EE2310 Introduction to Programming
2020 Fall – Assignment #1

- **Due: Nov 28th, 2020 (Sat), 11:59pm**
- **Late assignments incur 10% penalty per day up to 3 days.**
- **No submission is allowed as of Dec 2nd 0:00am.**

Submission

1. You must submit their source file at NTHU OJ (Problem#12990) https://acm.cs.nthu.edu.tw/problem/12990/ 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 in a single **.c** file.
4. Compress your README and source files in a single zipped file named `YourStudentID.zip` and submit this zipped file onto the **eLearn system**.

Task Description

In this assignment, you are to write a program to help a player play the game Bingo. In our version of Bingo, each player is given a card containing an *n*x*n* array of numbers. The numbers are in the range 0 to 99 and will not repeat. An example of a 5x5 array is shown below.

5	13	14	6	97
67	17	85	23	1
12	99	33	65	7
2	49	48	20	57
50	38	73	88	41

Then the referee will randomly select a number from 0 to 99 and each player will search for the number in his array. If the number is present, he will mark it off. This is repeated until some player has a complete row, column or diagonal marked off. That player will then declare himself a winner and the game will be stopped. For example, suppose the sequence of numbers chosen by the referee is:

```
12, 99, 70, 23, 49, 21, 1, 50, 7, 88, 17, 45, 97, 65, 33.
```

Then the player holding the above card will mark the corresponding entries on his table:

5	13	14	6	~~97~~
67	~~17~~	85	~~23~~	~~1~~
~~12~~	~~99~~	~~33~~	~~65~~	~~7~~
2	~~49~~	48	20	57
~~50~~	38	73	~~88~~	41

As soon as he marks off the number 33, he has a diagonal and a row completed. Therefore, he declares himself a winner. Notice that some numbers (e.g., 70 and 23) in the sequence are not present in the table.

Suppose on the other hand, the sequence of numbers is:

```
12, 99, 70, 23, 49, 21, 1, 50, 7, 88, 17, 45, 97
```

when some other player has completed a row/column/diagonal, the game is stopped.

Your program will input an integer *n* in the range 0 to 99. This represents the number of rows and also the number of columns of the table. This is followed by n lines of integers, each line containing n integers. This represents the *n*x*n* table. Finally, there will be a sequence of integers in the range 0 to 99 followed by a -1 value, each value on a separate line. You may assume that the numbers in the sequence do not repeat and that the sequence of numbers will be ended (with a -1) as soon as a row, column or diagonal is completed. A sample input is given below.

Your program will then output “won” or “lost” depending on whether the player holding the card wins on that sequence or not. Then output the table and mark the numbers with star(s) in the row/column/diagonal that is completely matched. Notice that in the sample output below, the numbers 17, 1 and 88 are not marked because they are not involved in a completed row, column or diagonals.

Your table should have *n* columns and *n* rows. Each column is 3-character wide and there should be 1 blank space between two columns. Column 1 starts from the left. If a number should be marked, put enough stars (*) in front of the number so that the stars and the number together is 3-character wide.

Sample Input (correct version, difference shown in red)

```
5
5 13 14 6 97
67 17 85 23 1
12 99 33 65 7
2 49 48 20 57
50 38 73 88 41

12 99 70 23 49 21 1 50 7 88 17 45 97 65 33 -1
```

Sample Output

```
won
  5  13  14   6 *97
 67  17  85 *23   1
*12 *99 *33 *65 **7
  2 *49  48  20  57
*50  38  73  88  41
```

Guidelines

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

2. Correctness:

   Your program output will be checked automatically by NTHU OJ. Therefore, you must follow the exact output format specified for this assignment (including newline character '\n').

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 (which we haven't taught yet so there shouldn't be any problem about this).
   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