# Leetcode 2510.
# Check if There is a Path With Equal Number of 0's And 1's

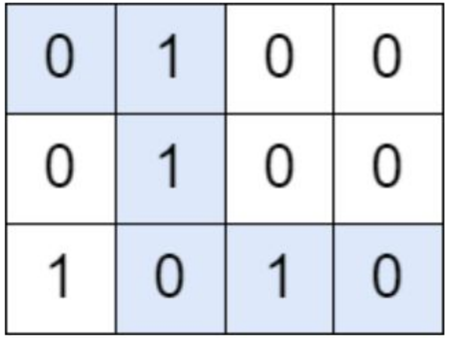
#Array #Dynamic Programming #Matrix

## Problem Description

You are given a **0-indexed** m x n **binary** matrix grid. You can move from a cell **(row, col)** to any of the cells **(row + 1, col)** or **(row, col + 1).**

Return **true** if there is a *path* from **(0, 0)** to **(m - 1, n - 1)** that **visits** an **equal** number of 0's and 1's. Otherwise return false.

**Example 1:**



**Input:** grid = [[0,1,0,0],[0,1,0,0],[1,0,1,0]]
**Output:** true

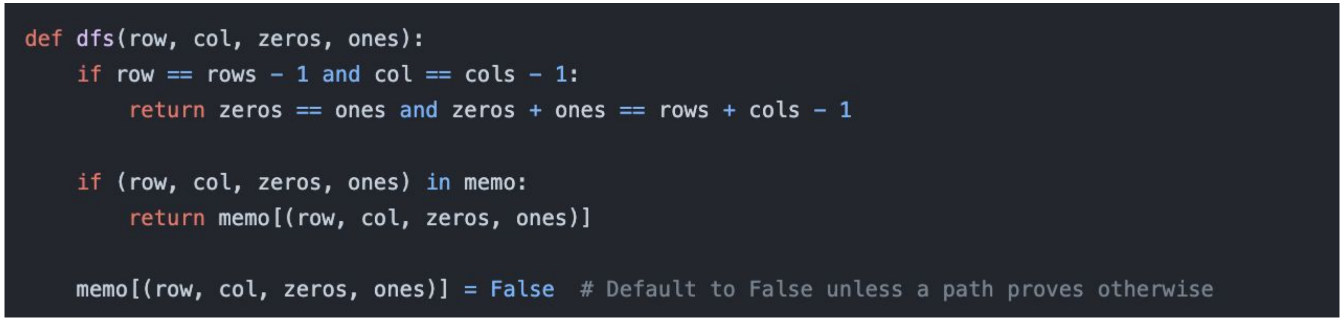
Explanation: The path colored in blue in the above diagram is a valid path because we have 3 cells with a value of 1 and 3 with a value of 0. Since there is a valid path, we return true.

# Dynamic Programming: Depth-First Search (DFS) with memoization

The dfs function is defined to recursively explore possible paths from the current position (row, col) with a given count of zeros and ones encountered.

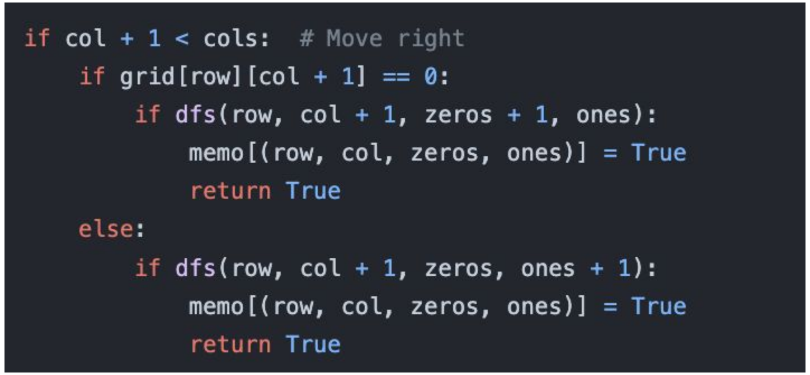
def dfs(**row, col,** zeros, ones):

- Current position **(row, col)**
- # of zeros
- # of ones

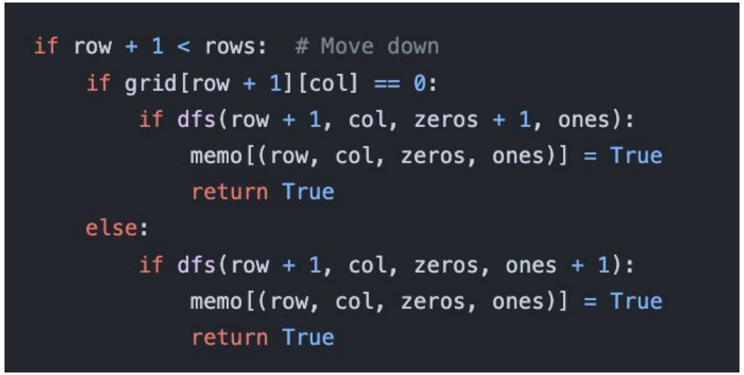


```python
def dfs(row, col, zeros, ones):
    if row == rows - 1 and col == cols - 1:
        return zeros == ones and zeros + ones == rows + cols - 1

    if (row, col, zeros, ones) in memo:
        return memo[(row, col, zeros, ones)]

    memo[(row, col, zeros, ones)] = False  # Default to False unless a path proves otherwise
```

