Question - 1

Distinct Characters Count (Java Stream)

In this challenge, use the Java *Stream*, *Predicate*, and *Function* to find the number of distinct characters in the given names which start with a given prefix. The complete implementation of *CharactersCount* class is given. It has the following two fields:

- *name* describes the name string.
- *distinctCharacterCount* describes the number of distinct characters.

Create the following two classes:

- *Filter* class with method *Predicate<String> nameStartingWithPrefix(String prefix)* that returns a predicate to check whether a name string starts with the given prefix or not.
- *Mapper* class with method *Function<String, CharactersCount> getDistinctCharactersCount()* that returns a mapper function to return a *CharactersCount* class object that correspond to the given name string.

The locked stub code in the editor validates the correctness of the *Filter* and *Mapper* classes implementation.

Constraints

- There exists at least one name for the given prefix.

▼ Input Format For Custom Testing

The only line of input contains the *prefix* string.

▼ Sample Case 0

Sample Input For Custom Testing

```
aa
```

Sample Output

```
"aaryanna" has 4 distinct characters.
"aayanna" has 3 distinct characters.
```

Explanation

For the given names ["aaryanna", "aayanna", "airianna", "alassandra", "allanna", "allannah", "allessandra", "allianna", "allyanna", "anastaisa", "anastashia", "anastasia", "annabella", "annabelle", "annebelle"], *"aaryanna"* and *"aayanna"* starts with the prefix *"aa"*.

- *"aaryanna"* has four distinct characters: *'a'*, *'n'*, *'r'*, and *'y'*.
- *"aayanna"* has four distinct characters: *'a'*, *'n'*, and *'y'*.

Question - 2

Roll the String

~~A~~ single *roll* operation on a string is a circular increment of each character by one. Looking at the English alphabet, characters in the range ascii[a-z], *a* becomes *b*, *b* becomes *c*, and *z* becomes *a*.

Help

Given an array of integers named *roll*, 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)*.

Example

s = 'abz'

roll = [3, 2, 1]

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 the operations, the final value of *s* is *'dda'*.

Function Description

Complete the function *rollTheString* in the editor below.

rollTheString has the following parameter(s):

string s: the string to operate on

int roll[n]: an array of integers indicating the number of items in *s* to roll

Returns:

string : the resulting string after all roll operations have been performed

Constraints

- Each character in *s* is a character in the range ascii[a-z].
- *1 ≤ length of s ≤ 10^5*
- *1 ≤ n ≤ 10^5*
- *1 ≤ roll[i] ≤ length of s*, where *0 ≤ 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

```
STDIN   Function
-----   -------------------
abz  →  s = 'abz'
1    →  roll[] size n = 1
3    →  roll = [3]
```

Sample Output 0

```
bca
```

Explanation 0

Roll forward the first 3 characters in the substring *s[0] … s[2]*, so *abz* becomes *bca*.

▼ Sample Case 1

Sample Input 1

```
STDIN    Function
-----    -------------------
vwxyz →  s = 'vwxyz'
5     →  roll[] size n = 5
1     →  roll = [1, 2, 3, 4, 5]
2
3
4
5
```

Sample Output 1

```
aaaaa
```

Explanation 1

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***.