**Strings**

1. Perform the following operations on strings object
2. Initialize the string object str1 with the following sentence “string class is part of C++ library”
3. Initialize another string object str2 with from 1st index 12 characters, using constructors.
4. Print 1 and 2
5. Print the first element , last element and middle element of str1
6. Append “object” in str2
7. Print the index of “Hello ” in the object str3=”Hello Hello Hello” using find member function
8. Remove last Hello from str3
9. Replace Last Hello with your name which is read from the keyboard
10. Print str3
11. Given string containing just characters check if ‘(‘ is there , there should be ‘)’ also in the string, only then the input is valid.

Test case

Example 1 :

**Input:** "(cn,xn,xncxmcxmc.xmc.x)"

**Output:** true

**Input:** "()"

**Output:** true

**Input:** "(fjdljfldj fldfdl"

**Output:** False

1. Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Note that an empty string is also considered valid.

**Example 1:**

**Input:** "()"

**Output:** true

**Example 2:**

**Input:** "()[]{}"

**Output:** true

**Example 3:**

**Input:** "(]"

**Output:** false

**Example 4:**

**Input:** "([)]"

**Output:** false

**Example 5:**

**Input:** "{[]}"

**Output:** true

1. Given a string, find the first non-repeating character in it and return its index. If it doesn't exist, return -1.

**Examples:**

s = "leetcode"

return 0.

s = "loveleetcode"

return 2.

1. Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

**Example 1:**

**Input:** ["flower","flow","flight"]

**Output:** "fl"

**Example 2:**

**Input:** ["dog","racecar","car"]

**Output:** ""

**Explanation:** There is no common prefix among the input strings.

**Note:**

All given inputs are in lowercase letters a-z.

1. A Program to check if strings are rotations of each other or not

Given a string s1 and a string s2, write a snippet to say whether s2 is a rotation of s1?  
(eg given s1 = ABCD and s2 = CDAB, return true, given s1 = ABCD, and s2 = ACBD , return false)

**Algorithm:** areRotations(str1, str2)

1.Create a temp string and store concatenation of str1 to

str1 in temp.

temp = str1.str1

2. If str2 is a substring of temp then str1 and str2 are

rotations of each other.

Example:

str1 = "ABACD"

str2 = "CDABA"

temp = str1.str1 = "ABACDABACD"

Since str2 is a substring of temp, str1 and str2 are

rotations of each other.