C++ 小学组

1.竞赛排名(rank)

题目描述:

阳光中学正在举办一年一度的数学竞赛。今年的竞赛吸引了 n 名学生参加，每名学生都有一个独特的编号(从 1 到 n)。比赛结束后，老师们需要根据学生的得分来确定最终排名。不过，学校的排名规则有点特殊:得分相同的学生会获得相同的名次，持且下一个名次会跳过相应的人数。例如，如果有两名学生并列第一，那么下一名学生的名次就足第三名，而不是第二名。

现在按顺序给定 n 名学生的编号和得分，请按同样顺序输出名学生的排名

输入输出格式

输入格式:

第一行包含一个整数 n，表示参赛学生人数。

第二行包含 n 个整数，其中第 i 个数 a_i 表示编号为 i 的同学的得分。

输入:

第一行包含一个整数 n，表示参赛学生人数。

第二行包含 n 个整数，其中第 i 个数 a_i 表示编号为 i 的同学的得分。

输出:

输出 n 行，第 i 行输出编号为 i 的学生的最终名次。

输入样例 1:

4

3 12 9 9

输出样例 1:

4

1

2

2

输入样例 2:

3

3 6 9

输出样例 2:

3

1

2

输入样例 3:

4

100 100 100 100

输出样例 3:

1

1

1

1

用时/内存:

1000MS/512MB

```
#include<bits/stdc++.h>
using namespace std;
int n;
int a[100005];
int b[100005];
int main()
{
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        b[i]=a[i];
    }
    sort(a+1,a+n+1);
    reverse(a+1,a+n+1);
    for(int i=1;i<=n;i++){
        int l=1,r=n;
        int m=0;
        int pos=0;
        while(l<=r){
            m=(l+r)/2;
            if(a[m]==b[i]){
                pos=m;
                r=m-1;
            }
            else if(a[m]>b[i]){
                l=m+1;
            }
            else{
                r=m-1;
            }
        }
        printf("%d\n",pos);
    }
    return 0;}
```

2.宝藏(treasure)

题目描述:

在遥远的古代，有一个隐藏在密林深处的宝藏。宝藏的入口处有两块古老的石碑，每块石碑上面各刻着一组神秘的数字序列。只有从小到大输入两块石碑都有的数字(重复的数字也需要输出)，才能打开宝藏的大门。聪明的探险家们，你们能解开这个谜题，找到宝藏吗?

输入:

第一行两个正整数 n,m 分别表示两块石碑上数字序列的长度。

第二行 n 个整数表示第一块石碑上的数字序列 A。

第三行 m 个整数表示第二块石碑上的数字序列 B。

输出:

一行，从小到大输出两块石碑都有的数字(重复的数字也要输出)。

数据保证至少存在一个数字在两块石碑的数字序列中均出现。

输入样例 1:

```
4 5
4 2 5 2
2 1 9 2 4
```

输出样例 1:

```
2 2 4
```

输入样例 2:

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

输出样例 2:

1

补充说明

【样例说明】

样例一:两个序列均出现了 2 个 2,1 个 4，故输出 2 2 4

样例二:两个序列均出现了 1 个 1，故输出 1。

【数据范围】

对于 30% 的致据，n,m<=10，1≤=a[i],b[i]<=10;

对于 60%的数据，n,m<=1000,1<=a[i],b[i]<=1000

对于 100% 的数据，n,m<=1e5,1<=a[i],b[i]<=1e9

用时/内存:

1000MS/512MB

```
#include<bits/stdc++.h>
using namespace std;
int n,m;
int a[100005],b[100005];
int r[100005],t=0;
int main(){
	cin>>n>>m;
	for(int i=1;i<=n;i++)
		cin>>a[i];
	for(int i=1;i<=m;i++)
		cin>>b[i];
	sort(a+1,a+1+n);
```

```
    sort(b+1,b+1+m);
    int i=1,j=1;
    while(i<=n && j<=m){
        if(a[i]<b[j]){
            i++;
        }else if(a[i]>b[j]){
            j++;
        }else{
            r[++t]=a[i];
            i++;
            j++;
        }
    }
    for(int i=1;i<=t;i++)
        cout<<r[i]<<' ';
    return 0;
}
```

3.破损的石板铭文(stone)

题目描述:

考古队在遗迹中发现了一块刻有 n 个数字的古老石板。据文献记载，这块石板原本是一句"回文神谕"(即正着读和反着读都一样的序列)。

然而由于风化，石板上混入了一些杂质数字。好在这些杂质数字非常特殊:它们全部都是同一种数字 x，你可以选择这种数字 x，并将石板上任意数量(零个、部分或全部)的该数字清除。

如果清除掉这些杂质后，剩下的数字能组成一句回文神谕，这块石板就被称为“圣物"。例如:[3,1,2,3,1]这块石板就是圣物。把里面的 3 全部删除，剩下的数[1,2.1]就是一个回文神谕。

[1,4,4,4,1]这块石板也是圣物。可以选择只删除中间的 4，剩下的[1,4,4,1]组成一个“回文神谕”

[1.2.1]也是圣物。它本身就是一个回文数组；

[1.2.3]不是圣物。请你判断，眼前的这块刻有几个数字石板 a 是否是传说中的圣物?

