

# 一维数组

## 数组基础

读入元素，逐个判断是否优秀：数组的解法：

```
#include <iostream>//1-1423-1 javacn
using namespace std;
int main() {
    int a[100] = {0}; // 定义长度为 100 的数组，最多存放 100 个人的成绩
    int n, i, c = 0; // n: 代表了数组中实际存储元素的个数
    cin >> n; // 读入 n 个成绩的值
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
        if (a[i] >= 90)
        {
            c++;
        }
    }
    cout << c;
    return 0;
}
```

#include <iostream>//1-1423-2 javacn 非数组的解法：

```
using namespace std;
int main() {
    int n, i, c = 0, x;
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> x; // 读入 n 个成绩的值
        if (x >= 90) { c++; }
    }
    cout << c;
    return 0;
}
```

由于第一个数和最后一个数不可能是支撑数，那么我们循环的下标范围  $i = 1 \sim n-2$ 。

对于下标为  $i$  的元素  $a[i]$  而言，左侧的数为  $a[i-1]$ ，右侧的数为  $a[i+1]$ 。

```
#include<bits/stdc++.h> //2-1153
using namespace std;

/*
思路：
有 n 个数，下标就是  $0 \sim n-1$  找支撑数（比左右都大的数）
由于不可能是第 1 个，也不可能是最后一个
那么可能的支撑数的下标： $1 \sim n-2$ 
*/
int main()
{
    int a[100]; // 存储每个数
    int i, n;
    cin >> n; // 读入实际有几个数
    // 读入这 n 个数，这 n 个数的下标  $0 \sim n-1$ 
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }

    // 输出支撑数：支撑数的下标是从  $1 \sim n-2$ 
    for (i = 1; i <= n - 2; i++)
    {
        // 当前数 a[i]，左边数 a[i-1]，右边数 a[i+1]
        if (a[i] > a[i-1] && a[i] > a[i+1])
        {
            cout << a[i] << endl;
        }
    }
    return 0;
}
```

思路:

- 1、求出身高的和;
- 2、求平均身高;
- 3、求哪些人的身高超过平均身高;

```
#include<bits/stdc++.h>//3-1155
using namespace std;
// 定义数组长度要比题目需要的多定义 10 个
// 将变量定义到 main 的外面
// 初始值不是随机, 整数初始值都是 0
int a[20];
int n, i;
double s = 0;
double v;// 存放平均值
int main()
{
    cin>>n;
    for (i = 0; i < n; i++)
    {
        cin>>a[i];

        s = s + a[i]; // 求和
    }

    v = s / n; // 平均
    cout<<"AVE="<<fixed<<setprecision(1)<<v<<endl;

    for (i = 0; i < n; i++) // 求哪些人的身高超过平均身高
    {
        if(a[i] > v)
        {
            cout<<i+1<<": "<<a[i]<<" ";
        }
    }
    return 0;
}
```

将读入的序号的平方除以 7 的余数都是 1 这类数去除掉（不输出），输出剩余的元素

`#include <iostream>`//4-1156-1 数组的解法:

```
using namespace std;
```

```
int main() {
```

```
    /* 将数组中 序号的平方除以 7 的余数都是 1 这类数去除掉，输出剩余的元素
```

```
    思路：循环数组的每个元素，如果是满足  $a[i] * a[i] \% 7 \neq 1$ 
```

```
    说明是好的基因，输出 */
```

```
    int a[210], n, i;
```

```
    cin >> n; // 读入数组实际元素的个数
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        cin >> a[i];
```

```
        // 判断如果不是异形基因，则输出
```

```
        if (a[i] * a[i] % 7 != 1)
```

```
        {
```

```
            cout << a[i] << " "; // 每输出一个数空一格
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

`#include <iostream>`//4-1156-2 非数组的解法:

```
using namespace std;
```

```
int main() {
```

```
    int x, n, i;
```

```
    cin >> n; // 读入数组实际元素的个数
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        cin >> x;
```

```
        if (x * x % 7 != 1)
```

```
        { // 判断如果不是异形基因，则输出
```

```
            cout << x << " "; // 每输出一个数空一格
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

分两个循环，第一个循环输出所有的奇数，第二个循环输出所有的偶数。

```
#include<iostream>//5-1158
using namespace std;
int main()
{
    int a[40], n;
    cin>>n;
    for (int i=0; i<n; i++)
    {
        cin>>a[i];
    }

    for (int i=0; i<n; i++)
    {
        // 输出奇数
        if(a[i]%2==1)    cout<<a[i]<<" ";
    }
    cout<<endl;

    for (int i=0; i<n; i++)
    {
        // 输出偶数
        if(a[i]%2==0)    cout<<a[i]<<" ";
    }
    return 0;
}
```

```

#include <bits/stdc++.h>//6-1160 hyb
using namespace std;
int main()
{
    int n, a[110], i;
    double s = 0;
    cin>>n;
    for (i=1; i<=n; i++)
    {
        cin>>a[i];
        s = s + a[i];
    }
    if (s>100)
    {
        s = 100 + (s - 100)*0.9;
    }
    cout<<fixed<<setprecision(2)<<s;
    return 0;
}

```

```

#include<bits/stdc++.h>//7-1169 非官方
using namespace std;
int a[20], i;
int main()
{
    for (i=1; i<=10; i++)
    {
        cin>>a[i];
    }
    sort(a+1, a+10+1);
    for (i=10; i>=1; i--) cout<<a[i]<<" ";
    return 0;
}

```

`#include <bits/stdc++.h> //8-1174-1 javacn` 读入 n 个数，将奇数和偶数分别求和。

```
using namespace std;
int main() {
    int n, a[5010], i, s1 = 0, s2 = 0;
    cin >> n; // 读入 n 个数，将这 n 个数中的奇数和偶数分别求和
    for (i = 0; i < n; i++) {        cin >> a[i];    }
    for (i = 0; i < n; i++)
    {
        if (a[i] % 2 == 1)
        {
            s1 = s1 + a[i];
        }
        else
        {
            s2 = s2 + a[i];
        }
    }
    cout << s1 << endl << s2;
    return 0;
}
```

```
#include <bits/stdc++.h> //8-1174-2
using namespace std;
int n, suma, sumb, i;
int main() {
    cin >> n;
    int x;
    for (i = 1; i <= n; i++)
    {
        cin >> x;
        if (x % 2 == 0) suma += x;
        else sumb += x;
    }
    cout << sumb << endl << suma << endl;
    return 0;
}
```

```

#include<iostream>//9-1176
#include<cmath>
using namespace std;
int a[1010];
bool sushu(int n)//判断素数函数
{
    bool f = true;
    for(int i = 2; i <= sqrt(n); i++)
    {
        if(n % i == 0)
        {
            f = false;
            break;
        }
    }
    if(n <= 1) f = false; //判断特殊情况
    return f;
}

```

```

int main()
{
    int n;
    while(cin>>n && n!=0)
    {
        int k=0;

        for(int i=2; i<=n; i++)
        {
            if(sushu(i)==true) k++;
        }
        cout<<k<<endl;
    }
    return 0;
}

```

另外一种写法:

```

int main()
{
    while(1)
    {
        int n;
        int k=0;
        cin>>n;
        if(n==0) return 0;
        for(int i=1; i<=n; i++)
        {
            if(sushu(i) k++;
        }
        cout<<k<<endl;
    }
    return 0;
}

```

```
#include<bits/stdc++.h>//10-1218 475214719
using namespace std;
int main()
{
    int a[101], n, min, cnt=0;
    double sum=0;
    cin>>n;
    for (int i=1;i<=n;i++)
        cin>>a[i];
    min=a[1];
    for (int i=2;i<=n;i++)
        if(min>a[i]) min=a[i]; // 找出最小苹果

    for (int i=1;i<=n;i++)
        if(min==a[i]) cnt++; // 最小苹果个数

    for (int i=1;i<=n;i++)
        if(a[i]!=min)
            sum=sum+a[i]; // 不是最小苹果总重量

    cout<<fixed<<setprecision(1)<<sum*1.0/(n-cnt);
    return 0;
}
```

```
#include<iostream>//11-1231 zhengjia
#include<iomanip>
using namespace std;
int a[110], c1, c2;
double ans, sum;
int main()
{
    int n;
    cin>>n;
    for (int i=1; i<=n; i++)
    {
        cin>>a[i];
        sum+=a[i];
    }
    ans=sum/n;
    for (int i=1; i<=n; i++)
    {
        if (a[i]>=ans)
        {
            c1++;
        }
        else
        {
            c2++;
        }
    }
    cout<<fixed<<setprecision(1)<<ans<<" "<<c1<<" "<<c2;
    return 0;
}
```

```

#include<iostream>//12-1316-1 zhengjia
#include<iomanip>
using namespace std;
int m, n;
double sum;
double a[30];
int main()
{
    cin>>m>>n;
    for (int i=1;i<=n;i++)
    {
        cin>>a[i];
        sum+=a[i];
    }
    sum=sum/n*m;
    cout<<fixed<<setprecision(1)<<sum;
}

