Codeforces Round #753（Div3）A~D 题解

A. Linear Keyboard

题意

给定一个确定顺序的、由26个小写字母组成的键盘，每组给出一个单词，求出手敲完该单词所运动的距离。

思路

首先创建两个字符串a和s分别储存键盘和单词。

如果只是将键盘存在字符串中，那在过程中想寻找特定字母的位置，就需要遍历字符串寻找，数据量大的情况下有可能会超时。

所以可以创建一个整型数组k，用其下标为0到25存储a到e的对应位置，随后模拟手移动过程中查询 `k[s[i]-'a']` 的值进行减法运算即可。

总结

做题时需要有对储存的数据进行预处理操作的意识。

也可以用map做，自己应该尽快去学习c++相关的知识，而不是继续在c++程序中用c做题。

AC代码

Problem	Lang	Verdict	Time	Memory
A - Linear Keyboard	GNU C++17	**Accepted**	0 ms	0 KB

```
#include<bits/stdc++.h>
using namespace std;
int t;
char a[30];
char s[55];
int k[30];
void solve()
{
    scanf("%s", a);
    scanf("%s", s);
    int sum = 0;
    memset(k, 0, sizeof(k));
    for (int i = 0; i < strlen(a); i++) {
        k[a[i] - 'a']=i;
    }
    for (int i = 1; i < strlen(s); i++) {
        sum+=abs(k[s[i]-'a'] - k[s[i - 1]-'a']);
    }
    printf("%d\n", sum);
}

int main()
{
    scanf("%d", &t);
```

```
    while (t--) {
        solve();
    }
    return 0;
}
```

B. Odd Grasshopper

题意

给定初始位置 x_0 和运动次数 n，运动的规则是如果身处的坐标为偶数，运动方向为向左，不然运动方向为向右；而运动的距离，则为其对应的运动次数，即第一次运动一个单位.....第 n 次运动 n 个单位。求出最终位置。

共 t 次测试，每次给出一组 x_0 和 n，数据范围如下：

$$t(1 \le t \le 10^4) \cdots\cdots x_0(-10^{14} \le x_0 \le 10^{14}) \cdots\cdots n(0 \le n \le 10^{14})$$

思路

可以发现，x_0 和 n 的数据范围都很大，超过了int的范围，需要注意使用long long定义变量。

另外，如此大的数据也可以提醒我们，题目中的操作应该是有周期性规律的。

于是可以发现，$T = 4$，即每当四次操作后，会回到原处。

所以将 n 取余4后，分类讨论即可。

总结

当见到操作次数或数据极大时，就应该马上意识到极有可能存在周期性或有一定规律。

应当注意变量的范围，思考运算过程中会不会超int。

AC代码

Problem	Lang	Verdict	Time	Memory
B - Odd Grasshopper	GNU C++17	**Accepted**	15 ms	0 KB

```
#include<bits/stdc++.h>
using namespace std;
int t;
long long n;
long long x;

void solve()
{
    scanf("%lld%lld", &x,&n);
    long long left = n % 4;
    if (x % 2) {
        if (left == 3)x -= n+1;
        if (left == 2)x -= 1;
        if (left == 1)x += n;
    }
```

```
    else {
        if (left == 3)x += n+1;
        if (left == 2)x += 1;
        if (left == 1)x -= n;
    }
    printf("%lld\n", x);
}

int main()
{
    scanf("%d", &t);
    while (t--) {
        solve();
    }
    return 0;
}
```

C. Minimum Extraction

题意

给出一个数组，可对其进行如下操作;

选择数组中最小的元素，使其余元素分别减这个元素，随后删除这个元素（若有不止一个相同的最小元素，仅删除一个）。问经过任意数量的操作后，每次操作后数组中最小的元素出现过的最大值是多少。

t 组测试，数组中有 n 个元素，数组元素用 a_i 表示，数据范围如下：

$$t(1 \le t \le 10^4) \cdots\cdots n(1 \le n \le 2 \cdot 10^5) \cdots\cdots a_i(-10^9 \le a_i \le 10^9)$$

