1. Odd String Difference

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

File Edit Format Run Options Window Help

```python
def find_unique_string(words):
    # Function to compute the difference array for a given word
    def compute_difference_array(word):
        return [ord(word[i + 1]) - ord(word[i]) for i in range(len(word) - 1)

    # Dictionary to count the occurrences of each difference array
    difference_dict = {}

    # List to store the difference arrays for comparison later
    differences = []

    # Compute the difference arrays for all words and count their occurrence
    for word in words:
        diff_array = compute_difference_array(word)
        diff_tuple = tuple(diff_array)
        differences.append(diff_array)
        if diff_tuple in difference_dict:
            difference_dict[diff_tuple] += 1
        else:
            difference_dict[diff_tuple] = 1

    # Find the unique difference array
    unique_diff = None
    for diff in difference_dict:
        if difference_dict[diff] == 1:
            unique_diff = diff
            break

    # Find the word corresponding to the unique difference array
    for i in range(len(words)):
        if tuple(differences[i]) == unique_diff:
            return words[i]

# Example usage
words = ["adc", "wzy", "abc"]
print(find_unique_string(words))  # Output: "abc"
```

IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
abc
>>>
```

2. Words Within Two Edits of Dictionary

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

File Edit Format Run Options Window Help

```python
def words_within_two_edits(queries, dictionary):
    # Helper function to count the number of differing characters
    def count_edits(word1, word2):
        return sum(1 for a, b in zip(word1, word2) if a != b)

    # List to store the result
    result = []

    # Check each query word
    for query in queries:
        # Check against each word in the dictionary
        for dict_word in dictionary:
            if count_edits(query, dict_word) <= 2:
                result.append(query)
                break  # No need to check further if we found a match

    return result

# Example usage
queries = ["word", "note", "ants", "wood"]
dictionary = ["wood", "joke", "moat"]
print(words_within_two_edits(queries, dictionary))  # Output: ["word", "note"
```

IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
['word', 'note', 'wood']
>>>
```

3. Destroy Sequential Targets

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

File Edit Format Run Options Window Help

```python
from collections import defaultdict

def min_seed_to_destroy_max_targets(nums, space):
    remainder_count = defaultdict(int)
    min_value_for_remainder = {}

    for num in nums:
        remainder = num % space
        remainder_count[remainder] += 1

        if remainder not in min_value_for_remainder:
            min_value_for_remainder[remainder] = num
        else:
            min_value_for_remainder[remainder] = min(min_value_for_remainder

    max_count = max(remainder_count.values())

    min_seed = float('inf')

    for remainder in remainder_count:
        if remainder_count[remainder] == max_count:
            min_seed = min(min_seed, min_value_for_remainder[remainder])

    return min_seed

# Example usage
nums = [3, 7, 8, 1, 1, 5]
space = 2
print(min_seed_to_destroy_max_targets(nums, space))  # Output: 1
```

IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
1
>>>
```

4. Next Greater Element IV

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

```python
def find_second_greater(nums):
    n = len(nums)
    answer = [-1] * n
    stack1 = []
    stack2 = []

    for i in range(n - 1, -1, -1):
        # Process stack2 to find the second greater element
        while stack2 and nums[stack2[-1]] <= nums[i]:
            stack2.pop()

        if stack2:
            answer[i] = nums[stack2[-1]]

        # Process stack1 to maintain candidates for second greater element
        while stack1 and nums[stack1[-1]] <= nums[i]:
            stack2.append(stack1.pop())

        stack1.append(i)

    return answer

# Example usage
nums = [2, 4, 0, 9, 6]
print(find_second_greater(nums))
```

IDLE Shell 3.12.1

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
[6, 6, 6, -1, -1]
>>>
```

5. Average Value of Even Numbers That Are Divisible by Three

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

```python
def average_even_divisible_by_three(nums):
    # Step 1: Filter numbers that are even and divisible by 3
    filtered_numbers = [num for num in nums if num % 2 == 0 and num % 3 == 0]

    # Step 2: Calculate the sum and count of these numbers
    if not filtered_numbers:
        return 0  # If there are no such numbers, return 0

    total_sum = sum(filtered_numbers)
    count = len(filtered_numbers)

    # Step 3: Calculate the average and round down
    average = total_sum // count

    return average

# Example usage
nums = [1, 3, 6, 10, 12, 15]
print(average_even_divisible_by_three(nums))  # Output: 9
```

IDLE Shell 3.12.1

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
9
>>>
```

6. Most Popular Video Creator

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

```python
def find_most_popular_creator(creators, ids, views):
    from collections import defaultdict

    # Dictionaries to store total views and most popular video per creator
    total_views = defaultdict(int)
    most_popular_video = {}

    # Populate total views and most popular video per creator
    for creator, video_id, view in zip(creators, ids, views):
        total_views[creator] += view
        if creator not in most_popular_video:
            most_popular_video[creator] = (view, video_id)
        else:
            # Compare views or lexicographical order of video ID if views are
            if view > most_popular_video[creator][0] or (view == most_popula
                most_popular_video[creator] = (view, video_id)

    # Determine the highest popularity
    max_views = max(total_views.values())

    # Collect all creators with the highest popularity
    result = []
    for creator in total_views:
        if total_views[creator] == max_views:
            result.append([creator, most_popular_video[creator][1]])

    return result

# Example usage
creators = ["alice", "bob", "alice", "chris"]
ids = ["one", "two", "three", "four"]
views = [5, 10, 5, 4]

print(find_most_popular_creator(creators, ids, views))  # Output: [["alice",
```

IDLE Shell 3.12.1

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
[['alice', 'one'], ['bob', 'two']]
>>>
```

7. Minimum Addition to Make Integer Beautiful

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

```python
def sum_of_digits(number):
    return sum(int(digit) for digit in str(number))

def find_min_x(n, target):
    x = 0
    while sum_of_digits(n + x) > target:
        x += 1
    return x

# Example usage
n = 38
target = 10
print(find_min_x(n, target))
```

IDLE Shell 3.12.1

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
2
>>>
```

8. Split Message Based on Limit

Duplicate.py - C:\Users\sleva\Desktop\Duplicate.py (3.12.1)

File Edit Format Run Options Window Help

```python
def split_message(message, limit):
    # Helper function to calculate the length of the suffix "<a/b>"
    def suffix_length(a, b):
        return len(f"<{a}/{b}>")

    n = len(message)

    # Determine the number of parts needed
    for b in range(1, n + 1):
        if suffix_length(b, b) > limit:
            return []  # Impossible to create even a single valid part

        total_suffix_length = sum(suffix_length(i, b) for i in range(1, b + 1
        max_message_length = limit * b - total_suffix_length

        if max_message_length >= n:
            # Found the correct number of parts
            result = []
            part_index = 1
            current_index = 0

            while current_index < n:
                # Calculate the length of the current part
                current_suffix = f"<{part_index}/{b}>"
                max_part_length = limit - len(current_suffix)

                part_message = message[current_index:current_index + max_part
                result.append(part_message + current_suffix)

                current_index += max_part_length
                part_index += 1

            return result

    return []  # If no valid splitting found

# Example usage
message = "this is really a very awesome message"
limit = 9
print(split_message(message, limit))  # Output: ["thi<1/14>","s i<2/14>","s r
```

Ln: 41 Col: 0

IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\sleva\Desktop\Duplicate.py
['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>']
>>>
```

Ln: 6 Col: 0