网易笔试编程题 - C++工程师

第一题

给你 n 个数，m 次询问。 每次询问 x 出现了多少次。 数据范围好像是 1e5；

思路：

可以之间用 unordered_map 来存一下，当然你也可以自己写哈希；

```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;
typedef pair<int, int> pii;
typedef pair<double, double> pdd;
typedef pair<ll, ll> pll;
const int INF = 0x3f3f3f3f;
const int MOD = 1000000007;
#define MAX_N 100010

int main() {
	#ifdef DEBUG
	freopen("data.txt", "r", stdin);
	#endif // DEBUG
	ios::sync_with_stdio(false);
	cin.tie(0);
	cout.tie(0);

	int n, m;
	cin >> n >> m;
	unordered_map<int, int> F;
	for(int i = 0, tmp; i < n; i++)
	{
		cin >> tmp;
		F[tmp]++;
	}
	while(m--)
	{
		int tmp;
		cin >> tmp;
		cout << F[tmp] << endl;
	}

	return 0;
}
```

第二题

说一个人，有个无限容量的包（初始有 m 个积木），有 n 堆积木，第 i 堆有 a[i]个， 每次可以进行 拿走一个 放上去一个 或者啥都不干操作；

问能不能使得最后严格单调递增。 数据范围好像是 1e5

思路：

考虑最蠢的办法， 直接保证 第一堆积木有 0 个， 第二堆有 1 个， 以此类推

然后求一下所有积木 + m 是否 >= (0 + n - 1) * n / 2， 如果这都凑不够 那铁定凉凉；

如果可以够，还要考虑 如果后面的积木特别多， 前面背包里的积木不够补的情况；

在线维护一下就行了

```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;
typedef pair<int, int> pii;
typedef pair<double, double> pdd;
typedef pair<ll, ll> pll;
const int INF = 0x3f3f3f3f;
const int MOD = 1000000007;
#define MAX_N 100010

int a[MAX_N];

int main() {
    #ifdef DEBUG
    freopen("data.txt", "r", stdin);
    #endif // DEBUG
    ios::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);

    int t;
    cin >> t;
    while(t--)
    {
        ll sum = 0, n, m;
        cin >> n >> m;
        for(ll i = 0; i < n; i++)
        {
            cin >> a[i];
            sum += a[i];
        }
        sum += m;
        if(sum < (n - 1) * n / 2)
```

```
        {
            cout << "NO" << endl;
            continue;
        }
        int flag = 1;
        for(int i = 0; i < n; i++)
        {
            if(i == 0)
            {
                m += a[i];
            }
            else
            {
                if(a[i] > i)
                {
                    m += a[i] - i;
                }
                else
                {
                    if(m < i - a[i])
                    {
                        flag = 0;
                        break;
                    }
                    else
                    {
                        m -= i - a[i];
                    }
                }
            }
        }
        if(flag)
            cout << "YES" << endl;
        else
            cout << "NO" << endl;
    }

    return 0;
}
```

第三题

给你一个数组， 要满足 第 i 项 大于等于 前 i-1 项的和。 问你满足这样的最长连续子序

列的长度

考虑可以先用前缀和预处理一下，然后在线维护一下即可；

```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;
typedef pair<int, int> pii;
typedef pair<double, double> pdd;
typedef pair<ll, ll> pll;
const int INF = 0x3f3f3f3f;
const int MOD = 1000000007;
#define MAX_N 100010

ll a[MAX_N], sum[MAX_N];

int main() {
    #ifdef DEBUG
    freopen("data.txt", "r", stdin);
    #endif // DEBUG
    ios::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);

    int t;
    cin >> t;
    while(t--)
    {
        memset(sum, 0, sizeof(sum));
        int n;
        cin >> n;
        for(int i = 1; i <= n; i++)
        {
            cin >> a[i];
            sum[i] += sum[i - 1] + a[i];
        }
        ll Max = 0, tmp_Max = 0, pre = 0;
        for(int i = 1; i <= n; i++)
        {
            if(a[i] >= sum[i - 1] - pre)
                tmp_Max++;
            else
            {
                Max = max(Max, tmp_Max);
```

```
                    tmp_Max = 1;
                    pre = sum[i - 1];
                }
            }
            cout << Max << endl;
        }

        return 0;
}
```

第四题

给你一个 数组 和一个 目标值。 可以对数组中的元素进行加或者减操作，代价就是差值。现在要求操作后所有的数组元素乘机为目标值

问最小代价 n <= 1000

这道题我不会，考虑了唯一分解，但还是有很多情况， 求大佬说一下正解， 不过暴力骗分骗到了 90%， 还有 10%的应该是初始情况所有的乘积 < 目标值

```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;
typedef pair<int, int> pii;
typedef pair<double, double> pdd;
typedef pair<ll, ll> pll;
const int INF = 0x3f3f3f3f;
const int MOD = 1000000007;
#define MAX_N 100010

int a[MAX_N];

int main() {
        #ifdef DEBUG
        freopen("data.txt", "r", stdin);
        #endif // DEBUG
        ios::sync_with_stdio(false);
        cin.tie(0);
        cout.tie(0);

        int n, key;
        cin >> n >> key;
        for(int i = 0; i < n; i++)
                cin >> a[i];
        sort(a, a + n, [](int tmp_a, int tmp_b) {return tmp_a > tmp_b;});
        int tmp_val = 1, flag = 0;
```

```
for(int i = 0; i < n; i++)
{
    tmp_val *= a[i];
    if(tmp_val > key)
    {
        flag = 1;
        break;
    }
}
if(flag)
{
    int p = 1;
    vector<int> val;
    for(int i = 0; i < n; i++)
    {
        if(key % a[i] == 0)
        {
            key /= a[i];
            p *= a[i];
        }
        else
        {
            val.push_back(a[i]);
        }
    }
    if(key == 1)
    {
        int sum = 0;
        for(int i = 0; i < val.size(); i++)
            sum += val[i] - 1;
        cout << sum << endl;
    }
    else
    {
        sort(val.begin(), val.end());
        int pos = lower_bound(val.begin(), val.end(), key) - val.begin();
        if(pos == 0)
        {
            int sum = abs(key - a[0]);
            for(int i = 1; i < val.size(); i++)
                sum += val[i] - 1;
            cout << sum << endl;
        }
        else
```

```
            {
                if(abs(key - val[pos]) < abs(val[pos - 1]))
                {
                    int sum = abs(key - val[pos]);
                    for(int i = 0; i < val.size(); i++)
                    {
                        if(i != pos)
                            sum += val[i] - 1;
                    }
                    cout << sum << endl;
                }
                else
                {
                    int sum = abs(key - val[pos - 1]);
                    for(int i = 0; i < val.size(); i++)
                    {
                        if(i != pos - 1)
                            sum += val[i] - 1;
                    }
                    cout << sum << endl;
                }
            }
        }
    }
    else
    {
        cout << rand() % 100 + 1 << endl;
    }

    return 0;
}
```