输入:

输入共 2 行。第一行一个数字 n，表示序列的长度。

第二行 n 个数字，表示石板 a 上刻的 n 个数字

输出:

如果 a 是圣物，输出 YES；否则，输出 NO。

输入样例 1:

5

1 4 4 1 4

输出样例 1:

YES

输入样例 2:

6

1 5 3 2 4 1

输出样例 2:

NO

数据范围

对于 40%的数据，1≤n≤2000,1≤a≤n

对于 100%的数据，1≤n≤2*10^5. 1≤a≤n

用时/内存:

1000MS/512MB

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 2e5+50;
int n;
int a[MAXN];
signed main() {
	cin >> n ;
	for (int i = 1; i <= n; i++) {
		cin >> a[i];
	}
	int l = 1, r = n;
	while (a[l] == a[r]) {
		l++;
		r--;
	}
	if (l >= r) {
		cout << "YES";
		return 0;
	}
	int t1 = a[l];
	int t2 = a[r];l = 1, r = n;
	bool f1= 1,f2 = 1;
	while (l < r) {
		if (a[r] == t1) r--;
		else if (a[l] == t1) l++;
		else if (a[l] == a[r]) l++, r--;
		else {
			f1 = 0;
			break;
		}
	}l = 1, r = n;
	while (l < r) {
		if (a[r] == t2) r--;
		else if (a[l] == t2) l++;
```

```
            else if (a[l] == a[r]) l++, r--;
            else {
                f2 = 0;
                break;
            }
        }
        if (f1 == 0 && f2 == 0) {
            cout << "NO";
        }
        else {
            cout << "YES";
        }
}
```

4.巧克力礼盒(chocolate)

题目描述:

甜品店老板有 n 块印有小写字母的巧克力，现在他想选择一些巧克力分装到 k 个精美的礼盒中送给顾客(不必将 n 个巧克力全部分装到 k 个盒子中)。

老板是个完美主义者，他要求:

1、每个礼盒里的巧克力字母经过排列后，必须是一个回文串。

2、所有的 k 个礼盒都必须至少分到一块巧克力。

3、为了公平起见，老板希望这 k 个礼盒中，含巧克力块数最少的礼盒，其数量也要尽可能地多。

你能帮老板算出，这个“最少块数"的最大值是多少吗?

补充说明:回文串是正读和反读都一样的字符串。比如:abcba 是回文串，abc 不是。

输入:

输入共 2 行。

第一行两个数字 n 和 k，表示印有小写字母的巧克力的块数和礼盒个数。

第二行一个长度为 n 的字符串，该字符串只包含小写字母，表示 n 个印有小写字母的巧克力。

输出:

一行一个整数，表示含巧克力块数最少的礼盒的最多数量。

输入样例 1:

8 2

bxyaxzay

输出样例 1:

6 3

输入样例 2:

Wwfyfy

输出样例 2:

2

补充说明

【样例说明】

在样例一中，s="bxyaxzay”，k=2。一种可行的分配方案如下:bxyaxzay(字母 z 的巧克力没有被装到盒子里)，染色后

交换第一个盒子的两个字符(下标 1 和 4)，得到 axybxzay

交换要二个盒子的两个字符(下标 5 和 8)，得到 axybyzax

现在，对于每个盒子，从左到右写出对应的字符，得到两个字符串 aba 和 xyyx。它们都是回文串，最短的长度为 3。可以证明、最短回文串的最大长度无法超过 3

【数据范围】

对于 10%的数据，K=1

另外存在 10%的教据，k=n

另外存在 10%的数据，保证 n 个巧克力只印有一种字母

对于 100%的数据，1≤k≤n:2*10^5，字符串只有小写字母组成

用时/内存:

1000MS/512MB

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 2e5+50;
int n, k, sum, ans;
int a[250];
signed main() {
	cin >> n >> k;
	for (int i = 1; i <= n; i++) {
		char c;
		cin >> c;
		a[c]++;
	}
	for (int i = 'a'; i <= 'z'; i++) {
		sum += a[i] / 2;
		ans += a[i] % 2;
	}
	int res = sum / k;
	ans += (sum % k) * 2;
	res *= 2;
	if (ans >= k) res++;
```

```
    cout << res;
}
```

5.自习室的节能灯(light)

题目描述:

小明在自习室看书，自习室的灯被系统设定为在 0 时开启，m 时准时关闭。系统还预设了 n 个自动切换状态的时间点。

小明发现，只有在灯亮着的时候他才能专注看书。

幸运的是，控制面板出了一个小故障，允许小明在剩余的空闲时间点中，额外手动技下一次开关(最多只能一次)。当然，小明也可以选择不按下开关,这一按会改变之后所有的灯光状态。

小明想尽可能多地争取看书时间。请你计算，如果小明采取最佳策略，最多能让灯亮多长时间?

