Preparation for Amazon's SDE Coding Assessment

I have included the problem statements from a past assessment that has since been archived. This should give you a better idea of what to expect when you take the online coding assessment! *See below for the Problem Statements →*

Problem #1 Problem Statement

Amazon is building a way to help customers search reviews quicker by providing real-time suggestions to search terms when the customer starts typing. When given a minimum of two characters into the search field the system will suggest at most three keywords from the review word repository. As the customer continues to type in the reviews search bar the relevant keyword suggestions will update automatically.

- Write an algorithm that will output a maximum of three keyword suggestions after each character is typed by the customer in the search field.
- If there are more than three acceptable keywords, return the keywords that are first in alphabetical order.
- Only return keyword suggestions after the customer has entered two characters.
- Keyword suggestions must start with the characters already typed.
- Both the repository and the *customerQuery* should be compared in a **case-insensitive way**.

Input

The input to the method/function consists of three arguments: ***numreviews***, an integer representing the number of various keywords from the Amazon review comment section; ***repository***, a list of unique strings representing the various keywords from the Amazon review comment section; ***customerQuery***, a string representing the full search query of the customer.

Output

Return a list of a list of stri he system as the customer types each character of the ithout deleting or removing any characters. If an output

Example

Input:

numreviews = 5
repository = ["mobile", "mouse", "moneypot", "monitor", "mousepad"]
customerQuery = "mouse"

Output:

["mobile", "moneypot", "monitor"]
["mouse", "mousepad"]
["mouse", "mousepad"]
["mouse", "mousepad"]

Explanation:

- The chain of words that will generate in the search box will be mo, mou, mous, mouse
  and each line from output shows the suggestion of "mo", "mou", "mous", "mouse", respectively in each line.
- For the keyword suggestions made by the system that are generated for 'mo', the matches that will be generated are:["mobile", "mouse", "moneypot", "monitor", "mousepad"]
- Alphabetically, they will be reordered to ["mobile", "moneypot", "monitor", "mouse", "mousepad"].
  Thus the keyword suggestions made by the system are ["mobile", "moneypot", "monitor"].

Briefly describe your approach for solving this code question.

What is the run time complexity of your solution for this code question?

Problem #2 Problem Statement

AWS Cloudfront serves files in multiple physical locations in order to minimize client latency. Cloudfront can be visualized as a 2D grid of servers. When Amazon wants to host a file on Cloudfront, the file needs to be distributed to all servers. The servers are in the form of a 2D array of 0s and 1s, where the value 0 represents a server that has yet to receive the file and 1 represents a server that already has the file.

Amazon will initially send the file to a handful of (but not all) servers based on expected utilization. A server, upon receiving a file, will then send the file to adjacent servers that don't yet have the file. An adjacent server is either on the left, right, above or below a given server. To conserve bandwidth, once a server receives a file, it will wait 1 hour before sending the file to adjacent servers.

Given the 2D array representing the existence of a new file on each server, write an algorithm that will determine the minimum number of hours required to send the file to all the available servers.

Input

The input to the function/method consists of three arguments: rows, an integer representing the number of rows in the grid; columns, an integer representing the number of columns in the grid; grid, an integer array representing the current status of its servers. The value 0 represents a server that has yet to receive the file and 1 represents a server that already has the file.

Output

Return an integer representing the minimum number of hours required to send the file to all the available servers. Return -1 if all the available servers cannot receive the file.

Note - ***Diagonally placed c***

Example

Input:
rows = 4
columns = 5
grid =

```
[[0, 1, 1, 0, 1],
 [0, 1, 0, 1, 0],
 [0, 0, 0, 0, 1],
 [0, 1, 0, 0, 0]]
```

Output:
2

Explanation:

At the end of the first hour, the status of the servers :

```
1, 1, 1, 1, 1
1, 1, 1, 1, 1
0, 1, 0, 1, 1
1, 1, 1, 0, 1
```

at the end of the second hour, the status of the servers :

```
1, 1, 1, 1, 1
1, 1, 1, 1, 1
1, 1, 1, 1, 1
1, 1, 1, 1, 1
```

By the end of two hours, all the servers are up to date.

Briefly describe your approach for solving this code question.

What is the run time complexity of your solution for this code question?