2563. Count the Number of Fair Pairs

Given a **0-indexed** integer array `nums` of size `n` and two integers `lower` and `upper`, return *the number of fair pairs*.

A pair `(i, j)` is **fair** if:

- `0 <= i < j < n`, and
- `lower <= nums[i] + nums[j] <= upper`

Example 1:

```
Input: nums = [0,1,7,4,4,5], lower = 3, upper = 6
Output: 6
Explanation: There are 6 fair pairs: (0,3), (0,4), (0,5), (1,3), (1,4), and (1,5).
```

Example 2:

```
Input: nums = [1,7,9,2,5], lower = 11, upper = 11
Output: 1
Explanation: There is a single fair pair: (2,3).
```

Constraints:

- `1 <= nums.length <= 10^5`
- `nums.length == n`
- `-10^9 <= nums[i] <= 10^9`
- `-10^9 <= lower <= upper <= 10^9`

2563. Count the Number of Fair Pairs

```
/*
   Sorting+Binary search
   Time complexity: O(n log n)
   Space complexity: O(1)
*/
typedef long long ll;
typedef std::vector<int> vi;

class Solution {
   public:
      ll countFairPairs(vi& nums, int lower, int upper) {
         std::sort(nums.begin(),nums.end());
         int n=nums.size();
         ll ans=0;
         for(int i=0;i<n-1;++i){
            int up=std::upper_bound(nums.begin()+i+1,nums.end(),upper-nums[i])-nums.begin();
            int low=std::lower_bound(nums.begin()+i+1,nums.end(),lower-nums[i])-nums.begin();
            ans+= (up-low);
         }
         return ans;
      }
};
```