1. Ciallo (∠・ω<)⌒☆

输出 Ciallo 即可

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#define ll long long
#define maxn 200005
using namespace std;

void solve(){
	cout<<"Ciallo"<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a=1;
	//cin>>a;
	while(a--){
		solve();
	}
	return 0;
}
```

2: 我是谁?

发现输出一共 43 行可先写如下代码，再输入要输出的内容:

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;

void solve() {
	string s="";
    for(int i=0;i<43;i++){
	string t;
	getline(cin,t);
	s+="cout<<"+t+"<<endl\n";
	}
	cout<<s<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	//cin >> a;
	while (a--) {
		solve();
	}
	return 0;
}
```

再复制输出的内容粘贴到 solve 函数内即可。

3: 不要 0!

如题模拟即可，全部输出一样的符合要求的数也可

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
void solve() {
	int n;
	cin >> n;
	for (int i = 0; i < n; i++) {
		cin >> sz[i];
	}
	for (int i = 0; i < n; i++) {
		cout << 1 << " ";
	}
	cout << endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	//cin >> a;
	while (a--) {
		solve();
	}
	return 0;
}
```

4: 旅行者的甜甜花酿鸡

最优解一定是 X*N 或 Y*N/3+X*N%3 比较两者大小，输出小的那个即可

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
void solve() {
	int x,y,z;
	cin>>x>>y>>z;
	int u=x*z;
	int o=z/3*y+z%3*x;
	cout<<min(u,o)<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	//cin >> a;
	while (a--) {
		solve();
	}
	return 0;
}
```

5: 小祥的计算器

模拟题，判断是否出现连续两个 0，并更新下标即可

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#define ll long long
#define maxn 2000005
using namespace std;
void solve(){
	string a;
	cin>>a;
	int i=0,ans=0;
	while(i<a.length()){
		if(i<a.length() && a[i]=='0' && a[i+1]=='0'){
			i++;
		}
		ans++;
		i++;
	}
	cout<<ans<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a=1;
	//cin>>a;
	while(a--){
		solve();
	}
	return 0;
}
```

6: 欢迎回家

对数组排序（sort()函数即可），可以证明如果相邻两个数的差的绝对值大于等于 2 便无法满足条件，输出”NO”，反之“YES”

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
void solve() {
	int n;
	cin>>n;
	for(int i=0;i<n;i++){
		cin>>sz[i];
	}
	sort(sz,sz+n);
	for(int i=1;i<n;i++){
		if(sz[i]-sz[i-1]>1){
			cout<<"NO"<<endl;
			return;
		}
	}
	cout<<"YES"<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	cin >> a;
	while (a--) {
		solve();
	}
	return 0;
```

```
}
```

7.源石虫的自我修养

观察到如果同一列上全是 1 源石虫便无法到达终点。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
void solve() {
	int n;
	cin>>n;
	string s,t;
	cin>>s>>t;
	for(int i=0;i<n-1;i++){
		if(s[i]==t[i] && s[i]=='1'){
			cout<<"NO"<<endl;
			return;
		}
	}
	cout<<"YES"<<endl;
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	cin >> a;
	while (a--) {
		solve();
```

```
    }
    return 0;
}
```

8: DDDG

模拟题，从左到右依次遍历每一列，找下一个该出现的字母。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
string s[200005];
void solve() {
    int n,m;
    cin>>n>>m;
    for(int i=0;i<n;i++){
        cin>>s[i];
    }
    string t="DDDG";
    int xb=0;
    for(int i=0;i<m;i++){
        for(int j=0;j<n;j++){
            if(s[j][i]==t[xb]){
                xb++;
                break;
            }
        }
        if(xb>=4){
```

```
                cout<<"YES"<<endl;
                return;
            }
        }
        cout<<"NO"<<endl;
}
int main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        cin >> a;
        while (a--) {
            solve();
        }
        return 0;
}
```

9: Monster 的电脑

稍微麻烦一点的模拟题，找题意模拟即可。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int x[200005];
int y[200005];
void solve() {
        int n,p1,p2,t;
        cin>>n>>p1>>p2>>t;
        for(int i=0;i<n;i++){
```

```
            cin>>x[i]>>y[i];
        }
        int ans=0;
        for(int i=0;i<n;i++){
            if(i!=0){
                ans+=(x[i]-y[i-1])*p2;
            }
            if(i!=n-1){
                y[i]+=min(t,x[i+1]-y[i]);
            }
            ans+=(y[i]-x[i])*p1;

        }
        cout<<ans<<endl;

