Contents

Strings

1. Definition

- Array of characters or Sequence/Stream of characters.

2. Ascii Values

a. Capital Letters

A – 65, B – 67,................................Z – 90

b. Small Letters

a – 97, b – 98,................................z – 122

c. Digit Characters

0 – 48, 1 – 49,................................9 – 57

3. Programs

a. tolower()

Convert each character of A into lowercase characters if it exists. If the lowercase of a character does not exist, it remains unmodified.

The uppercase letters from A to Z are converted to lowercase letters from a to z respectively.

Return the **lowercase** version of the given character array.

Example Input

Input 1:

A = ['S', 'c', 'A', 'l', 'e', 'r', 'A', 'c', 'a', 'D', 'e', 'm', 'y']

Input 2:

A = ['S', 'c', 'a', 'L', 'e', 'r', '#', '2', '0', '2', '0']

Example Output

Output 1:

['s', 'c', 'a', 'l', 'e', 'r', 'a', 'c', 'a', 'd', 'e', 'm', 'y']

Output 2:

['s', 'c', 'a', 'l', 'e', 'r', '#', '2', '0', '2', '0']

b. **toupper()**

Convert each character of A into Uppercase character if it exists. If the Uppercase of a character does not exist, it remains unmodified.

The lowercase letters from a to z is converted to uppercase letters from A to Z respectively.

Return the uppercase version of the given character array.

Example Input

Input 1:

A = [‘S’, ‘c’, ‘A’, ‘L’, ‘E’, ‘r’, ‘A’, ‘c’, ‘a’, ‘D’, ‘e’, ‘m’, ‘y’]

Input 2:

A = [‘S’, ‘c’, ‘a’, ‘L’, ‘e’, ‘R’, ‘#’, ‘2’, ‘0’, ‘2’, ‘0’]

Example Output

Output 1:

[‘S’, ‘C’, ‘A’, ‘L’, ‘E’, ‘R’, ‘A’, ‘C’, ‘A’, ‘D’, ‘E’, ‘M’, ‘Y’]

Output 2:

[‘S’, ‘C’, ‘A’, ‘L’, ‘E’, ‘R’, ‘#’, ‘2’, ‘0’, ‘2’, ‘0’]

c. **Toggle Case**

You are given a character string A having length N, consisting of only lowercase and uppercase latin letters.

You have to toggle case of each character of string A. For e.g 'A' is changed to 'a', 'e' is changed to 'E', etc.

d. **Sort**

Given an array A. Sort this array

e. **Simple Reverse**

Given a string A, you are asked to reverse the string and return the reversed string.
A = "academy"
"ymedaca"

f. Reverse the String

You are given a string A of size N.
Return the string A after reversing the string word by word.

NOTE:

- A sequence of non-space characters constitutes a word.
- Your reversed string should not contain leading or trailing spaces, even if it is present in the input string.
- If there are multiple spaces between words, reduce them to a single space in the reversed string.

Example Input

Input 1:

A = "the sky is blue"

Input 2:

A = "this is ib"

Example Output

Output 1:

"blue is sky the"

Output 2:

"ib is this"

g. Longest Palindromic Substring

Given a string A of size N, find and return the **longest palindromic substring** in A.
Substring of string A is `A[i...j]` where 0 <= i <= j < len(A)

Palindrome string:
A string which reads the same backwards. More formally, A is palindrome if reverse(A) = A.

Incase of conflict, return the substring which occurs first (with the least starting index).

Example Input

Input 1:

A = "aaaabaaa"

Input 2:

A = "abba

Example Output

Output 1:

"aaabaaa"

Output 2:

"abba"

h. Longest Common Prefix

Given the array of strings **A**, you need to find the longest string **S,** which is the prefix of **ALL** the strings in the array.

The longest common prefix for a pair of strings **S1** and **S2** is the longest string **S** which is the prefix of both **S1** and **S2**.

Example: the longest common prefix of `"abcdefgh"` and `"abcefgh"` is `"abc"`.

Example Input

Input 1:

A = ["abcdefgh", "aefghijk", "abcefgh"]

Input 2:

A = ["abab", "ab", "abcd"];

Example Output

Output 1:

"a"

Output 2:

"ab"

i. Isalnum()

You are given a function `isalpha()` consisting of a character array **A**.

Return **1** if all the characters of a character array are alphanumeric **(a-z, A-Z, and 0-9)** else, return **0**.

j. String Operations

Akash likes playing with strings. One day he thought of applying following operations on the string in the given order:

- Concatenate the string with itself.
- Delete all the uppercase letters.
- Replace each vowel with '#'.

You are given a string **A** of size N consisting of lowercase and uppercase alphabets. Return the resultant string after applying the above operations.

NOTE: 'a' , 'e' , 'i' , 'o' , 'u' are defined as vowels.

k. **Change Characters**

You are given a string **A** of size **N** consisting of lowercase alphabets.

You can change at most **B** characters in the given string to any other lowercase alphabet such that the number of distinct characters in the string is minimized.

Find the **minimum** number of **distinct** characters in the resulting string.

Example Input

A = "abcabbccd"
B = 3

Example Output

2

Example Explanation

We can change both 'a' and one 'd' into 'b'.So the new string becomes "bbcbbbccb".
So the minimum number of distinct character will be 2.

l. **Count Occurrences**

Find the number of occurrences of bob in string A consisting of lowercase English alphabets.

m. **Check Anagrams**

You are given two lowercase strings **A** and **B** each of length **N**. Return 1 if they are anagrams to each other and 0 if not.

Note : Two strings A and B are called anagrams to each other if A can be formed after rearranging the letters of B.

Example Input

Input 1:
A = "cat"
B = "bat"
Input 2:
A = "secure"
B = "rescue"

Example Output

Output 1: 0
Output 2: 1