CS2121/9643 – Project 2
due Mar. 22, 2016 (latest to submit: Mar. 25)

1. (100pt) Write a Python program `solvemaze.py` to find a way out of a maze. (Problem 7.4, page 219 of textbook; you need to implement only `findpath()` and `draw()` for the existing code or reimplement everything yourself.)

   The maze is given in a file to be read in the command line. You can either type in a terminal

   ```
   python solvemaze.py <maze.txt>
   ```

   or, in Canopy,

   ```
   run solvemaze.py <maze.txt>
   ```

   The `<maze.txt>` file contains the maze encoded as follows:

   - the first line contains the size (rows, columns)
   - the next two lines give the starting and ending positions, resp.
   - then the maze is given as a two dimensional array where the walls are marked with `*`'s; below is an example: (the start and end positions are also shown; they need not be given; also, the non-walls cells may be dots or just spaces; the only thing you can reply upon is the position of the walls as `*`'s):

   ```
   5 5
   4 1
   3 4
   *****
   *-*-*
   *---*
   *-*-E
   *S***
   ```

   If there is a solution, then the program will print a solution where the path from start to end is marked with `x`'s, as shown below (`o`'s represent failed attempts):

   ```
   Path found:
   *****
   *o*o*
   *xxx*
   *x*xx
   *x***
   ```

   If there is no solution, as it happens for this maze:

   ```
   5 5
   4 1
   3 4
   *****
   *-*-*
   *-*-*
   *-*-E
   *S***
   ```

   then the program prints:

```
There is no way out!
*****
*o*-*
*o*-*
*o*--
*o***
```

Here is a more interesting example that has a solution:

```
19 19
17 0
11 18
*******************
*-*---------------*
*-*-***-*-*********
*---*---*---------*
*-***-*********-*-*
*-*---*-------*-*-*
*-*****-*****-***-*
*---*-------*-*---*
***-*-*****-*-*-*-*
*---*-*---*-*-*-*-*
***-*-*-*-*-*-*-*-*
*---*---*-*-*---*-E
*-*******-*******-*
*-------*-*-------*
*****-*-*-*****-*
*---*-*-*---*---*-*
*-***-*-*****-*-*-*
S-----*-------*---*
*******************
```

and here is a solution:

```
Path found:
*******************
*o*xxxxxxxoooooooo*
*o*x***-*x*********
*xxx*---*xxxxxxxxx*
*x***-*********-*x*
*x*---*xxxxxxx*-*x*
*x*****x*****x***x*
*xxx*xxx----*x*xxx*
***x*x*****-*x*x*o*
*oox*x*xxx*-*x*x*o*
***x*x*x*x*-*x*x*o*
*xxx*xxx*x*-*xxx*xx
*x*******x*******x*
*xxxxx--*x*xxxxxxx*
*****x*-*x*x*****-*
*ooo*x*-*xxx*---*-*
*o***x*-*****-*-*-*
xxxxxx*-------*---*
*******************
```

Three test files are provided, corresponding to the above examples: `maze1.txt`, `maze2.txt`, and `maze3.txt`. Note that your solution found by your program may be different.

The structure of the code will not be considered for grading but it is strongly advised to implement clear logic, use meaningful names, and provide useful comments.

Note Submit your solution on `owl.uwo.ca`: include all necessary Python files as well as a `readme` file explaining how you ran the program. You may want to include the solutions you obtained to the test files given.