```

```

#include<iostream>//12-1316-2 wangpeng
using namespace std;
int main()
{
    int m , n;
    double sum = 0 ; // sum 负责求 n 个数的和
    cin >> m >> n ;
    float a[n+5];
    for ( int i = 0 ; i < n ; i++)
    {
        // 边读入边求和
        cin >> a[i];
        sum += a[i] ;
    }
    double agv = sum *1.0 / n ; // 求出 n 个数的平均值
    double z = m * agv ; // 推理出总数
    printf("%. 1f", z); // 保留 1 位小数
    return 0;
}

```

拿到数字 x 的概率 = 数字 x 的个数 / 总数字的个数，第一步：求出数字 x 在数组中出现的次数，第二步：概率 = x 出现的次数 / 总元素个数

```
#include <bits/stdc++.h>//13-1354-1 javacn
using namespace std;
int main() {
    int a[1000]; // 存储读入的数字
    int i, n, c = 0, x; // c 统计 x 在数组中出现的次数
    cin >> n; // 读入的数字的数量
    for (i = 0; i < n; i++)    cin >> a[i];    // 读入 n 个数
    cin >> x;    // 读入 x, 表示要找的数
    for (i = 0; i < n; i++)
    {
        // 求出 x 在数组中出现了几次
        if (a[i] == x)    c++;
    }
    cout << fixed << setprecision(2) << c * 1.0 / n;    // 输出 x 出现的概率
    return 0;
}
```

```
#include <iostream> //13-1354-2    zhengjia    // 下标写法
#include <iomanip>
using namespace std;
int n, x, x1;
double sum;
int a[110];
int main()
{
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> x;
        a[x]++;
    }
    cin >> x1;
    sum = a[x1] * 1.0 / n;
    cout << fixed << setprecision(2) << sum;
}
```

求数组元素的和，根据总和求方差，根据方差判断哪组零件更标准。

```
#include<bits/stdc++.h>//14-1357-1    哪个厂家的零件更标准?    javacn
```

```
using namespace std;
```

```
double fc(int a[], int s, int n) // 求数组的方差
```

```
{  
    double p, r = 0; //p: 平均  r: 方差  
    p=s/n;  
    for (int i=1; i<=n; i++)  
    {  
        r+=(a[i]-p)*(a[i]-p);  
    }  
    return r;  
}
```

```
int n, a[110], b[110], s1, s2, ch1, ch2;
```

```
int main()
```

```
{  
    cin>>n;  
    // 读入的同时求和  
    for (int i=1; i<=n; i++)  
    {  
        cin>>a[i];  
        s1=s1+a[i];  
    }  
    for (int i=1; i<=n; i++)  
    {  
        cin>>b[i];  
        s2=s2+b[i];  
    }  
    // 判断方差  
    if (fc(a, s1, n)<fc(b, s2, n))    cout<<"jia";  
    else    cout<<"yi";  
    return 0;  
}
```

```
#include<bits/stdc++.h>//14-1357-2 哪个厂家的零件更标准?
```

```
using namespace std;
```

```
int a[100]; int b[100]; int main()
```

```
{
```

```
    int n, s1=0, s2=0;
```

```
    double g=0, h=0;
```

```
    double f, c;
```

```
    scanf("%d", &n);
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

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

```
        s1=s1+a[i];
```

```
    }
```

```
    f=s1*1.0/n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        g=g+(a[i]-f)*(a[i]-f);
```

```
    }
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

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

```
        s2=s2+b[i];
```

```
    }
```

```
    c=s2*1.0/n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        h=h+(b[i]-f)*(b[i]-f);
```

```
    }
```

```
    if (g>h)    {    printf("yi");    }
```

```
    if (g<h)    {    printf("jia");    }
```

```
    return 0;
```

```
}
```

```
#include<iostream>//15-1388
using namespace std;
float a[10];
int main()
{
    int n, i;
    int g=0;
    for (i=1; i<=10; i++)
    {
        scanf("%f ", &a[i]);
    }
    scanf("%d", &n);
    n=n+30;
    for (i=1; i<=10; i++)
    {
        if (n>=a[i])
        {
            g++;
        }
    }
    printf("%d", g);
    return 0;
}
```

```
#include<bits/stdc++.h>//16-1397 完美偶数
```

```
using namespace std;
```

```
bool oushu1(int n)//判断位数是否偶数
```

```
{  
    bool r=true;  
    int c=0;  
    while(n!=0) {    c++;  n=n/10;    }  
    if(c%2!=0)      r=false;  
    return r;  
}
```

```
bool oushu2(int n)//判断每个位数是否偶数
```

```
{  
    bool r=true;  
    int x;  
    while(n!=0)  
    {  
        x=n%10;  
        if(x%2!=0) {    r=false;    return 0;    }  
        n=n/10;  
    }  
    return r;  
}
```

```
int main() {
```

```
    int i,n,a[100];
```

```
    cin>>n;
```

```
    for(i=0;i<n;i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        if(a[i]%2==0&&oushu1(a[i])==true&&oushu2(a[i])==true)
```

```
        {
```

```
            cout<<a[i]<<endl;
```

```
        }  
    }  
    return 0;
```

```
}
```

```

#include<bits/stdc++.h>//17-1401    fide
using namespace std;
int main()
{
    int a[105]={0}, n, cnt = 0;
    cin >> n;
    for (int i = 1; i <= n; i++)
        cin >> a[i];
    for (int i = 2; i <= n-1; i++)
    {
        if (a[i] > a[i-1] && a[i] > a[i+1])
            cnt++;
    }
    cout << cnt << endl;
    return 0;
}

```

```

#include<algorithm>//18-1009    zhengjia
#include<iostream>
#include<cstdlib>
#include<cstdio>
using namespace std;
int a[100];
int main()
{
    int s;
    cin>>s;
    int a[s];
    for (int i=1;i<=s;i++)
        cin>>a[i];
    for (int i=s;i>=1;i--) {
        cout<<a[i]<<" ";
    }
    return 0;
}

```

解法二：短除法拆位判断

`#include <iostream>`//19-1425-1 javacn 推荐学习

`using namespace std;`

`int a[1010], n, x, t, s = 0, c = 0;`

`int main()`

```
{  
    cin>>n;    // 读入  
    for(int i = 0; i < n; i++)  
    {  
        cin>>a[i];  
    }  
    cin>>x;  
  
    // 逐个判断是否含有 x  
    for(int i = 0; i < n; i++)  
    {  
        //a[i] 还要用，不能短除除到 0  
        t = a[i];  
        while(t != 0)  
        {  
            // 含有 x  
            if(t % 10 == x)  
            {  
                s = s + a[i];  
                c++;  
                break;  
            }  
  
            t = t / 10;  
        }  
    }  
  
    cout<<c<<" "<<s;  
    return 0;  
}
```

思路：1、读入每个数；2、循环每个数，判断该数有没有数字 x；

拆出该数的每一位（本题 x 是 1~9 之间，且 a[i] 最多是 4 位数，可以直接拆个十百千位，也可以用短除法拆违，如果 x 是 0~9，或者 a[i] 的位数不确定，用短除法拆违）；

如果有，计数、求和；

解法一：逐个数拆位判断

```
#include<bits/stdc++.h>//19-1425-2 javacn 仅供参考
```

```
using namespace std;
```

```
int a[1010];
```

```
//c: 满足条件的数有几个
```

```
//s: 满足条件的数的和是多少
```

```
int n, i, c = 0, h = 0, x;
```

```
int g, s, b, q;
```

```
int main()
```

```
{
```

```
    cin>>n;// 读入数组中实际元素的个数
```

```
    for (i = 0; i < n; i++) {    cin>>a[i]; }
```

```
    cin>>x;    // 循环判断是否有数字 x
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        //a[i] = 1234, 看着一个数字拆位
```

```
        q = a[i] / 1000;
```

```
        b = a[i] / 100 % 10;
```

```
        s = a[i] / 10 % 10;
```

```
        g = a[i] % 10;
```

```
        if(q == x || b == x || s == x || g == x)
```

```
        {    // 判断是否有数字 x
```

```
            //cout<<a[i]<<endl;
```

```
            c = c + 1;// 计数
```

```
            h = h + a[i];
```

```
        }
```

```
    }
```

```
    cout<<c<<" "<<h;
```

```
    return 0;
```

```
}
```

```

#include <bits/stdc++.h>//20-1426
using namespace std;
int main()
{
    int n, i, c1=0, c2=0, c3=0, c4=0, a[110];
    cin>>n;
    for (int i = 0; i < n; i++)
    {
        cin>>a[i];
        if(a[i]<=18)                {    c1++; }
        else if(a[i]>=19&&a[i]<=35)  {    c2++; }
        else if(a[i]>=36&&a[i]<=60)  {    c3++; }
        else                        {    c4++; }
    }

    cout<<c1<<" " <<fixed<<setprecision(2) <<c1*1.0/n*100<<"%"<<endl;
    cout<<c2<<" " <<fixed<<setprecision(2) <<c2*1.0/n*100<<"%"<<endl;
    cout<<c3<<" " <<fixed<<setprecision(2) <<c3*1.0/n*100<<"%"<<endl;
    cout<<c4<<" " <<fixed<<setprecision(2) <<c4*1.0/n*100<<"%"<<endl;
    return 0;
}

