CS1083 Assignment # 8 - Winter 2023

Due: Wednesday, 6 November before 4:30 pm in the Desire2Learn dropbox. (See submission instructions below).

The purpose of this assignment is:

- Review 2D Arrays
- Practice Recursion

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during an extra help session. If your question is not answered during an extra help session, you may contact your course professor.

Topographic Map

A topographic map is a map that shows various information about the land of a given area. One such bit of information that may be displayed is the altitude of points. For this assignment, we will be describing these maps in terms of a 2D array of numbers and characters. Our maps will consist of numbers 1 -> 9 with 1 being the lowest altitude and 9 being the highest altitude. Our maps will also denote rivers with the character "R" and lakes with the character "L".

Example Topographic Map:

9	8	7	7	7	6	6	5	1	1	R	R
8	8	7	6	5	L	L	3	1	1	R	R
7	7	7	6	5	L	3	3	1	1	R	R
6	6	6	6	5	4	3	3	5	6	R	R
5	4	5	5	L	L	3	6	7	7	R	R
3	3	3	4	L	L	4	6	7	7	1	R
1	2	2	4	2	2	5	6	7	7	1	R
1	2	2	6	6	6	6	6	7	7	1	1
1	1	1	1	1	1	3	2	3	7	1	1
1	1	1	1	2	1	3	1	2	7	2	1
R	R	R	R	1	1	4	1	1	6	L	L
R	R	R	R	R	1	1	1	5	5	L	L
R	R	R	R	R	R	R	R	1	5	6	L

Flooding

Flooding can occur anytime there is a rainfall. Rainfalls can be categorized as a number from 1 -> 6 based on severity with 1 being less severe and 6 being more severe. Depending on the severity level of the rainfall, rivers and lakes will flood differently. If a rainfall occurs, rivers will flood all adjacent areas with an altitude less than or equal to the rainfall severity. On the other hand, lakes will only flood adjacent altitude areas in increments of 2. See table below for a breakdown.

Rainfall Severity	Rivers will flood tiles:	Lakes will flood tiles:
1	1	No flooding
2	1, 2	1, 2
3	1, 2, 3	1, 2
4	1, 2, 3, 4	1, 2, 3, 4
5	1, 2, 3, 4, 5	1, 2, 3, 4
6	1, 2, 3, 4, 5, 6	1, 2, 3, 4, 5, 6

For example, if a level 3 rainfall occurs, the map above will be changed to the following:

9	8	7	7	7	6	6	5	R	R	R	R
8	8	7	6	5	L	L	R	R	R	R	R
7	7	7	6	5	L	R	R	R	R	R	R
6	6	6	6	5	4	R	R	5	6	R	R
5	4	5	5	L	L	R	6	7	7	R	R
R	R	R	4	L	L	4	6	7	7	R	R
R	R	R	4	L	L	5	6	7	7	R	R
R	R	R	6	6	6	6	6	7	7	R	R
R	R	R	R	R	R	R	R	R	7	R	R
R	R	R	R	R	R	R	R	R	7	R	R
R	R	R	R	R	R	4	R	R	6	L	L
R	R	R	R	R	R	R	R	5	5	L	L
R	R	R	R	R	R	R	R	R	5	6	L

Notice that when a flood occurs, rivers will be expanded first; however, rivers should not replace lakes that were already there.

Also note that due to these rules, altitudes of 7 and higher will never flood regardless of the rainfall severity.

Traversal

In the event of an emergency occurring, emergency vehicles may not be able to reach the location where the emergency occurred due to water cutting off the location.

