Check If Array Pairs Are Divisible By K

solved by	Senan
Platform	LeetCode
difficulty	Medium
# Serial	1497
tags	Array Maths Vector
language	C++
solved on	@October 1, 2024
link	https://leetcode.com/problems/check-if-array-pairs-are-divisible-by-k/description/
☑ Completion	☑

Intuition

The goal is to determine whether it's possible to pair up elements from the array such that the sum of each pair is divisible by `k`. If we can group the elements in such a way that their remainders (when divided by `k`) cancel each other out, then it is possible to rearrange the array as desired.

Approach

1. **Mod Function**: To handle negative numbers and ensure the remainder is always non-negative, we use the custom `mod` function: `((n % k) + k) % k`.
2. **Remainder Count**: We create a `vector<int> arrange(k, 0)` to count how many numbers from `arr` have a remainder of `i` when divided by `k`.
3. **Validation**:
   - The remainder `0` must have an even count, as these elements must be paired with each other.
   - For other remainders `i`, the count of elements with remainder `i` must be equal to the count of elements with remainder `k-i`, as they will cancel each other out when summed.
4. **Final Check**: If all conditions are met, we return `true`, otherwise `false`.

Complexity

Time Complexity:

The time complexity is determined by iterating over the array and performing constant-time operations for each element.

- Iterating through the array takes $O(n)$, where n is the size of `arr`.
- Validating the remainders takes $O(k)$, where k is the modulus divisor.

Thus, the overall time complexity is: $O(n + k)$

Space Complexity:

We are using a vector of size to store the remainder counts.

Thus, the space complexity is: $O(k)$

Code

```
class Solution {
private:
    int mod(int n, int k) {
        return ((n % k) + k) % k;
    }
public:
    bool canArrange(vector<int>& arr, int k) {
            // Can't pair odd number of elements
        if(arr.size() % 2 == 1) return false;

                // To store count of remainders
        vector<int> arrange(k, 0);

        // Count occurrences of each remainder
        for(int num : arr) {
            arrange[mod(num, k)]++;
        }

        // Check if numbers with remainder 0 can form pairs
        if(arrange[0] % 2 != 0) return false;

        // Check if remainder i and remainder k-i have the same count
        int i = 1, j = k - 1;
        while(i < j) {
            if(arrange[i++] != arrange[j--]) return false;
        }

        return true;
    }
};
```