```

思路：基本格式

```
if(有不及格)
```

```
{  
    输出 no;  
}
```

```
}
```

```
else
```

```
{  
    if(优秀科目 >=5) 一等  
    else if(优秀科目有 3 科或 4 科) 二等  
    else if(优秀科目有 2 科) 三等  
    else 输出 no;  
}
```

```
}
```

```
#include<bits/stdc++.h>//21-1429 javacn
```

```
using namespace std;
```

```
int a[25], n, i;
```

```
int c1 = 0, c2 = 0; //c1 统计优秀科目数量 //c2 统计不及格的科目数量
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    for(i = 0; i < n; i++)
```

```
    {
```

```
        cin>>a[i]; // 统计优秀及不及格科目数量
```

```
        if(a[i] >= 90) c1++;
```

```
        else if(a[i] < 60) c2++;
```

```
    }
```

```
    if(c2 != 0) cout<<"no"; // 如果有不及格，则不能参评奖学金
```

```
    else
```

```
    {
```

```
        if(c1 >= 5) cout<<1;
```

```
        else if(c1 == 3 || c1 == 4) cout<<2;
```

```
        else if(c1 == 2) cout<<3;
```

```
        else cout<<"no";
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//22-1450-1 javacn 推荐学习
```

```
using namespace std;
```

```
/*1. 准备一个空数组，用来存放满足条件的数； 默认长度为 0；
```

```
2. 读入 n 个整数，每读入一个整数 t，短除法求 t 各个位的和；
```

```
如果和是 x，说明满足条件，存入数组（数组长度 +1）、求和； */
```

```
int a[10010], x, t, n, s, t2, r = 0;
```

```
int len = 0; // 数组长度
```

```
int main()
```

```
{
```

```
    cin>>x>>n; // 读入 n 个数
```

```
    for (int i = 0; i < n; i++)
```

```
    {
```

```
        cin>>t; // 读入一个数
```

```
        t2 = t;
```

```
        s = 0;
```

```
        while (t != 0)
```

```
        {
```

```
            s = s + t % 10;
```

```
            t = t / 10;
```

```
        }
```

```
        if (s == x)
```

```
        {
```

```
            a[len] = t2;
```

```
            r = r + t2; // 求和
```

```
            len++;
```

```
        }
```

```
    }
```

```
    cout<<r<<" " <<len<<endl;
```

```
    sort(a, a+len); // 排序
```

```
    for (int i = 0; i < len; i++)    cout<<a[i]<<" ";
```

```
    return 0;
```

```
}
```

```

#include<bits/stdc++.h>//22-1450-2
using namespace std;
int a[10005];
int main()
{
    int x, n, i;
    int g, s, b, q, ge=0, sum=0;
    scanf("%d%d", &x, &n);
    for (i=0; i<n; i++) { scanf("%d", &a[i]); }
    for (i=0; i<n; i++)
    {
        g=a[i]%10;
        s=a[i]/10%10;
        b=a[i]/100%10;
        q=a[i]/1000;
        if (g+s+b+q==x)
        {
            sum=sum+a[i];
            ge++;
        }
        else
        {
            a[i]=0;
        }
    }
    printf("%d %d\n", sum, ge);
    sort(a, a+n);
    for (i=0; i<n; i++)
    {
        if (a[i]!=0) { cout<<a[i]<<" "; }
    }
    return 0;
}

```

```
#include<bits/stdc++.h> //23-1582
using namespace std;
/*
假设有 n 个数
前一半的下标:  $0 \sim n/2-1$ 
后一半的下标是:  $n/2 \sim n-1$ 
*/
int n, a[1010];
int main()
{
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> a[i];
        // 按题意修改数组元素的值
        if (i <= n / 2)
        {
            a[i] = a[i] * 2;
        }
        else
        {
            a[i] = a[i] + 1;
        }
    }
    // 输出
    for (int i = 1; i <= n; i++)
    {
        cout << a[i] << " ";
    }
    return 0;
}
```

```
#include<bits/stdc++.h> //24-1738 csdn
```

```
using namespace std;
```

```
int main()
```

```
{  
    int a[1001];  
    int i, n, s1=0, s2=0;  
    cin>>n;  
    for (i=1; i<=n; i++)  
    {  
        cin>>a[i];  
    }  
    for (i=1; i<=n; i++)  
    {  
        if (i%2!=0)  
        {  
            s1=s1+a[i];  
        }  
        else  
        {  
            s2=s2+a[i];  
        }  
    }  
    cout<<"KING"<<" "<<s1<<endl;  
    cout<<"WIN"<<" "<<s2<<endl;  
    if (s2>s1)  
    {  
        cout<<"WIN"<<endl;  
    }  
    else  
    {  
        cout<<"KING"<<endl;  
    }  
}
```

```

#include<bits/stdc++.h>//25-1804
using namespace std;
int n, a[110];
int c1=0, c2=0, c3=0; // 统计直角、锐角、钝角的数量
int main()
{
    cin>>n;
    for (int i = 0; i < n; i++) // 读入每个数，逐个判断是直角、锐角还是钝角
    {
        cin>>a[i];
        if(a[i] == 90)        c1++;
        else if(a[i] < 90)   c2++;
        else                  c3++;
    }
    cout<<c1<<" "<<c2<<" "<<c3;
    return 0;
}

```

```

#include<bits/stdc++.h>//26-1808 marsshu
using namespace std;
int a[110];
int main() {
    int n;
    cin>>n;
    int sum10=0;
    int sum20=0;
    for (int i=0; i<n; i++)
    {
        cin>>a[i];
        if(a[i]==10)    sum10++;
        else            sum20++;
    }
    if (sum10>sum20)    cout<<"10";
    else                cout<<"20";
    return 0;
}

```

```
#include<bits/stdc++.h>//27-1817 zhengjia
```

```
using namespace std;
```

```
int a[20];
```

```
int main()
```

```
{  
    int n, i, p=0;  
    scanf("%d", &n);  
    for (i=0; i<n; i++)  
    {  
        scanf("%d", &a[i]);  
    }  
    for (i=0; i<n/2; i++)  
    {  
        if(a[i]!=a[n-i-1])  
        {  
            p++;  
        }  
    }  
    if(p==0)        printf("YES");  
    else            printf("NO");  
    return 0;  
}
```

```
#include <bits/stdc++.h>//28-1830 fide
```

```
using namespace std; int a[101];
```

```
int main()
```

```
{  
    int n, sum = 0;  
    cin >> n;  
    for (int i = 1; i <= n; i++) {  
        cin >> a[i];  
        sum += a[i] * a[i];  
    }  
    cout << sum ;  
    return 0;  
}
```

# 数组找数

```
#include <bits/stdc++.h> //1-1154-1
using namespace std;
int a[110], n, x;
int main()
{
    cin>>n;    // 读入
    for(int i = 1; i <= n; i++)
    {
        cin>>a[i];
    }
    cin>>x;    // 求 x 第一次出现的位置

    for(int i = 1; i <= n; i++)
    {
        if(a[i] == x)
        {
            cout<<i;
            return 0;
        }
    }

    cout<<-1; // 如果没找到, 在循环结束后才能判断找不到的情况
    return 0;
}
```

- (1) 循环过程中，我们不能确定找不到，循环结束后才能确定找不到；
- (2) 注意采用标记的思想，循环结束后要确定找不到，我们可以使用标记；

```
#include <bits/stdc++.h> //1-1154-2  javacn
using namespace std;
int a[110], n, x;
int main()
{
    cin>>n;    // 读入
    for (int i = 1; i <= n; i++)
    {
        cin>>a[i];
    }
    cin>>x;    // 求 x 第一次出现的位置

    bool f = false; // 表示没有找到
    for (int i = 1; i <= n; i++)
    {
        if(a[i] == x)
        {
            cout<<i;
            f = true;
            break;
        }
    }

    if(f == false)    // 如果没找到，在循环结束后才能判断找不到的情况
    {
        cout<<-1;
    }
    return 0;
}
```

