算法2阶段

第1讲 差分与前缀和

引例1：区间和

给出数组 a 的元素 a[1], a[2],… , a[n] 的值, 进行 m 次询问，每次问你区间 [L,R] 数之和为多少?

样例输入

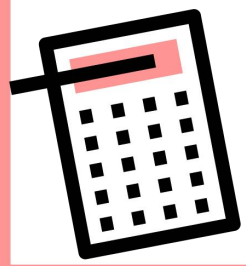
```
6
1 -6 5 -4 2 4
3
1 2
3 6
2 5
```

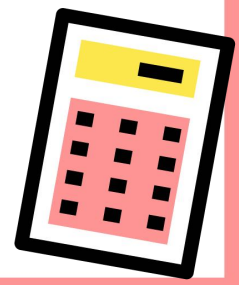
样例输出

```
-5
7
-3
```

数据范围

$2 \le m, n \le 10000$；$1 \le L < R \le n$；$-1000 \le a[i] \le 1000$





```cpp
#include<iostream>
using namespace std;
const int MAXN = 1e4 + 5;
int a[MAXN],n,m;
int main() {
    cin >> n;
    for (int i = 1;i <= n; i++)
        cin >> a[i];
    cin >> m;
    while (m--) {
        int l,r,sum = 0;
        cin >> l >> r;
        for (int i = l; i <= r; i++)
            sum += a[i];
        cout << sum << endl;
    }
    return 0;
}
```

时间复杂度：O(MN)

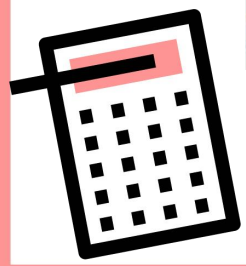
冗余分析：

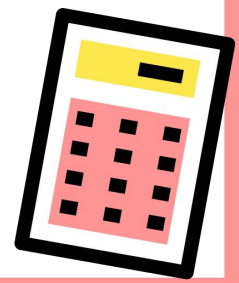
2~5:

a[2] + a[3] + ... + a[5]

1~9:

a[1] + a[2] + a[3] + ... + a[5] + ... + a[9]



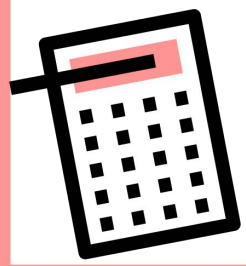


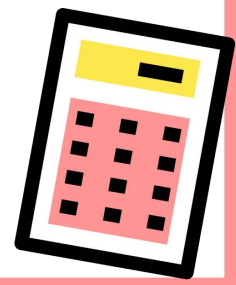
前缀和：数据预处理方法之一，使用于大量计算区间和。

s[i] = a[1] + a[2] + a[3] + ... + a[i-1] + a[i]

下标	1	2	3	4	5	6	7
数字a[]	4	9	10	3	6	1	8
前缀和s[]							

s[i] = s[i-1] + a[i]



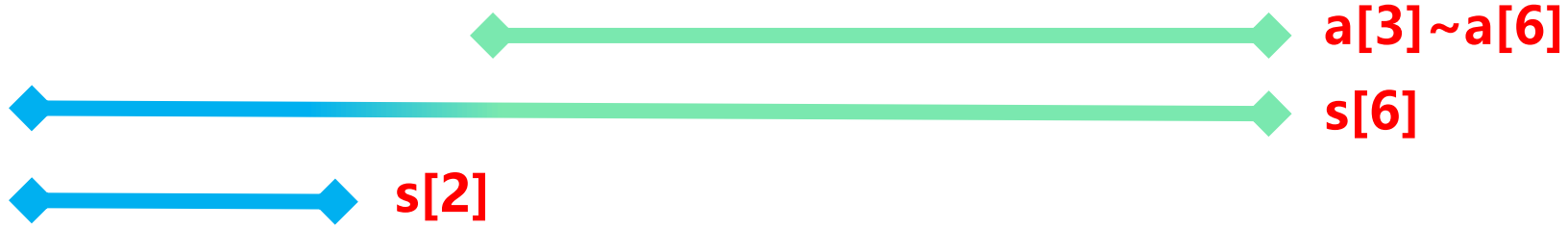


前缀和：数据预处理方法之一，使用于大量计算区间和。

s[i] = a[1] + a[2] + a[3] + ... + a[i-1] + a[i]

