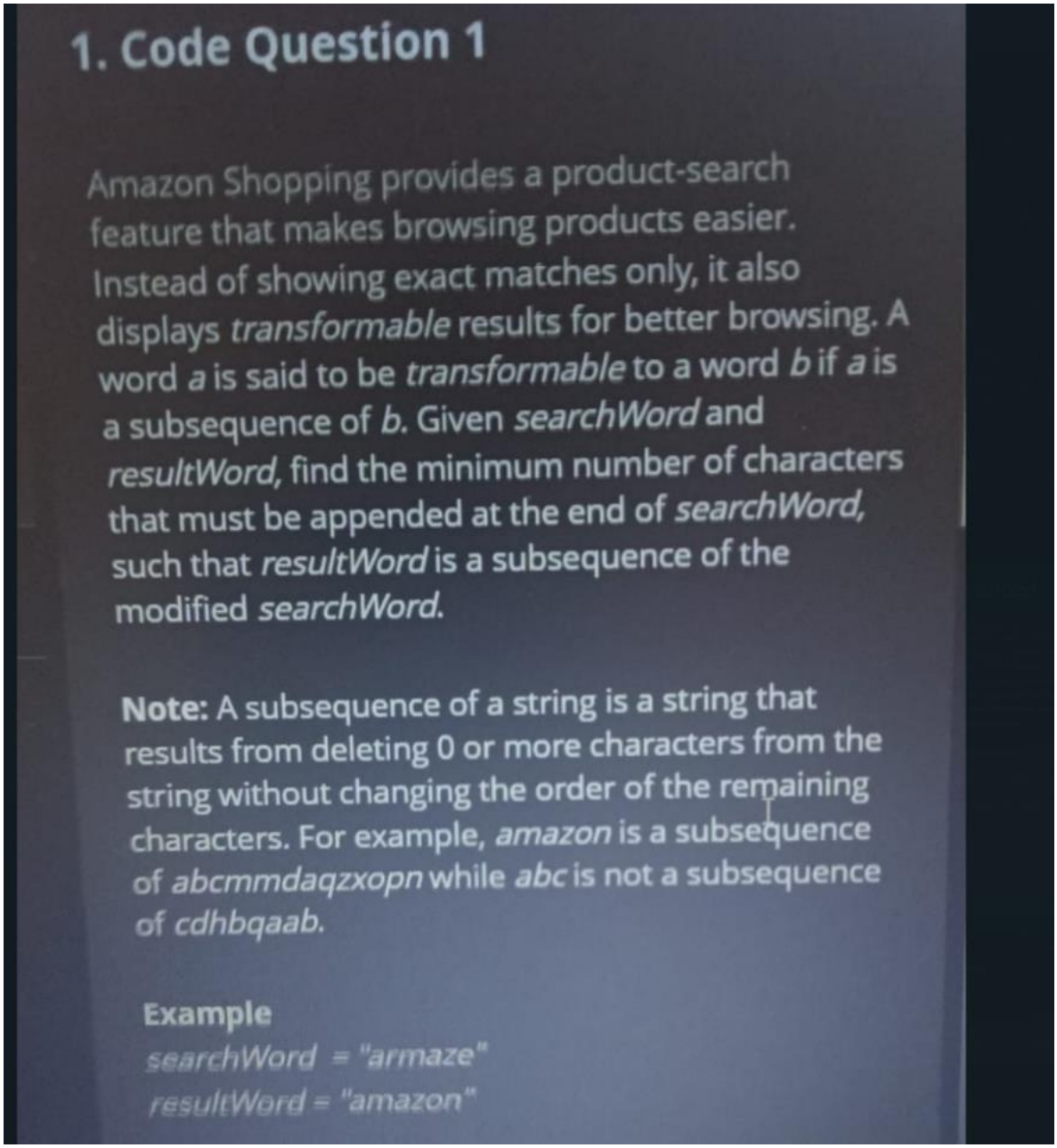
Amazon intern

1st



1. Code Question 1

Amazon Shopping provides a product-search feature that makes browsing products easier. Instead of showing exact matches only, it also displays *transformable* results for better browsing. A word *a* is said to be *transformable* to a word *b* if *a* is a subsequence of *b*. Given *searchWord* and *resultWord*, find the minimum number of characters that must be appended at the end of *searchWord*, such that *resultWord* is a subsequence of the modified *searchWord*.

Note: A subsequence of a string is a string that results from deleting 0 or more characters from the string without changing the order of the remaining characters. For example, *amazon* is a subsequence of *abcmmdaqzxopn* while *abc* is not a subsequence of *cdhbqaab*.

Example

searchWord = "armaze"

resultWord = "amazon"

hackerrank.com

Example

searchWord = "armaze"

resultWord = "amazon"

Add 2 characters, 'on', to *searchWord* to make *resultWord*, a subsequence of *searchWord*.