思路

可以将问题转化为求：将数组从小到大排序后，相邻元素差值的最大值为多少。

总结

做题时最难的是看到问题的本质，需要提高转化问题的能力。

AC代码

Problem	Lang	Verdict	Time	Memory
C - Minimum Extraction	GNU C++17	**Accepted**	78 ms	800 KB

```
#include<bits/stdc++.h>
using namespace std;
int t;
int n;
int a[200010];

void solve()
{
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
```

```
    {
        scanf("%d", &a[i]);
    }
    sort(a, a + n);
    int Max = a[0];
    for (int i = 0; i < n-1; i++) {
        Max = max(a[i + 1] - a[i],Max);
    }
    printf("%d\n", Max);
}

int main()
{
    scanf("%d", &t);
    while (t--) {
        solve();
    }
    return 0;
}
```

D. Blue-Red Permutation

题意

给出一个数组，包含 n 个元素，每个元素有两个属性：值和颜色，颜色分为蓝色或红色。

可有如下操作：

（1）：选择任意数量的蓝色元素，使它们的值均减一。

（2）：选择任意数量的红色元素，使它们的值均加一。

问能不能使数组最终变成 1 到 n 的一个排列？

数据范围如下：

测试组数 $t\ (1 \leq t \leq 10^4)$，元素个数 $n(1 \leq n \leq 2 \cdot 10^5)$，元素 $a_i(-10^9 \leq a_i \leq 10^9)$

思路

也就是说，蓝色元素可以变成比它小的任意值，红色元素可以变成比它大的任意值。

那么可以很容易地发现，蓝色元素的最大值需要大于等于蓝色元素的个数，同理红色元素的最小值也需要小于等于n 减去红色元素的个数。

可以进一步发现，每个元素都需要符合类似的条件，不然出现类似于蓝色1、蓝色1，显然是不行的，第二个蓝色1需要大于等于2。

那么这里的个数是什么？就是将蓝色的元素从小到大排序后，该元素所处的位置，第 n 个元素要大于等于 n；同理，红色元素也是类似的条件，相反即可。

所以将红蓝元素分别保存在两个数组里，排序后再遍历判断即可，全部符合输出yes，有一个不符合就可以结束遍历输出no了。

总结

注意对数据的预处理操作，有时是解决问题的关键。

AC代码

Problem	Lang	Verdict	Time	Memory
D - Blue-Red Permutation	GNU C++17	**Accepted**	78 ms	2600 KB

```cpp
#include<bits/stdc++.h>
using namespace std;
int t;
int n;
int a[200010];
int a1[200010];
int a2[200010];
char s[200010];

void solve()
{
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
    scanf("%s", s);//储存类似于“BRBR”的字符串
    int k1 = 0, k2 = 0;
    for (int i = 0; i < n; i++) {
        if (s[i] == 'B') {
            a1[k1] = a[i];
            k1++;
        }
        else {
            a2[k2] = a[i];
            k2++;
        }
    }
    sort(a1, a1 + k1);//默认为从小到大排序
    sort(a2, a2 + k2,greater<int>());//greater才是从大到小排序

    //测试程序
    /*for (int i = 0; i < k1; i++)
        printf("%d ", a1[i]);
    printf("\n");
    for (int i = 0; i < k2; i++)
        printf("%d ", a2[i]);*/

    int flag = 1;
    for (int i = 0; i < k1; i++) {
        if (a1[i] < i+1) {
            flag = 0; break;
        }
    }
    for (int i = 0; i < k2; i++) {
        if (flag == 0)break;
        if (a2[i] > n-i) {
            flag = 0; break;
        }
```

```
    }
    if (flag)printf("YES\n");
    else printf("NO\n");
}

int main()
{
    scanf("%d", &t);
    while (t--) {
        solve();
    }
    return 0;
}
```

---

by syj

tip：希望能有一次a了E