1. Reversing String

Given an English sentence, reverse the order of characters in each word within the sentence while still preserving whitespace and initial word order. For example, given the input "the Noah’s Ark Lab is located in Hong Kong and Shenzhen", the output is "eht s’haoN krA baL si detacol ni gnoH gnoK dna nehznehS".

```
    String reverseWords(String s){}
```

2. Root-to-leaf of binary tree

Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum. Note: A leaf is a node with no children.

Example:

Given the below binary tree and sum = 22,

```
      5
     / \
    4   8
   /   / \
  11  13  4
 /  \      \
7    2       1
```

return true, as there exist a root-to-leaf path 5->4->11->2 which sum is 22.

Functions:

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *      int val;
 *      TreeNode *left;
 *      TreeNode *right;
 *      TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    bool hasPathSum(TreeNode* root, int sum) {

    }
};
```

3. Find the Longest Substring Without Repeating Characters

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

Example 1:

Input: "abcabcbb"

Output: 3

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

Example 2:

Input: "bbbbb"

Output: 1

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

Example 3:

Input: "pwwkew"

Output: 3

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

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