STAGE ONE

1. Given an array of numbers, return **indices** of two numbers such that they add up to a specific target.

You may assume that each input would have ***exactly*** one solution, and you may not use the *same* element twice.

Example:

Given nums = [2, 7, 11, 15], target = 9,

Because nums[**0**] + nums[**1**] = 2 + 7 = 9,
return [**0**, **1**].

2. Given a 32-bit signed integer, reverse digits of an integer.

Example 1:

Input: 123
Output: 321

Example 2:

Input: -123
Output: -321

Example 3:

Input: 120
Output: 21

3. Determine whether an integer is a palindrome. An integer is a palindrome when it reads the same backward as forward.

Example 1:

Input: 121

Output: true

Example 2:

Input: -121

Output: false

Explanation: From left to right, it reads -121. From right to left, it becomes 121-. Therefore it is not a palindrome.

Example 3:

Input: 10

Output: false

Explanation: Reads 01 from right to left. Therefore it is not a palindrome.

4. 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.

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

5. Merge two sorted linked lists and return it as a new list. The new list should be made by splicing together the nodes of the first two lists.

Example:

Input: 1->2->4, 1->3->4

Output: 1->1->2->3->4->4

6. The count-and-say sequence is the sequence of integers with the first five terms as following:

```
1.     1
2.     11
3.     21
4.     1211
5.     111221
```

`1` is read off as `"one 1"` or `11`.
`11` is read off as `"two 1s"` or `21`.
`21` is read off as `"one 2`, then `one 1"` or `1211`.

Given an integer *n*, generate the *n*th term of the count-and-say sequence.

Note: Each term of the sequence of integers will be represented as a string.

Example 1:

Input: 1

Output: "1"

Example 2:

```
Input: 4
Output: "1211"
```

7. Given two binary trees, write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical and the nodes have the same value.

Example 1:

```
Input:     1         1
          / \       / \
         2   3     2   3

        [1,2,3],   [1,2,3]

Output: true
```

Example 2:

```
Input:     1         1
          /           \
         2             2

        [1,2],     [1,null,2]

Output: false
```

Example 3:

```
Input:     1         1
          / \       / \
         2   1     1   2

        [1,2,1],   [1,1,2]

Output: false
```

8. Given two sorted integer arrays *nums1* and *nums2*, merge *nums2* into *nums1* as one sorted array.

Note:

- The number of elements initialized in *nums1* and *nums2* are *m* and *n* respectively.
- You may assume that *nums1* has enough space (size that is greater or equal to *m* + *n*) to hold additional elements from *nums2*.

Example:

Input:

nums1 = [1,2,3,0,0,0], m = 3

nums2 = [2,5,6],   n = 3

Output: [1,2,2,3,5,6]

9. Given two sorted integer arrays *nums1* and *nums2*, merge *nums2* into *nums1* as one sorted array.

Note:

- The number of elements initialized in *nums1* and *nums2* are *m* and *n* respectively.
- You may assume that *nums1* has enough space (size that is greater or equal to *m* + *n*) to hold additional elements from *nums2*.

Example:

Input:

nums1 = [1,2,3,0,0,0], m = 3

nums2 = [2,5,6],   n = 3

Output: [1,2,2,3,5,6]

10. Given a non-negative integer `num`, repeatedly add all its digits until the result has only one digit.

Example:

Input: 38

Output: 2

Explanation: The process is like: 3 + 8 = 11, 1 + 1 = 2.

Since 2 has only one digit, return it.

Follow up:
Could you do it without any loop/recursion in O(1) runtime?