Assignment 07 Solutions

Question 1

Given two strings s and t, *determine if they are isomorphic*.

Two strings s and t are isomorphic if the characters in s can be replaced to get t.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character, but a character may map to itself.

Example 1:

Input: s = "egg", t = "add"

Output: true

In [25]:
```
def is_isomorphic(s, t):
    if len(s) != len(t):
        return False

    s_map = {}
    t_map = {}

    for s_char, t_char in zip(s, t):
        if s_char in s_map:
            if s_map[s_char] != t_char:
                return False
        else:
            s_map[s_char] = t_char

        if t_char in t_map:
            if t_map[t_char] != s_char:
                return False
        else:
            t_map[t_char] = s_char

    return True
```

In [26]:
```
s = "egg"
t = "add"
print(is_isomorphic(s, t))
```

```
True
```

Question 2

Given a string num which represents an integer, return true *if* num *is a strobogrammatic number*.

A **strobogrammatic number** is a number that looks the same when rotated 180 degrees (looked at upside down).

Example 1:

Input: num = "69"

Output:

true

In [37]:
```
def is_strobogrammatic(num):
    strobogrammatic_map = {
        '0': '0',
        '1': '1',
        '6': '9',
        '8': '8',
        '9': '6'
    }

    left = 0
    right = len(num) - 1

    while left <= right:
        digit_left = num[left]
        digit_right = num[right]

        if digit_left not in strobogrammatic_map or strobogrammatic_map[digit_left] != digit_right:
            return False
```

```
        left += 1
        right -= 1

    return True
```

In [38]:
```
num = "69"
print(is_strobogrammatic(num))
```

```
True
```

Question 3

Given two non-negative integers, num1 and num2 represented as string, return *the sum of* num1 *and* num2 *as a string*.

You must solve the problem without using any built-in library for handling large integers (such as BigInteger). You must also not convert the inputs to integers directly.

Example 1:

Input: num1 = "11", num2 = "123"

Output:

"134"

In [61]:
```
def addStrings(num1, num2):
    i = len(num1) - 1
    j = len(num2) - 1
    carry = 0
    result = ""

    while i >= 0 or j >= 0:
        digit1 = int(num1[i]) if i >= 0 else 0
        digit2 = int(num2[j]) if j >= 0 else 0
        current_sum = digit1 + digit2 + carry
        carry = current_sum // 10
        result += str(current_sum % 10)
        i -= 1
        j -= 1

    if carry > 0:
        result += str(carry)

    return result[::-1]
```

In [62]:
```
num1 = "11"
num2 = "123"
print(addStrings(num1, num2))
```

```
134
```

Question 4

Given a string s, reverse the order of characters in each word within a sentence while still preserving whitespace and initial word order.

Example 1:

Input: s = "Let's take LeetCode contest"

Output: "s'teL ekat edoCteeL tsetnoc"

In [75]:
```
def reverseWords(s):
    words = s.split()
    reversed_words = [word[::-1] for word in words]
    reversed_sentence = ' '.join(reversed_words)
    return reversed_sentence
```

In [76]:
```
s = "Let's take LeetCode contest"
print(reverseWords(s))
```

```
s'teL ekat edoCteeL tsetnoc
```

Question 5

Given a string s and an integer k, reverse the first k characters for every 2k characters counting from the start of the string.

If there are fewer than k characters left, reverse all of them. If there are less than 2k but greater than or equal to k characters, then reverse the first k characters and leave the other as original.

Example 1:

Input: s = "abcdefg", k = 2

Output:

"bacdfeg"

In [85]:
```
def reverseStr(s, k):
    s = list(s)

    for i in range(0, len(s), 2 * k):
        s[i:i+k] = reversed(s[i:i+k])

    return ''.join(s)
```

In [86]:
```
s = "abcdefg"
k = 2
print(reverseStr(s, k))
```

```
bacdfeg
```

Question 6

Given two strings s and goal, return true *if and only if* s *can become* goal *after some number of shifts** on* s.

A **shift** on s consists of moving the leftmost character of s to the rightmost position.

- For example, if s = "abcde", then it will be "bcdea" after one shift.

Example 1:

Input: s = "abcde", goal = "cdeab"

Output:

true

In [93]:
```
def rotateString(s, goal):
    if len(s) != len(goal):
        return False

    s_concat = s + s

    if goal in s_concat:
        return True
    else:
        return False
```

In [94]:
```
s = "abcde"
goal = "cdeab"
print(rotateString(s, goal))
```

```
True
```

Question 7

Given two strings s and t, return true *if they are equal when both are typed into empty text editors*. '#' means a backspace character.

Note that after backspacing an empty text, the text will continue empty.

Example 1:

Input: s = "ab#c", t = "ad#c"

Output: true

Explanation:

Both s and t become "ac".

In [107…
```
def backspaceCompare(s, t):
    def process_string(s):
        processed = []
        for c in s:
            if c != '#':
                processed.append(c)
            elif processed:
                processed.pop()
```

```
        return ''.join(processed)

    processed_s = process_string(s)
    processed_t = process_string(t)

    return processed_s == processed_t
```

In [108…
```
s = "ab#c"
t = "ad#c"
print(backspaceCompare(s, t))
```

```
True
```

Question 8

You are given an array coordinates, coordinates[i] = [x, y], where [x, y] represents the coordinate of a point. Check if these points make a straight line in the XY plane.

