You have been asked by a newspaper to write weekly word searches for a games column. Being the tech-savvy beast that you are, you've decided to write a computer program that automatically generates the content for you every week so that you don't have to lift a finger.

The newspaper has requested that your program be able to generate word searches of any number of rows or columns. They send you a file with a list of words each week that they would like included in the game. The catch is that they want you to be able to generate several versions of the word search for editors to pick from. This includes various surprise sizes each week that they spring on you days before the deadline.

You are really keen on writing this program and compiling it once so that you don't ever have to rewrite your code. You also want to be able to generate the searches directly from the file. The file starts with the number of rows and columns on the first line and then has a word per line after this. The file sometimes has multiple sets of words in it. There is an empty line separating each new word search. Every time a new wordsearch starts, the number of rows and columns is listed on the first line.

Your program should have these 3 functions (at least, to get you started):

char generate_word_search(const int rows, const int columns) –** create a randomly generated multi-dimensional space in memory of lowercase characters from a-z

void print_word_search(char word_search, int rows, int columns) –** print the wordsearch to standard output with each character separate by spaces and each row on its own line

void insert_word(char, string word, int row, int column) –** add a word to the word search starting at a particular space in memory. This part does not need to be fancy, you may just insert the word starting at this location without checking if another word is overwritten. It is up to you what direction the word should be written.

Notice that these functions use pointer notation to pass and return array parameters. Please refrain from using array notation in the function heads for this lab. Also note that to return a new array from a function, it is essential to use dynamic memory allocation and generate the memory for the data structure on the heap. Please also, remember to delete your arrays before closing your program to free up the memory.