这一道题还是数组问题，我们就用数组来解决吧

最多 20 个数所以将数组 a 设为 21 个元素

然后每一次都判读值是否大于最大值，成立就把最大值设为这个值

同时也判断值是否小于最小值，成立就把最小值设为这个值

注意！这里最小值可以设为值的最大 + 1，也就是  $32767 + 1 = 32768$ ~

最后分别输出最大值和最小值就可以 AC 啦 ~

```
#include <bits/stdc++.h> //2-1152
using namespace std;
int main()
{
    int n, a[21];
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> a[i];
    }

    int max = a[1], min = a[1];

    for (int i = 1; i <= n; i++)
    {
        if (a[i] > max)
            max = a[i];

        if (a[i] < min)
            min = a[i];
    }
    cout << max << " " << min;
    return 0;
}
```

```
#include<bits/stdc++.h>//3-1168-1 zhengjia
using namespace std;
int a[20];
int main()
{
    int n, g=0;
    scanf("%d", &n);
    float s;
    for (int i=0; i<n; i++)
    {
        scanf("%d", &a[i]);
        g=g+a[i];
    }
    int max=a[0];
    int min=a[0];
    for (int i=0; i<n; i++)
    {
        if(a[i]>max)
        {
            max=a[i];
        }
    }
    for (int i=0; i<n; i++)
    {
        if(a[i]<min)
        {
            min=a[i];
        }
    }
    s=(g-max-min)*1.0/(n-2);
    printf("%.2f", s);
    return 0;
}
```

求数组的总和、最大、最小，然后按题意求平均。

```
#include<bits/stdc++.h>//3-1168-2 javacn
using namespace std;
int a[20],n;
// 将最大数设置为整数的最小值
// 最小数，设置为整数的最大值
int ma=INT_MIN,mi=INT_MAX;
double s;
int main()
{
    int i;
    cin>>n;
    for(i=0; i<n; i++)
    {
        cin>>a[i];
        // 打擂台求最大、最小
        if(a[i] > ma) ma = a[i];
        if(a[i] < mi) mi = a[i];

        s = s + a[i];// 求和
    }

    cout<<fixed<<setprecision(2)<<(s - ma - mi)/(n-2);
    return 0;
}
```

```
#include<bits/stdc++.h>//4-1170 javacn
using namespace std;
/*
1. 求最大数
2. 求哪些位置的数是最大数
*/
int main()
{
    int n, i, a[20], max=INT_MIN;// 求最大数 ;
    cin>>n;
    for (i=0; i<n; i++)
    {
        cin>>a[i];
        if(a[i]>max)
        {
            max=a[i];
        }
    }
    // 求哪些位置的数是最大的
    for (i=0; i<n; i++)
    {
        if(a[i]==max)
        {
            cout<<i+1<<endl;
        }
    }
    return 0;
}
```

```
#include <bits/stdc++.h>//5-1212
using namespace std;
int a[210], n, ma, mi, t;
int main()
{
    cin>>n;
    for (int i = 0; i < n; i++)          cin>>a[i];
    // 找最小数下标和最大数下标
    mi = 0;
    ma = 0;
    for (int i = 1; i < n; i++)
    {
        if(a[i] < a[mi])          mi = i;

        if(a[i] > a[ma])          ma = i;
    }

    swap(a[mi], a[0]); // 对调下标为 min 的数和下标为 0 的数
    swap(a[ma], a[n-1]); // 对调下标为 max 的数和下标为 n-1 的数
    for (int i = 0; i < n; i++)
    {
        cout<<a[i]<<" ";
    }
    return 0;
}
```

应付款 = ( 总和 - 最贵的书价 + 最贵的书价 \* 0.9 ) \* 0.9，因此本题求总和和最大数。

```
#include<bits/stdc++.h>//6-1427
using namespace std;
int a[110], n, ma=INT_MIN, s=0;

int main()
{
    cin>>n;
    for (int i=0; i<n; i++)
    {
        cin>>a[i];
        s=s+a[i]; // 求和
        // 打擂台求最大
        if(a[i]>ma) ma=a[i];
    }

    //( 总和 - 最贵的书价 + 最贵的书价 * 0.9 ) * 0.9
    cout<<fixed<<setprecision(1)<<(s-ma+ma*0.9)*0.9;
    return 0;
}
```

```
#include<bits/stdc++.h>>//7-1428 javacn
using namespace std;
/*
找数组的最大数位置、最小数位置
*/
int a[110],n,ma,mi;
int main()
{
    cin>>n;
    // 下标从 1 开始用
    for(int i=1; i<=n; i++)
    {
        cin>>a[i];
    }
    // 假设第 1 个数是最大的、最小的
    ma = 1,mi = 1;//ma,mi 存: 下标
    // 循环其余的数
    for(int i=2; i<=n; i++)
    {
        if(a[i]<a[mi])
        {
            mi = i;
        }

        if(a[i]>a[ma])
        {
            ma = i;
        }
    }
    cout<<ma<<" "<<mi;

    return 0;
}
```

```

#include<iostream>//8-1581 马里奥的银币1  改自 zhengjia
#include<cstdio>
#include<cstdlib>
using namespace std;
int a[1005];
int main()
{
    int n;
    int max=a[0];
    scanf("%d",&n);
    for (int i=0;i<n;i++)    scanf("%d",&a[i]);
    for (int i=0;i<n;i++)
    {
        if(a[i]>max)    max=a[i];
    }
    for (int i=0;i<n;i++)
    {
        if(a[i]==max)
        {
            if(a[i]%2==0)
            {
                a[i]=a[i]*2;
                for (int i=0;i<n;i++)    printf("%d ",a[i]);
            }
            else if(a[i]%2==1)
            {
                a[i]++;
                for (int i=0;i<n;i++)    printf("%d ",a[i]);
            }
        }
    }
    return 0;
}

```

```
#include<iostream>//9-1583 马里奥的银币2 zhengjia
```

```
#include<cstdio>
```

```
using namespace std;
```

```
int a[1010];
```

```
int main()
```

```
{
```

```
    int n;
```

```
    scanf("%d",&n);
```

```
    for(int i=0;i<n;i++)        scanf("%d",&a[i]);
```

```
    int min=a[0];
```

```
    int max=a[0];
```

```
    for(int i=0;i<n;i++)
```

```
    {
```

```
        if(a[i]>=max)        max=a[i];
```

```
        if(a[i]<=min)        min=a[i];
```

```
    }
```

```
    for(int i=0;i<n;i++)
```

```
    {
```

```
        if(a[i]==max)        a[i]*=2;
```

```
        if(a[i]==min)        a[i]++;
```

```
    }
```

```
    for(int i=0;i<n;i++)        printf("%d ",a[i]);
```

```
    return 0;
```

```
}
```

```

#include <bits/stdc++.h> //10-1584 马里奥花银币          javacn
using namespace std;
/*
最小金额花光、最大金额花 1 个
消费 = 最小值 * 最小值个数 + 最大值
1、求最大、最小数、总和
2、求最小数有几个
3、计算消费金额，剩余金额
*/
//s: 总和 c: 最小数有几个
int n, a[1010], ma = INT_MIN, mi = INT_MAX, s = 0, c = 0;
int main()
{
    int i;
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
        s = s + a[i]; // 求和
        if (a[i] > ma)    ma = a[i];
        if (a[i] < mi)    mi = a[i];
    }

    for (i = 0; i < n; i++) // 计算最小数的个数
    {
        if (a[i] == mi)
        {
            c++;
        }
    }

    cout << ma + mi * c << endl; // 输出消费金额
    cout << s - (ma + mi * c);
    return 0;
}

```

```
#include <bits/stdc++.h> //11-1585-1 马里奥找中等的银币 推荐学习 javacn
```

```
using namespace std;
```

```
/*
```

思路:

1. 求平均

2. 打擂台求离平均值最近的数

```
*/
```

```
double mi = INT_MAX; // 离平均值最小的差值
```

```
int r; // 离平均值最近的数
```

```
double v; // 平均
```

```
int a[1010], s = 0, n;
```

```
int main()
```

```
{
```

```
    cin >> n;
```

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

```
    {
```

```
        cin >> a[i];
```

```
        s = s + a[i];
```

```
    }
```

```
    v = s * 1.0 / n; // 平均
```

```
    // 求离平均值最近的数
```

