Max Subarray sum in C++

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

int maxsub(int arr[], int n) {
    int res = arr[0];
    int maxEnding = arr[0];
    for (int i = 1; i < n; i++) {
        maxEnding = max(maxEnding + arr[i], arr[i]);
        res = max(res, maxEnding);
    }
    return res;
}

int main() {
    int arr[] = {-3, 8, -2, 4, -5, 6};
    int n = sizeof(arr) / sizeof(arr[0]);
    cout << maxsub(arr, n) << endl;
    return 0;
}
```

Input:

arr[] = {-3, 8, -2, 4, -5, 6}
n = 6

📓 **Variables:**

- res: Stores the **maximum subarray sum found so far**
- maxEnding: Stores the **maximum subarray sum ending at the current index**

🔁 **Dry Run Table:**

i	arr[i]	maxEnding = max(maxEnding + arr[i], arr[i])	res = max(res, maxEnding)
0	-3	maxEnding = -3	res = -3
1	8	max(-3 + 8, 8) = 8	res = 8
2	-2	max(8 - 2, -2) = 6	res = 8
3	4	max(6 + 4, 4) = 10	res = 10
4	-5	max(10 - 5, -5) = 5	res = 10
5	6	max(5 + 6, 6) = 11	res = 11

✅ **Final Output:**

11

11