**SHEET- 20**

**Strings**

1. **Longest Substring Without Repeating Characters**

Given a string s, find the length of the longest substring without repeating characters.

Example 1:

Input: s = "abcabcbb"

Output: 3

Explanation: The answer is "abc", with the length of 3.

Example 2:

Input: s = "bbbbb"

Output: 1

Explanation: The answer is "b", with the length of 1.

Example 3:

Input: s = "pwwkew"

Output: 3

Explanation: The answer is "wke", with the length of 3.

Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.

1. **Generate Parentheses**

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

Example 1:

Input: n = 3

Output: ["((()))","(()())","(())()","()(())","()()()"]

Example 2:

Input: n = 1

Output: ["()"]

1. **Decode String**

Given an encoded string, return its decoded string.

The encoding rule is: k[encoded\_string], where the encoded\_string inside the square brackets is being repeated exactly k times. Note that k is guaranteed to be a positive integer.

You may assume that the input string is always valid; there are no extra white spaces, square brackets are well-formed, etc.

Furthermore, you may assume that the original data does not contain any digits and that digits are only for those repeat numbers, k. For example, there will not be input like 3a or 2[4].

Example 1:

Input: s = "3[a]2[bc]"

Output: "aaabcbc"

Example 2:

Input: s = "3[a2[c]]"

Output: "accaccacc"

Example 3:

Input: s = "2[abc]3[cd]ef"

Output: "abcabccdcdcdef"

Constraints:

* 1 <= s.length <= 30
* s consists of lowercase English letters, digits, and square brackets '[]'.
* s is guaranteed to be a valid input.
* All the integers in s are in the range [1, 300].

1. **Largest Number**

Given a list of non-negative integers nums, arrange them such that they form the largest number and return it.

Since the result may be very large, so you need to return a string instead of an integer.

Example 1:

Input: nums = [10,2]

Output: "210"

Example 2:

Input: nums = [3,30,34,5,9]

Output: "9534330"

Constraints:

* 1 <= nums.length <= 100
* 0 <= nums[i] <= 109

1. **Longest Palindromic Substring**

Given a string s, return the longest palindromic substring in s.

Example 1:

Input: s = "babad"

Output: "bab"

Explanation: "aba" is also a valid answer.

Example 2:

Input: s = "cbbd"

Output: "bb"

Constraints:

* 1 <= s.length <= 1000
* s consist of only digits and English letters.