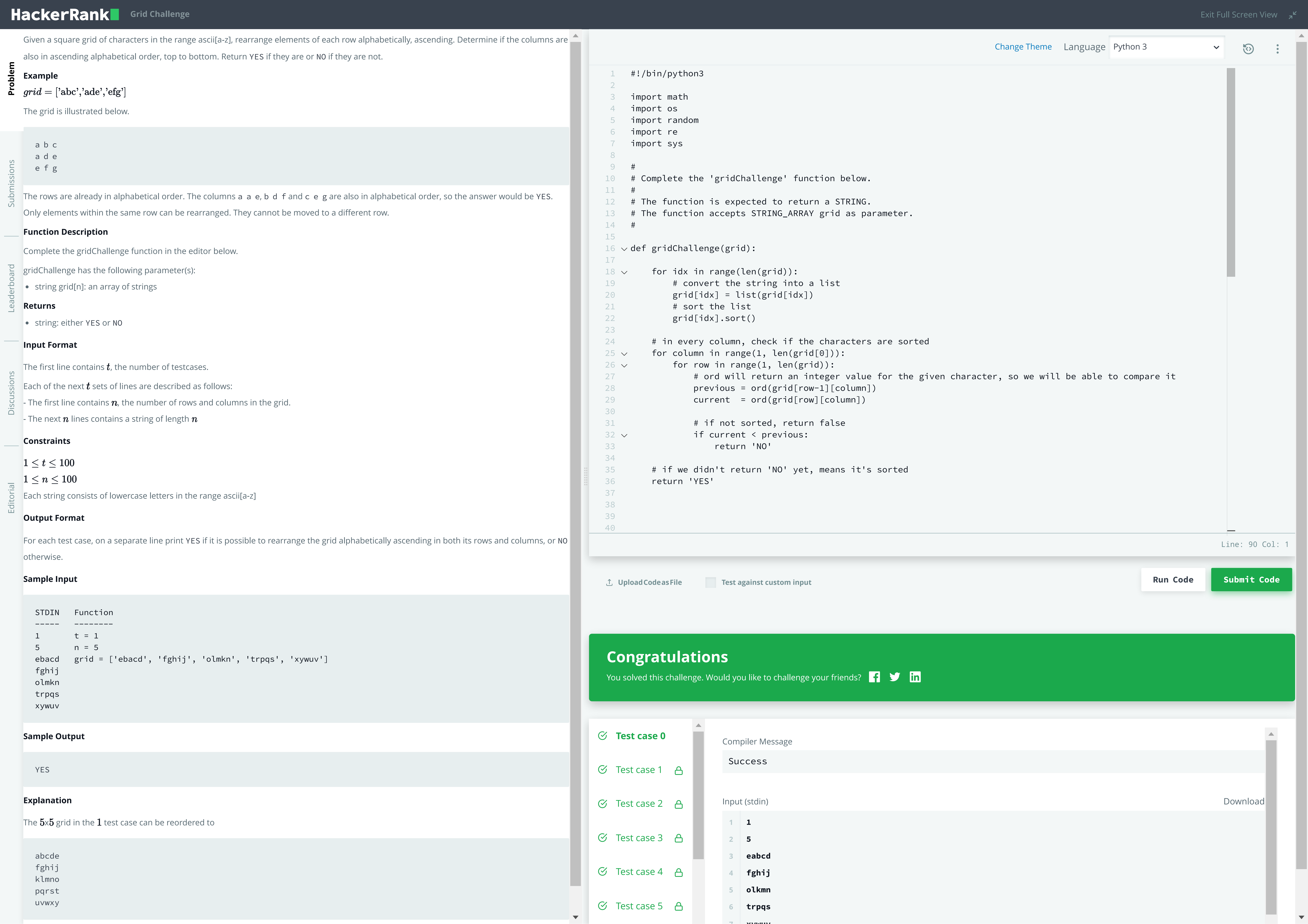

Given a square grid of characters in the range ascii[a-z], rearrange elements of each row alphabetically, ascending. Determine if the columns are also in ascending alphabetical order, top to bottom. Return YES if they are or NO if they are not.

Example

$grid = ['abc', 'ade', 'efg']$

The grid is illustrated below.

```
a b c
a d e
e f g
```

The rows are already in alphabetical order. The columns `a a e`, `b d f` and `c e g` are also in alphabetical order, so the answer would be YES. Only elements within the same row can be rearranged. They cannot be moved to a different row.

Function Description

Complete the gridChallenge function in the editor below.

gridChallenge has the following parameter(s):

- string grid[n]: an array of strings

Returns

- string: either YES or NO

Input Format

The first line contains t, the number of testcases.

Each of the next t sets of lines are described as follows:

- The first line contains n, the number of rows and columns in the grid.

- The next n lines contains a string of length n

Constraints

$1 \leq t \leq 100$
$1 \leq n \leq 100$
Each string consists of lowercase letters in the range ascii[a-z]

Output Format

For each test case, on a separate line print YES if it is possible to rearrange the grid alphabetically ascending in both its rows and columns, or NO otherwise.

Sample Input

```
STDIN   Function
-----   --------
1       t = 1
5       n = 5
ebacd   grid = ['ebacd', 'fghij', 'olmkn', 'trpqs', 'xywuv']
fghij
olmkn
trpqs
xywuv
```

Sample Output

```
YES
```

Explanation

The 5x5 grid in the 1 test case can be reordered to

```
abcde
fghij
klmno
pqrst
uvwxy
```

```python
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'gridChallenge' function below.
#
# The function is expected to return a STRING.
# The function accepts STRING_ARRAY grid as parameter.
#

def gridChallenge(grid):

    for idx in range(len(grid)):
        # convert the string into a list
        grid[idx] = list(grid[idx])
        # sort the list
        grid[idx].sort()

    # in every column, check if the characters are sorted
    for column in range(1, len(grid[0])):
        for row in range(1, len(grid)):
            # ord will return an integer value for the given character, so we will be able to compare it
            previous = ord(grid[row-1][column])
            current  = ord(grid[row][column])

            # if not sorted, return false
            if current < previous:
                return 'NO'

    # if we didn't return 'NO' yet, means it's sorted
    return 'YES'
```

Line: 90 Col: 1

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

Input (stdin) Download

```
1
5
eabcd
fghij
olkmn
trpqs
```