|  |
| --- |
| [**Codeforces Round 859 (Div. 4)**](https://codeforces.com/contest/1807) |

[Problem - E - Codeforces](https://codeforces.com/contest/1807/problem/E)

我的思路：每次可以打印一个序列，系统返回一个值，根据这个值与前缀和的大小比较可以知道，要查找的目标在不在这个序列中。当时不知道为什么，认为打印左半边，答案不在左半边的话就要打印mid+1，r，然后根据答案在哪半边把程序写了两个分支

题解思路：默认打印左半边，l到mid，对应二分中<=和 >，再根据系统返回情况改l，r就行。**解题的关键在于将操作与二分的情况联系起来**

题解用的递归二分，return 自身，不断缩小范围调用自己，递到l == r时开始return，因为其他递的过程都已经到了“归”的一步，所以就一直return 到最初的一层函数，return得到的答案到main函数中

#include <iostream>

#include <cstring>

#include <cstdio>

#include <algorithm>

using namespace std;

const int N = 2e5 + 10;

typedef long long ll;

int f[N];int n;

// 默认左半边，如果不等于前缀和，说明就是在这个区间，猜对了，继续左半边

// 反之则是在右半边，猜小了，往右边

int find\_special\_stone(int l, int r) {

if (l == r) {

return l;

}

int mid = (l + r) / 2;

cout << "? " << mid-l+1 << " ";

for (int i = l; i <= mid; i++) {

cout << i << " ";

}

cout << endl;

int total\_weight;

cin >> total\_weight;

if (total\_weight != f[mid] - f[l-1]) {

return find\_special\_stone(l, mid);

} else {

return find\_special\_stone(mid+1, r);

}

}

int main(){

int t = 0;

cin >> t;

while(t--){

cin >> n;

f[0] = 0;

for(int i = 1;i <= n;i++){

scanf("%d",&f[i]);

f[i] += f[i-1];

// cout << f[i] << " ";

}

int l = find\_special\_stone(1,n);

printf("! %d\n",l);

cout.flush();

}

return 0;

}

用于测试

**1**

**5**

**1 2 3 4 5**

**7**

**4**

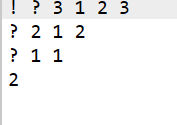
**1**

二刷：卡壳点：

**output limit exceed，输出的时候for循环i <= mid 写成l <= mid，**

**递归输出结果异常**

**理论上，要开long long，数据太弱，所以都ac**



**用cout递归二分时没有先储存好递归结果再cout**

导致



程序先输出了！再输出递归结果（仅在cout中会获得值并立刻立刻输出

Printf中则会计算出所有结果后输出

但是endl可以刷新缓冲区，不用再写一个刷新的语句。

#include <iostream>

#include <algorithm>

#include <cstring>

#include <cstdio>

using namespace std;

typedef long long ll;

const int N = 2e5 + 10;

ll f[N];

int dfs(int l,int r){

if(l == r) {

return l;

}

int mid = l + r >> 1;

printf("? %d",mid - l + 1);

for(int i = l;i <= mid;i++){

if(i == mid) cout << " " << mid << endl;

else cout << " " << i;

}

ll x;

scanf("%lld",&x);

if(x == f[mid] - f[l - 1]){

return dfs(mid+1,r);

}

else return dfs(l,mid);

}

int main(){

int t;

cin >> t;

while(t--){

int n;

scanf("%d",&n);

for(int i = 1;i <= n;i++){

scanf("%lld",&f[i]);

f[i] += f[i-1];

}

int k = dfs(1,n);

cout << "! " << k << endl;

}

return 0;

}

三刷：

卡壳点是遗漏了多组输入的说明，题目看太快了（英文题看着烦？）

整体较顺利

[云剪贴板 - 洛谷 | 计算机科学教育新生态 (luogu.com.cn)](https://www.luogu.com.cn/paste/rtblmmu5)

四刷，秒了