ASSIGNMENT 7

1.

```
def isomorphic_strings(s, t):
    if len(s) != len(t):
        return False

    s_to_t = {}
    t_to_s = {}

    for ch_s, ch_t in zip(s, t):
        if ch_s in s_to_t:
            if s_to_t[ch_s] != ch_t:
                return False
        else:
            if ch_t in t_to_s:
                return False
            s_to_t[ch_s] = ch_t
            t_to_s[ch_t] = ch_s

    return True
s = "egg"
t = "add"
result = isomorphic_strings(s, t)
print(result)
```

2.

```
def is_strobogrammatic(num):
    valid_pairs = {"00", "11", "69", "96", "88"}
    left, right = 0, len(num) - 1

    while left <= right:
        pair = num[left] + num[right]
        if pair not in valid_pairs:
            return False
        left += 1
        right -= 1

    return True
num = "69"
result = is_strobogrammatic(num)
print(result)
```

3.

```
def add_strings(num1, num2):
    i, j = len(num1) - 1, len(num2) - 1
    result = ""
    carry = 0

    while i >= 0 or j >= 0:
        digit_sum = carry
        if i >= 0:
            digit_sum += ord(num1[i]) - ord('0')
            i -= 1
        if j >= 0:
            digit_sum += ord(num2[j]) - ord('0')
            j -= 1

        carry = digit_sum // 10
        result = str(digit_sum % 10) + result

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

    return result
```

```
num1 = "11"
num2 = "123"
result = add_strings(num1, num2)
print(result)
```

4.

5.

```
def reverseStr(s, k):
    result = ""
    for i in range(0, len(s), 2*k):
        result += s[i:i+k][::-1] + s[i+k:i+2*k]
    return result
s = "abcdefg"
k = 2
result = reverseStr(s, k)
print(result)
```

6.

```
def canShift(s, goal):
    if len(s) != len(goal):
        return False
    s_shifted = s + s
    return goal in s_shifted
s = "abcde"
goal = "cdeab"
result = canShift(s, goal)
print(result)
```

7.

```
def backspaceCompare(s, t):
    def buildString(string):
        stack = []
        for char in string:
            if char != '#':
                stack.append(char)
            elif stack:
                stack.pop()
        return ''.join(stack)

    return buildString(s) == buildString(t)
s = "ab#c"
t = "ad#c"
result = backspaceCompare(s, t)
print(result)
```

8.

```
def checkStraightLine(coordinates):
    n = len(coordinates)
    if n <= 2:
        return True

    x1, y1 = coordinates[0]
    x2, y2 = coordinates[1]
    for i in range(2, n):
        x, y = coordinates[i]
        if (y - y1) * (x2 - x1) != (y2 - y1) * (x - x1):
            return False

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