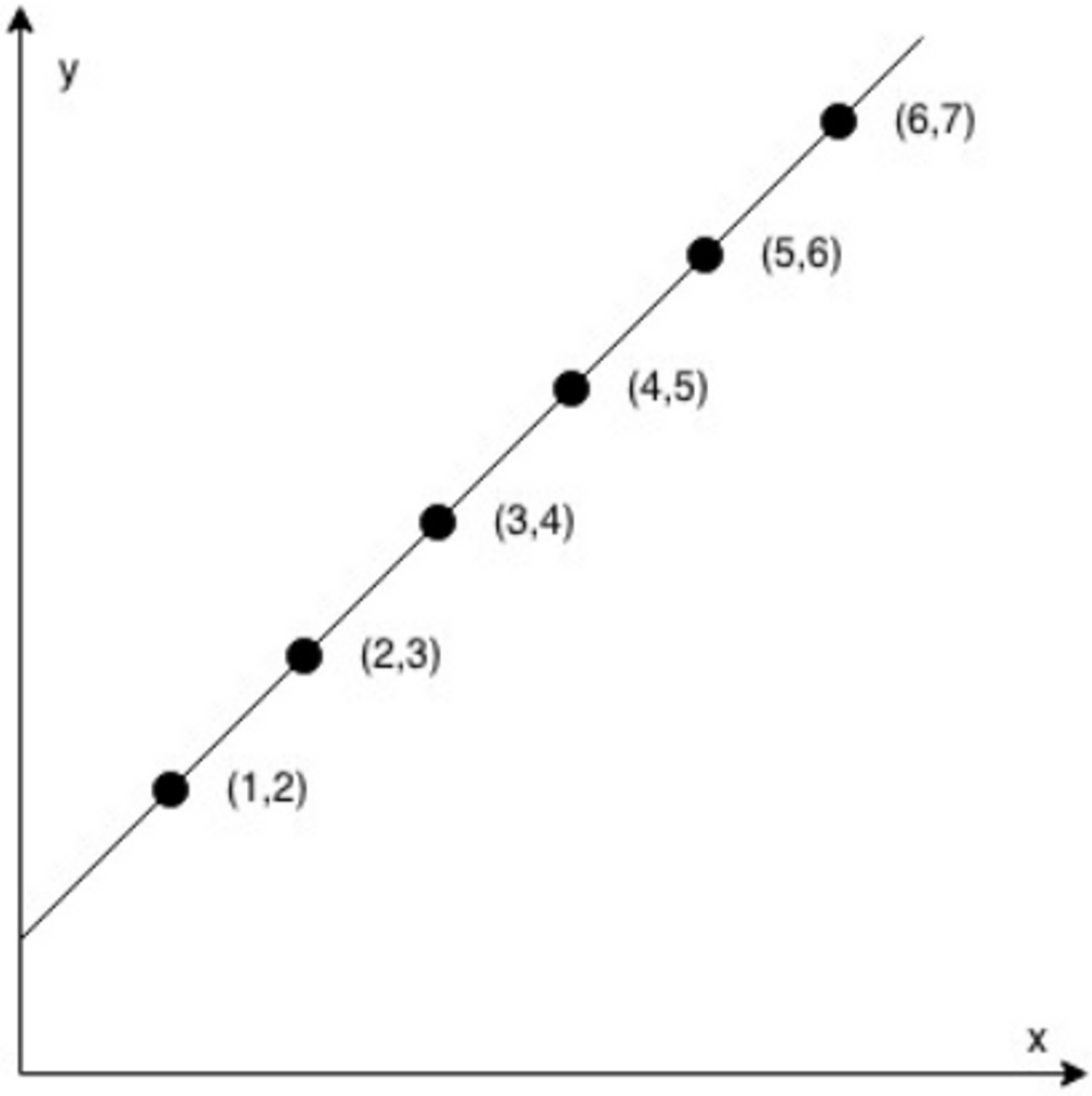
Input: coordinates = [[1,2],[2,3],[3,4],[4,5],[5,6],[6,7]]

Output: true

Example 1:

In [146…

Out[146]:



In [125…
```
def checkStraightLine(coordinates):
    x1, y1 = coordinates[0]
    x2, y2 = coordinates[1]
    if x2 - x1 == 0:
```

```
        slope = float('inf')
    else:
        slope = (y2 - y1) / (x2 - x1)

    for i in range(2, len(coordinates)):
        x1, y1 = coordinates[i - 1]
        x2, y2 = coordinates[i]
        if x2 - x1 == 0:
            current_slope = float('inf')
        else:
            current_slope = (y2 - y1) / (x2 - x1)

        if current_slope != slope:
            return False

    return True
```

In [127…

```
coordinates = [[1, 2], [2, 3], [3, 4], [4, 5], [5, 6], [6, 7]]
print(checkStraightLine(coordinates))
```

```
True
```

In [ ]: