ITU COMPUTER ENGINEERING DEPARTMENT

BLG 233E DATA STRUCTURES

HOMEWORK -1

Due Date : 10th of October, 2017

In this homework, you are supposed to write a program which solves a word puzzle.

The program accepts two input files named as: “puzzle.txt” and “searchedwords.txt”. The first file, “puzzle.txt”, contains the word puzzle consisting of fifteen columns and fifteen rows. Also, the words which will be found on the puzzle exist in the second file, “searchedwords.txt”.
The program aims to find searched words in the word puzzle by doing some searching operations in certain directions: vertical, horizontal and diagonal.

Program Work Flow

1. First, read the information from “puzzle.txt” and store all characters in a 2 dimensional puzzle array.

   A sample puzzle and the 2-D array are given below respectively.

```
A S F G V E D H K L E R D G M
K G V A N D E R V A L S S M A
A C L M S R I I E E A L E N D
Y D G A I S T P T E D T G R J
L K J D E A G D O F A H E A U
O L E C D A F M O L E K U L S
U J L A L V I U A G D I A K K
L N E J H H J S M H D I L M O
S B M N A I B S K J H U P C V
T V E A V D V I J R L K K O A
B C N G T R N E O N U A F V L
N E T F A O A R O W U E J Z E
M W U U Y J M S L T S T U E N
M D I I K E E A J S Z J Y G T
A S L K L N V K G A C L K J K
```

Definition of the 2-D array is given as:

```
char** puzzleArray;
```

2. Read the second file named as "searchedwords.txt". In this file you will find some words and their identification numbers (id). These numbers also indicate the priority order of the words. This means that you must start searching with the word having the highest priority and continue to words having the lowest priority. You are also supposed to calculate the length of each searched word and store this information during the runtime.

   Definition of the struct is:

```
struct searchedWord {
int id;
char* word;
int wordLength;
};
```

3. As the third step, you are supposed to check whether the word exists in the puzzle or not, by considering the priority order of searched words. You are allowed to do searching operations in certain ways: Vertical, Horizontal and Diagonal.

   In vertical check, each row is searched from right to left and vice versa.
   In horizontal check, each column is searched from up to down and vice versa.
   In diagonal check, words are searched diagonally.

   During the searching operations, you do **NOT** make any changes on the original puzzle array. Definitions of the check functions are given as:

```
public bool verticalSearch   (char** puzzleArray, searchedWord* word)
public bool horizontalSearch (char** puzzleArray, searchedWord* word)
public bool diagonalSearch   (char** puzzleArray, searchedWord* word)
```

   You have to traverse puzzle array with pointers.
   When you want to access any character in the puzzle, you cannot use array indexes. That is, you must use the **pointer arithmetics**. For example:
   puzzleArray[i][j+1] is **NOT** allowed, therefore puzzleArrayPtr++ must be used.

4. Up to now, you checked all searched words whether they exist in the puzzle or not. For now, you should report in **which indexes and direction** searched words are found in the puzzle. These information are written to an output file called as "foundWordsInfo.txt"

   An example for the output of this program:

WORD1	HORIZONTAL	[3][5] – [3][9]
WORD2	VERTICAL	[1][6] – [9][6]
WORD3	DIAGONAL	[9][11] – [5][7]

5. As the last step, you should replace characters of found words with empty characters in the original puzzle array. Then, you should write this version of the puzzle to the output file called as "lastVersionPuzzle.txt". An example output file is:

Submission

1. Make sure you write your name and number in all of the files of your project, in the following format:

```
/* @Author
 * Student Name: <student_name>
 * Student ID : <student_id>
 * Date: <date>
 */
```

2. Use comments wherever necessary in your code to explain what you did.

3. **Compile the code in the Secure Shell Client (SSH) before you send your homework.**

4. After you make sure that everything is compiled smoothly, archive all files into a zip file. Submit this file through www.ninova.itu.edu.tr. Ninova enables you to change your submission before the submission deadline.

   Do not miss submission deadline. **Do not** leave your submission until the last minute. The submission system tends to become less responsive due to high network traffic.

   HOMEWORKS SENT VIA E-MAIL WILL NOT BE GRADED.

Academic dishonesty including but not limited to cheating, plagiarism and collaboration is unacceptable and subject to disciplinary actions. Your homeworks will be checked with a plagiarism checker system, any student found guilty will receive 0 as his/her grade for the homework and subject to disciplinary actions.

If you have any question about the homework, contact the teaching assistant **Osman Ali DURNA** via e-mail **(durna16@itu.edu.tr)** or in **BAAL**.