计算a[3]+...+a[6]的和

下标	1	2	3	4	5	6	7
数字a[]	4	9	10	3	6	1	8
前缀和s[]	4	13	23	26	32	33	41

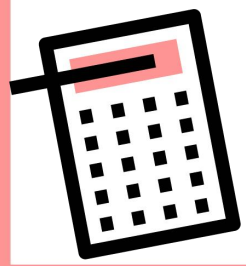


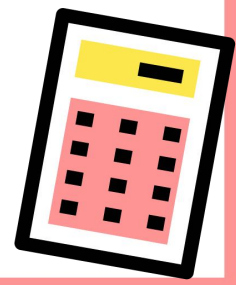
前缀和：数据预处理方法之一，使用于大量计算区间和。

s[i] = a[1] + a[2] + a[3] + ... + a[i-1] + a[i]

下标	1	2	3	4	5	6	7
数字a[]	4	9	10	3	6	1	8
前缀和s[]	4	13	23	26	32	33	41

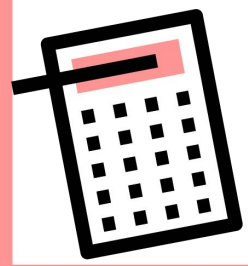
a[l] + a[l+1] + ... + a[r] = s[r] - s[l-1]



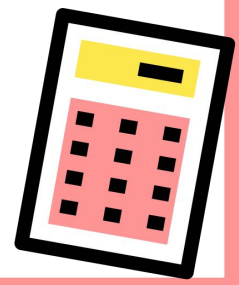


```cpp
#include<iostream>
using namespace std;
const int MAXN = 1e4 + 5;
int a[MAXN],s[MAXN],n,m;
int main()
{
    cin >> n;
    for (int i = 1;i <= n; i++) {
        cin >> a[i];
        s[i] = s[i-1] + a[i];
    }

    cin >> m;
    while (m--) {
        int l,r;
        cin >> l >> r;
        cout << a[r] - a[l-1] << endl;
    }
    return 0;
}
```



时间复杂度：O(N)



P1147
连续自然数和

对一个给定的自然数M，求出所有的连续的自然数段（每一段至少有两个数），这些连续的自然数段中的全部数之和为M。

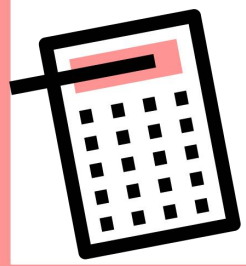
例子：1998+1999+2000+2001+2002=10000，所以从1998到2002的一个自然数段为M=10000的一个解。

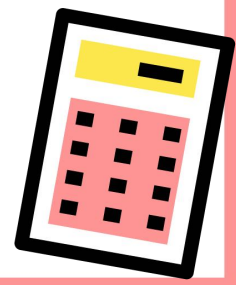
样例输入

10000

样例输出

18 142
297 328
388 412
1998 2002





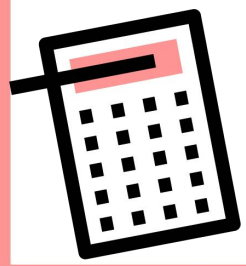
解题思路:

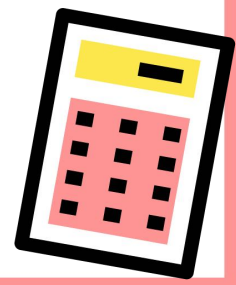
1、连续区间和 --> 前缀和

2、枚举区间头尾

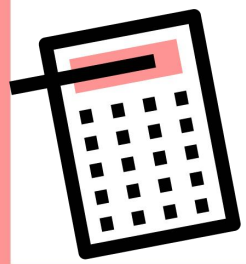
3、区间和超过M时结束（优化）

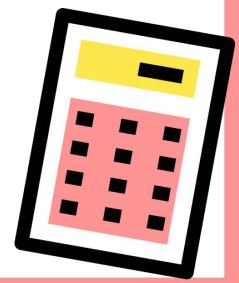
ps:固定区间尾,二分查找区间头（进一步优化）





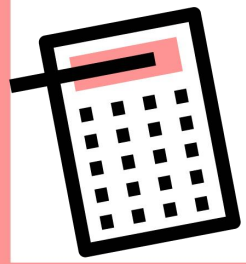
```cpp
//P1147 连续自然数和
#include<iostream>
#include<algorithm>
using namespace std;
const int MAXN = 2e6 + 5;
int s[MAXN];
int main()
{
    int m;
    cin >> m;
    for (int i = 1; i <= m; i++)
        s[i] = s[i-1] + i;
    for (int i = 1; i <= m; i++)
    {
        for (int j = i + 1; j <= m; j++)
        {
            if (s[j] - s[i-1] == m)
            {
                cout << i << " " << j << endl;
            }
            if (s[j] - s[i-1] > m)   break;  //优化
        }
    }
    return 0;
}
```

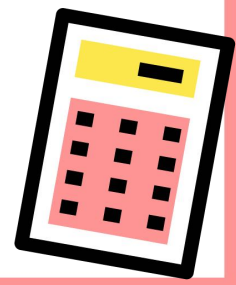




P5638
光骓者的荣耀

小K打下的江山一共有n个城市，城市i和城市i+1有一条双向高速公路连接，走这条路要耗费时间ai。
小K为了关心人民生活，决定定期进行走访。他每一次会从1号城市到n号城市并在经过的城市进行访问。其中终点必须为城市n。
不仅如此，他还有一个传送器，传送半径为k，也就是可以传送到i−k和i+k。如果目标城市编号小于1则为1，大于n则为n。
但是他的传送器电量不足，只能传送一次，况且由于一些原因，他想尽量快的完成访问，于是就想问交通部部长您最快的时间是多少。
注意：他可以不访问所有的城市，使用传送器不耗费时间。





输入格式

两行,第一行n,k。
第二行n-1个整数,第i个表示ai。

输出格式

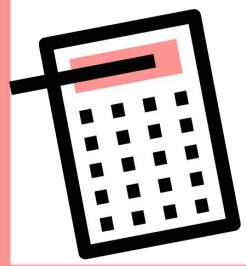
一个整数，表示答案。

输入样例

4 0
1 2 3

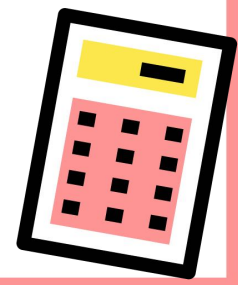
输出格式

6



数据范围

n≤10^6 k ≤10^6 ai≤10^12

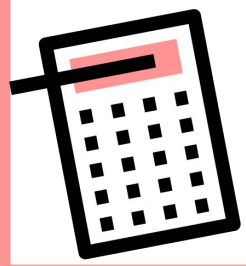


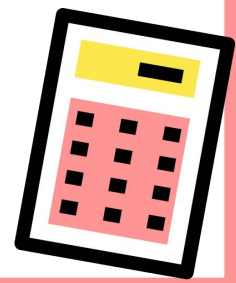
解题思路:

1、耗费时间 = 总时间 - 跳过路程的时间

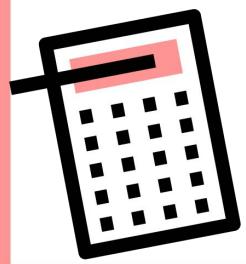
2、求跳过路程最大值

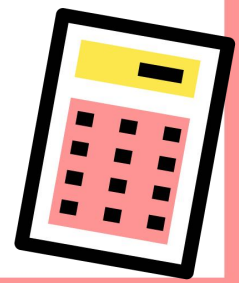
3、跳过路程(区间和) --> 前缀和





```cpp
//P5638 光骓者的荣耀
#include<iostream>
using namespace std;
const int MAXN = 1e6 + 5;
long long a[MAXN],s[MAXN];
int main()
{
    int n,k;
    cin >> n >> k;
    for (int i = 1; i <= n - 1; i++)
    {
        cin >> a[i];
        s[i] = s[i-1] + a[i];
    }
    long long ans = 0;
    for (int i = k; i <= n - 1; i++)  //跳过区间终点
    {
        ans = max(ans,s[i] - s[i-k]);
    }
    cout << s[n - 1] - ans << endl;
    return 0;
}
```





引例2：
区间操作

给出数组 a 的元素 a[1], a[2],… , a[n] 的值, 进行 m 次操作，第i次操作使区间 [L,R] 的数增加ci，问操作之后的数组a的每个数字是多少?

