P1182 数列分段 Section II

题目描述

对于给定的一个长度为N的正整数数列 A_1~N_A，现要将其分成 M（M≤N）段，并要求每段连续，且每段和的最大值最小。

关于最大值最小：

例如一数列 4 2 4 5 1要分成 3 段。

将其如下分段：

[4 2][4 5][1]

第一段和为 6，第 2 段和为 9，第 3 段和为 1，和最大值为 9。

将其如下分段：

[4][2 4][5 1]

第一段和为 4，第 2 段和为 6，第 3 段和为 6，和最大值为 6。

并且无论如何分段，最大值不会小于 6。

所以可以得到要将数列 4 2 4 5 1 要分成 3 段，每段和的最大值最小为 6。

输入格式

第 1行包含两个正整数 N,M。

第 2 行包含 N个空格隔开的非负整数 Ai，含义如题目所述。

输出格式

一个正整数，即每段和最大值最小为多少。

输入样例

```
5 3
4 2 4 5 1
```

输出样例

```
6
```

解析

基本的二分答案。题目中有一个很重要的一点：最大子段和的最小值必然是全局的最大值。怎么理解呢，看下面的例子：

2 2
1 100

2个数据分两段，很明显最小的子段和就是100。因为无论你怎么划分，都不可能小于100。所以，我们在二分的时候，左边界即为全局最大值，有边界则为全局整数和。

编码

```
#include<bits/stdc++.h>

using namespace std;
int const MaxN = 1e5 + 5;
long long n, m, a[MaxN];

long long CountNum(long long x) {
    //默认从分了一组开始
    long long cnt = 1;
    long long sum = 0;
    //尝试放入标记牌
    for (int i = 1; i <= n; i++) {
        //如果能够合并就合到一起进行分组
        if (sum + a[i] <= x) {
            sum += a[i];
        }
        //无法合并，新建一个分组
        else {
            sum = a[i];
            cnt++;
        }
    }
    return cnt;
}

long long Search(long long left, long long right) {
    while (left <= right) {
        long long mid = left + (right - left) / 2;
        long long num = CountNum(mid);
        //数量合适，但是要寻找尽可能短的距离，因此右边界左移
        if (num == m) {
            right = mid - 1;
        }
            //分出的组数大于目标数，说明间距太小，应该增加间距
        else if (num > m) {
            left = mid + 1;
        }
            //分出的组数小于目标数，说明间距太大，应该减少间距
```

```
        else if (num < m) {
              right = mid - 1;
          }
     }
     return left;
}


int main() {
     //累计总和
    long long rightN = 0;
     cin >> n >> m;
     long long leftN = 0;
     for (int i = 1; i <= n; i++) {
          cin >> a[i];
          rightN += a[i];
          //计算左边界
        leftN = max(leftN,a[i]);
     }
     //计算最大子段和
    cout << Search(leftN, rightN);
     return 0;
}
```