Sliding Window Algorithm (i, j, n)

C++ Code Implementation:

```
#include <iostream>
#include <vector>

using namespace std;

void slidingWindow(vector<int>& nums, int k) {
    int n = nums.size();  // Total size of array
    int i = 0, j = 0;     // i -> left pointer, j -> right pointer
    int sum = 0;          // Example: sum of elements in the window

    while (j < n) {  // Expand window by moving j
        sum += nums[j];

        // If window size reaches k, process the window
        if (j - i + 1 == k) {
            cout << "Window [" << i << ", " << j << "]: Sum = " << sum << endl;

            // Shrink window from the left
            sum -= nums[i];
            i++;
        }

        j++;  // Move right pointer forward
    }
}

int main() {
    vector<int> nums = {1, 3, 2, 6, 4, 5, 7};
    int k = 3;  // Window size
    slidingWindow(nums, k);
    return 0;
}
```

Time Complexity:

```
- Best Case: O(N)
- Average Case: O(N)
- Worst Case: O(N) (Each element is processed at most twice)
```