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

```
    {
```

```
        // 差值更小, 或者差值相等, 但数值更小
```

fabs 求绝对值

```
        if (fabs(a[i] - v) < mi)
```

```
        {
```

```
            mi = fabs(a[i] - v);
```

```
            r = a[i];
```

```
        }
```

```
    }
```

```
    cout << r;
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//11-1585-2 马里奥找中等的银币
```

```
using namespace std;
```

```
int main() {
```

```
    int a[1010], n;
```

```
    int sum=0, num=0;
```

```
    double ave=0, k, min=INT_MAX;
```

```
    cin>>n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        sum=sum+a[i];
```

```
    }
```

```
    ave=(sum*1.0)/n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        if(a[i]>ave)
```

```
        {
```

```
            k=a[i]-ave;
```

```
        }
```

```
    else
```

```
    {
```

```
        k=ave-a[i];
```

```
    }
```

```
        if(k<min)
```

```
        {
```

```
            min=k;
```

```
            num=i;
```

```
        }
```

```
    }
```

```
    cout<<a[num];
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//12-1723-1 奇偶数选大王
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, a[1000];
```

```
    int max1=0, max2=0;
```

```
    cin>>n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        if(a[i]%2==0&& a[i]>=max2)
```

```
        {
```

```
            max2=a[i];
```

```
        }
```

```
        else if(a[i]%2==1&& a[i]>max1)
```

```
        {
```

```
            max1=a[i];
```

```
        }
```

```
    }
```

```
    sort(a, a+n);
```

```
    cout<<max1<<" "<<max2<<endl;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cout<<a[i]<<" ";
```

```
    }
```

```
    return 0;
```

```
}
```

```

#include<bits/stdc++.h>//12-1723-2
using namespace std;
int main()
{
    int n, a[1010];
    cin>>n;
    for (int i=0;i<n;i++)
    {
        cin>>a[i];
    }
    int max=a[0];
    int max1=a[0];
    for (int i=0;i<n;i++)
    {
        if(a[i]%2==0)
        {
            if(a[i]>max)
            {
                max=a[i];
            }
        }
        else
        {
            if(a[i]>max1)
            {
                max1=a[i];
            }
        }
    }
    cout<<max1<<" "<<max<<endl;
    sort(a, a+n);
    for (int i=0;i<n;i++)
    {
        cout<<a[i]<<" ";
    }
    return 0;
}

```

先求最大最小，再去除所有的最大最小以后求平均。

```
#include<bits/stdc++.h>//13-1724 采摘活动 javacn
using namespace std;
int a[1010],n,r;
int ma=INT_MIN;
int mi=INT_MAX;
int s = 0;// 总和
int main()
{
    cin>>n;
    r=0;
    int i;
    for (i=0; i<n; i++)
    {
        cin>>a[i];
        s=s+a[i];
        if(a[i]>ma) ma=a[i];
        if(a[i]<mi) mi=a[i];
    }
    for (i=0; i<n; i++) // 去掉所有的最大最小，同时统计有几个
    {
        if(a[i]==ma)
        {
            s=s-a[i];
            r++;
        }
        if(a[i]==mi)
        {
            s=s-a[i];
            r++;
        }
    }
    cout<<s<<endl<<fixed<<setprecision(1)<<s*1.0/(n-r); // 求平均
    return 0;
}
```

```
#include<bits/stdc++.h>//14-1809 最贵商品和最便宜商品之差 javacn
using namespace std;
/*
求数组的最大数和最小数的差
*/
//ma 代表最大数, mi 代表最小数
int a[110], n, ma, mi;
int main()
{
    cin>>n;
    for (int i = 0; i < n; i++)
    {
        cin>>a[i];
    }

    ma = a[0];// 假设第 1 个数最大
    mi = a[0];

    // 循环剩余的元素, 打擂台找最大、最小数
    for (int i = 1; i < n; i++)
    {
        if(a[i] > ma)
        {
            ma = a[i];
        }

        if(a[i] < mi)
        {
            mi = a[i];
        }
    }

    cout<<ma - mi;
    return 0;
}
```

```
#include<bits/stdc++.h>//15-1810-1 最贵商品和最便宜商品分别有几个 jkcx001
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, a[2000], sum1=0, sum2=0;
```

```
    int max=INT_MIN, min=INT_MAX;
```

```
    cin>>n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        if(a[i]>max) max=a[i];
```

```
        if(a[i]<min) min=a[i];
```

```
    }
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        if(a[i]==max) sum1++;
```

```
        if(a[i]==min) sum2++;
```

```
    }
```

```
    cout<<sum1<<" "<<sum2;
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//15-1810-2 最贵商品和最便宜商品分别有几个 jkcx001
```

```
using namespace std;
```

```
int main()
```

```
{  
    int n, a[2000], sum1=0, sum2=0;  
  
    cin>>n;  
    for (int i=0;i<n;i++)  
    {  
        cin>>a[i];  
    }  
    int max=a[0], min=a[0];  
    for (int i=0;i<n;i++)  
    {  
        if(a[i]>max) max=a[i];  
        if(a[i]<min) min=a[i];  
    }  
    for (int i=0;i<n;i++)  
    {  
        if(a[i]==max) sum1++;  
        if(a[i]==min) sum2++;  
    }  
    cout<<sum1<<" "<<sum2;  
    return 0;  
}
```

```
#include <bits/stdc++.h>//16-1811-1 买不买
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i, a[120], k=0;
```

```
    cin>>n;
```

```
    for (i=1; i<=n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        k=k+a[i];
```

```
    }
```

```
    sort(a+1, a+n+1) ;
```

```
    if(k<=100)        cout<<"No"<<endl<<k;
```

```
    if(k>100)        cout<<"Yes"<<endl<<k-a[n];
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h>//16-1811-2 买不买
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i, a[120], k=0, max=INT_MIN;
```

```
    cin>>n;
```

```
    for (i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        if(a[i]>max) max=a[i];
```

```
        k=k+a[i];
```

```
    }
```

```
    if(k<=100)        cout<<"No"<<endl<<k;
```

```
    if(k>100)        cout<<"Yes"<<endl<<k-max;
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//17-1813 找出最大的圆 marsshu
```

```
using namespace std;
int main()
{
    int n, a[1000];
    int max=0, k=0;
    cin>>n;
    for (int i=1; i<=n; i++)
    {
        cin>>a[i];
        if(a[i]>max)
        {
            max=a[i];
            k=i;
        }
    }
    printf("%.2lf", max*max*3.14);
    printf(" %d", k); //%d 前面是空格
    return 0;
}
```

```
#include<bits/stdc++.h>//18-1814 卖苹果的最大利润 javacn
```

```
using namespace std; // 打擂台求最大数下标，然后求出最大利润。
int a[110], j, m, x, n;
int ma=1; // 假设下标为 1 的数是最大数
int main()
{
    cin>>m>>x>>n;
    for (int i=1; i<=n; i++) // 我们下标从 1 开始，比较方便求位置
    {
        cin>>a[i];
        if(a[i]>a[ma]) ma=i; // 打擂台找最大数下标
    }
    cout<<ma<<" "<<(a[ma]-x)*m; // 输出结果
    return 0;
}
```

```
#include<bits/stdc++.h> //19-1815 最后一次出现该数的位置 jkcx001
```

```
using namespace std;
```

```
int n, x, a[101], idx=-1;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

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

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    cin>>x;
```

```
    for (int i=n; i>=1; i--)
```

```
    {
```

```
        if (a[i]==x)
```

```
        {
```

```
            idx=i;
```

```
            break;
```

```
        }
```

```
    }
```

```
    cout<<idx;
```

```
    return 0;
```

```
}
```

# 元素移动

```
#include<bits/stdc++.h>//1-1157 最小数  hasome
using namespace std;
int n, a[210], i, mi=INT_MAX, t, p;
int main()
{
    cin>>n;
    for (i=1; i<=n; i++)
    {
        cin>>a[i];
        if(a[i]<mi) // 打擂台求最小数，只记第一个
        {
            mi=a[i];
            p=i;
        }
    }
    swap(a[1], a[p]); // 交换
    cout<<p<<endl;
    for (i=1; i<=n; i++) cout<<a[i]<<" ";
    return 0;
}
```

```
#include<iostream>//2-1159 数组元素的移动
using namespace std;
int main() {
    int n, i, a[15], x;
    cin>>n;
    for (i=1; i<=n; i++)    cin>>a[i];
    cin>>x;
    int t=a[x];
    for (i=x; i<n; i++)    a[i]=a[i+1];
    a[n]=t;
    for (i=1; i<=n; i++)    cout<<a[i]<<" ";
    return 0;
}
```

```
#include <bits/stdc++.h>//3-1211 数组元素的插入 dragoncatter
```

```
using namespace std;
```

```
int a[20];
```

```
int main()
```

```
{
```

```
    int n, x, y;
```

```
    cin >> n;
```

```
    for (int i=1; i<=n; i++)        cin >> a[i];
```

```
    cin >> x >> y;
```

```
    for (int i=n; i>=x; i--)        a[i+1]=a[i];
```

```
    a[x]=y;
```

```
    for (int i=1; i<=n+1; i++)      cout << a[i] << " ";
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//4-1161-1 元素插入有序数组
```

```
using namespace std;
```

```
int a[1005];
```

```
int main()
```

```
{
```

```
    int i, n, m;
```

```
    cin >> m ;
```

```
    cin >> n;
```

```
    for (int i = 0; i < n; i++)
```

```
        cin >> a[i];
```

```
    a[n] = m;
```

```
    sort(a, a+n+1);
```

```
    for (int i = 0; i <= n; i++)
```

```
        cout << a[i] << " ";
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//4-1161-2 元素插入有序数组
```

```
using namespace std;
```

```
int a[1005];
```

```
int main()
```

```
{
```

```
    int i, n, m, p;
```

```
    cin >> m ;
```

```
    cin >> n;
```

```
    for (i = 0; i < n; i++) cin >> a[i];
```

```
    for (i = 0; i < n; i++)// 找到插入的位置
```

```
    {
```

```
        if(a[i]>m)
```

```
        {
```

```
            p=i;
```

```
            break;
```

```
        }
```

```
    }
```

```
    for (i = n; i >= p+1; i--)// 数字向后移动
```

```
    {
```

```
        a[i]=a[i-1];
```

```
    }
```

```
    a[p] = m;
```

```
    for (i = 0; i <= n; i++)
```

```
        cout << a[i] <<" ";
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//5-1162 数组元素的删除 hasome
```

```
using namespace std;
const int N=20;
int x, n, a[N], i;
int main()
{
    cin>>n;
    for (i=1; i<=n; i++)    cin>>a[i];
    cin>>x;
    for (i=x; i<n; i++)
    {
        a[i]=a[i+1];
    }
    for (i=1; i<n; i++)    cout<<a[i]<<" ";
    return 0;
}
```

```
#include <bits/stdc++.h>//6-1213-1 删除数组最小数 javacn 推荐学习
```

```
using namespace std;// 解法一：输出除了最小数以外的数
int main()
{
    int a[110], n, mi, i;
    cin>>n;
    for (i = 0; i < n; i++)    cin>>a[i];
    mi = a[0];
    for (i = 1; i < n; i++)    // 查找最小数
    {
        if(a[i] < mi)    mi = a[i];
    }

    for (i = 0; i < n; i++)    // 输出除了最小数以外的数
    {
        if(a[i] != mi)    cout<<a[i]<<" ";
    }
    return 0;
}
```