样例输入

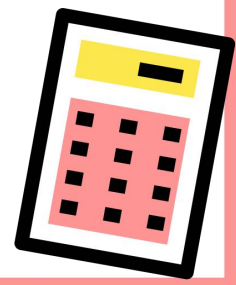
6
1 -6 5 -4 2 4
3
1 2 1
3 6 2
2 5 -1

样例输出

2 -6 6 -3 3 6

数据范围

$2 \le m, n \le 10000$；$1 \le L < R \le n$；$-1000 \le a[i],ci \le 1000$



```cpp
#include<iostream>
using namespace std;
const int MAXN = 1e4 + 5;
int a[MAXN],n,m;
int main() {
    cin >> n;
    for (int i = 1;i <= n; i++)
        cin >> a[i];
    cin >> m;
    while (m--) {
        int l,r,c;
        cin >> l >> r >> c;
        for (int i = l; i <= r; i++)
            a[i] += c;
    }
    for (int i = 1; i <= n; i++)
        cout << a[i] << " ";
    return 0;
}
```

时间复杂度：O(MN)

冗余分析：

2~5:

a[2]、a[3]、a[4]、a[5]

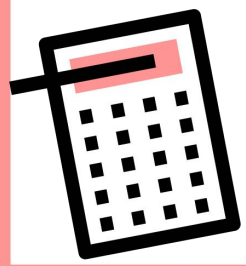
1~9:

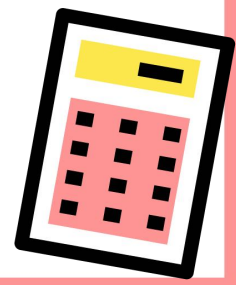
a[1]、a[2]、a[3]、…、a[5]、…、a[9]

差分：数据预处理方法之一，使用于大量区间操作。
差分可以看出是前缀和的逆运算，即差分数组d[]的前缀和是原数组a[]。

下标	1	2	3	4	5	6	7
数字a[]	4	9	10	3	6	1	8
差分d[]							

