Clash Of Coders Round 3 Questions

1.Given two integer arrays A and B of size N.

There are **N** gas stations along a circular route, where the amount of gas at station **i** is **A[i]**.

You have a car with an unlimited gas tank and it costs **B[i]** of gas to travel from station **i** to its next station **(i+1)**. You begin the journey with an empty tank at one of the gas stations.

Return the minimum starting gas station’s index if you can travel around the circuit once, otherwise return -1.

You can only travel in one direction. **i to i+1, i+2, … n-1, 0, 1, 2..** Completing the circuit means starting at **i** and ending up at **i** again.

Input Format

The first argument given is the integer array A.
The second argument given is the integer array B.

Output Format

Return the minimum starting gas station's index if you can travel around the circuit once, otherwise return -1.

For Example

Input 1:
A = [1, 2]
B = [2, 1]
Output 1:
1
Explanation 1:

If you start from index 0, you can fill in A[0] = 1 amount of gas. Now your tank has 1 unit of gas. But you need B[0] = 2 gas to travel to station 1.

If you start from index 1, you can fill in A[1] = 2 amount of gas. Now your tank has 2 units of gas. You need B[1] = 1 gas to get to station 0. So, you travel to station 0 and still have 1 unit of gas left over. You fill in A[0] = 1 unit of additional gas, making your current gas = 2. It costs you B[0] = 2 to get to station 1, which you do and complete the circuit.

2.Find the **longest increasing subsequence** of a given array of integers, **A**.
In other words, find a subsequence of array in which the subsequence’s elements are in strictly increasing order, and in which the subsequence is as long as possible.
This subsequence is not necessarily contiguous, or unique.
In this case, we only care about the **length** of the longest increasing subsequence.

Input Format:

The first and the only argument is an integer array A.

Output Format:

Return an integer representing the length of the longest increasing subsequence.

Constraints:

1 <= length(A) <= 2500
1 <= A[i] <= 2000

Example :

Input 1:
A = [1, 2, 1, 5]

Output 1:
3

Explanation 1:
The sequence : [1, 2, 5]

Input 2:
A = [0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15]

Output 2:
6

Explanation 2:
The sequence : [0, 2, 6, 9, 13, 15] or [0, 4, 6, 9, 11, 15] or [0, 4, 6, 9, 13, 15]

3.Given a string **A**, partition **A** such that every substring of the partition is a palindrome.

Return the **minimum** cuts needed for a palindrome partitioning of **A**.

Input Format:

The first and the only argument contains the string A.

Output Format:

Return an integer, representing the answer as described in the problem statement.

Constraints:

1 <= length(A) <= 501

Examples:

Input 1:
  A = "aba"

Output 1:
  0

Explanation 1:
  "aba" is already a palindrome, so no cuts are needed.

Input 2:
  A = "aab"

Output 2:
  1

Explanation 2:
  Return 1 since the palindrome partitioning ["aa","b"] could be produced using 1 cut.

4.Given a string **A** and a dictionary of words **B**, determine if **A** can be segmented into a space-separated sequence of one or more dictionary words.

Input Format:

The first argument is a string, A.
The second argument is an array of strings, B.

Output Format:

Return 0 / 1 (0 for false, 1 for true) for this problem.

Constraints:

1 <= len(A) <= 6500
1 <= len(B) <= 10000
1 <= len(B[i]) <= 20

Examples:

Input 1:
A = "myinterviewtrainer",
B = ["trainer", "my", "interview"]

Output 1:
1

Explanation 1:
Return 1 (corresponding to true) because "myinterviewtrainer" can be segmented as "my interview trainer".

Input 2:
A = "a"
B = ["aaa"]

Output 2:
0

Explanation 2:
Return 0 (corresponding to false) because "a" cannot be segmented as "aaa".

5.Given two words (start and end), and a dictionary, find the shortest transformation sequence from start to end, such that:

- Only one letter can be changed at a time
- Each intermediate word must exist in the dictionary

If there are multiple such sequence of shortest length, return all of them. Refer to the example for more details.

Example :

Given:

start = "hit"
end = "cog"
dict = ["hot","dot","dog","lot","log"]

Return

```
 [
   ["hit","hot","dot","dog","cog"],
   ["hit","hot","lot","log","cog"]
 ]
```

6.Count of integers in a range which have even number of odd digits and odd number of even digits

Given a range **[L, R]**, the task is to count the numbers which have even number of odd digits and odd number of even digits. For example,

1. 8 has 1 even digit and 0 odd digit – Satisfies the condition since 1 is odd and 0 is even.
2. 545 has 1 even digit and 2 odd digits – Satisfies the condition since 1 is odd and 2 is even.
3. 4834 has 3 even digits and 1 odd digit – Does not satisfy the condition since there are odd numbers(i.e 1) of odd digits.

Examples:

Input: L = 1, R = 9
Output: 4
2, 4, 6 and 8 are the only integers from the
given range that satisfy the given conditions.
Input: L = 1, R = 19
Output: 4
Input: L = 123, R = 984
Output: 431

7.Find the Nth Mosaic number

Given an integer **N**, the task is to find the **Nth** Mosaic number. A Mosaic number can be expressed as follows:

If **N = A^a * B^b * C^c …** where **A**, **B**, **C**.. are the prime factors of **N** then the Nth Mosaic number will be **A * a * B * b * C * c ….**

Examples:

Input: N = 8
Output: 6
8 can be expressed as 2^3.
So, the 8th Mosaic number will be 2 * 3 = 6

Input: N = 36
Output: 24
36 can be expressed as 2^2 * 3^2.
2 * 2 * 3 * 2 = 24

8.Given a set of distinct integers, S, return all possible subsets.

