Python Code Explanation

Overview

We have a list that can contain positive and negative integers. We need to find two elements such that their sum is closest to zero.

Agenda

- Problem Statement
- Solution Logic using Pseudocode
- Relevant Data Structures
- Possible Edge Cases
- Evaluating Solution Efficiency

Problem Statement: You're given a list of integers, both positive and negative. The task is to find two elements in the list whose sum is closest to zero.

Solution Logic using Pseudocode:

Sort the List:

- Sorting helps in efficiently finding pairs with the closest sum.

Initialize Variables:

- `closest_sum`: Tracks the closest sum found so far.
- `closest_pair`: Keeps the pair of elements associated with the closest sum.

Iterate Through Sorted List:

- For each adjacent pair of numbers in the sorted list:
  - Calculate their sum.
  - If this sum has a smaller absolute value than `closest_sum`, update `closest_sum` and `closest_pair`.

Return Closest Pair:

- After iterating through the entire list, return the pair of elements whose sum is closest to zero.

Relevant Data Structures:

- List: To store the input list of integers.
- Variables: To store the closest sum found and the pair of elements associated with it.

Possible Edge Cases:

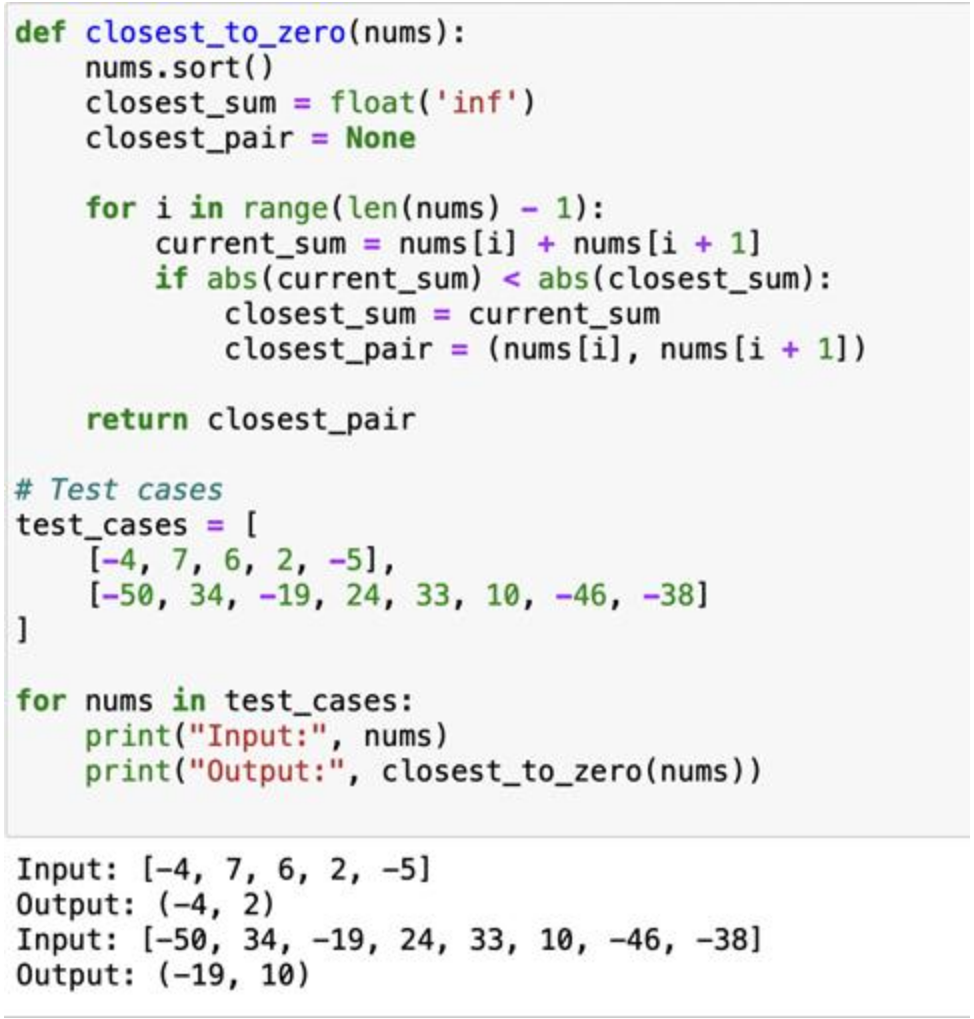
- Empty List: Handle the case when the input list is empty.
- Single Element List: If the list contains only one element, there's no pair to consider.
- Duplicate Elements: Ensure the solution handles duplicate elements appropriately.

Evaluating Solution Efficiency:

- **Time Complexity:** O(n log n) due to sorting the list.
- **Space Complexity:** O(1) as the solution only uses a constant amount of extra space.

Efficiency Evaluation:

- Test the solution with large input lists to ensure it runs within a reasonable time frame.
- Compare its runtime with alternative approaches, if available, to validate its efficiency.



```python
def closest_to_zero(nums):
    nums.sort()
    closest_sum = float('inf')
    closest_pair = None

    for i in range(len(nums) - 1):
        current_sum = nums[i] + nums[i + 1]
        if abs(current_sum) < abs(closest_sum):
            closest_sum = current_sum
            closest_pair = (nums[i], nums[i + 1])

    return closest_pair

# Test cases
test_cases = [
    [-4, 7, 6, 2, -5],
    [-50, 34, -19, 24, 33, 10, -46, -38]
]

for nums in test_cases:
    print("Input:", nums)
    print("Output:", closest_to_zero(nums))
```

```
Input: [-4, 7, 6, 2, -5]
Output: (-4, 2)
Input: [-50, 34, -19, 24, 33, 10, -46, -38]
Output: (-19, 10)
```

Video File

https://www.youtube.com/watch?v=tP7GUzI0lI4

Thank You

Biplab Mondal