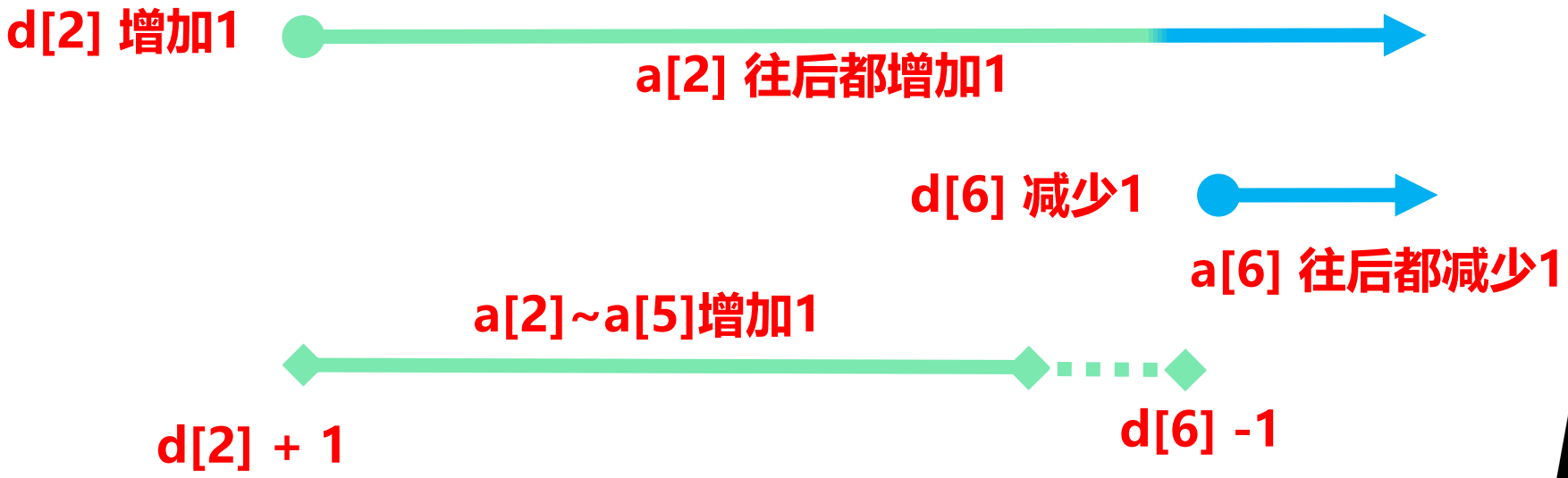
$$d[i] = a[i] - a[i-1]$$





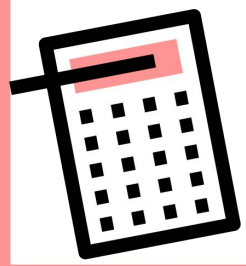
在区间a[2]~a[5]上，所有数字增加1

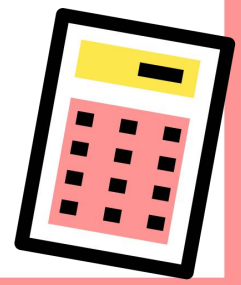
下标	1	2	3	4	5	6	7
数字a[]	4	9	10	3	6	1	8
差分d[]	4	5	1	-7	3	-5	7



```cpp
#include<iostream>
using namespace std;
const int MAXN = 1e4 + 5;
int a[MAXN],d[MAXN];
int n,m;
int main(){
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
        d[i] = a[i] - d[i-1];  //差分预处理
    }
    cin >> m;
    for (int i = 1; i <= m; i++) {
        cin >> l >> r >> c;
        d[l] += c;
        d[r+1] -= c;
    }
    for (int i = 1; i <= n; i++) {
        a[i] = a[i-1] + d[i];
        cout << a[i] << " ";
    }
    return 0;
}
```

时间复杂度：O(N)





P2367
语文成绩

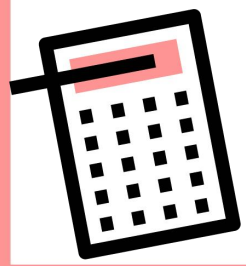
语文老师总是写错成绩，所以当她修改成绩的时候，总是累得不行。她总是要一遍遍地给某些同学增加分数，又要注意最低分是多少。你能帮帮她吗?

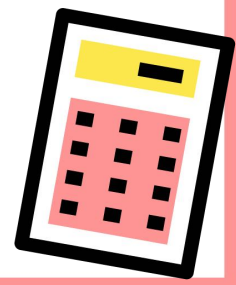
输入格式

第一行有两个整数n,p代表学生数与增加分数的次数。
第二行有n个数,a1~an代表各个学生的初始成绩。
接下来p行,每行有三个数,x,y,z,代表给第x个到第y个学生每人增加z分。

输出格式

输出仅一行,代表更改分数后,全班的最低分。





样例输入

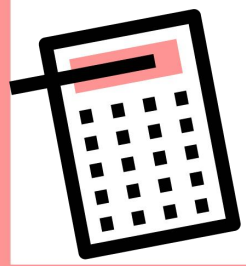
3 2
1 1 1
1 2 1
2 3 1

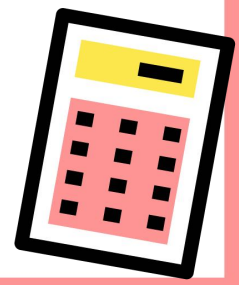
样例输出

2

数据范围

$n \le 5 \times 10^6, p \le n$



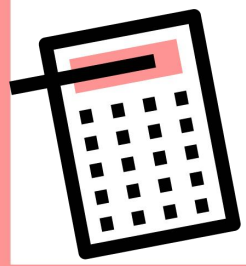


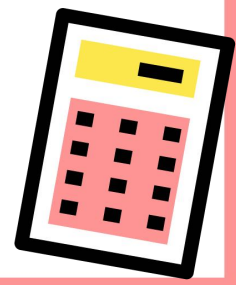
解题思路:

