EE231002 Introduction to Programming

Lab07. Magic Square

Due: Nov. 2, 2019

A magic square is an $N \times N$ matrix that fills with integers from 1 to N^2 and satisfies the following conditions.

1. All the row sums are equal,
2. All the column sums are equal,
3. The sums of the two diagonals are equal to the row sum and column sum.

For example, a 4×4 magic square are shown below.

16	8	9	1
10	2	15	7
3	13	4	14
5	11	6	12

Note that all the row sums, column sums and the two diagonal sums are equal to 34. In fact, given an $N \times N$ magic square, it is easy to show that the row sums, column sums and diagonal sums are all equal to $\dfrac{N(N^2+1)}{2}$.

In this lab, you will write a C program to read in a partially filled 4×4 matrix and then fill the remaining blanks to make it a magic square. It is possible that more than one magic squares can be formed given the partially filled matrix and your program should find all of them. Four matrices are provided for you to test your program: `m48.dat`, `m47.dat`, `m46.dat`, and `m45.dat`. Each file contains a 4×4 matrix, with `0` as a unfilled space. You can use `unix` input redirection as the following to read the content of a file.

```
$ ./a.out < m48.dat
```

In this way, your `scanf` will read from the file `m48.dat` instead of from the keyboard.

Example of program execution.

Given an 4×4 input file, `test.dat`, as below,

```
  16   0   9   0
  10   0  15   0
   0  13   0  14
   0  11   0  12
```

Your program should execute to get

```
$ ./a.out < test.dat
Solution 1:
  16   8   9   1
  10   2  15   7
   3  13   4  14
   5  11   6  12

Number of solutions found: 1
```

For this example, only one solution is found. However, it is possible that the given input files may have more than one solution.

Notes.

1. Create a directory **lab07** and use it as the working directory.
2. Name your program source file as **lab07.c**.
3. The first few lines of your program should be comments as the following.

   ```
   // EE231002 Lab07. Magic Square
   // ID, Name
   // Date:
   ```

4. After finishing editing your source file, you can execute the following command to compile it,

   ```
   $ gcc lab07.c
   ```

   If no compilation errors, the executable file, **a.out**, should be generated, and you can execute it by typing

   ```
   $ ./a.out < m48.dat
   ```

5. After you finish verifying your program, you can submit your source code by

   ```
   $ ~ee231002/bin/submit lab07 lab07.c
   ```

   If you see a "submitted" message, then you are done. In case you want to check which file and at what time you submitted your labs, you can type in the following command:

   ```
   $ ~ee2310/bin/subrec
   ```

   It will show the last few submission records.

6. You should try to write the program as efficient as possible. The format of your program should be compact and easy to understand. These are part of the grading criteria.