**Problem Name: Two Sum Less Than K**

**Topics:**

**Companies:**

**Level:** Easy

**Language:** C++

**Problem Statement**: Given an array nums of integers and integer k, return the maximum sum such that there exists i < j with nums[i] + nums[j] = sum and sum < k. If no i, j exist satisfying this equation, return -1.

**Input Format:**

First line of the input contain integer n (size of list)

Second line contain n space separated integer list values.

Last line contain integer value pos representing value of node to delete.

Ex:

5

1 2 3 4 5

1

**Output Format:** Print linked list after removing node having value pos

**Constraints:**

* The number of the nodes in the given list is in the range [2, 1000].
* -1000 <= Node.val <= 1000
* The value of each node in the list is **unique**.
* The node to be deleted is **in the list** and is **not a tail** node

**Examples:**

**Input:** head = [4,5,1,9], node = 5

**Output:** [4,1,9]

**Explanation:** You are given the second node with value 5, the linked list should become 4 -> 1 -> 9 after calling your function.

**Brute force Solution:**

# Explanation: Two Pointers

Sort and use two pointers to try all reasonable combinations. This is a generic solution for an unconstrained problem.

**Code:**

**Time Complexity**: O(NlogN)

**Space Complexity:** O(n) for the sorted array (assuming we cannot modify the input)

**Optimized Solution:**

# Explanation: Counting Sort

Since 1 <= A[i] <= 1000, we can apply the counting sort, which takes linear time. Then, we use two indexes to search for a pair, like in the solution above.

**Code:**

**Time Complexity**: O(n+m) where m is the maximum value (1000 for this problem).

**Space Complexity:** O(m)