NumberofSubArrayswithGCDequaltoK **in C++**

```cpp
#include <iostream>
#include <vector>
using namespace std;

class NumberofSubArrayswithGCDequaltoK {
public:
    int subarrayGCD(vector<int>& nums, int k) {
        int count = 0;
        int n = nums.size();

        for (int sp = 0; sp < n; sp++) {
            int ans = 0;
            for (int ep = sp; ep < n; ep++) {
                ans = gcd(ans, nums[ep]);

                if (ans < k) {
                    break;
                }
                if (ans == k) {
                    count++;
                }
            }
        }

        return count;
    }

    int gcd(int a, int b) {
        if (a == 0) {
            return b;
        }
        return gcd(b % a, a);
    }
};

int main() {
    NumberofSubArrayswithGCDequaltoK solution;

    // Hard-coded input
    vector<int> nums = {2, 4, 6, 8, 3, 9};
    int k = 3;

    int result = solution.subarrayGCD(nums, k);
    cout << "Number of subarrays with GCD equal to "
<< k << ": " << result << endl;

    return 0;
}
```

Input:

nums = {2, 4, 6, 8, 3, 9}
k = 3

We'll check **all subarrays** and see how many have GCD = 3.

📊 Dry Run Table

sp	Subarray	ans (GCD)	Matches k?
0	[2]	2	✗
0	[2, 4]	2	✗
0	[2, 4, 6]	2	✗
0	[2, 4, 6, 8]	2	✗
0	[2, 4, 6, 8, 3]	1	✗ (GCD < k) – break
1	[4]	4	✗
1	[4, 6]	2	✗
1	[4, 6, 8]	2	✗
1	[4, 6, 8, 3]	1	✗ (GCD < k) – break
2	[6]	6	✗
2	[6, 8]	2	✗
2	[6, 8, 3]	1	✗ (GCD < k) – break
3	[8]	8	✗
3	[8, 3]	1	✗ (GCD < k) – break
4	[3]	3	✓
4	[3, 9]	3	✓
5	[9]	9	✗

✓ Final Count

We found **2 subarrays** where the GCD is exactly 3:

- [3]
- [3, 9]

🧠 Explanation of Logic

You're using a **nested loop**:

- Outer loop: start point sp
- Inner loop: end point ep
- You maintain a running GCD of the subarray
- If GCD < k, you **break** early (smart optimization)
- If GCD == k, increment the counter

	And your GCD function is correct, based on the Euclidean algorithm. 🧪 **Output:** Number of subarrays with GCD equal to 3: 2
Number of subarrays with GCD equal to 3: 2	