}
int main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        cin >> a;
        while (a--) {
            solve();
        }
        return 0;
}
```

10: 怎么绘世呢

可以发现要么是奇数位上的数是红色，要么是偶数位上的数是红色。注意到 d 要么是奇数位上所有数的最大公因数要么是偶数位上所有数的最大公因数。分别求得 d1 和 d2 并分别判断能否满足条件

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
```

```
#include<string.h>
#include<queue>
#define ll long long
#define int ll
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
int gcd(int a,int b){
	return b==0?a:gcd(b,a%b);
}
void solve() {
	int n;
	cin>>n;
	int o=0,j=0;
	for(int i=0;i<n;i++){
		cin>>sz[i];
		if(i%2==0){
			o=gcd(o,sz[i]);
		}
		else{
			j=gcd(j,sz[i]);
		}
	}
	int flag1=1,flag2=1;
	for(int i=0;i<n;i++){
		if(i%2==0){
			if(sz[i]%j==0){
				flag1=0;
			}
		}
		else{
			if(sz[i]%o==0){
				flag2=0;
			}
		}
	}
	if(flag1==0 && flag2==0){
		cout<<0<<endl;
	}
	else if(flag2==0){
		cout<<j<<endl;
	}
	else{
```

```
            cout<<o<<endl;
        }
}
signed main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        cin >> a;
        while (a--) {
            solve();
        }
        return 0;
}
```

11: 使徒，袭来！

可以造一个长度为 N 的数组 a，第 i 个数就代表第 i 座城墙有几个炮塔守护，取最小值即可。每次输入 L 和 R 便对数组 L 到 R 的数+1 会导致超时，我们可以使用差分的思想，即在 a[L]++,在 a[R+1]--，最后用前缀和。时间复杂度 O(n)。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[1000005];
int sum[1000005];
void solve() {
        int n, m;
        cin >> n >> m;
        for (int i = 0; i < m; i++) {
            int l, r;
```

```
            cin >> l >> r;
            sz[l]++;
            sz[r+1]--;
        }
        int mn = 1e9;
        for (int i = 1; i <= n; i++) {
            sum[i] = sum[i - 1] + sz[i];
            mn = min(mn, sum[i]);
        }
        cout << mn << endl;
}
int main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        //cin >> a;
        while (a--) {

            solve();
        }
        return 0;
}
```

12: 我现在就要玩《我的世界》

考虑运用并查集，把朋友放到一个集合里面，把每一组朋友的根节点设成节点权值最小的那个。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define int ll
#define maxn 2000005
```

```
#define endl '\n'
using namespace std;
int ff[200005];
int sz[200005];
int find(int a){
	if(ff[a]==a){
		return ff[a];
	}
	return ff[a]=find(ff[a]);
}
int flag[200005];
void solve() {
	int n,m;
	cin>>n>>m;
	for(int i=1;i<=n;i++){
		cin>>sz[i];
		ff[i]=i;
	}
	while(m--){
		int u,v;
		cin>>u>>v;
		if(sz[find(u)]<sz[find(v)]){
			ff[find(v)]=find(u);
		}
		else{
			ff[find(u)]=find(v);
		}
	}
	int ans=0;
	for(int i=1;i<=n;i++){
		if(!flag[find(i)]){
			ans+=sz[find(i)];
			flag[find(i)]=1;
		}
	}
	cout<<ans<<endl;
}
signed main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
	//cin >> a;
	while (a--) {
		solve();
	}
```

```
    return 0;
}
```

13: 奶龙的谎言

运用 dp，dp[i][1]代表左边第 i 个奶龙是诚实的奶龙的配置数，dp[i][0]代表左边第 i 个奶龙是调皮的奶龙的配置数,由于两个调皮的奶龙不能站在一起，dp[i][0]=dp[i-1][1]。再考虑第 i 个是诚实的奶龙，如果前一个奶龙是诚实的，那么应该 arr[i]==arr[i-1]，所以 dp[i][1]=dp[i][1]+dp[i-1][1]，如果前面的奶龙是调皮的，那么应该 arr[i]==arr[i-2]+1，所以 dp[i][1]=dp[i][1]+dp[i-2][0]。初始时 dp[1][0]=1，如果 arr[1]==0，那么 dp[1][1]=1，否则为 0。最后只需要 dp[n][1]+dp[n][0]对 998244353 取模即可。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
const int mod=998244353;
int dp[200005][2];
int sz[200005];
void solve() {
    int n;
    cin>>n;
    for(int i=1;i<=n;i++){
        dp[i][1]=0;
        dp[i][0]=0;
        cin>>sz[i];
    }
    dp[1][0]=1;
    dp[1][1]=(sz[1]==0);
    for(int i=2;i<=n;i++){
```

