

Word Search

time limit per test: 1 second
memory limit per test: 976.6 megabytes
input: standard input
output: standard output

points: 100 · problem code: wordsearch · author: kahootist

Problem Statement

You are playing word search!

The rules of the game is simple: You are given an $n \times m$ grid of letters. You want to find if a given word, w , exists on a straight line in the grid of letters.

For example, if the grid you are playing on is as follows:

```
WGTNPOTSFXQQ
OVOEWPCNRR LX
BDZVNXWLCXJM
NBEZPRLGXHVU
OTTPDIMJSYPJ
XNEKTUFLSFVF
JKPYJHJCYVPR
FLAUQHFWKVVA
RIKDMPJFNUJG
VTDVCECFCPXK
```

and if you're asked to find the word `DEPTH` from it, you would be able to find it with the starting letter `D` starting from 2nd column, 3rd row:

```
.....
.....
.D.....
..E.....
...P.....
....T.....
.....H.....
.....
.....
.....
```

The word can go in any direction as long as it is in a straight line: East, North, South, West, North-East, North-West, South-East, or South-West.

As a smart kid like you are who even attends MAPS workshops, you have decided to write yourself a little algorithm that could solve word searches for you.

Input

Your first line will contain n and m as space-separated integers. n and m represents the size of the letter grid.

Your next line will contain w , a string of the word you are trying to search for.

Your next m lines will contain n letters each, representing a single row of the letter grid.

Output

The string `FOUND` if the word w is present in the letter grid, `nope` otherwise.

Constraints

For all test cases:

- $1 \leq n, m \leq 10^4$

- $1 \leq |w| \leq \max(n, m)$; that is, the word w will always be able to fit on the grid.

Example 1

Input

```
12 10
DEPTH
WGTNPOTSFXQQ
OVOEWPCNRRLX
BDZVNXWLCXJM
NBEZPRLGXHVU
OTTPDIMJSYPJ
XNEKTUFLSFVF
JKPYJHJCYVPR
FLAUQHFWKVVA
RIKDMPJFNUJG
VTDVCECFCPXK
```

Output

```
FOUND
```

Example 2

Input

```
4 7
MONASH
QKNP
PAOA
SNMA
UHAJ
UNHS
MBSY
AOAE
```

Output

```
nope
```

Example 3

Input

```
40 36
SUPERCALIFRAGILISTICEXPIALIDOCIOUS
OOTFOBAIZIWIUDEKMUYTWIAAZLQEPQQYVFILJEVX
DNRRGLJRHJDFILAWZXC PJGGOFDKAXDLIDFDLLZUK
WXGCCMCWJIBTTDDTQSBFNFWRNQZCNAXUVKAHNNUO
MWAHYIAWGAFCEPFKLUJNKZKIRZMZVXKZRTJNNADR
BXNOKPUNCAECBZEAKOGRCTAUYLMZDRGQKFOLDBJZ
QITUYPONZRHCBBCFIWYNQORTGOXEDAVHZTNIZND
ETGOBLKTMLAZJGTUACRJRRANJYTHCRJTHLJYYXIJ
MBPVRUOKWGPJHOUZOUUIFWZBNAEJQB JMC IYNQRZ
OENVWIOOCAMXIXBAMDFNKHFNBAAPLCGUJGMHEKDO
HTQE0JJIGMJQVIIFGIEERMYODPCAPYXFZMPNATC
BHGHWLAEFXKFLONNELRJHJWBURAQQJIO0FKXBHXC
HHHKENXWFLVFFRPAXADFZIBHOTQJZFWJJKYGXCD0
WJRMEWXP TZQOHAMQUIEAQYEPJZBZLCNRQXLOBQRU
PMRWAKNVDGRHDGQFGPOMTJPF0FFWHYBRXATHPHLD
OVIFBNPYUCFJIKWNNXOYYNAEQGAQDKKNRCZOTGAX
IKBULMKXLYWQWWINKELBBWVFPTQYEPHKKKULADLT
CYRFRHIDQWPPMXARZCGDEMHTTJCURIXVGGRIFFCG
OGBBJKFKBFFXWABJEIEKPJGVWVLIQFVXRQOQDOOU
KRKQRBADPECOAEKJTEAXAHVIYUKNCNVEZDFHJEE
QMNRRNARCYFGNLYXCSXEZVGRYCXRI LUKYINDMGE
GEDBIBZQRJBJMXKMRIXCEWVLLROVEPCRKWUVVELG
MXHPMJMDOZAFYCUUTLYLCXXFBFKPXGAEBQCYPZRV
WRMPVNIHYPFZXBDDOI XBHPVKAJCKKVIBVAQMRPV
HBNYGBKFVCWTPZREPGGDRBHQDIAXIRFMKNTDPOLQ
TJRVIYOAREOZQDEJOAYIQVXWYMPNJTWUTKGRKBBK
DVCDIEMRAATXLOJIERTGKYARTLLHITDRPRMVGBUB
CMXILELKURQKBDHVTFHQLCKNJAMGNRFPALGRVAIJ
GKLEKYJPIEFDWNIKCICFIQMZACWALKYDAXOUPRWN
RGRZBHCWHTORTXVZNLYTFWEAXBBUWHRVJ PYWHARP
PVTHFGGKFJWVADPVGAA00NDNWJWLAZMFVXVJMAZFK
LPLYIFXIJJKLBMVJRCCJUJMPVFQTQXMOBQWVEFYX
LDHRJBZKCAEJVVVPZGRIZZETCNZIYCQACBMWGCAEJ
MKJTIQKZJGGKKNNAHEPTGKAZJWNVFNKMRPMOBFJN
OCIWIMHTGJZIXPULNPWYIGNQYRONTAUIPXXUXAVP
EFREVDIOYVIKXQTBUIJGWD TNIIHHERDXVODYWYRI
RINBUWOQZXBKFRVVKSYXFBJLHFIIAHMMVJRGLTCO
```

Output

```
FOUND
```

Python Template

```
n, m = map(int, input().split())
word = input()
board = [list(input()) for _ in range(m)]

# print(...)
```