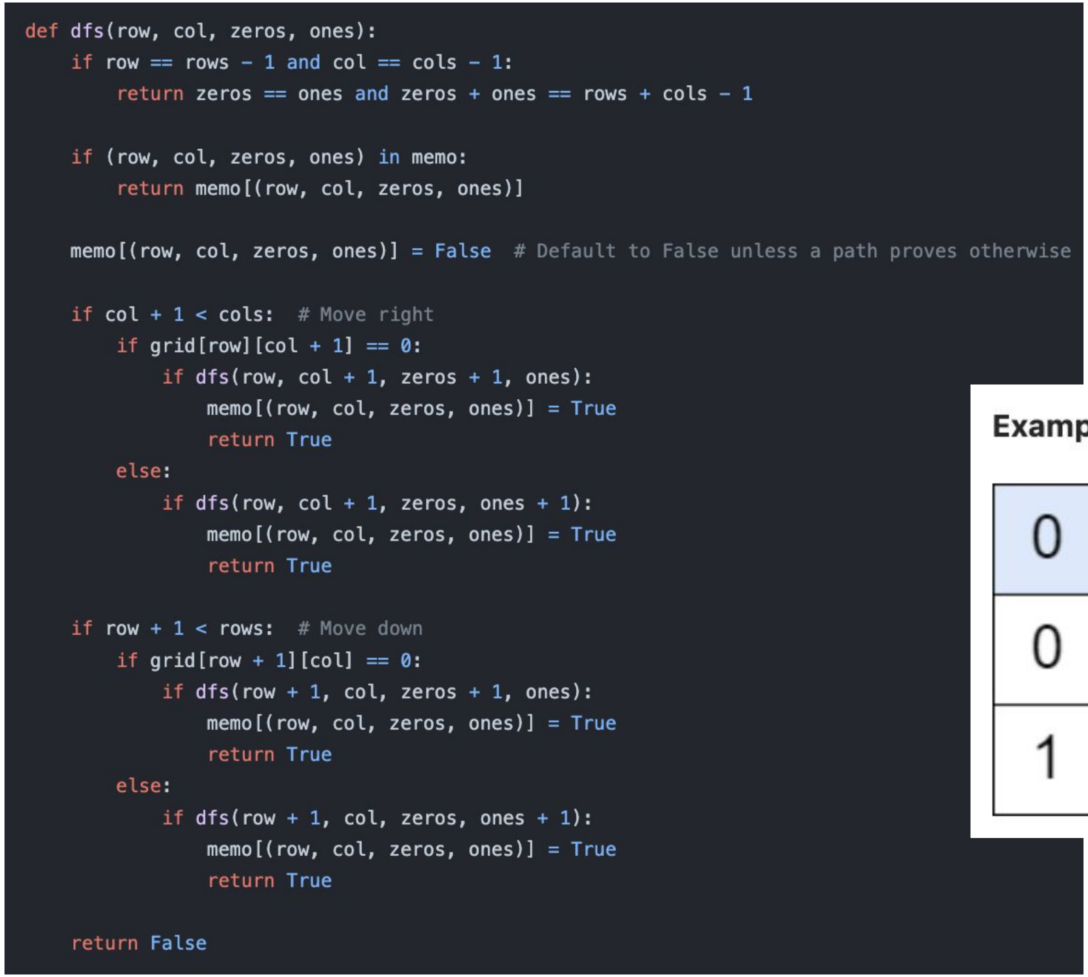
## Dynamic Programming:
## Depth-First Search (DFS) with memoization



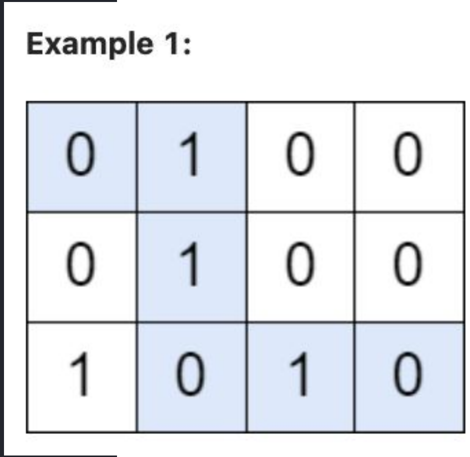
```python
if col + 1 < cols:  # Move right
    if grid[row][col + 1] == 0:
        if dfs(row, col + 1, zeros + 1, ones):
            memo[(row, col, zeros, ones)] = True
            return True
    else:
        if dfs(row, col + 1, zeros, ones + 1):
            memo[(row, col, zeros, ones)] = True
            return True
```



```python
if row + 1 < rows:  # Move down
    if grid[row + 1][col] == 0:
        if dfs(row + 1, col, zeros + 1, ones):
            memo[(row, col, zeros, ones)] = True
            return True
    else:
        if dfs(row + 1, col, zeros, ones + 1):
            memo[(row, col, zeros, ones)] = True
            return True
```



```python
def dfs(row, col, zeros, ones):
    if row == rows - 1 and col == cols - 1:
        return zeros == ones and zeros + ones == rows + cols - 1

    if (row, col, zeros, ones) in memo:
        return memo[(row, col, zeros, ones)]

    memo[(row, col, zeros, ones)] = False  # Default to False unless a path proves otherwise

    if col + 1 < cols:  # Move right
        if grid[row][col + 1] == 0:
            if dfs(row, col + 1, zeros + 1, ones):
                memo[(row, col, zeros, ones)] = True
                return True
        else:
            if dfs(row, col + 1, zeros, ones + 1):
                memo[(row, col, zeros, ones)] = True
                return True

    if row + 1 < rows:  # Move down
        if grid[row + 1][col] == 0:
            if dfs(row + 1, col, zeros + 1, ones):
                memo[(row, col, zeros, ones)] = True
                return True
        else:
            if dfs(row + 1, col, zeros, ones + 1):
                memo[(row, col, zeros, ones)] = True
                return True

    return False
```



**Example 1:**

| | | | |
|---|---|---|---|
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 |

# 감사합니다!

THANK YOU