For example, in the scenario below from the previous example:

	A	B	C	D	E	F	G	H	I	J	K	L
1	9	8	7	7	7	6	6	5	R	R	R	R
2	8	8	7	6	5	L	L	R	R	R	R	R
3	7	7	7	6	5	L	R	R	R	R	R	R
4	6	6	6	6	5	4	R	R	5	6	R	R
5	5	4	5	5	L	L	R	6	7	7	R	R
6	R	R	R	4	L	L	4	6	7	7	R	R
7	R	R	R	4	L	L	5	6	7	7	R	R
8	R	R	R	6	6	6	6	6	7	7	R	R
9	R	R	R	R	R	R	R	R	R	7	R	R
10	R	R	R	R	R	R	R	R	R	7	R	R
11	R	R	R	R	R	R	4	R	R	6	L	L
12	R	R	R	R	R	R	R	R	5	5	L	L
13	R	R	R	R	R	R	R	R	R	5	6	L

If emergency vehicles are initially located at position (1, A) and an emergency occurs at location (13, K), then the emergency vehicle **can drive there without issue**.

If emergency vehicles are initially located at position (1, A) and an emergency occurs at location (11, G), then the emergency vehicles **cannot get to the location by driving.**

Programming

For this assignment, you will be programmatically determining whether emergency vehicles can drive from a location to another location after flooding occurs. All programming may be done in a single .java file for this assignment.

Assume that no bad input will be given as input unless stated otherwise.

Step 1: Reading in the Map

When a user runs the program, they should be able to run it in the format of:

```
java EmerTraversal FileName.csv
```

Where the file contains a comma-delimited list of numbers or characters representing the graphs. Each new line is a new row while commas separate the columns.

The first 2 rows of the file will state the number of rows and columns respectively.

You must read in the map from this file and store it in a 2D array. An example file for the map shown in the introduction steps can be found on D2l labelled "Map1.csv". Assume that the graphs will always be in the correct format with no missing data. Handle the appropriate exception(s) that ***must*** be handled in this step.

Tips:

- You may want to create a static method here that will print out your map to make sure it reads in properly.
- Be certain that the reading in is occurring properly before moving onto the next step.

Step 2: Rainfall Occurs

Next the program should ask the user for a rainfall severity level which will be inputted through the terminal. Again, you can assume that the user will not input any bad data (i.e., only values 1->6). Iteratively traverse your map and when you encounter a river tile, expand it using a recursive method (described below). Once all rivers have been expanded, iteratively traverse your map a second time, expanding lakes this time. Print the array after all flooding has been performed.

Recursive Method

Create a static recursive method that will take a 2D array, a rainfall severity level, the row and column number to start at, and a character: either 'R' or 'L'. This method should recursively expand rivers and lakes to fill all adjacent tiles. Expand rivers and lakes by replacing the numeric data with either 'R' or 'L' if the tile should be flooded. ***Do not consider diagonal tiles as adjacent.*** This method should not return anything as all changes should occur to the original array.

Tips:

- It may be useful to print the map in each recursion call for testing purposes and then can delete this line later.
- Testing with a smaller map in the beginning may save time.
- First check if the position you want to potentially flood exists, i.e., does it even exist? Tiles such as -1, 0 do not exist.

Step 3: Emergency Vehicle

Finally, the program should ask the user for the row and column that the emergency vehicles are located at and the row and column that the emergency occurred at. Rows will always be input first with columns being input second. See example output below for a better understanding of the style of this input. Rows and columns are labelled with integers starting at 0. Position (0,0) represents the upper left corner of the map. After reading in these positions, see if there is a path between the two points that does not involve crossing any water. Print the result of the pathfinding method.

Recursive Method

Create a static recursive method that will take a 2D array for the map, a 2D array of Booleans to check if a position has already been searched, and 4 integers representing the current location and the location the emergency vehicles are trying to get to. This method should recursively check if there is a pathway between the two points without crossing water. Once you check a position, you should label it as checked by updating the 2D array of Booleans. This method should return true if a safe pathway exists and false otherwise.

Once again, diagonal paths are not considered. Assume vehicles can go north, south, east, or west, but cannot move 2 directions in one step.

Tips:

- It may be useful to print the checked position array in each recursion call for testing purposes and this can be deleted later.
- This method will have some similar code to the flooding method but will not be identical.
- Here, we just want to check if a path exists, and we are not interested in the actual steps of the pathway.
- Again here, it is likely easier to test on a smaller map first.

