CS 2213-001 Advanced Programming

Instructor Dr. Turgay Korkmaz

Homework 3
Due date: check BB
!!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!
Total 5 points

A **word search** is a game where letters of words are hidden in a grid (2D char array), that usually has a rectangular or square shape. The objective of this game is to find and mark all the words hidden inside the grid. The words may be hidden horizontally (left-to-right →), vertically (top-to-bottom ↓) or diagonally (left-top-to-right-bottom ↘).

You are asked to implement a program that perform **horizontal, vertical, and diagonal** search.

- The 2D char array can be read from a file. But since we didn't see files yet, we want you to just declare a 2D char array and initialize it in your program. Here is an example:
```
/* yourprog.c */
#define ROW 3  /* these numbers will be larger */
#define COL 4  /* in an actual program */

main()
{
  char g[ROW][COL] = {  {'a','b','c','d'},
                        {'d','c','b','a'},
                        {'x','y','z','d'}};
  ...
```
- The words that a user wants to search will be given as command line arguments. User can give as many words as he/she wants. For example,
```
> yourprog  bcd  bd cy abcdef
```
- As the output, your program will try to find out if each of the words given in the command line argument appears **horizontally, vertically**, or **diagonally** in the given grid g (which is just a 2D array of characters, rows or columns are NOT null terminated). If a word appears in the grid then your program should print how it appears as well as the index values for the beginning row and column of the word in the grid. A word may appear more than once, just print the information about first appearance.

- For the above example, your program should generate the following output
```
bcd appears horizontally starting at g[0][1]

bd appears diagonally starting at g[1][2]

cy appears vertically starting at g[1][1]

abcdef does not appear in g
```

What to return: !!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!

1. Follow the problem solving methodology, and use top-down design technique to solve the problem(s) in a modular manner (for some hints: check Fall 2011 Midterm 1 and Final which are available at the class webpage under samples link). Then convert your solution(s) to a C program. Include /* comments */ so that we can understand your solution(s). You can name your program as hw03.c
2. Compile and run it. Copy/paste the result in an output file, say out3.txt. Or simply use script thing as in hw2…. we may modify your program to test it with a new array g and may change symbolic constants ROW and COL, in that case your program should still work.... we will not change anything else...
3. Zip your files hw03.c and out03.txt as a .zip file
4. Go to BB Learn, and submit your .zip file as an attachment before the deadline.

/* Don't forget to include comments about the problem, yourself and each major step in your program! */

You must submit your work using Blackboard Learn and respect the following rules:

1) All assignments must be submitted as either a zip or tar archive file unless it is a single pdf file.
2) Assignments must include all source code.
3) Assignments must include an output.txt file which demonstrates the final test output run by the student.
4) If your assignment does not run/compile, the output.txt file should include an explanation of what was accomplished, what the error message was that prevented the student from finishing the assignment and what the student BELIEVES to be the underlying cause of the error.