输入:

第一行两个整数，n,m;

以下一行有 n 个整数，即改变时间点

输出:

一个整数，即最大亮着的时间

输入样例 1:

3 10

4 6 7

输出样例 1:

8

输入样例 2:

2 12

1 10

输出样例 2:

9

补充说明

【样例说明】

样例一中，在 5 时刻改变一次灯的状态，此时灯在 0-4,5-6,7-10 这三个时间段是开着的，总时长足 4+1+3=8

样例二中，在 2 时刻改变一次灯的状态，此时灯在 0-1,2-10 这两个时间段是开着的，总时长是 1+8=9

【数据范围】

对于 30%的数据:$1 \le n \le 1000$；

对于 100%的致据，$1 \le n \le 10^5$,$1 \le m \le 10^9$。

用时/内存:

1000MS/512MB

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 2e5+50;
int n, m, cnt_odd, cnt_cap;
int a[MAXN], c[MAXN], odd[MAXN], cap[MAXN];
```

```
signed main() {
    cin >> n >> m;
    for (int i = 1 ; i <= n; i++) {
        cin >> a[i];
    }
    for (int i = 1; i <= n; i++) {
        c[i] = a[i] - a[i - 1];
    }
    c[n + 1] = m - a[n];
    for (int i = 1; i <= n + 1; i++) {
        if (i % 2 == 1) odd[i] = c[i] + odd[i - 1], cap[i] = cap[i - 1];
        else cap[i] += c[i] + cap[i - 1], odd[i] = odd[i - 1];
    }
    int ans = odd[n + 1];
    for (int i = 1; i <= n + 1; i++) {
        if (c[i] <= 1) continue;
        int res = odd[i - 1];
        res += cap[n + 1] - cap[i];
        res += c[i] - 1;
        ans = max(ans, res);
    }
    cout << ans;
}
```

6.外卖员的电量管理(energy)

题目描述:

外卖员 小 Z 接到了一个长街任务，街上有 n 个配送点。第 i 个配送点会消耗电动车 a_i,的电量。小 Z 可以选择从第 l 个点开始工作，一直到第 r 个点结束。

车的电瓶有一个特殊的“自动过热保护”机制:初始过热值 g=0。

每经过一个配送点，过热值 g 增加 a_i。

如果 g 严格大于安全阈值 x，保护系统会强制关机冷却，过热值 g 瞬间重置为 0。

为了保证效率，小 Z 希望工作结束时，车子仍处于“热机“状态(即 g>0)以便继续前往下一条街。

请问有多少种[l,r]段落，能让他在完成任务后过热值 g 不为 0？

输入:

第一行包括两个整数 n,x,分别表示有 n 个配送点和安全阈值 x;

第二行包括 n 个整数，依次表示 a1,a2...an。

输出:

对于每个测试用例，输出一个整数，即符合题目条件的方案数。

输入样例 1:

```
4 2
1 1 1 1
```

输出样例 1:

```
8
```

输入样例 2:

```
6 3
1 2 1 4 3 8
```

输出样例 2:

```
10
```

补充说明

【样例说明】

样例一中，满定条件的(1，r)有(1.1).(1.2),(1.4),(2.2),(2.3).(3,3),(3,4),(4.4);

【数据范围】

对于30%的数据:$1 \le n \le 100$；

对于100%的致据，$1 \le n \le 2*10^5$, ,$1 \le x \le 10^9$，$1 \le a_i \le 10^9$

用时/内存:

1000MS/512MB

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 2e5 + 50;
long long n, x, ans;
long long s[MAXN];
long long dp[MAXN];

signed main() {
  cin >> n >> x;
  for (int i = 1; i <= n; i++) {
    long long v;
    cin >> v;
    s[i] = s[i - 1] + v;
  }
  for (int i = n; i >= 1; i--) {
    long long target = s[i - 1] + x;
    int idx = upper_bound(s + 1, s + n + 1, target) - s;

    if (idx <= n) {
      dp[i] = 1 + dp[idx + 1];
    } else {
      dp[i] = 0;
    }
    ans += dp[i];
  }
  cout << n * (n + 1) / 2 - ans << endl;
```

```
  return 0;
}
```