word. Allows new variable to replace the letters with \_, but still allow original variable holding the word to remain untouched. User new variable in for loop to replace letters with \_

1. The program indicates to the user how many letters are in the word

* Get the size of the chosen word and print it out to show the user, print out word of \_ instead of letters to show user individual spaces as well

1. The user is asked to guess a letter

* ask user for a guess and receive answer

1. If the letter is in the word, the letter is displayed in the correct position of the word with all previously guessed correct letters

* if statement to find letter in the word, if found, update word contained by new variable from step 2 to replace the \_ with the correctly guessed letter, create new variable for the user’s guess to be assigned to a string variable to allow .replace (function to work), but still keeping the user’s guess as char
* probably use for loop, place if statement inside a for loop to allow program to find the guessed letter at specific index in the word.

1. If the letter is not in the word, display the letter indicating it is not in the word with all previously guessed letters that are not in the word

* If statement to determine if letter is not in word, print out explanation that the user’s guess is not in the word. Increment number of incorrect guesses in this if statement
* Create another vector, declare it at the top of main, to create list of missed letters. if guess is not in word, add it to this vector inside this if statement.
* Print out vector create function for printing this vector

1. The program displays how many guesses have been made, with how many correct and incorrect guesses.

* Keep track of total, correct, and incorrect guesses, adding one to each, each time the type of guess is made, reset each back to 0 after game finishes—declare the variables as 0 at beginning of do while loop—before user makes guesses. Did not do at first and each total kept going up when playing again. Increment total by 1 after while loop in step 6 begins, which is after user makes the first guess—total adds one after every guess. Increment number of letters found by 1 at end of if statement from step 4a.

1. The program continues to ask the user for guesses until all the letters in the word are guessed correctly.

* Create loop around if statements determining a correct or incorrect guess to allow user to keep guessing, loop begins after the first guess is made, once user reaches max amount of incorrect guesses, if statement with that condition is executed, user loses, the word is revealed and loop ends, stopping user from guessing more letters. Loop condition should be if the user has a smaller number of incorrect guesses than the max number of incorrect guesses, loop continues.
* Check before iterating through loop again to stop loop if each letter of word has been guessed correctly, use if statement to check if there are any \_ left in the word contained in the new variable. If not, user wins and loop breaks
* Print updated word contained by new variable from step 2 and ask user for another guess and receive input, both occur at end of while loop after all if statements

7. When all letters of the word are guessed correctly,

a. the program tells the user they have correctly guessed the word

* Print out statement to tell user they figured out the word—inside if statement above in step 6/last bullet point

b. and indicates the number of guesses it took

* Print the total guesses from step 5—inside if statement above in step 6/last bullet point

8. The program then asks the user if they would like to try again or quit

* Ask user to play again, receive answer, clear vector of the wrong guesses to be ready to let the user play again

a. If the user indicates they want to continue, the program chooses a different word randomly and the play continues

* If statement at start of overall loop from step 1 to take an affirmative user answer to let them play the game again

b. If the user indicates they want to quit, the program thanks them for playing and quits.

* If statement at the end of loop from step 1 to take a negative user answer and print out thanks for playing statement and ends program