Distinct Elements in each Window in C++

```cpp
#include <iostream>
#include <unordered_map> // for unordered_map

using namespace std;

void printDistinct(int arr[], int n, int k) {
    unordered_map<int, int> m; // Declaration of unordered_map to store element frequencies

    // Count frequencies of first window
    for (int i = 0; i < k; i++) {
        m[arr[i]]++;
    }

    // Print the size of the map for the first window
    cout << m.size() << " ";

    // Process subsequent windows
    for (int i = k; i < n; i++) {
        // Remove the element that is moving out of the window
        m[arr[i - k]]--;

        // Remove the element from map if its count becomes zero
        if (m[arr[i - k]] == 0) {
            m.erase(arr[i - k]);
        }

        // Add the new element to the map
        m[arr[i]]++;

        // Print the size of the map for the current window
        cout << m.size() << " ";
    }
}

int main() {
    int arr[] = {10, 10, 5, 3, 20, 5};
    int n = sizeof(arr) / sizeof(arr[0]); // Calculate the size of the array
    int k = 4; // Size of the window

    // Call the function to print distinct elements in every window of size k
    printDistinct(arr, n, k);

    cout << endl;

    return 0;
}
```

Input

```
arr[] = {10, 10, 5, 3, 20, 5}
n = 6
k = 4
```

📋 Dry Run Table (Sliding Window)

Window Index	Elements in Window	Frequencies Map (unordered_map)	Distinct Count
[0–3]	10, 10, 5, 3	{10: 2, 5: 1, 3: 1}	3
[1–4]	10, 5, 3, 20	{10: 1, 5: 1, 3: 1, 20: 1}	4
[2–5]	5, 3, 20, 5	{5: 2, 3: 1, 20: 1}	3

✅ Final Output

```
3 4 3
```

Output:
3 4 3