CSC301 HW5

Instructions:

1. Download all the filxes from the dropbox for HW5.
2. Start up Eclipse with the same workspace as you used for HW1.
3. Inside the src folder, create a hw5 package.
4. Select all the .java files you downloaded and copy and paste them into the hw5 package in Eclipse.
5. You should now be able to edit the Solver class. This is the only file you should modify (other than creating more tests in the HW5Test file). The majority of the other classes are straight from the book's code and are meant to be used as is.
6. The Solver class has a single static method called solve that you must implement. It should behave as follows:
    - It takes a single char[][] argument which I have called grid. The grid is a square array (every row and every column have the same size), and each cell contains either 's', 'f', '*', or ' ' (space character).
    - You must find a shortest path (there may be more than one so any shortest path is correct) from the 's' to the 'f', while avoiding all the '*' characters.
    - You return the path as a string consisting solely of 'U', 'D', 'L', and 'R' characters.
    - 'U' moves you up one row. So if you are currently at grid[3][7], 'U' moves you to grid[2][7].
    - 'D' moves you down one row. So if you are currently at grid[3][7], 'D' moves you to grid[4][7].
    - 'L' moves you left one column. So if you are currently at grid[3][7], 'L' moves you to grid[3][6].
    - 'R' moves you right one column. So if you are currently at grid[3][7], 'R' moves you to grid[3][8].

Restrictions:

- The only classes you may modify are Solver.java and HW5Test.java.
- You may not modify the header for the solve function in Solver.java and your solution must be completely contained within the Solver class. In other words, I should be able to replace the given Solver.java file with your Solver.java file and then run all my tests on your submission.

Hints:

- As described in the comments inside the solve function, you want to make use of the book's Graph class and BreadthFirstPaths class to solve this. This means you will need to find a way to translate the problem into a graph problem, run BFS on the graph, and then translate the result from the BFS back into a sequence of 'U', 'D', 'L', and 'R'.

- Since BFS finds shortest paths, you don't need to do anything special to find a shortest path besides using BFS.
- I've included a class called GridUtilities. It's meant to help you generate grids for testing your solution. See the current HW5Test class for examples of how you can use it.

Submission:

Submit your Solver.java file and your HW5Test.java file.

Grading:

Your grade will be based on how your code performs on a JUnit test I will design. I will post my JUnit test after the submission deadline. You will then have a chance to resubmit. The resubmission will be graded the same way as the original but a 15 penalty will be assessed. This means the highest possible grade on the resubmission is 85. Your final HW5 grade will be the larger of the two grades.

Note that it is very difficult to get partial credit on this assignment. Most likely your code will either pass all the tests or will fail most if not all the tests. In other words, your grade will most likely be 100 or it will be less than 30, so make sure to thoroughly test your solution.