解法二：求最小数下标，从最小数下标开始将元素前移

```
#include <bits/stdc++.h> //6-1213-2 javacn
```

```
using namespace std;
```

```
int main()
{
    int a[110], n, mi, i;
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    // 查找最小数下标
    mi = 0;
    for (i = 1; i < n; i++)
    {
        if (a[i] < a[mi])
        {
            mi = i;
        }
    }
    // 从最小数的下标开始将元素前移
    for (i = mi; i < n - 1; i++)
    {
        a[i] = a[i + 1];
    }
    // 少了一个元素
    n--;
    for (int i = 0; i < n; i++) // 输出
    {
        cout << a[i] << " ";
    }
    return 0;
}
```

```

#include <bits/stdc++.h> //7-1214      在最大数后面插入一个新数      hasome
using namespace std;
int a[110], n, i, p, ma=INT_MIN, y;
int main()
{
    cin>>n;

    for (i=1; i<=n; i++)      // 找出最大的数的下标
    {
        cin>>a[i];
        if(a[i]>ma)
        {
            ma=a[i];
            p=i;      //p 是最大数下标
        }
    }

    for (i=n; i>p; i--)      // 从最大的数开始所有的数后移一位
    {
        a[i+1]=a[i];
    }

    cin>>y;      // 插入这个数
    a[p+1]=y;

    for (i=1; i<=n+1; i++)
    {
        cout<<a[i]<<" ";
    }
    return 0;
}

```

```
#include <bits/stdc++.h> //8-1217 小明排队做操迟到 hasome
```

```
using namespace std;
```

```
int a[1010], n, x, y, i, p;
```

```
int main()
```

```
{
```

```
    cin>>n>>x>>y;
```

```
    for (i=1; i<=n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        if(a[i]==x) p=i;
```

```
    }
```

```
    for (i=n; i>p; i--)
```

```
    {
```

```
        a[i+1]=a[i];
```

```
    }
```

```
    a[p+1]=y;
```

```
    n++;
```

```
    for (i=1; i<=n; i++)
```

```
    {
```

```
        cout<<a[i]<<" ";
```

```
    }
```

```
    return 0;
```

```
}
```

```

#include <bits/stdc++.h> //9-1232    换位置    javacn
using namespace std;
// 思路：找到最大数和最小数下标，交换这两个下标对应的数
int main()
{
    //max 和 min 代表最大、最小数下标
    int n, i, max, min, a[1000];
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }

    /*
    max 用来存储最大数下标
    假设第 1 个数最大，第 1 个数下标为 0
    假设 max=0
    */
    max = 0; // 假设下标为 0 的数是最大的
    min = 0; // 假设下标为 0 的数是最小的
    for (i = 1; i < n; i++)
    {
        // 如果下标为 i 的数 > 下标为 max 的数
        if (a[i] > a[max])    max = i;
        if (a[i] < a[min])    min = i;
    }

    swap(a[max], a[min]);

    for (i = 0; i < n; i++) // 输出
    {
        cout << a[i] << " ";
    }
    return 0;
}

```

# 数组排序

本题可以用各类排序算法排序，也可以使用 sort 函数排序。

解法一：sort 函数排序

```
#include <bits/stdc++.h> //1-1010-1 数组元素的排序    javacn
using namespace std;
int main()
{
    int a[20], i, n;
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    // 对数组排序
    sort(a, a+n);
    // 输出
    for (i = 0; i < n; i++)
    {
        cout << a[i] << " ";
    }
    return 0;
}
```

## 解法二：冒泡排序

```
#include <iostream>//1-1010-2 javacn
using namespace std;
int main()
{
    int i,n,a[10],j,t;
    cin>>n;// 读入数组长度和数组每个数
    for(i = 0;i < n;i++)    {    cin>>a[i];    } // 读入数组每个元素
/* 循环排序的轮数，n个数排序 n-1 轮
*/ 这里的 i 代表排序的轮数
for(i = 1;i <= n - 1;i++)
{
    /*
    第 i 轮从下标为 0 的元素比较到下标为 n-i 的元素
    但下标循环只能从 0 循环到 n-i-1  */
    for(j = 0;j <= n - i - 1;j++)
    {
        /* 如果前面的元素 > 后面的元素，交换它们
        让较大的数，向后移动，变为本轮最后一个数
        */
        if(a[j] > a[j + 1])
        {
            t = a[j];
            a[j] = a[j + 1];
            a[j + 1] = t;
            // 交换 a[j] 和 a[j+1]，还可以通过函数 swap(a[j], a[j+1]) 来实现
        }
    }
}
for(i = 0;i < n;i++)    // 输出排序的结果
{
    cout<<a[i]<<" ";
}
return 0;
}
```

1. 对于每个读入的数，可以用普通的拆位方法，也可以用短除法拆位求各个位的和；
2. 对于各个位的和进行排序，排序可以用冒泡、快排，也可以用 sort 函数排序；

```
#include <bits/stdc++.h>//2-1166 数的排序 javacn
```

```
using namespace std;
```

```
int a[20], n, s;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    int i;
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
        s = 0; // 每个 a[i] 都要求和，单独清零
```

```
        // 求 a[i] 的各个位的和
```

```
        while (a[i] != 0)
```

```
        {
```

```
            s = s + a[i] % 10;
```

```
            a[i] = a[i] / 10;
```

```
        }
```

```
        a[i] = s;
```

```
    }
```

```
    // 排序
```

```
    sort(a, a+n);
```

```
    // 输出
```

```
    for (int i = 0; i < n; i++)
```

```
    {
```

```
        cout<<a[i]<<" ";
```

```
    }
```

```
    return 0;
```

```
}
```

```

#include<bits/stdc++.h>//3-1172-1 寻找第K大数字
using namespace std;
int a[10010], n, i, k;

int main()
{
    cin>>n>>k;
    for (i=1; i<=n; i++) cin>>a[i];
    sort(a+1, a+n+1, greater<int>());    // 从大到小排序
    cout<<a[k]<<endl;
    return 0;
}

```

```

#include <iostream>//3-1172-2 寻找第K大数字  javacn
using namespace std;
int main() {
    int a[1010], n, i, j, t, k; // 数组大小按照题目的要求定义，最好多定义 10 个
    cin>>n>>k;
    for (i=0; i<n; i++) {    cin>>a[i]; }
    for (i=1; i<=n-1; i++)
    {
        for (j=0; j<=n-i-1; j++)
        {
            // 按照从大到小排序
            if(a[j]<a[j+1])
            {
                t=a[j];
                a[j]=a[j+1];
                a[j+1]=t;
            }
        }
    }
    cout<<a[k-1]; // 第 1 大的数，下标是 0，第 2 大的数下标是 1，第 3 大的数下标是 2
    return 0; // 第 k 大的数，下标是 k-1
}

