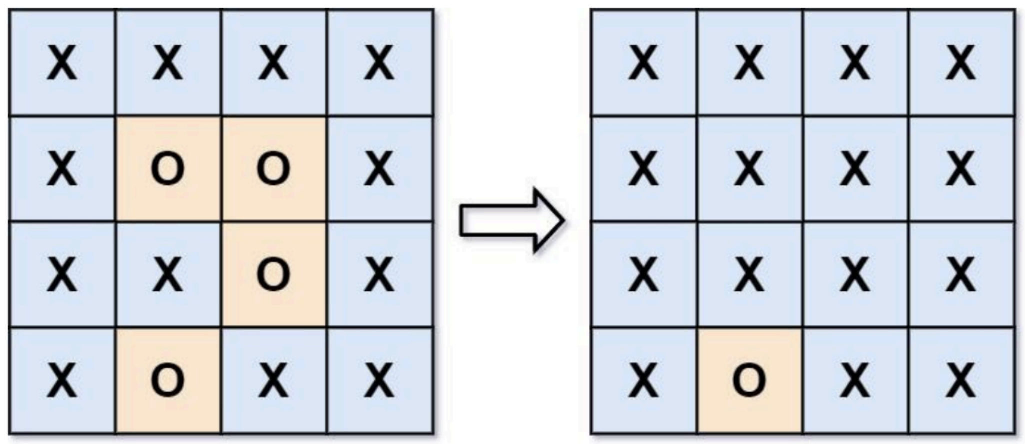
Python Assignment 1

Problem Statement:

Given an m*n matrix board containing 'X' and 'O', Capture all regions that are 4-directionally surrounded by 'X'.
A region is captured by flipping all 'O's into 'X's in that surrounded region.



Input: board = [["X","X","X","X"],["X","O","O","X"],["X","X","O","X"],["X","O","X","X"]]
Output: [["X","X","X","X"],["X","X","X","X"],["X","X","X","X"],["X","O","X","X"]]

Approach:
I used Depth-First Search (DFS) to find and mark 'O' cells connected to the border. By starting DFS from 'O' cells on the border, I identified regions that cannot be surrounded by 'X'. Then, I flipped internal 'O' cells to 'X', preserving border 'O' cells. This approach modifies the matrix, ensuring only surrounded 'O' regions are changed to 'X'.

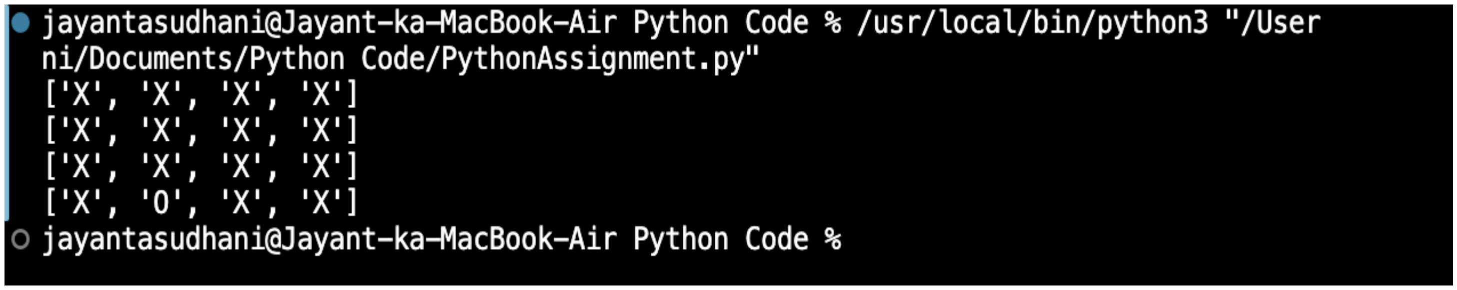
Time Complexity:

Overall time complexity is **O(m * n)**, where
"m" is the number of rows
"n" is the number of columns in the input matrix.

Code:

```
class Solution(object):
   def solve(self, board):
       m = len(board)
       n = len(board[0])

       # Depth-First Search function to mark connected 'O' cells
       def dfs(i, j):
           # Base case: check if the cell is out of bounds or not 'O'
           if(i < 0 or j < 0 or i >= m or j >= n or board[i][j] != 'O'):
               return
           # Mark the current 'O' cell as visited
           board[i][j] = '#'
           # Explore adjacent cells in all four directions
           dfs(i + 1, j)
           dfs(i, j + 1)
           dfs(i - 1, j)
           dfs(i, j - 1)

       # Step 1: Mark 'O' cells on the border as visited
       for i in range(m):
           for j in range(n):
               # If the current cell is on the border and contains 'O'
               if((i == 0 or i == m - 1 or j == 0 or j == n - 1) and
board[i][j] == 'O'):
                   dfs(i, j)

       # Step 2: Flip 'O' cells to 'X' and restore marked cells to 'O'
       for i in range(m):
           for j in range(n):
               if board[i][j] == 'O':
                    board[i][j] = 'X'

        # Step 3: Restore marked cells ('#') to 'O'
        for i in range(m):
            for j in range(n):
                if board[i][j] == '#':
                    board[i][j] = 'O'

#Main method to demonstrate working of code
def main():
   # Example input
   input_board = [
       ["X", "X", "X", "X"],
       ["X", "O", "O", "X"],
       ["X", "X", "O", "X"],
       ["X", "O", "X", "X"]
   ]

   solution = Solution()
   solution.solve(input_board)

   # Display the modified board
   for row in input_board:
       print(row)



if __name__ == "__main__":
   main()
```

Output:



```
● jayantasudhani@Jayant-ka-MacBook-Air Python Code % /usr/local/bin/python3 "/User
  ni/Documents/Python Code/PythonAssignment.py"
  ['X', 'X', 'X', 'X']
  ['X', 'X', 'X', 'X']
  ['X', 'X', 'X', 'X']
  ['X', 'O', 'X', 'X']
○ jayantasudhani@Jayant-ka-MacBook-Air Python Code %
```