```
            dp[i][0]=dp[i-1][1];
            if(sz[i]==sz[i-1]){
                dp[i][1]=(dp[i-1][1]+dp[i][1])%mod;
            }
            if(sz[i]==sz[i-2]+1){
                dp[i][1]=(dp[i][1]+dp[i-1][0])%mod;
            }
        }
        cout<<(dp[n][0]+dp[n][1])%mod<<endl;
}
int main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        cin >> a;
        while (a--) {
            solve();
        }
        return 0;
}
```

14: 琪亚娜，出击！

先考虑给出的数组是不是一个排列(即一个长度为 n 的数组，1 到 n 只能出现一次)。如果不是，那么，如果是物体 A，这个点到任何点的距离必定为零，如果是物体 B 的话则不是 0。如果是一个排列，那么我们找到 1 和 n 出现的下标，并正反询问这两个下标，如果两次询问结果不同，那么一定是物体 A，如果相同，则再看询问得到的数，如果小于 n-1 那一定是物体 A，反之则是物体 A。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
```

```
#define endl '\n'
using namespace std;
int sz[200005];
int ask(int a,int b){
	cout<<"? "<<a<<" "<<b<<endl;
	cout.flush();
	int y;
	cin>>y;
	return y;
}
void solve() {
	int n;
	cin>>n;
	int flag=0;
	map<int,int>mp;
	int mx=0,mn=0;
	for(int i=1;i<=n;i++){
		cin>>sz[i];
		mp[sz[i]]++;
		if(sz[i]==1){
			mn=i;
		}
		if(sz[i]==n){
			mx=i;
		}
	}
	int ks=0,js=0;
	for(int i=1;i<=n;i++){
		if(mp[i]==0){
			ks=i;
			flag=1;
		}
		else{
			js=i;
		}
	}
	if(flag){
		int o=ask(ks,js);
		if(o==0){
			cout<<"! A"<<endl;
			cout.flush();
			return;
		}
		else{
```

```
                cout<<"! B"<<endl;
                cout.flush();
                return;
            }
        }
        else{
            int p=ask(mx,mn);
            int q=ask(mn,mx);
            if(p!=q){
                cout<<"! A"<<endl;
                cout.flush();
                return;
            }
            else{
                if(p<n-1){
                    cout<<"! A"<<endl;
                    cout.flush();
                    return;
                }
                else{
                    cout<<"! B"<<endl;
                    cout.flush();
                    return;
                }
            }
        }

}
int main() {
    ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
    int a = 1;
    cin >> a;
    while (a--) {
        solve();
    }
    return 0;
}
```

15: 爱莉希雅的游戏

一道经典的博弈论，尼姆游戏。

直接给结论：

先手必胜当且仅当 $a_1 \oplus a_2 \oplus \cdots \oplus a_n \neq 0$

具体证明可以看这篇

https://www.zhihu.com/question/26934313/answer/32768976853?share_code=VIa5AFZk00yh&utm_psn=1961762421128558365

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
int sz[200005];
void solve() {
	int n;
	cin>>n;
	int sum=0;
	for(int i=0;i<n;i++){
		int y;
		cin>>y;
		sum^=y;
	}
	if(sum==0){
		cout<<"Elysia"<<endl;
	}
	else{
		cout<<"Mei"<<endl;
	}
}
int main() {
	ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
	int a = 1;
```

```
    cin >> a;
    while (a--) {
        solve();
    }
    return 0;
}
```

16: 你本可以中彩票的……

考虑每一次抽奖对答案的贡献，记 s 为分数总和，初始时 $s = \sum_{i=1}^{m} f_i$。

设每一对球选中的概率 $p = \frac{2}{n(n-1)}$

每一轮选择时，选择到的分数的期望为 p*s，分数总和增加的期望为 p*m。

```
#include<iostream>
#include<map>
#include<algorithm>
#include<vector>
#include<stack>
#include<set>
#include<math.h>
#include<string>
#include<deque>
#include<iomanip>
#include<string.h>
#include<queue>
#define ll long long
#define maxn 2000005
#define endl '\n'
using namespace std;
const ll mod=1e9+7;
ll ksm(ll a,ll b){
    ll temp=a;
    ll ans=1;
    while(b){
        if(b%2==1){
```

```
                ans=(ans*temp)%mod;
            }
            temp=(temp*temp)%mod;
            b=b>>1;
        }
        return ans%mod;
}
void solve() {
        ll n,m,k;
        cin>>n>>m>>k;
        ll sum=0;
        ll p=m;
        while(m--){
            ll u,v,f;
            cin>>u>>v>>f;
            sum+=f;
            sum=sum%mod;
        }
        ll ans=0;
        ll pp=ksm(n*(n-1)%mod,mod-2)%mod;
        while(k--){
            ans=(ans+2*sum*pp)%mod;
            sum=(sum+2*p*pp)%mod;
        }
        cout<<ans<<endl;
}
int main() {
        ios::sync_with_stdio(0); cin.tie(0), cout.tie(0);
        int a = 1;
        cin >> a;
        while (a--) {
            solve();
        }
        return 0;
}
```

17: 题目越短越简单？

防 ak 题，这题 cf3000 分，我也不会

原题 https://codeforces.com/contest/2081/problem/G1

题解 https://codeforces.com/blog/entry/140702