# README File

Step 1. Start the Game.

Step 2. Initialize variables

* rounds=5(used to count the amount of games)
* userScore=0 (used to count up the userScore when attempting a word)
* playagain=’n’ (used as an exit point to the loop of continue game)
* bestword (left blank until the word with the most amount of characters will be stored to this variable)
* array myLetters[100] (as specified in question. 100 characters) see note\*\*
* random(0,99) (this is a random function to pick a random number between 0-99)
* RandomLetterCounter=0(used as a counter to generate 9 random letters)
* GameLetters (from the array list myLetters[100] the actual game letters for the user to create a word will be this variable)

Step 3. Is rounds > 0 (if yes proceed to step 4. If NOT proceed to step 15)

Step 4. Is random letter counter equal to 9 (it was initialized as 0. So therefor it must run a loop 9 times to choose 9 random letters)

Step 5. The variable position is created out of picking a random number between 0-99.

The 9 or[randomLettercounter] game letters are then generated from the array list myLetters[postion] from the random number generator 0-99. Once 9 letters the loop is equal to 9 (from the Random letterCounter ++). 9 letters are generated go to step 6.

Step 6. Print out the Game letters [position] i.e random(0,99)

Step 7 Ask the user to create a word from the 9 random letters

Step 8. Read in the users attempt to variable userword. IF the GameLetters characters contain more than one of the same characters from myLetters, print “The game Letters can only contain one of the same letters”. Go back to step 4. If the GameLetters do contain ONLY one of each letter from myLetters[100] array go to step 9.

Step9. A counter is set(set to 0) up to iterate over the users string input. Also a variable userword.length is set up to count the amount of letters

Step 10. Is the counter(of 0) equal to userword.length? if not got to step 11. If Yes go to step 13

Step 11. Is userword[counter] in game letters? (this is checking of the letters the user put in were all included in the game letters). If yes the counter increments by 1 to check every letter of the users word. If not go to step 12.

Step 12. If the user creates a word out of letters that were not included in the game letters(9 random letters) the output “INVALID WORD. TRY AGAIN” and reverts back to step 4 to regenerate 9 more random letters.

Step 13. Now a check is being made to see if the usersword.length is less than bestword.length. If NOT then userword.length becomes the value of score and rounds decrements by 1 and reverts to step 3. If yes proceed to step 14. And GameLetters is also decremented from myLetters[100] array. This is to prevent the same letters from being generator in the following rounds

Step 14. If the userword.length is greater than the bestword.lenght then the userword becomes the bestword (based on its character count) and also the score is the value of the userword.length. rounds decrements by 1 and reverts to step 3. And GameLetters is also decremented from myLetters[100] array. This is to prevent the same letters from being generator in the following rounds

Step 15. Once rounds decrement by 5(ie 5 games happen and there are 0 games left) The user is prompted to play again

Step 16. The users input is read in

Step 17. If the user entered ‘n’ then score is outputted along with the bestword stored. If the user entered an value other than ‘n’ proceed to step 19

Step 18. The game ends.

Step 19.if the user had entered anything other than ‘n’ to continue then they chose to play the game again. This reverts the user to step 1. Where all variables are set to default value and the game begins from the start.