

## Problem of the Week – *Word Search in 2D Matrix(Easy)*

*This problem was asked by Microsoft.*

### Problem Description:

#### Scenario:

In many document processing, word puzzles, and image processing systems, scanning a grid for a target pattern is a common task. You are given a **2D matrix of characters**, and a **target word**. Your task is to **check if the word exists in the matrix** either:

- Left-to-right (horizontally)
  - Top-to-bottom (vertically)
- 

### Input Format:

- A 2D character matrix of size **M x N**
  - A string representing the **target word**
- 

### Output Format:

- Return `true` if the word exists in the matrix (either row-wise left to right or column-wise top to bottom)
  - Else return `false`
- 

### Constraints:

- $1 \leq M, N \leq 100$
  - Word length  $\leq \max(M, N)$
  - All characters are uppercase English letters
- 

### Example Input:

```
matrix = [  
    ['F', 'A', 'C', 'I'],  
    ['O', 'B', 'Q', 'P'],  
    ['A', 'N', 'O', 'B'],  
    ['M', 'A', 'S', 'S']  
]  
word = "FOAM"
```

---

## Example Output:

True

---

## Explanation:

- "FOAM" appears in the **first column**:  $F \rightarrow O \rightarrow A \rightarrow M$  (top to bottom)
  - "MASS" appears in the **last row**:  $M \rightarrow A \rightarrow S \rightarrow S$  (left to right)
- 

## Approach Hint:

1. **Scan all rows** for the word (convert to string and check substring)
  2. **Scan all columns** by transposing the matrix or looping column-wise
  3. Use simple loops or string slicing — no need for backtracking
- 

## Expected Time Complexity:

- $O(M \times N)$
- 



## Sample Starter Code (Python):

```
def word_search(matrix, word):
    rows = len(matrix)
    cols = len(matrix[0])

    # Check rows
    for row in matrix:
        if word in ''.join(row):
            return True

    # Check columns
    for col in range(cols):
        col_str = ''.join(matrix[row][col] for row in range(rows))
        if word in col_str:
            return True



    return False

# Sample Test
matrix = [
    ['F', 'A', 'C', 'I'],
    ['O', 'B', 'Q', 'P'],
```

```
    ['A', 'N', 'O', 'B'],  
    ['M', 'A', 'S', 'S']  
]  
print(word_search(matrix, "FOAM")) # Output: True  
print(word_search(matrix, "MASS")) # Output: True  
print(word_search(matrix, "FACE")) # Output: False
```

---

## Practice Links:

-  [Leetcode – Word Search I \(Extended Problem\)](#)
-  [GeeksforGeeks – Search a Word in a 2D Grid of Characters](#)