Leisha Figuerres
CS112
06 November 2024

Project02 Test Document

For this project, I started off by accounting for the following cases I wanted to test:

- Missing file ~ Desired output “ERROR ! File not found”
- Not rectangular: All lines have to be the same length so if one or more is off
- If the maze is unsolvable: no S or E
- If the maze is unsolvable: more than one S or E : >1 S or E
- Unallowed characters, only X, O and S E, and - are allowed
- Maze is unsolvable: S and E are not connected
- Maze that is solvable with S and E in the middle and not at edges
- Regular mazes that are solvable

To test each of these cases, I started creating mazes. At first, I made around 10-15 of them to test: nonrectangular mazes, unsolvable mazes, mazes with an inappropriate number of S’s and E’s or no S’s and E’s, mazes with characters like “#” in the maze, mazes with S and E in the middle, and regular solvable mazes.

I first built my logic for moving through the actual maze going one direction at a time starting at right first and going to others if one doesn’t work with a while loop. No backtrack logic was added yet so I tested with the basic maze given to us in class: (maze1.txt)

```
--------------------
SOOOOOO-------------
------O------OOOOOO-
------OOOOOOOO----O-
------------------O-
EOOOOOOOOOOOOOOOOOO-
--------------------
```

Once I mastered the basic movement, I started working on backtracking. I tested the other mazes I created that required backtracking and tweaked my code based on the behaviors shown in them. At first, I was able to find dead ends but not backtrack multiple times.

(maze21)

```
SOOOO-------
---O---O----
---O---O----
---OOOOOOO--
---------OOE
------------
```

Once I successfully solved that maze (which took a while), I created more complex mazes and ran into multiple issues when testing other mazes like maze32 which tested another case of backtracking after going up and reaching a dead end.

```
---------
-O---O---
-O-O-O-O-
-S---O---
-O-O-O-E-
-------O-
---------
```

This maze showed up as unsolvable and would get stuck at the bolded O and not make its way down so I knew I had an issue with backtracking. I also tested these mazes :

(maze11)

```
S----O----O----
OOOOO--O----O--
---O---O----O--
---O-O----O----
---O-OOOOOOOOOE
---------------
```

(maze30)

```
-----------
---O-O-O---
---O---O---
---O-O-O---
---S---E---
---O-O-O---
---O---O---
-----------
```

(maze40)

```
-----------
-SO-O-O-O-O
-OO-OOO-O-O
-O---O---OO
-OOOO-OOO-O
---O-----O-
-OOOOOOOO-E
----------
```

These mazes gave me ArrayIndexOutOfBounds exceptions. After some help and getting new perspectives from the T.A, I realized that looking at previous indices was the cause of all this and after getting rid of that in my backtracking logic and focusing solely on when to use backtracking (no other paths and the only other path is a revisited queue), I was able to successfully mark up and solve the mazes that gave me a hard time.

I kept creating more complex solvable and unsolvable mazes to test and would document results and ones that were successful/gave me issues. After fixing my issues, I tested the project in its entirety by going through every single maze I created (about 32 of them) to test each of the issues I listed in the beginning. I wanted to see if my backtracking and marking were correct before adding a cleanup method and not printing out a maze if it's unsolvable. Here are some of my results:

(maze11) Tested having to backtrack all the way to the end because there's no solution

```
S----O----O----
bbbbb--O----O--
---b---O----O--
---b-O----O----
---b-OOOOOOOOOE
---------------
```

(maze30) Tested having to backtrack all the way to the end because there's no solution

```
-----------
---b-O-O---
---b---O---
---b-O-O---
---S---E---
---O-O-O---
---O---O---
-----------
```

(maze44)

```
ERROR ! No solution found.
-----------
-Sb-O-O-O-O
-bb-OOO-O-O
-b---O---OO
-bbbb-OOO-O
---b-----O-
-bbbbbbbb-E
-----------
```

Maze13 Tested having a S and E in the middle

```
----------------
------O----O----
---O---O----O---
-S3----O----E---
--23333333334bb-
----------------
```

After I tested that my marking and solving logic was correct I added my cleanup method, refined my error messages, and tested every single maze and error scenario I listed again. (I did this multiple times) Here are some examples:

Test case: Nonexistent file

input: maze16:

output: ERROR ! File is not found. **program terminates

Test case: not rectangular

maze2

```
SOOOOOO
----O------OOOOOO-
------OOOOOOOO----O-
------------------O-
EOOOOOOOOOOOOOOOOOO-
--------------------
```

output: ERROR ! Not rectangular. ** program terminates

Test Case: No E or S

maze4

```
--------O------
-O---O---------
----OOOOOO--O--
----O----------
---O--OOOOOOOOO
---------------
```

Output: ERROR ! Missing a start, or end, or both ** program terminates

Test case: Multiple E’s

maze7

```
S----O----O----
-OO---O------O-
---O---OOOOO---
-------O-------
EOOOOOOOOOOOO-E
```

Output: ERROR ! Multiple start or end characters. ** program terminates

Test Case: Invalid characters

Maze9

```
S----O----O----
-OOO-----O-----
---O-OOO-------
```

```
-----O---------
EOOOOOO#OOOOOO-
---------------
```

Output: ERROR ! Invalid Characters. ** program terminates

Test Case: Unsolvable S not connected to any O’s

Maze10

```
S----O----O----
--O---O---O----
----O-OOOOO----
----O----------
EOOOOOOOOOOOOOO
```

Ouput: ERROR ! No solution found. ** program terminates

Test Case: S and O’s are connected but had to backtrack to start

Mazes 11, 30, 44.

<u>Test cases: Solvable mazes</u>

Maze 46

```
--------------
-O-O-O-O-O-O-O
-O---O-OOO-O-O
-O-XXXXXOO-O-O
-O-X---X---O-O
-O-XXX-XXXXX-O
-O---S-O---E-O
-OOOOOOO-O-O-O
-O-------O-O-O
-OOOOOOOOO-O-O
--------------
```

Maze 43

maze 43

```
-OOOOOOOO--
-SXXXXXXX--
-OO---OOX--
-O-O-O-OXX-
-OOOOO---X-
-----O-OOXO
-OOOOO---E-
----------
```

Maze 33

```
-----------
---O-O-O---
---O---O---
-XXXXXXXXX-
-S-------E-
-OOOOOOOOO-
---O---O---
---O-O-O---
-----------
```