The shortest route challenge
University of Nottingham team

Scenario:

The school of Computer Science is preparing a booklet for 1st year students induction. They want to include maps showing the shortest route from one room (lecture theatre, etc.) to another. To do this, they ask you to write a computer program to calculate both the length of the shortest route and the route itself.

Challenge:

A plan of a floor is given in an input file. The first line consists of 2 numbers N and M ($2 \leq N, M \leq 500$), which describe the size of the map (N x M).

In the following 2 lines, the coordinates of start and finish points are given. The first number in each line means the X coordinate of a point while the second represents the Y coordinate. For example, the very first point would have the coordinate of "1 1"

In the fourth line, there are N numbers separated by a white space in each of the following M lines. The numbers below describe each of the map's fields:

```
0 - Empty field
1 - Wall
2 - Switch - if you go to this field, all open doors within the
    distance of 1 are closed while all closed doors are opened (the
    state of the doors which are diagonal to the switch will not be
    changed)
3 - Open door
4 - Closed door
```

Note that the start and finish points will have the value of 0.

Output:

If a route exists, output file should consist of 2 lines:

The first line represents the length of the shortest route as an integer.

The second line represents the route itself.

The route should be given as a set of directions denoted by a sequence of letters D (for "down"), L (for "left"), U (for "up"), R (for "right") which should not be separated by any other symbols (spaces, commas, etc.).

Only one route should be provided, even if there is more than one possible route with the shortest length. If there is no possible route in a given plan, output file should simply consist of 0.

Notes:

There are 10 input files (route01.in – route10.in) which you can use for testing. Another extra 10 Maps would be used for Marking.

The length of the route and the route itself will be marked separately, thus you will gain marks even if you can calculate only the length of the shortest route.

Some of the first tests do not contain any doors or switches, thus you will gain some marks even if you cannot implement doors and switches.

All moves to the next point will add 1 to the length of the route. Note that it is not possible to move diagonally.

Typical Input and output example:

Your program should take two arguments – input and output filenames

e.g., **java RouteFinder route0.in route0.out**

Input file `route0.in`	Output file `route0.out`
10 10 2 2 4 9 1 0 0 0 1 1 1 1 1 1 1 **0** 0 2 3 2 2 **4** 0 1 1 **0** 0 0 1 0 0 1 0 1 1 **2** 0 0 1 0 0 1 0 1 1 **4** 1 1 1 0 0 1 0 1 1 **0** 0 0 0 0 2 3 0 1 1 **0 0 0 0 0 0** 1 0 1 1 1 1 1 1 1 **0** 1 0 1 1 0 0 **0 0 0 0** 1 0 1 1 1 1 1 1 1 1 1 1 1	15 DDDDRRRRRDDDLLL

Marking Scheme:

70%: Dynamic Correctness

- 20 test files – 3.5 marks for each
    - Output File created and not empty (0.5 Mark)
    - Shortest route length (1 Mark)
    - Shortest route (1.5 Mark)
    - If the route's length equals the length given by the user in the first line(0.5 Mark)
    - If no route exists in a given map, all marks will be given for length and route in case of printing out 0.

Note: Your program needs to finish in 30 seconds Max for each map.

10 %: Typographic tests

- Correct Indentation (2 Mark)
- Classes, variable and methods Names (3 Marks)
- Comments (4 Marks)
- Good use of blank lines to identify related lines of code (1 Mark)

20%: Feature tests

- Correct Access modifiers (4 Marks)
- Correct use of constants (4 Marks)
- Using ArrayLists and generics (4 Marks)
- General quality of code, good use of classes, methods etc. (8 Marks)