|  |  |
| --- | --- |
| **Kadane Max Sum Subarray C++** | |
| #include <iostream>  using namespace std;  int maxSubArraySum(const int arr[], int n) {      int currentSum = arr[0]; // Initialize current sum and overall sum      int overallSum = arr[0];        for (int i = 1; i < n; i++) {          if (currentSum >= 0) {              currentSum += arr[i]; // Add current element to current sum if positive          } else {              currentSum = arr[i]; // Start new subarray if current sum is negative          }            if (currentSum > overallSum) {              overallSum = currentSum; // Update overall sum if current sum is greater          }      }        return overallSum; // Return maximum sum found  }  int main() {      const int arr[] = {5, 6, 7, 4, 3, 6, 4}; // Input array      int n = sizeof(arr) / sizeof(arr[0]); // Determine the number of elements in the array      cout << maxSubArraySum(arr, n) << endl; // Output maximum sum of subarray      return 0;  } | **Dry Run with Given Input** Given array:  {5,6,7,4,3,6,4} ****Step 2.1: Initialize Variables**** currentSum = arr[0] = 5  overallSum = arr[0] = 5 ****Step 2.2: Iterate Through Array****  | **Index (i)** | **Element (arr[i])** | **currentSum** | **overallSum** | | --- | --- | --- | --- | | 0 | 5 | 5 | 5 | | 1 | 6 | (5 + 6) = 11 | **11** | | 2 | 7 | (11 + 7) = 18 | **18** | | 3 | 4 | (18 + 4) = 22 | **22** | | 4 | 3 | (22 + 3) = 25 | **25** | | 5 | 6 | (25 + 6) = 31 | **31** | | 6 | 4 | (31 + 4) = 35 | **35** |  **Step 3: Final Answer** Maximum Subarray Sum = 35 |
| Output:- 35 | |