CS262, Lab Assignment 7:
Crossword Puzzle

Due: Sunday, April 3 at 11:59 pm ET

Description:

In this lab, you will write a C program to find the position of W words in a N x M crossword puzzle that has been previously solved. The values for W, N, M range from `5` up to `100`.

Preparation:

To code the program, you will use: File I/O, the `string.h` library, the `malloc` (or `calloc`) function for dynamic memory; and processing of two-dimensional arrays to find the solution. Some useful functions that could be used for this lab are:

`strlen(cs)`	Return length of cs without including end character
`strcat(s,ct)`	Concatenate string `ct` to end of string `s`
`strstr(cs,ct)`	Return pointer to first occurrence of string `ct` in cs, or `NULL` if not present
`strcpy(s,ct)`	Copy string `ct` to string s, including `\0`
`strtok(s,ct)`	Splits `s` into tokens, each delimited by a character from `ct`

A page that provides some examples of string functions is available at:
https://beginnersbook.com/2014/01/c-strings-string-functions/

Note `strstr()` returns a pointer (i.e. a memory address) NOT the index of the array on which the "sub" string was found. To get that index, it's necessary to perform a subtraction between the memory address returned by `strstr()` and the memory address of the start of the array.

Instructions:

The source file for this assignment will be named `lab7_<username>_<labsection>.c`

When running your program, you will enter the name of the *input* file *on the command line*.
The input filename must be given, otherwise show an Error Msg! and exit the program.
Here is the command you should run when executing your program:

```
lab7_<username>_<labsection> <input_file>
```

The program will print the result in an output file with ***the same input filename*** but extension `.out` (e.g. for the input file `test.in` the output file is `test.out`).

Input:

Your program will open the input file for reading. The first row in the input file contains the values of W, N and M separate by a blank space.

Each of the next W rows correspond to every word that we want to find in the solved crossword puzzle. Finally, the last N rows of the file is a two-dimensional array that corresponds to a solved crossword puzzle. **Assume** that the crossword puzzle contains only lowercase letters and blank spaces; and the searched words don't contain blank spaces.

Process:

The program uses `malloc` (or `calloc`) to allocate dynamic memory for a two-dimensional array that stores the solved crossword puzzle and other for the words that we want to find. Do not forget to add an additional column for the character `\0` to indicate the end of a string.

The function `main()` uses a loop statement for calling the function `searchWord()` to find the position of each word. `searchWord()` has the parameters: *word*, the *crossword* puzzle, *and* the array *position*. If necessary `searchWord()` could receive additional parameters and calling other functions to find the words.

When a word if found the information is store in the char array position. The first char indicates the orientation: `h` (horizontal) or `v` (vertical) and after the coordinates `[row][col]` of the matrix where stats the first letter of the word. The coordinates are between brackets. Remember that the index of the first element of an array is `0`.

*Note that for this particular crossword doesn't consider the diagonal orientation.

Hint: A vertical search could be performed using a transpose matrix; thus, the vertical words could be searched in same way as the horizontal ones but considering that the rows are now the columns and vice versa. More about transpose matrix can be found at:

https://www.khanacademy.org/math/linear-algebra/matrix-transformations/matrix-transpose/v/linear-algebra-transpose-of-a-matrix

Output:

The result will be printed in an output file. The first line displays the value W. The next W rows consists of two columns separated with a tab. The first column is the searched *word* and the second column its *position* on the crossword.

Run example:

Suppose that we want to find the words: *nail, taco, name, men, next, can* in the following solved crossword puzzle:

	n	e	x	t	
n	a	m	e	c	t
		e		a	a
		n		n	c
	n	a	i	l	o

The puzzle as given in the input file will contain only blank spaces and letters.
In the above example, the line in the input file for the first line of the puzzle will look like this:\

```
next
```

To give the example more visibility, if we were to replace each blank character with a '_' character for the first line of the crossword puzzle, the data would look like this:

```
_next_
```

However, the above line is just an example. The spaces for a line in the input file will be represented by actual space ' ' characters.

For this example, the output file will contain:

```
6
nail h[4][1]
taco v[1][5]
name h[1][0]
men v[1][2]
next h[0][1]
can v[1][4]
```

Makefile:

Modify the Makefile you used for lab5 so that it works for this assignment. Replace the `-OS` flag by `-O2`.

Submission:

You will submit a typescript file similar to the one of previous Labs:

1. Create a typescript file named `lab7_typescript_<username>_<labsection>`
2. Show that you are logged onto Zeus
3. Show the content of your source code.
4. Compile the code using your `Makefile`
5. Show a detailed listing of the cwd (Type: `ls -l`)
6. Run the code using the `test.in` as input file
7. Show the content of `test.out`
8. Remove the executable using your `Makefile`
9. End the typescript
10. Be sure your directory ONLY contains the *source file*, *script* and `Makefile`
11. Verify your typescript file is correct, then change (`cd`) to the directory above
12. Create a `tarfile` of your `lab7_<username>_<labsection>` directory
13. Submit the `tarfile` to Blackboard
14. Verify that your submitted `tarfile` can be extracted and it's the right `tarfile`.

Congratulations! You have completed your assignment