**Overview:**

An n x n game board is populated with integers, one nonnegative integer per square. The goal is to travel along any legitimate path from the upper left corner to the lower right corner of the board. The integer in any one square dictates how large a step away from that location must be. If the step size would advance travel off the game board, then a step in that particular direction is forbidden. All steps must be either to the right or toward the bottom.

Consider the 4 x 4 board shown in Figure 1. Figure 2 shows the three paths from the start (upper left) to the target (lower right), with the irrelevant numbers in each removed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | 2 | 3 | 3 | 1 | | 1 | 2 | 1 | 3 | | 1 | 2 | 3 | 1 | | 3 | 1 | 1 | 0 | | Figure 1 | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | 2 |  | 3 |  | |  |  |  |  | |  |  |  |  | |  |  | 1 | 0 | |  | | | | | |  |  |  |  | | --- | --- | --- | --- | | 2 |  |  |  | |  |  |  |  | | 1 | 2 |  | 1 | |  |  |  | 0 | | Figure 2 | | | | | |  |  |  |  | | --- | --- | --- | --- | | 2 |  |  |  | |  |  |  |  | | 1 |  |  |  | | 3 |  |  | 0 | |  | | | | |

**Program Input (from file named in arg[0])**

The input file contains data for one to thirty boards, followed by a final line containing only the integer -1. The data for a board starts with a line containing a single positive integer n, 4 <= n <= 35, which is the number of rows in this board. This is followed by n rows of data. Each row contains n single digits, 0-9, with no spaces between them.

**Program Output (to stdout)**

The output consists of one line for each board, containing a single integer, which is the number of paths from the upper left corner to the lower right corner.

**Example Input**

4

2331

1213

1231

3110

4

3332

1213

1232

2120

5

11111

11111

11111

11111

11111

-1

**Example Ouput**

3 0 70

**Program Design**

This is one of those problems where there is a Graph implicit in the data but the actual verticies and edges are not explicitly given. Nevertheless, a graph search algorithm is quite useful for finding paths. Use the recursive depth first search algorithm given in the text to find the total number of paths from the upper left square to the lower right square.

Your programs will be limited to 1 minute of run time. [Think!?]

**On the due date at the beginning of class turn in:**

* A hard copy of your source code (see the web page for additional requirements).
* A hard copy of your test runs showing output correctness for your own data.
* A hard copy of the Javadocs for your classes, or the equivalent if you are using a language other than Java.
* Suggestions for improving this lab assignment (optional).
* Your source (.java ) files via [Submit](https://hub.cs.plu.edu/submit/).