Function Description

Complete the function *findMinimumCharacters* in the editor below.

findMinimumCharacters has the following parameters:

string searchWord: the search word, to which characters are appended

string resultWord: the result word, which should be a subsequence of *searchWord*

Returns

int: the minimum number of characters to be appended to *searchWord*, to make *resultWord transformable* to *searchWord*.

Constraints

- $1 \leq |searchWord|, |resultWord| \leq 10^5$ where, $|x|$ represents the length of the string x.
- *searchWord* and *resultWord* consist of lowercase

▼ **Sample Case 0**

Sample Input For Custom Testing

STDIN		FUNCTION
abcz	→	searchWord = "abcz"
azdb	→	resultWord = "azdb"

Sample Output

```
2
```

Explanation

Append 'd' and 'b' -> *searchWord'* = "abczdb".

'azdb' is a subsequence of *searchWord'* = "abczdb".

▼ **Sample Case 1**

Sample Input For Custom Testing

STDIN		FUNCTION
ab	→	searchWord = "ab"
abd	→	resultWord = "abd"

Sample Output

```
1
```

Explanation

Append 'd' -> *searchWord'* = "abd".

2nd

2. Code Question 2

Several satellites provide observational black and white images which are stored in data centers at Amazon Web Services (AWS).

A black and white image is composed of pixels and is represented as an *(n · m)* grid of cells. Each pixel can have a value of `0` or `1`, where `0` represents a white pixel and `1` represents a black pixel. The *greyness* of a cell *(i, j)* is affected by the pixel values in the i^{th} row and the j^{th} column. More formally, the *greyness* of the cell *(i, j)* is the difference between the number of black pixels in the i^{th} row and the j^{th} column and the number of white pixels in the i^{th} row and the j^{th} column.

Find the maximum *greyness* among all the cells of the grid.

Note: The value of cell *(i, j)* is counted both in the i^{th} row and in the j^{th} column.

Example
pixels = ['101', '001', '110']

The *n x m = 3 x 3* grid of pixels looks like this:

The *n x m = 3 x 3* grid of pixels looks like this:

1	0	1
0	0	1
1	1	0

The *greyness* of the cell (1, 1) is calculated as:

Number of 1s in 1st row = 2
Number of 1s in 1st column = 2
Number of 0s in 1st row = 1
Number of 0s in 1st column = 1
Thus,
greyness = (2 + 2) - (1 + 1) = 2

1	0	1
0	0	1
1	1	0

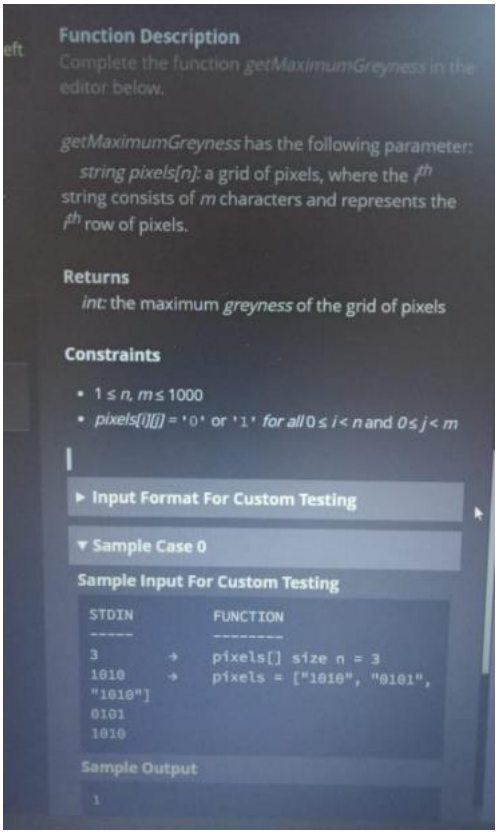
The *greyness* of each cell is:

2	0	2
0	-2	0
2	0	2

The maximum achievable *greyness* is 2.

Function Description
Complete the function *getMaximumGreyness* in the editor below.

getMaximumGreyness has the following parameter:
string pixels[n]: a grid of pixels, where the i^{th} string consists of *m* characters and represents the



Function Description

Complete the function *getMaximumGreyness* in the editor below.

getMaximumGreyness has the following parameter:

string pixels[n]: a grid of pixels, where the i^{th} string consists of *m* characters and represents the i^{th} row of pixels.

Returns

int: the maximum *greyness* of the grid of pixels

Constraints

- $1 \leq n, m \leq 1000$
- *pixels[i][j]* = `'0'` or `'1'` *for all* $0 \leq i < n$ and $0 \leq j < m$

▸ **Input Format For Custom Testing**

▾ **Sample Case 0**

Sample Input For Custom Testing

```
STDIN           FUNCTION
-----           --------
3           →   pixels[] size n = 3
1010        →   pixels = ["1010", "0101",
"1010"]
0101
1010
```

Sample Output

```
1
```