https://course.acciojob.com/idle?question=ca9e2848-0a0f-4b74-9fc0-048f5e4a7295

- MEDIUM
- Max Score: 40 Points

Word Search

You are given a `m x n` grid of characters `board` and a string `word`, your task is to find if the `word` exists in the grid, return `true` if it exists, else return `false`.

The word can be constructed from letters of the adjacent cells, where the adjacent cells are horizontally or vertically neighbouring.

Note

Same letter can not be used more than once.

Input Format

The first line of input contains `m` and `n`, representing the number of rows and collumns, respectively.

The next `m` lines contains `n` space seperated characters, representing the grid.

The next line contains the string representing `word`, which we have to search.

Output Format

Return true or false, if the word exists in the grid.

Example 1

Input

```
3 4
A B C E
S F C S
A D E E
ABCCED
```

Output

```
true
```

Explanation

The word ABCCED is present in the grid strating from index (0,0)->(0,1)->(0,2)->(1,2)->(2,2)->(2,1).

Example 2

Input

```
3 4
A B C E
S F C S
A D E E
ABCB
```

Output

```
false
```

Explanation

The word ABCB is not present in the grid.

Constraints

1 <= m,n <= 6

1 <= |word| <= 15

Topic Tags

- Recursion

My code

```
// n java
import java.util.*;
class Solution
```

```
{
        static boolean seartch(char arr[][],String s,int r,int c,int i,int
j,int ind,int vis[][])
 {
         if(ind>=s.length())//keep first because if string found on
boundry
                  return true;
        if(i<0 || i>r-1 ||j <0 ||j>c-1 || vis[i][j]==1 )
                return false;

         if(s.charAt(ind)!=arr[i][j])
                  return false;
         vis[i][j]=1;
         boolean a=seartch(arr, s,r,c,i,j-1,ind+1,vis);
          boolean b=seartch(arr, s,r,c,i,j+1,ind+1,vis);
          boolean f=seartch(arr, s,r,c,i+1,j,ind+1,vis);
          boolean d=seartch(arr, s,r,c,i-1,j,ind+1,vis);
          vis[i][j]=0;
         return (a || b || f|| d);

 }
   public boolean solve(char board[][],String word)
   {
       // your code here

       int r=board.length;
              int c=board[0].length;
              int vis[][]=new int[r][c];
              for(int i=0;i<r;i++)
                     for(int j=0;j<c;j++)
```

```
                if(seartch(board, word,r,c,i,j,0,vis))
                        return true;
                return false;
    }

}
class Main {

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int m=sc.nextInt();
        int n=sc.nextInt();
        char board[][]=new char[m][n];
        for(int i=0;i<m;i++)
        {
            for(int j=0;j<n;j++)
            board[i][j]=sc.next().charAt(0);
        }
        sc.nextLine();
        String word=sc.nextLine();
        Solution obj=new Solution();
        boolean ans=obj.solve(board,word);
        if(ans==true)
        System.out.println("true");
        else
        System.out.println("false");
        sc.close();
    }
}
```