Examples

Note: Example input is run with "Map1.csv" which can be found on D2l.

Test Case 1:

```
Input rainfall severity level:
3

9 8 7 7 7 6 6 5 R R R R
8 8 7 6 5 L L R R R R R
7 7 7 6 5 L R R R R R R
6 6 6 6 5 4 R R 5 6 R R
5 4 5 5 L L R 6 7 7 R R
R R R 4 L L 4 6 7 7 R R
R R R 4 L L 5 6 7 7 R R
R R R 6 6 6 6 6 7 7 R R
R R R R R R R R 7 R R
R R R R R R R R 7 R R
R R R R R R 4 R R 6 L L
R R R R R R R R 5 5 L L
R R R R R R R R 5 6 L

Row Coordinate of Vehicles:
0
Column Coordinate of Vehicles:
0
Row Coordinate of Emergency:
10
Column Coordinate of Emergency:
6

No land path to the emergency exists
```

Test Case 2:

```
Input rainfall severity level:
3

9 8 7 7 7 6 6 5 R R R R
8 8 7 6 5 L L R R R R R
7 7 7 6 5 L R R R R R R
6 6 6 6 5 4 R R 5 6 R R
5 4 5 5 L L R 6 7 7 R R
R R R 4 L L 4 6 7 7 R R
R R R 4 L L 5 6 7 7 R R
R R R 6 6 6 6 6 7 7 R R
R R R R R R R R 7 R R
R R R R R R R R 7 R R
R R R R R R 4 R R 6 L L
R R R R R R R R 5 5 L L
R R R R R R R R 5 6 L

Row Coordinate of Vehicles:
12
Column Coordinate of Vehicles:
10
Row Coordinate of Emergency:
0
Column Coordinate of Emergency:
0

There does exist a land path to the emergency
```

Test Case 3:

```
Input rainfall severity level:
1

9 8 7 7 7 6 6 5 R R R R
8 8 7 6 5 L L 3 R R R R
7 7 7 6 5 L 3 3 R R R R
6 6 6 6 5 4 3 3 5 6 R R
5 4 5 5 L L 3 6 7 7 R R
3 3 3 4 L L 4 6 7 7 R R
R 2 2 4 2 2 5 6 7 7 R R
R 2 2 6 6 6 6 6 7 7 R R
R R R R R R 3 2 3 7 R R
R R R R 2 R 3 R 2 7 2 R
R R R R R R 4 R R 6 L L
R R R R R R R R 5 5 L L
R R R R R R R R 5 6 L

Row Coordinate of Vehicles:
0
Column Coordinate of Vehicles:
0
Row Coordinate of Emergency:
10
Column Coordinate of Emergency:
6

There does exist a land path to the emergency
```

Test Case 4:

```
Input rainfall severity level:
6

9 8 7 7 7 R R R R R R R
8 8 7 R R L L R R R R R
7 7 7 R R L R R R R R R
R R R R R R R R R R R R
R R R R L L R R 7 7 R R
R R R R L L R R 7 7 R R
R R R R R R R R 7 7 R R
R R R R R R R R 7 7 R R
R R R R R R R R 7 7 R R
R R R R R R R R 7 7 R R
R R R R R R R R 7 7 R R
R R R R R R R R R R L L
R R R R R R R R R R L L
R R R R R R R R R R R L

Row Coordinate of Vehicles:
1
Column Coordinate of Vehicles:
1
Row Coordinate of Emergency:
4
Column Coordinate of Emergency:
8

No land path to the emergency exists
```

Your electronic submission (submitted via Desire2Learn) will consist of two files. Name your files YourName-fileName.extension, e.g. JohnSmith-as8.zip, JohnSmith-as8.pdf:

1. A single pdf file containing a listing of the code for your program.
2. A zip file containing all your Java classes.