Question - 1
Maximum among neighbours

Given a list of non negative numbers, find the peak in that list and return the index of the peak. A peak element is an element that is greater than its neighbours. You can assume that no 2 consecutive elements in the array are equal.
If there are multiple peaks in the array, return the index of first peak element.

Example 1:
Input:
[12,23,35,17]
Output: 2

Example 2:
[123,2343,1323,35656,5232,6342,4232]
Output: 1

Question - 2
Compact Array

Given a sorted integer array with no duplicates, compact the array based on continuous range of numbers. If there are no such ranges available, output the list of strings where each element is a string notation of the number.
Input : integer array
Output: List of strings

Example 1
Input : [1,2,3,6,7,8,10,15]
Output :
1 to 3
6 to 8
10
15
Explanation : 1,2,3 form a continuous range and hence is compacted to "1 to 3".Same goes for 6,7,8

Example 2
Input: [10,20,30,40]
Output:
10
20
30
40
Explanation: None of the elements in the array form a continuous range

Question - 3

Roll the String

We define a single *roll* operation on a string to be the circular increment of each character by one. In other words, each character is *rolled* forward and overwritten with the next alphabetic character. Looking at the English alphabet, characters in the range ascii[a-z], *a* becomes *b*, *b* becomes *c*, and *z* becomes *a*.

Given an array of integers named *roll*, we want to perform a roll operation on the first *roll[i]* characters of *s* for each element *i* in the array. Given a zero indexed string, an operation *roll[i]* affects characters at positions *0* through *(roll[i]-1)*.

For example, if string *s = abz* and *roll = [3, 2, 1]*, we perform the following sequence of operations:

- *roll[0] = 3*: Roll all three characters so *abz* becomes *bca*.
- *roll[1] = 2*: Roll the first two characters so *bca* becomes *cda*.
- *roll[2] = 1*: Roll the first character so *cda* becomes *dda*.

After performing all the operations, the final value of *s* is *dda*.

Function Description

Complete the function *rollTheString* in the editor below. The function must return the resulting string after all roll operations have been performed.

rollTheString has the following parameter(s):

s: the string to operate on

roll[roll[0],...roll[n-1]]: an array of integers indicating the number of items in *s* to roll

Constraints

- Each character in *s* is a character in the range ascii[a-z].
- $1 \leq |s| \leq 10^5$
- $1 \leq n \leq 10^5$
- $1 \leq roll[i] \leq |s|$, where $0 \leq i < n$.

▼ Input Format Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains a string *s*.
The next line contains an integer *n*, the size of the array *roll*.
The next *n* lines each contain an element *roll[i]* where 0 ≤ i < n.

▼ Sample Case 0

Sample Input 0

```
abz
1
3
```

Sample Output 0

```
bca
```

Explanation 0

We want to perform the operation on *s = abz* described in *roll = [3]*. For our first (and only) operation, we roll forward all characters in the substring *s[0] … s[2]* (which ends up being the entire string) so *abz* becomes *bca*.

▼ **Sample Case 1**

Sample Input 1

```
vwxyz
5
1
2
3
4
5
```

Sample Output 1

```
aaaaa
```

Explanation 1

We want to perform the *n = 5* operations on *s = vwxyz* described in *roll = [1, 2, 3, 4, 5]*:

- *roll[0] = 1*: Roll forward all characters in the substring *s[0] … s[1−1]*, so ***v***wxyz becomes ***w***wxyz.
- *roll[1] = 2*: Roll forward all characters in the substring *s[0] … s[2−1]*, so ***ww***xyz becomes ***xx***xyz.
- *roll[2] = 3*: Roll forward all characters in the substring *s[0] … s[3−1]*, so ***xxx***yz becomes ***yyy***yz.
- *roll[3] = 4*: Roll forward all characters in the substring *s[0] … s[4−1]*, so ***yyyy***z becomes ***zzzz***z.
- *roll[4] = 5*: Roll forward all characters in the substring *s[0] … s[5−1]*, so ***zzzzz*** becomes ***aaaaa***.

Question - 4

Exencode String

Given a string S, you need to generate an encoded version of this string.
The encoding rules are as follows:
- If you encounter an alphabet in the input string, add it to the output.
- If you encounter a digit 'd', repeat the current output 'd-1' times and append it to the current output.
- If you encounter any character other than alpha numeric, you can ignore it and proceed to the next character.

The digits in the input string can be negative, in which case you will append the current output string once to the output.
If you encounter 0 or 1, do not append anything to the current output.

Example 1:
Input: cisco

Output: cisco

Example 2:
Input: cisco2India
Output: ciscociscoIndia

Example 3:
Input: cisco1India-1Bangalore
Output: ciscoIndiaciscoIndiaBangalore