COMP 482 Project 2: Puddles
Due: Friday March 11, 2355
Points: 30 points possible

Overview: Your program will be given a "map" of wet and dry locations and you will determine how many seperate puddles (contiguous wet areas) are depicted.

Details: The input will come from a file called input.txt which will be placed in the same directory as your java file. The first line of the file will have two integer values R and C which will be the number of rows and columns in the map (simple rectangular region that can be represented by a 2-D array). Each of the next R lines will consist of C zeroes or ones. A zero will indicate water and a one will indicate NOT water (grass?). Two zeroes will be considered to be in the same puddle if they are either directly left/right of each other (ie (i,j) and (i, j+1)) or directly above/below each other (ie (i,j) and (i+1, j)) or can be connected via repeated applications of these rules.

Another (more technical) way to picture the situation is that you begin with a graph having $R \cdot C$ vertices each having label (i,j) where $0 \leq i < R$ and $0 \leq j < C$ and are divided into two sets "Water" and "Not Water". Vertex (i,j) shares an edge with the vertices above, below, left, and right, but not diagonal). In other words, (i,j) is adjacent to vertices (i-1,j), (i,j-1), (i,j+1), and (i+1, j). You then remove all "Not Water" vertices and their incident edges.

Your program will output the number of puddles (or in the more technical description the number of components in the graph).

The obvious way to way to design the program is to (repeatedly) run a version of BFS. This was described and demonstrated in class (either 2/17 or 2/19), but you may use any reasonable technique you understand.

You can discuss the algorithm to be used with anyone and consult any source (books, internet, etc). However, for this project, you are expected to write the code on your own with limited or no assistance from the professor, no assistance from others, and limited or no assistance from other sources (books, internet, etc). To clarify, you can seek assistance in understanding the task, but "your code" should be written by you: not written by others, not copied from others, not copied from books/internet.

Picky, but required specifications: Your project must:

- be submitted via canvas.
- consist of 1 or more dot-java files (no class files, zip files, input files or other files should be submitted). Each file must have your name and which project you are submitting as comments on the first 2 lines.
- not be placed into any package (for the java pedants, it must be in the default package).
- be designed and formatted reasonably (correct indentation, no excessively long lines, no excessively long methods, has useful method/variable names, etc)
- have one file called Project2.java.
- compile using the command 'javac Project2.java'.
- run using the command 'java Project2',
- accept input from a file called input.txt in the same directory as the java file(s) formatted precisely as described above.
- accomplishes the goal of the project. In other words, the output should be the correct answer, computed in a valid way, formatted correctly.
- be submitted on time (early and multiple times is fine).

For each listed item that you fail to follow, expect to lose at least 5 points. In particular, submitting via anything other than canvas will result in a 0 and submitting more that a week late will also result in a 0.

Sample execution: If input.txt contains

```
5 10
1 1 1 1 1 1 0 1 1 1
1 1 1 1 1 1 0 1 1 1
1 1 0 1 1 1 0 1 1 1
1 1 0 0 0 0 0 1 1 1
1 1 1 0 0 0 0 1 1 1
```

then the output should be

```
1
```

If input.txt contains

```
10 15
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 1 1 1 1 1 1 0 1 0 1
1 1 1 1 1 0 1 1 1 1 1 0 1 0 1
1 1 1 1 1 1 0 1 1 1 1 0 1 0 1
1 1 1 1 1 1 1 0 1 1 1 0 1 0 1
1 1 1 1 1 1 1 1 0 1 1 0 1 0 1
1 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1
1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1
```

the the output should be

```
12
```

Stray Thoughts:
I suggest you finish you finish and submit your project at least several days in advance. This way you have time and opportunity to ask any last questions and verify that what you upload satisfies the requirements. There is nothing wrong with working on the project for a day and uploading your code, working for another day and uploading your improved code, ..., working for another day and uploading your final version. In fact there are advantages: you have a fairly reliable place that keeps your versions and even if you get busy at the last moment you have still uploaded your best version.
Your project should be written and understood by you. Helping or receiving help from others to figure out what is allowed/required is fine, but copying code is not. Significant shared source code indicates that you either did not write/understand what you submitted or you assisted another in submitting code they did not write/understand.