```

```
#include<bits/stdc++.h>//4-1175 语文成绩 hasome
```

```
using namespace std;
int n, a[160], sum, i;
int main() {
    cin>>n;
    for (i=1; i<=n; i++)
    {
        cin>>a[i];
        sum+=a[i];
    }
    cout<<sum<<endl;
    printf("%. 2f\n", sum*1. 0/n);
    sort(a+1, a+n+1);
    for (i=n; i>=1; i--)
    {
        cout<<a[i]<<" ";
    }
    return 0;
}
```

```
#include <bits/stdc++.h>//5-1221 优秀成绩的平均分 hasome
```

```
using namespace std;
int a[110], n, i;
double ans;
int main()
{
    cin>>n;
    for (i=1; i<=n; i++)    cin>>a[i];
    sort(a+1, a+n+1);    // 从小到大排序
    for (i=n; i>n-5; i--)
    {
        ans=ans+a[i];
    }
    printf("%. 1f", ans/5);
    return 0;
}
```

```
#include <bits/stdc++.h> //6-1233 求中位数 zhanghaotian
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n;
```

```
    cin>>n;
```

```
    int a[n];
```

```
    float med;
```

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

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    sort(a+1, a+n+1, less<int>());
```

```
    if (n%2==0)
```

```
    {
```

```
        med=(a[n/2]+a[n/2+1])/2.0;
```

```
    }
```

```
    else
```

```
    {
```

```
        med=a[(n+1)/2];
```

```
    }
```

```
    printf("%.1f", med);
```

```
    return 0;
```

```
}
```

```

#include<bits/stdc++.h>//7-1242 javacn
using namespace std;
int n, k, ans=0;
int a[10010]= {0};

int main()
{
    cin>>n>>k;
    for (int i=1; i<=n; i++) cin>>a[i];
    sort(a+1, a+n+1);
    ans=a[n+1-k]-a[k];

    bool f = true;// 假设是素数
    for (int i = 2; i <= sqrt(ans); i++) // 判断素数
    {
        if(ans % i == 0)
        {
            f = false;
            break;
        }
    }
    if(ans <= 1) f = false;

    if(f) cout<<"YES"<<endl;
    else cout<<"NO"<<endl;
    cout<<ans;
    return 0;
}

```

```
#include <bits/stdc++.h>//8-1399 学员的名次 zhanghaotian
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, x;
```

```
    cin>>n;
```

```
    int a[n];
```

```
    for (int i=1; i<=n; i++)    cin>>a[i];
```

```
    sort(a+1, a+n+1, greater<int>());
```

```
    cin>>x;
```

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

```
    {
```

```
        if(x==a[i])    cout<<i;
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//9-1451-1 n个一位数能够组成的最大数 一柯 正确解法
```

```
using namespace std;
```

```
int a[12], c, b=1;
```

```
int main()
```

```
{
```

```
    int n, m;
```

```
    int i;
```

```
    cin>>n;
```

```
    for (i=1; i<=n; i++)
```

```
        cin>>a[i];
```

```
    sort(a+1, a+1+n, less<int>());
```

```
    for (i=1; i<=n; i++)
```

```
    {
```

```
        c=c+a[i]*b;
```

```
        b=b*10;
```

```
    }
```

```
    cout<<c;
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h> //9-1451-2 n 个一位数能够组成的最大数
```

```
using namespace std;  
int a[12];  
int main()  
{  
    int n, m, i;  
    cin >> n;  
    for (i = 0; i < n; i++)        cin >> a[i];  
    sort(a, a+n, greater<int>());  
    for (i = 0; i < n; i++)  
    {  
        cout << a[i];  
    }  
    return 0;  
}
```

```
#include<iostream> //9-1451-3 n 个一位数能够组成的最大数      jiangyf70
```

```
using namespace std;  
int a[10];  
int main()  
{  
    int n, m;  
    cin >> n;  
    for (int i = 0; i < n; i++)  
    {  
        cin >> m;  
        a[m]++; // 加入 0-9 下标数组计算个数  
    }  
    for (int i = 9; i >= 0; i--)  
    {  
        for (int j = 0; j < a[i]; j++)  
            cout << i; // 倒序输出, 没有的不输出  
    }  
    return 0;  
}
```

`#include<bits/stdc++.h> //10-1452-1` n个一位数能够组成的最小数 一柯 正确解法

```
using namespace std;
```

```
int a[15], c, b=1;
```

```
int main()
```

```
{
```

```
    int n;
```

```
    int i;
```

```
    cin>>n;
```

```
    for (i=1; i<=n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    sort(a+1, a+1+n, less<int>());
```

```
    if(a[1]==0)
```

```
    {
```

```
        for (i=2; i<=n; i++)
```

```
        {
```

```
            if(a[i]>a[1])
```

```
            {
```

```
                swap(a[i], a[1]);
```

```
                break;
```

```
            }
```

```
        }
```

```
    }
```

```
    for (i=n; i>=1; i--)
```

```
    {
```

```
        c=c+a[i]*b;
```

```
        b=b*10;
```

```
    }
```

```
    cout<<c;
```

```
    return 0;
```

```
}
```

将所有的数按照从小到大排序，注意，如果第 1 个数是 0，那么要和数组中的第 1 个非 0 元素交换。

`#include <bits/stdc++.h>` //10-1452-2 n 个一位数能够组成的最小数 [推荐学习 javacn](#)

```
using namespace std;
int main()
{
    int n, k, i, a[1010];
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }

    sort(a, a+n);

    if(a[0] == 0) // 如果第 1 个元素是 0，和第 1 个非 0 元素交换
    {
        for (i = 0; i < n; i++)
        {
            if(a[i] != 0)
            {
                swap(a[0], a[i]);
                break;
            }
        }
    }

    for (i = 0; i < n; i++)
    {
        cout << a[i];
    }
    return 0;
}
```

```

#include<bits/stdc++.h>//11-1453 橘子排队 fideI2021
using namespace std;
int main() {
    int n, i, a[205], s = 0;
    cin >> n;
    for(i = 1; i <= n; i++)
    {
        cin >> a[i];
        s += a[i];
    }
    sort(a+1, a+n+1); // 先处理排序
    s = s - a[1] - a[n]; // 再处理最大值和最小值
    printf("%.1f\n", s * 1.0 / (n - 2));
    for(i = 2; i < n; i++)        cout << a[i] << " ";
    return 0;
}

```

```

#include <bits/stdc++.h>//12-1454 允许并列的排名
using namespace std;
int a[1000000], b[1000000], c[1000000]; /*
int main() {                               80 50 50 30    成绩
    int n, m ;                               1  2  3  4    下标
    cin>>n;                                   1  2  2  4    排名
    for (int i=1; i<=n; i++)    cin>>a[i];   */
    cin>>m;
    sort(a+1, a+n+1, greater<int>());
    for (int i=1; i<=n; i++)
    {
        if(a[i]==m)
        {
            cout<<i;
            return 0;
        }
    }
    return 0;
}

```

```

#include<bits/stdc++.h>//13-1455-1 允许并列的排名2 推荐学习 javacn
using namespace std; //1. 对数组降序排序
int a[1010], b[1010], n; //2. 求每个元素在排序后的数组中第一次出现的位置
int main() {
    cin>>n;
    for (int i = 1; i <= n; i++)
    {
        cin>>a[i];
        b[i] = a[i];
    }
    sort(a+1, a+n+1, greater<int>());
    for (int i = 1; i <= n; i++) // 求 b[i] 在 a 数组中首次出现的位置
    {
        for (int j = 1; j <= n; j++)
        {
            if(b[i] == a[j])
            {
                cout<<j<<" ";
                break;
            }
        }
    }
    return 0;
}

```

```

#include<bits/stdc++.h>//13-1455-2 允许并列的排名2 仅供参考 zhengjia
using namespace std; int a[1010];
int main() {
    int n, r=0;
    scanf("%d", &n);
    for (int i=0; i<n; i++) scanf("%d", &a[i]);
    for (int i=0; i<n; i++)
    {
        int r=0;
        for (int j=0; j<n; j++) { if(a[i]<a[j]) r++; }
        printf("%d ", r+1);
    }
    return 0;
}

```

短除法拆出读入的 n 个数的每一位，存入数组，排序。

```
#include<bits/stdc++.h>//14-1458-1 粉碎数字 javacn
using namespace std;
//x 代表读入的每个数
//k 代表拆到 a 数组的每个数的下标
int a[4010],n,x,k = 0;
int main()
{
    cin>>n;
    // 读入 n 个数
    for (int i = 1;i <= n;i++) //a 数组，存储了 k 个一位数
    {
        cin>>x;
        // 将 x 的每一位存入数组
        while(x != 0)
        {
            k++;// 数组长度 +1
            // 注意：此处 a 数组下标从 1 开始用
            a[k] = x % 10;// 拆出 x 的个位
            x = x / 10;// 去掉个位
        }
    }

    sort(a+1, a+k+1);
    reverse(a+1, a+k+1); // 反转排序

    for (int i = 1; i <= k; i++) // 输出
    {
        cout<<a[i];
    }
    return 0;
}
```

