CS302

Assignment 2: Zelda

Description

Ah yes, Zelda. One of the greatest Nintendo games of all time. As always, you play as Link and if you played Breath of the Wild, you remember that you have to explore the terrain (unlock shrines, unlock divine beasts, etc) before you take on Calamity Ganon, otherwise you would have no chance to beat the game. As we know Link has been in stasis mode for over 100 years and Princess Zelda gives a tablet to Link to aid him in his journey. Design a program that runs on the tablet that reads in a 7x7 grid and finds a path that has Link visit every location from the start point to the end point, which allows Link to be powered up enough to defeat Calamity Ganon.

Since the path is not going to be trivial, we will need to use recursive backtracking (that will involve computing possibly every possible path until the right path is finally determined).

Input

Your program reads an input file (from standard input), and your program will use a filestream variable to read in the 7x7 character board, then the user (from standard) input will enter the starting and ending coordinates. You can assume all input will be valid.

Output

Output the board that displays the step counter (0-9 A-Z), Sample Run section for more details

Contents of Main

You will need to write a recursive function to traverse the grid, an idea of a function prototype can be seen below

```
bool foo(char grid[][7], int x, int y, int stepCounter);
```

You can add additional parameters as needed. The parameters I provided are the grid (where your recursive function will traverse), the x and y coordinates of the location the recursive function is currently on, and the step counter that is used to write to denote the current step (increments by 1 for every recursive function call). The function returns a boolean value, if a `true` is returned at any time that denotes your recursive function found a path, if a `false` is returned then the path taken by the recursive function did lead to the solution so a new path needs to be constructed.

You may also have global variables in this assignment (to avoid having too many parameters for the recursive function), so the start and end coordinates can be a global variable since they will not change throughout the duration of the program. Some details as to how you can think of the recursive function is explained below

1. When you enter the function, if all previously vacant spaces contains a step counter and the recursive function is currently on the destination coordinates, return `true` (a solution is found)
2. If the recursive function is currently on the final coordinate and not all the previously vacant spots are filled, then return `false` (you arrived to the final destination too early)
3. General Case: the function is currently at coordinate [x, y], you check for a vacant spot in any direction (left, right, up, and down), whichever direction you chose if the spot is vacant and not out of bounds, you place a step count on this spot and then call the function recursively (by passing in the updated parameters i.e. the x and y coordinates and the step counter).
4. Eventually the function returns back, if a `true` is returned then the path was found so relay this `true` value back and just return `true`, else unmark the location that you just went to, and try a different direction (same idea as in the last step), if all directions are tried and none of them led to a true being returned, then return `false` (this allows the previous function to try and re-compute its path)

Specifications

- Must use a recursive function
- Document your code
- There can be more than one solution for this assignment, so if your output does not match code grade but it is a valid path, we will not take off any points

Sample Run

This was updated after the video was recorded

```
% g++ main.cpp
% ./a.out

Enter config file: puzzle01.txt

Please enter starting coordinates: 0 1

Please enter end coordinates: 6 5

_ 0 1 A B _ _
_ _ 2 9 C _ _
_ 4 3 8 D _ _
_ 5 6 7 E _ _
_ _ _ G F _ _
_ _ I H _ _ _
_ _ J K L M _

% ./a.out

Enter config file: puzzle02.txt

Please enter starting coordinates: 4 0
```

```
Please enter end coordinates: 1 6


- - - - - - - -
- - 7 8 H I R
- - 6 9 G J Q
1 2 5 A F K P
0 3 4 B E L O
- - - C D M N
- - - - - - - -

% ./a.out

Enter config file: puzzle03.txt

Please enter starting coordinates: 6 6

Please enter end coordinates: 0 0

W V S R E D _
_ U T Q F C B
_ _ O P G 9 A
_ _ N I H 8 _
_ _ M J 6 7 _
_ _ L K 5 2 1
_ _ _ _ 4 3 0

% ./a.out

Enter config file: puzzle04.txt

Please enter starting coordinates: 0 6

Please enter end coordinates: 3 4

K L M N O P 0
J U T S R Q 1
I V _ _ _ _ 2
H W X Y Z _ 3
G F _ _ _ _ 4
D E _ _ _ _ 5
C B A 9 8 7 6
```

Submission

Submit the source files to code grade by the deadline

References

- Supplemental Video https://youtu.be/GqX3tkSvRYU
- Link to the top image can be found at https://www.pinclipart.com/maxpin/oJihRx/