问题 A：天宫课堂

题目大意：给出一段数字，每个数字出现一次代表对应数字的人多献出一朵花，求最后献花数等于对应数字的人的个数。

思路：基本模拟题，只需要开一个 0—100 的桶记录每个数字对应的献花数，最后再对桶进行遍历即可，需要注意的是虽然 a[i]有 10^9 但是 N 只有 100，所以能使数字和献花数对应起来的 i 的最大值是 100，所以超过 100 的部分可以直接忽略。
代码：

```
#include <iostream>
#include <cstring>
#include <algorithm>
using namespace std;
const int N=110;
int t[N];//桶
int main(){
	int n;
	cin>>n;
	for(int i=0;i<n;i++){
		int x;
		cin>>x;
		if(x<=100) t[x]++;
	}
	for(int i=0;i<=100;i++){//对 0 到 100 遍历即可
		if(t[i]==i) cout<<i<<" ";
	}
	return 0;
}
```

B. 二进制

题目大意：题目大意，给一个 t，代表数据组数，然后每组数据给一个字符串，代表二进制，然后求 n+1 和 n+3 的二进制，如果大于等于 32 位，就原样输出，如果不足，就添加前缀 0。

思路：模拟题，用 string n 来代表我的这个数字，然后从前往后遍历字符串，每一次遍历 n<<=1，如果字符串的当前位置是 1 的话，那么 n++，然后把 n 的每一位变成二进制。
写法 1：

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

int n;
string s;
char a='0';
void jinwei(){
    s[s.size()-1]++;//最低位+1
    for(int i=s.size()-1;i>0;i--){
        //从低位向高位一位一位处理
        if(s[i]=='2'){//是 2 就要进位
            s[i-1]++;//进位
            s[i]='0';//进位后自己清零
        }
        if(s[0]=='2'){//最高位需要进位
            s[0]='0';
            a++;
        }
    }
}

int main(){
    cin>>n;
    for(int i=1;i<=n;i++){
        cin>>s;
        jinwei();//+1
        if(a=='1')//最高位是 1，超过 32 位
            cout<<a;
        cout<<s<<endl;
        jinwei();
        jinwei();
        if(a=='1')
            cout<<a;
        cout<<s<<endl;
        a='0';

    }
}
```

写法 2：

```
#pragma GCC optimize(2)
#include<bits/stdc++.h>
#define _int128 lll
using namespace std;
typedef long long ll;
typedef pair<int, int> PII;
void quickcin() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
}
const int N = 100;
int t;
string n;
void print(stack<int> q) {
    if (q.size() >= 32) {
        while (!q.empty()) {
            cout << q.top();
            q.pop();
        }
    }
    else {
        for (int i = 1; i <= 32 - q.size(); i++) cout << 0;
        while (!q.empty()) {
            cout << q.top();
            q.pop();
        }
    }
    cout << "\n";

}
int main() {
    quickcin();
    cin >> t ;
    while (t--) {
        cin >> n;
        ll num=0;
        stack<int> q;
        for (int i = 0; i < n.size(); i++) {
            num <<= 1;
            if (n[i] == '1') num ++;
        }
```

```
        ll num1 = num + 1, num2 = num + 3;
        while (num1) {
            q.push(num1 & 1);
            num1 >>= 1;
        }
        print(q);
        while (!q.empty()) q.pop();
        while (num2) {
            q.push(num2 & 1);
            num2 >>= 1;
        }
        print(q);
    }
    return 0;
}
```

问题 C: 三角形

问题 D: 礼物

题目大意：有很多个小朋友，每个小朋友都有一个容量 v[i]，每个小朋友最多携带炸弹的数量为 v[i]/k;现在有多少种不同的送炸弹人的名单，满足这些人携带的炸弹总数大于等于 p。
思路：
快速幂+01 背包，先不管情况，把所有可能的情况的方案数算出（快速幂运），然后 01 背包 dp 计算携带炸弹数量小于 p 可能的方案数，最后两者相减即可（要取模）
代码：

```
#pragma GCC optimize(2)
#include<bits/stdc++.h>
#define _int128 lll
using namespace std;
typedef long long ll;
typedef pair<int, int> PII;
void quickcin() {
	ios::sync_with_stdio(false);
	cin.tie(0);
	cout.tie(0);
}
const int N = 3*1e3 + 10;
int v[N],zha[N];
int n, k, p;
ll sumn,mod= 998244353;
ll f[N * N];
int main() {
	quickcin();
	cin >> n ;
	for (int i = 1; i <= n; i++) cin >> v[i];
	cin >> k >> p;
	for (int i = 1; i <= n; i++) zha[i] = min(v[i] / k, p);//计算这个人的炸弹数量
	f[0] = 1;
	int sumn = n;
	for (int i = 1; i <= n; i++) {
```

```
            if (zha[i] == 0) sumn--;//如果这个人没有炸弹，那么 sumn--;
            else {
                for (int j = p; j >= zha[i]; j--) f[j] = (f[j] + f[j - zha[i]]) % mod;//求出 j 个炸弹数的情况
总数
            }
        }
        ll res = 1;
        for (int i = 1; i <= sumn; i++) res = res * 2 % mod;//算出所有的组合数量
        for (int i = 0; i <= p - 1; i++)res = ((res - f[i]) % mod + mod) % mod;//减去炸弹数不足的情况
        cout << res;
        return 0;}
```