Reverse String - Beginner (< 1 year)

Given an input (string, number or array) write a function that reverses it and returns the reversed input based on its original type. The .reverse() method cannot be used.

Example 1:
Input: “Reversing a string”
Output: “gnirts a gnisreveR”

Example 2:
Input: 123456
Output: 654321

Example 3:
Input: [“a”, “r”, “r”, “a”, “y”]
Output: [“y”, “a”, “r”, “r”, “a”]

Palindromes - Beginner (< 1 year)

A palindrome is a word or phrase that can be read the same forwards or backward. Write a function that takes strings, integers, and arrays and returns true if the input is a palindrome, and false if it is not.

Example 1:
Input: “racecar”
Output: true

Example 2:
Input: “coding”
Output: false

Example 3:
Input: “Tacocat”
Output: false (“T” is not equal to “t”)

Example 4 (Extra Credit):
Input: [1, 2, 3, 4, “4”, 3, 2, 1]
Output: false (4 is not equal to “4”)

Outermost Parentheses - Easy (1 - 2 years)

A valid parentheses string is either empty (""), "(" + A + ")", or A + B, where A and B are valid parentheses strings, and + represents string concatenation. For example, "", "()", "(())()", and "(()(()))" are all valid parentheses strings.

A valid parentheses string S is primitive if it is nonempty, and there does not exist a way to split it into S = A+B, with A and B nonempty valid parentheses strings.

Given a valid parentheses string S, consider its primitive decomposition: S = P_1 + P_2 + ... + P_k, where P_i are primitive valid parentheses strings.

Return S after removing the outermost parentheses of every primitive string in the primitive decomposition of S.

Example 1:

Input: "(()())(())"
Output: "()()()"
Explanation:
The input string is "(()())(())", with primitive decomposition "(()())" + "(())".
After removing outer parentheses of each part, this is "()()" + "()" = "()()()".

Example 2:

Input: "(()())(())(()(()))"
Output: "()()()()(())"
Explanation:
The input string is "(()())(())(()(()))", with primitive decomposition "(()())" + "(())" + "(()(()))".
After removing outer parentheses of each part, this is "()()" + "()" + "()(())" = "()()()()(())".

Example 3:

Input: "()()"
Output: ""
Explanation:
The input string is "()()", with primitive decomposition "()" + "()".
After removing outer parentheses of each part, this is "" + "" = "".

Robot Return to Origin - Easy (1 - 2 years)

There is a robot starting at position (0, 0), the origin, on a 2D plane. Given a sequence of its moves, judge if this robot ends up at (0, 0) after it completes its moves.

The move sequence is represented by a string, and the character moves[i] represents its ith move. Valid moves are R (right), L (left), U (up), and D (down). If the robot returns to the origin after it finishes all of its moves, return true. Otherwise, return false.

Note: The way that the robot is "facing" is irrelevant. "R" will always make the robot move to the right once, "L" will always make it move left, etc. Also, assume that the magnitude of the robot's movement is the same for each move.

Example 1:

Input: "UD"

Output: true

Explanation: The robot moves up once, and then down once. All moves have the same magnitude, so it ended up at the origin where it started. Therefore, we return true.

Example 2:

Input: "LL"

Output: false

Explanation: The robot moves left twice. It ends up two "moves" to the left of the origin. We return false because it is not at the origin at the end of its moves.

Medium/Hard (3+ Years)

Construct Binary Tree from Traversals

Given preorder and inorder traversal of a tree, construct the binary tree.

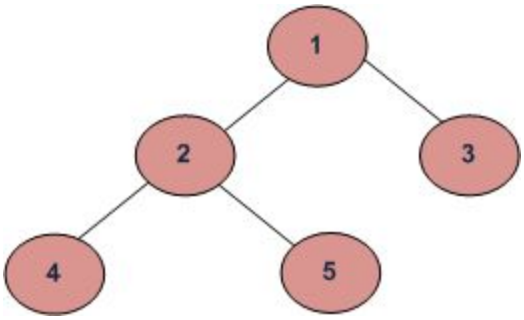
Note: You may assume that duplicates do not exist in the tree.

For example, given

```
preorder = [3,9,20,15,7], inorder = [9,3,15,20,7]
```

Return the following binary tree:

```
    3
   / \
  9  20
    /  \
   15   7
```



Notes:

Depth First Traversals:

(a) Inorder (Left, Root, Right) : 4 2 5 1 3
(b) Preorder (Root, Left, Right) : 1 2 4 5 3
(c) Postorder (Left, Right, Root) : 4 5 2 3 1

Find Islands

Given a 2d grid map of '1's (land) and '0's (water), count the number of islands. An island is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

Example 1:

```
Input:

11110

11010

11000

00000

Output: 1
```

Example 2:

```
Input:

11000

11000

00100

00011


Output: 3
```