Note:

- Elements in a subset must be in non-descending order.
- The solution set must not contain duplicate subsets.
- Also, the subsets should be sorted in ascending (lexicographic) order.
- The list is not necessarily sorted.

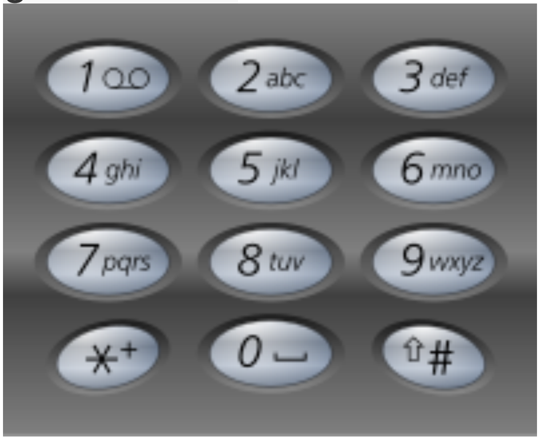
Example :

If S = [1,2,3], a solution is:

```
[
  [],
  [1],
  [1, 2],
  [1, 2, 3],
  [1, 3],
  [2],
  [2, 3],
  [3],
]
```

9.Given a digit string, return all possible letter combinations that the number could represent.

A mapping of digit to letters (just like on the telephone buttons) is given below.



The digit 0 maps to 0 itself.

The digit 1 maps to 1 itself.

Input: Digit string "23"

Output: ["ad", "ae", "af", "bd", "be", "bf", "cd", "ce", "cf"].

Make sure the returned strings are lexicographically sorted.

10.Given n pairs of parentheses, write a function to generate all combinations of well-formed parentheses of length 2*n.

For example, given n = 3, a solution set is:

"((()))", "(()())", "(())()", "()(())", "()()()"

Make sure the returned list of strings are sorted.

11.Minimum Characters required to make a String Palindromic

Given an string **A**. The only operation allowed is to insert characters in the beginning of the string.
Find how many minimum characters are needed to be inserted to make the string a palindrome string.

Input Format

The only argument given is string A.

Output Format

Return the minimum characters that are needed to be inserted to make the string a palindrome string.

For Example

Input 1:
A = "ABC"
Output 1:
2
Explanation 1:
Insert 'B' at beginning, string becomes: "BABC".
Insert 'C' at beginning, string becomes: "CBABC".

Input 2:
A = "AACECAAAA"
Output 2:
2
Explanation 2:
Insert 'A' at beginning, string becomes: "AAACECAAAA".
Insert 'A' at beginning, string becomes: "AAAACECAAAA".

12.Zigzag String

Asked in:
Paypal

The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

P.......A........H.......N
..A..P....L....S....I...I....G
....Y.........I........R

And then read line by line: PAHNAPLSIIGYIR
Write the code that will take a string and make this conversion given a number of rows:
string convert(string text, **int** nRows);
convert("PAYPALISHIRING", 3) should **return** "PAHNAPLSIIGYIR"
**Example 2 : **
ABCD, 2 can be written as
A....C
...B....D
and hence the answer would be ACBD.

13. Stringoholics

You are given an array **A** consisting of strings made up of the letters ‘a’ and ‘b’ only.
Each string goes through a number of operations, where:

1. At time 1, you circularly rotate each string by 1 letter.
2. At time 2, you circularly rotate the new rotated strings by 2 letters.
3. At time 3, you circularly rotate the new rotated strings by 3 letters.
4. At time i, you circularly rotate the new rotated strings by i % length(string) letters.

Eg: String is "abaa"

1. At time 1, string is "baaa", as 1 letter is circularly rotated to the back
2. At time 2, string is "aaba", as 2 letters of the string "baaa" is circularly rotated to the back
3. At time 3, string is "aaab", as 3 letters of the string "aaba" is circularly rotated to the back
4. At time 4, string is again "aaab", as 4 letters of the string "aaab" is circularly rotated to the back
5. At time 5, string is "aaba", as 1 letters of the string "aaab" is circularly rotated to the back

After some units of time, a string becomes equal to it’s original self. Once a string becomes equal to itself, it’s letters start to rotate from the first letter again (**process resets**). So, if a string takes **t** time to get back to the original, at time t+1 one letter will be rotated and the string will be it’s original self at 2**t** time.
You have to find the minimum time, where maximum number of strings are equal to their original self.
As this time can be very large, give the answer modulo 109+7.
Note: Your solution will run on multiple test cases so do clear global variables after using them.

Input:

A: Array of strings.

Output:

Minimum time, where maximum number of strings are equal to their original self.

Constraints:

1 <= size(A) <= 10^5
1 <= size of each string in A <= 10^5
Each string consists of only characters 'a' and 'b'
Summation of length of all strings <= 10^7

Example:

Input

A: [a,ababa,aba]

Output

4

String 'a' is it's original self at time 1, 2, 3 and 4.
String 'ababa' is it's original self only at time 4. (ababa => babaa => baaba => babaa => ababa)
String 'aba' is it's original self at time 2 and 4. (aba => baa => aba)

Hence, 3 strings are their original self at time 4.