EE2310 Introduction to Programming – Assignment 2
Due: Dec 20th, 2020 (11:59pm)

I. Submission

1. You must submit your source file containing the designated function at NTHU OJ (Problem#13035) https://acm.cs.nthu.edu.tw/problem/13035 by the deadline (like you did in the Lab classes).
2. You must prepare a **README file** in **.txt** format (~300 words, 2~3 paragraphs) that summarizes how you write your program, e.g. what your variables or functions are for, etc.
3. Prepare your source file (of the designated function) in a single **.c** file. No need to submit the main function.
4. Compress your README and source files in a single zipped file named `YourStudentID.zip` and submit this zipped file onto the **eLearn system**.
5. Late submission will incur 10% penalty per day up to 3 days. After that, assignment submission will be closed and no submissions will be accepted.

II. Task Description

Write a program that finds out a path out in a maze. For example, a 4*7 maze can be represented as follows:

```
1011110
1011000
1000011
0010110
```

Here, '0' represents that there is no obstacle (and '1' otherwise), so one can only visit the '0'-locations. The bottom-left and top-right locations are always '0', indicating there is always an entrance and an exit in the maze.

Your program needs to **find a way from the bottom-left to the top-right** and output the results with all nodes on the path marked '2' as follows:

```
0000002
0000222
0222200
2200000
```

We will provide the main function that deals with I/O. The input will be saved in the one-dimensional m×n dynamic character array called `maze` consisting of only '1' and '0'. The element of (i , j) will be saved at `maze+i*n+j`. All you need to do is find its path and save it at another one-dimensional character array called `result_maze`, which is declared in the main function. The element (i , j) also corresponds to the element `result_maze+i*n+j`, as shown in the sample output. Note that **you don't need to deal with the output. We already did that for you**. Just save your results in the array `result_maze`. If there is no way to the destination, save '`X`' at the beginning address of `result_maze`.

III. What you need to do?

You may need the following tools to do this assignment:

1. **You may need to use a stack** that stores all past history up to the current location. Your stack may need to double its size because the input may be more than the its size.
2. You may need to use another dynamic array to mark the locations you have already visited just to avoid possibly looping around forever and never finding the exit.

IV. Sample I/O

➢ Sample input (Case 1)

```
4 7
1011110
1011000
1000011
0010110
```

➢ Sample output (Case 1)

```
0000002
0000222
0222200
2200000
```

➢ **Sample input (Case 2)**

```
4 3
110
110
001
011
```

➢ **Sample output (Case 2)**

```
No way
```

V. Guidelines

1. Assessment: Correctness 70%, Program Style 30%.

2. Correctness:

   Your program output will be checked automatically by NTHU OJ. This time we already took care of the I/O and there should be no presentation errors anymore. Just save your result in the designated array.

3. Program Style:

   1. Your programs should be *well-commented*. All variables, functions, loops should be clearly explained both in the source code and briefly in the README file.
   2. Blocks in your source code should have *proper indentation* (縮排).
   3. Do not use any global variable to pass data in and out of a function.
   4. For a good program style, please refer to the Linux Kernel Coding Style at https://www.kernel.org/doc/html/v4.10/process/coding-style.html

This is not an easy assignment.
Please start early!!