CHANAKYA
UNIVERSITY

DSA
ASSIGNMENT-2

SUBMITTED BY - PRATEEK JOSHI

REG ID – 24UG00534

Problem statement

You are required to model a 2D grid of characters as a graph, where each cell acts as a node, and edges exist between horizontally and vertically adjacent cell (no diagonal connections). Given a target word, your program should search for occurrences of the word in the grid horizontally (left to right) or vertically (top to bottom). For each occurrence of the word not found, print the start node and end node coordinates. If there are no occurencess, print word not found.

The code:

```
#include <stdio.h>

#include <string.h>


#define MAX 100


void searchWord(char grid[MAX][MAX], int m, int n, char word[]) {
    int wordLen = strlen(word);
    int found = 0;


    // Horizontal search (left to right)
    for (int i = 0; i < m; i++) {
        for (int j = 0; j <= n - wordLen; j++) {
            int k;
            for (k = 0; k < wordLen; k++) {
                if (grid[i][j + k] != word[k])
                    break;
            }
            if (k == wordLen) {
                printf("Start: (%d, %d) End: (%d, %d)\n", i, j, i, j + wordLen - 1);
                found = 1;
            }
        }
    }
```

```
    // Vertical search (top to bottom)
    for (int j = 0; j < n; j++) {
        for (int i = 0; i <= m - wordLen; i++) {
            int k;
            for (k = 0; k < wordLen; k++) {
                if (grid[i + k][j] != word[k])
                    break;
            }
            if (k == wordLen) {
                printf("Start: (%d, %d) End: (%d, %d)\n", i, j, i + wordLen - 1, j);
                found = 1;
            }
        }
    }

    if (!found)
        printf("Word not found\n");
}

int main() {
    int m, n;
    char grid[MAX][MAX], word[MAX];

    printf("Enter number of rows (m): ");
    scanf("%d", &m);
    printf("Enter number of columns (n): ");
    scanf("%d", &n);

    printf("Enter the grid row by row (uppercase letters, no spaces):\n");
```

```
    for (int i = 0; i < m; i++) {
        scanf("%s", grid[i]);
        if (strlen(grid[i]) != n) {
            printf("Error: Each row must have exactly %d letters.\n", n);
            return 1;
        }
    }

    printf("Enter the word to search: ");
    scanf("%s", word);

    // Convert to uppercase (optional)
    for (int i = 0; word[i]; i++) {
        if (word[i] >= 'a' && word[i] <= 'z')
            word[i] -= 32;
    }

    printf("\nGrid:\n");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++)
            printf("%c ", grid[i][j]);
        printf("\n");
    }

    printf("\nOccurrences:\n");
    searchWord(grid, m, n, word);

    return 0;
}
```

Output:

Output

```
Enter number of rows (m): 8
Enter number of columns (n): 8
Enter the grid row by row (uppercase letters, no spaces):
PRATEEKP
KJSDHAKR
RAHULISA
VAMSHIDT
JOBDONEE
JOBNOTDE
IHHIJHJK
PRATEEKO
Enter the word to search: PRATEEK

Grid:
P R A T E E K P
K J S D H A K R
R A H U L I S A
V A M S H I D T
J O B D O N E E
J O B N O T D E
I H H I J H J K
P R A T E E K O

Occurrences:
Start: (0, 0) End: (0, 6)
Start: (7, 0) End: (7, 6)
Start: (0, 7) End: (6, 7)


=== Code Execution Successful ===
```