ASSIGNMENT-6

Problem 1: Odd String Difference

```
def findOddString(words):
    def get_diff_array(word):
        return [ord(word[i+1]) - ord(word[i]) for i in range(len(word) - 1)]

    diff_arrays = [get_diff_array(word) for word in words]
    diff_count = {}

    for diff in diff_arrays:
        diff_tuple = tuple(diff)
        if diff_tuple in diff_count:
            diff_count[diff_tuple] += 1
        else:
            diff_count[diff_tuple] = 1

    for word, diff in zip(words, diff_arrays):
        if diff_count[tuple(diff)] == 1:
            return word

words1 = ["adc", "wzy", "abc"]
words2 = ["aaa", "bob", "ccc", "ddd"]
print(findOddString(words1))
```
Output: "abc"

Problem 2: Words Within Two Edits of Dictionary

```
def wordsWithinTwoEdits(queries, dictionary):
    def within_two_edits(word1, word2):
        diff_count = sum(1 for a, b in zip(word1, word2) if a != b)
        return diff_count <= 2

    result = []
    for query in queries:
        for dict_word in dictionary:
            if within_two_edits(query, dict_word):
                result.append(query)
                break
    return result

queries1 = ["word", "note", "ants", "wood"]
dictionary1 = ["wood", "joke", "moat"]
queries2 = ["yes"]
dictionary2 = ["not"]
print(wordsWithinTwoEdits(queries1, dictionary1))
```
Output: ["word", "note", "wood"]

Problem 3: Destroy Sequential Targets

```
from collections import defaultdict

def destroySequentialTargets(nums, space):
    mod_count = defaultdict(int)
    for num in nums:
        mod_count[num % space] += 1
    max_count = max(mod_count.values())
    candidates = [num for num in nums if mod_count[num % space] == max_count]

    return min(candidates)

nums1 = [3, 7, 8, 1, 1, 5]
space1 = 2
nums2 = [1, 3, 5, 2, 4, 6]
space2 = 2
nums3 = [6, 2, 5]
space3 = 100
print(destroySequentialTargets(nums1, space1))
```
Output: 1

Problem 4: Next Greater Element IV

```
def nextGreaterElement(nums):
    n = len(nums)
    result = [-1] * n
    next_greater = [-1] * n

    for i in range(n - 1, -1, -1):
        for j in range(i + 1, n):
            if nums[j] > nums[i]:
                if next_greater[i] == -1:
                    next_greater[i] = j
                elif nums[j] < nums[next_greater[i]]:
                    next_greater[i] = j

    for i in range(n):
        if next_greater[i] != -1:
            result[i] = next_greater[next_greater[i]]
            if result[i] != -1:
                result[i] = nums[result[i]]

    return result

nums1 = [2, 4, 0, 9, 6]
nums2 = [3, 3]
print(nextGreaterElement(nums1))  # Output: [9, 6, 6, -1, -1]
```

Problem 5: Average Value of Even Numbers That Are Divisible by Three

```
def averageValue(nums):
    even_div_by_3 = [num for num in nums if num % 6 == 0]
    if not even_div_by_3:
        return 0
    return sum(even_div_by_3) // len(even_div_by_3)

nums1 = [1, 3, 6, 10, 12, 15]
nums2 = [1, 2, 4, 7, 10]
print(averageValue(nums1))
```
Output: 9

Problem 6: Most Popular Video Creator

```
from collections import defaultdict
```

```
def mostPopularCreator(creators, ids, views):
    popularity = defaultdict(int)
    max_views = defaultdict(lambda: (-1, ''))

    for creator, video_id, view_count in zip(creators, ids, views):
        popularity[creator] += view_count
        if (view_count > max_views[creator][0]) or (view_count == max_views[creator][0] and video_id < 
max_views[creator][1]):
            max_views[creator] = (view_count, video_id)

    max_popularity = max(popularity.values())
    result = [[creator, max_views[creator][1]] for creator in popularity if popularity[creator] == max_popularity]

    return result

creators1 = ["alice", "bob", "alice", "chris"]
ids1 = ["one", "two", "three", "four"]
views1 = [5, 10, 5, 4]
creators2 = ["alice", "alice", "alice"]
ids2 = ["a", "b", "c"]
views2 = [1, 2, 2]
print(mostPopularCreator(creators1, ids1, views1))
```
Output: [["alice", "one"], ["bob", "two"]]

Problem 7: Minimum Addition to Make Integer Beautiful

```
def minAdditionToMakeBeautiful(n, target):
    def digitsum(num):
        return sum(int(digit) for digit in str(num))

    x = 0
    while digitsum(n + x) > target:
        x += 1
    return x

n1 = 16
target1 = 6
n2 = 467
target2 = 6
n3 = 1
target3 = 1
print(minAdditionToMakeBeautiful(n1, target1))
```
Output: 4

Problem 8: Split Message Based on Limit

```
def splitMessage(message, limit):
    length = len(message)
    for b in range(1, length + 1):
        suffixlength = len(f"<{b}/{b}>")
        if suffixlength + 1 > limit:
            return []

        totallength = sum((len(f"<{a}/{b}>") for a in range(1, b + 1)))
        availablelength = limit * b - totallength

        if availablelength >= length:
            parts = []
            idx = 0
            for a in range(1, b + 1):
                suffix = f"<{a}/{b}>"
                part_length = limit - len(suffix)
                part = message[idx:idx + part_length]
                idx += part_length
                parts.append(part + suffix)
            return parts
    return []

message1 = "this is really a very awesome message"
limit1 = 9
message2 = "short message"
limit2 = 15
print(splitMessage(message1, limit1))
```

Output: ['thi<1/14>', 's i<2/14>', 's r<3/14>', 'eal<4/14>', 'ly <5/14>', 'a v<6/14>', 'ery<7/14>', ' aw<8/14>', 'eso<9/14>', 'me<10/14>', ' m<11/14>', 'es<12/14>', 'sa<13/14>', 'ge<14/14>']