分割数组的最大差值

题目描述：

给定一个由若干整数组成的数组 nums ，可以在数组内的任意位置进行分割，将该数组分割成两个非空子数组（即左数组和右数组），分别对子数组求和得到两个值，计算这两个值的差值，请输出所有分割方案中，差值最大的值。

输入描述：

第一行输入数组中元素个数 n，1 < n <= 100000

第二行输入数字序列，以空格进行分隔，数字取值为 4 字节整数

输出描述：

输出差值的最大取值

示例 1

输入：

```
6
1 -2 3 4 -9 7
```

输出：

```
10
```

说明：

将数组 nums 划分为两个非空数组的可行方案有：

左数组 = [1] 且 右数组 = [-2,3,4,-9,7]，和的差值 = | 1 - 3 | = 2

左数组 = [1,-2] 且 右数组 = [3,4,-9,7]，和的差值 = |-1 - 5 | = 6

左数组 = [1,-2,3] 且 右数组 = [4,-9,7]，和的差值 = | 2 - 2 | = 0

左数组 = [1,-2,3,4] 且 右数组 = [-9,7]，和的差值 = | 6 - (-2)| = 8

左数组 = [1,-2,3,4,-9] 且 右数组 = [7]，和的差值 = |-3 - 7 | = 10

最大的差值为 10

```
#include <iostream>

#include <vector>

#include <climits>
```

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

int main() {
    long long  n, num,sum =0;
    cin >> n;
    vector<long long> nums;
    vector<long long> prex;
    for(int i = 0; i < n; ++i) {
        cin >> num;
        nums.push_back(num);
        sum += num;
        if(i < n -1)
            prex.push_back(sum);
    }
    sort(prex.begin(), prex.end());
    long long  res = max(abs(2*prex[0] - sum), abs(2*prex.back() - sum));
    cout << res <<endl;
    return 0;

}
// 64 位输出请用 printf("%lld")
```