1、区间统一操作 --> 差分

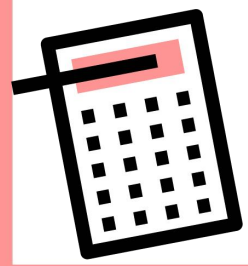
2、前缀和求最终成绩

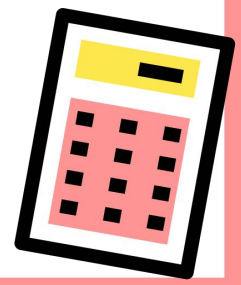
3、求最小值





```cpp
//P2367 语文成绩
#include<iostream>
using namespace std;
const int MAXN = 5e6 + 5;
int n,p;  //学生人数n  修改次数p
int a[MAXN],d[MAXN];  //原始成绩a[i]  差分d[i]
int main(){
    cin >> n >> p;
    for (int i = 1; i <= n; i++) {  //初始化差分数组
        cin >> a[i];
        d[i] = a[i] - a[i-1];
    }
    for (int i = 1; i <= p; i++) {  //区间操作,差分处理
        int x,y,z;
        cin >> x >> y >> z;
        d[x] += z;
        d[y+1] -= z;
    }
    int ans = 2e9;
    for (int i = 1; i <= n; i++) {  //前缀和求成绩,记录最小值
        a[i] = a[i-1] + d[i];
        ans = min(ans,a[i]);
    }
    cout << ans << endl;
    return 0;
}
```





课后练习
P6568 水壶

有n个容量无穷大的水壶，它们从1~n编号，初始时i号水壶中装有Ai单位的水。

你可以进行不超过k次操作，每次操作需要选择一个满足$1 \le x \le n-1$的编号x，然后把x号水壶中的水全部倒入x+1号水壶中。

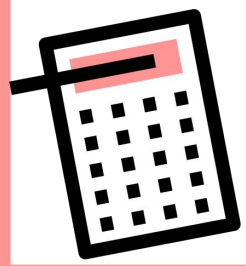
最后你可以任意选择恰好一个水壶，并喝掉水壶中所有的水。现在请你求出，你最多能喝到多少单位的水。

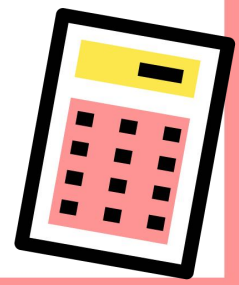
输入格式

第一行一个正整数n，表示水壶的个数。
第二行一个非负整数k，表示操作次数上限。
第三行n个非负整数，相邻两个数用空格隔开，表示水壶的初始装水量A1,A2,…,An。

输出格式

一行，仅一个非负整数，表示答案。





样例输入

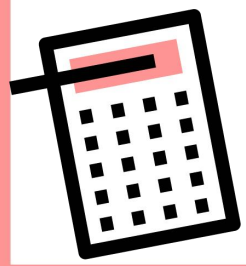
10
5
890 965 256 419 296 987 45 676 976 742

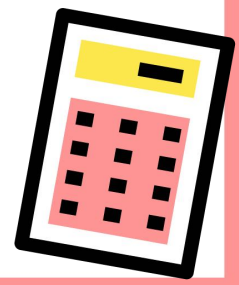
样例输出

3813

数据范围

$1 \le n \le 10^6, 0 \le k \le n-1, 0 \le A_i \le 10^3$



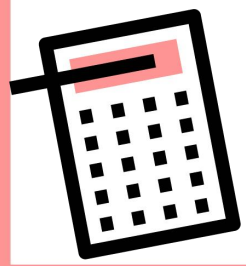


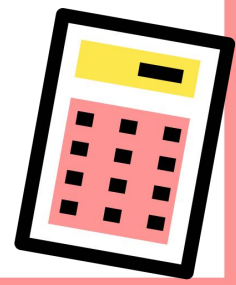
解题思路:

1、求长度为k+1的区间和最大值

2、区间和 --> 前缀和数据预处理

3、枚举求最大值





```cpp
//P6568 水壶
#include<iostream>
#include<algorithm>
using namespace std;
const int MAXN = 1e6 + 5;
int a[MAXN],s[MAXN];  //a[i]:第i个水壶水量  s[i]:a[i]的前缀和
int n,k;  //水壶数n 操作数k
int main()
{
    cin >> n >> k;
    for (int i = 1; i <= n; i++) {  //输入及数据处理
        cin >> a[i];
        s[i] = s[i-1] + a[i];
    }
    int ans = 0;  //记录长度为k+1的最大区间和
    for (int i = k + 1; i <= n; i++) {
        ans = max(ans,s[i] - s[i-k-1]);
    }
    cout << ans << endl;
    return 0;
}
```