Cosc 2P03: Advanced Data Structures in Java
Assignment #4
Due date: 5:00pm, Friday, July 6th

This assignment deals with graphs and path-finding within them.
You will write a program which allows the user to enter multiple levels of a text-based map, consisting of a 'start', an 'exit', 'walls', and 'floors'. The letter S represents the start. E is the exit. O (the letter 'oh') represents a 'floor' (meaning a path you can follow). X is a wall (meaning a block you cannot pass).

Your program will accept this ASCII-based map from Standard In (`System.in`), and convert it into a graph. It is up to you whether you wish to use an adjacency matrix or adjacency lists (linked or array-based). However, this step is not optional, and those are your only two options.

Ideally, your program will allow the user to find the 'fastest' path through the maze. In this context, 'fastest' means the fewest number of steps possible to get from the start to the exit, without crossing a wall. *Note: You may only move North, South, East, West, Up, or Down (diagonal travel is not allowed).*

If no valid path exists, your program should say so. You may assume 'sane' input, and that a start and exit are present (though there may not be a path connecting them). The maps do not 'wrap' (i.e. you can't exit the left edge to appear on the right edge), and you may *not* assume that a floor will have a boundary of walls.

Specifically, the input will be as follows:
An integer for the width of the map (W)
An integer for the height (length) of the map (H)
An integer for the depth (number of layers) of the map (D)
D×H lines of W characters (from {X,O,S,E}), terminated with newlines

For D layers, first the (H) lines of the top layer will be entered, then the lines of the second-highest, etc.

In place of any line of W characters, it is legal to include an entirely blank line (as such, any line will have W characters followed by a `\n`, or consist solely of a `\n`. Your code must be able to ignore such blank lines, but may never rely on their presence; an input might not have any blank lines at all).

You will then write a small report on the algorithms/methodologies you used. Include a quick description of how you translated the ASCII maze into a graph, and how you performed any path-finding/estimation algorithms. It will probably be approximately two pages (unless you're concise, in which case probably a page).

So you may prioritize the focuses of your work, the marking scheme is included below:
1. Transcribing a 2D ASCII puzzle to graph: 5 marks *
2. Showing *any* path (optimal or not) to the exit of a map: 10 marks
3. Finding the *length* of the shortest path: 10 marks
4. Actually finding that shortest path: 5 marks
5. Writeup: 10 marks
6. Discretionary (style, etc): 5 marks
Total: 45 marks

Note: You may find it easier to implement 2, 3, and 4 as separate menu options (particularly if you suspect you may not do #4). However, if you do actually complete #4, and have it report on the path's cost as part of that option, then that's sufficient to satisfy all three.

If it isn't already clear: You MUST actually transcribe the puzzle into a **graph. A 3D/2D char array is not sufficient. It's a graph problem, not an array problem. Transcribing is only worth 5 marks, but you can't get any of the other marks (other than writeup) unless you do this!*

An execution of one possible implementation is shown below:

```
Enter width: 8
Enter height: 5
Enter depth: 1
Enter maze below. Only rows of width 8 will be accepted.
XXXOOOOX
OXXXOXOE
OXXOOXXO
OSXOXXXO
XOOOXOOO

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 1
Not implemented. Use optimal instead.

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 3
Finding Solution...
Optimal Path Cost:      12

Optimal Path:
 S  E  E  N  N  E  N  N  E  E  S  E

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 5
Goodbye!
Enter width: 8
Enter height: 5
Enter depth: 1
Enter maze below. Only rows of width 8 will be accepted.
XXXOOOOX
OXXXOXOE
OXXOOXXO
OSXOXXXO
XOOOXOOO

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 1
Not implemented. Use optimal instead.
```

```
1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 3
Finding Solution...
Optimal Path Cost:      12

Optimal Path:
 S  E  E  N  N  E  N  N  E  E  S  E

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 4
Enter width: 8
Enter height: 5
Enter depth: 1
Enter maze below. Only rows of width 8 will be accepted.
XXXOOOOX
OXXXOXOE
OXXOOXXO
OSXOXXXO
XOOXXOOO

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 3
Finding Solution...
Exit not reachable.

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 4
Enter width: 5
Enter height: 5
Enter depth: 2
Enter maze below. Only rows of width 5 will be accepted.
XOOOX
OOXOE
XXXXX
OOOOO
XXXXX

XXXXX
OXXXX
OXXXX
OXXXO
XXSOO

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
```

```
4. Enter new puzzle
5. Quit
:> 3
Finding Solution...
Optimal Path Cost:      18

Optimal Path:
 E  E  N  U  W  W  W  W  D  N  N  U  E  N  E  E  S  E

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 4
Enter width: 5
Enter height: 5
Enter depth: 2
Enter maze below. Only rows of width 5 will be accepted.
XOXOX
OOXOE
XXXXX
OOOOO
XXXXX
XXXXX
OXXXX
OXXXX
OXXXO
XXSOO

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 3
Finding Solution...
Exit not reachable.

1. Solve suboptimally
2. Estimate optimal solution cost
3. Solve optimally
4. Enter new puzzle
5. Quit
:> 5
Goodbye!
```

Note that I did not implement the suboptimal search, because I did the optimal version. There's no reason for you to actually have suboptimal as an option if you do the same; I just did that to remind you that the suboptimal marks are earned via an optimal algorithm.

Submission Guidelines:
Your writeup should be in a `.pdf` file.

Electronic submission is required. Slap your files into a dedicated directory on sandcastle, ssh (or putty) in, and run 'submit2p03'.