```
#include<iostream>//14-1458-2 jiangyf70 仅供参考
```

```
using namespace std;// 利用 char 一位一位读入。
```

```
int a[10];
int main() {
    int n;
    cin >> n;
    char c;
    while(cin >> c) a[c-'0']++;
    for (int i = 9; i >=0; i--)
    {
        for (int j = 0; j < a[i]; j++)    cout << i;
    }
    return 0;
}
```

```
#include<bits/stdc++.h>//15-1473-1 去掉 x 个最高最低分后的平均分 zhengjia
```

```
using namespace std;
```

```
int a[10010];
int main() {
    int n, x;
    float p;
    int sum=0, da=0, xi=0;
    scanf ("%d%d", &n, &x);
    for (int i=0; i<n; i++)
    {
        scanf ("%d", &a[i]);
        sum+=a[i];
    }
    sort(a, a+n);
    for (int i=0; i<x; i++)    xi=xi+a[i];
    sort(a, a+n, greater<int>());
    for (int i=0; i<x; i++)    da=da+a[i];
    p=(sum-xi-da)*1.0/(n-x-x);
    printf("%.1f", p);
    return 0;
}
```

```
#include<bits/stdc++.h>//15-1473-2 去掉 x 个最高最低分后的平均分 475214719
```

```
using namespace std;
int main()
{
    int i, j, x, n, sum=0, a[10010];
    cin>>n>>x;
    for (i=1; i<=n; i++)    cin>>a[i];
    for (i=1; i<=n-1; i++)
        for (j=1; j<=n-i; j++)
            if (a[j]>a[j+1])
                swap(a[j], a[j+1]);

    for (i=x+1; i<=n-x; i++)    sum=sum+a[i];
    cout<<fixed<<setprecision(1)<<(sum*1.0)/(n-x*2)<<endl;
    return 0;
}
```

```
#include <iostream>//17-1498-1 宇宙总统? 推荐学习 kevinh
```

```
using namespace std;
int main() {
    int n, a[101]={0}, x, i;
    cin>>n;
    for (i=0; i<n; i++)
    {    // 统计每个编号票数
        cin>>x;
        a[x]+=1;
    }

    int ma=1;// 计算 1-100 编号票数最大值, 最大值相等, 则取编号最大数
    for (i=1; i<=100; i++)
    {
        if (a[i]>a[ma] || (a[i]==a[ma] && i>ma))
            ma=i;
    }

    cout<<ma<<endl;
    return 0;
}
```

```

#include<bits/stdc++.h>//17-1498-2 宇宙总统? 475214719
using namespace std;
int main()
{
    int i, j, m, max, maxid, n, sum=0, a[1005], b[1005]={0};
    cin>>n; //10
    for (i=1;i<=n;i++)
        cin>>a[i]; // 1 1 3 4 2 2 7 5 6 6

    for (i=1;i<=n-1;i++)
        for (j=1;j<=n-i;j++)// 冒泡法找出把输入的 n 个数从小到大排序
            if(a[j]>a[j+1]) swap(a[j], a[j+1]); // 1 1 2 2 3 4 5 6 6 7

    m=a[n]; // 最大的数赋值给 m m 是编号最大的数

    for (i=1;i<=m;i++)
        for (j=1;j<=n;j++)
            if(a[j]==i) b[i]++; // 分别对每个编号获得票数计数

    max=0;
    for (i=1;i<=m;i++)
        if(b[i]>=max)
        {
            max=b[i];
            maxid=i;
        }// 找出最大的数（最后面的那个最大数），并把最大数的下标赋值给 maxid

    cout<<maxid<<endl;
    return 0;
}

```

```
#include<bits/stdc++.h>//18-1812 排名第二高的成绩 fidel2021
```

```
using namespace std;
```

```
int n, i, a[105];
```

```
int main()
```

```
{
```

```
    cin >> n;
```

```
    for(i = 1; i <= n; i++)
```

```
        cin >> a[i];
```

```
    sort(a+1, a+n+1);
```

```
    for(i = n - 1; i >= 1; i--)
```

```
    {
```

```
        if(a[i] < a[n])
```

```
        {
```

```
            cout << a[i];
```

```
            break;
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

# 数组计数法

数组计数法模板题，由于读入的数在  $1 \sim 10$  之间，因此可以用长度至少为 11 的数组，统计每个数出现的次数。

总结，数组计数法的注意事项：

用来统计每个数出现多少次的数组，开多大是取决于读入数的范围，不是取决于读入了几个数；

下标为  $i$  的位置，存储的数  $a[i]$  代表的是值  $i$  出现的次数，因此下标  $i$  代表数值， $a[i]$  代表的是出现的次数；

循环统计结果，应该循环数值范围，而不是循环数的个数；

如果题目问：出现次数最多的数是哪个数，求最大数下标；

如果题目问：出现次数最多的数出现了几次，求最大数；

```
#include<bits/stdc++.h> //1-1884 求 n 个数中每个数出现的次数 javacn
using namespace std;
int a[20]; // 用来存储每个数出现的次数
int n, x;
int main()
{
    cin>>n;
    for (int i = 0; i < n; i++)
    {
        cin>>x;
        a[x]++; //x 是几，下标为几的格子就自增 1
    }
    for (int i = 1; i <= 10; i++) // 输出每个数出现的次数
    {
        if (a[i] != 0) // 如果值为 i 的数出现过了，输出该数，及出现的次数
        {
            cout<<i<<" "<<a[i]<<endl;
        }
    }
    return 0;
}
```

数组计数法统计每个数出现的次数，然后求数组的最大数下标。

注意：数组的下标  $i$  是统计的每个读入的数， $a[i]$  是值为  $i$  的数出现的次数。

```
#include<bits/stdc++.h> //2-1883 求 n 个数中出现次数最多的数 javacn
using namespace std;
int a[20]; // 用来存储每个数出现的次数
int n, x;
int main()
{
    cin>>n;
    // 循环读入数的数量
    for (int i = 1; i <= n; i++)
    {
        cin>>x;
        a[x]++;
    }

    int ma = 1; // 求：出现次数最多的数，也就是求最大数下标

    for (int i = 2; i <= 10; i++) // 循环剩余的数
    {
        if(a[i] > a[ma])
        {
            ma = i;
        }
    }

    cout<<ma;
    return 0;
}
```

数组计数法统计每个数出现的次数：

```
#include<bits/stdc++.h>//3-1725 声音识别 javacn
using namespace std;
int a[110]; // 用 a[1]~a[100] 记录 1~100 出现的次数
int n, t, c;

int main()
{
    int i;
    cin>>n;
    for (i = 1; i <= n; i++)
    {
        cin>>t; //t: 出现过的数字

        a[t]++;

        if(a[t] == 1) c++; // 如果 t 是第一次出现
    }

    cout<<c<<endl;
    for (i = 1; i <= 100; i++)
    {
        if(a[i] > 0)
        {
            cout<<i<<" "<<a[i]<<endl;
        }
    }
    return 0;
}
```

典型的：数组计数法 + 短除法拆位。

```
#include<bits/stdc++.h>//4-1178 COUNT 书的页码 javacn
```

```
using namespace std;
```

```
/*
```

```
求：1~n 的每个数，拆出每一位
```

```
统计 0~9 这些数出现的次数
```

```
*/
```

```
int a[20], n, t;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    // 循环每个数
```

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

```
    {
```

```
        t = i; // 不能拆 i, 会导致死循环
```

```
        while (t != 0)
```

```
        {
```

```
            a[t%10]++;
```

```
            t = t / 10; // 去掉个位
```

```
        }
```

```
    }
```

```
    // 输出每个数出现的次数
```

```
    for (int i = 0; i <= 9; i++)
```

```
    {
```

```
        cout<<a[i]<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

数组计数法，用数组统计每个数出现的次数，再求出计数数组的最大数。

使用数组计数法特别注意：

- (1) 如果要输出每个数出现的次数，注意循环范围是读入数的数值范围。
- (2) 在数组计数法中，下标  $i$  是统计的数，值  $a[i]$  代表的是  $i$  出现的次数。
- (3) 如果求出现次数最多的数出现了几次，求数组的最大数；

如果求出现次数最多的数是哪个数，求数组的最大数下标；

```
#include <bits/stdc++.h> //5-1180 数字出现次数 javacn
```

```
using namespace std;
```

```
int a[30], n, x, ma = INT_MIN;
```

```
int main()
```

```
{  
    // 读入 50 个数  
    for (int i = 1; i <= 50; i++)  
    {  
        cin >> x;  
        a[x]++;  
    }  
  
    // 求 0~19 下标范围内最大数  
    for (int i = 0; i <= 19; i++)  
    {  
        if (a[i] > ma) ma = a[i];  
    }  
  
    cout << ma;  
    return 0;  
}
```

由于每个数都在  $0 \sim 1000$  之间，所以可以用数组计数法来统计，统计每个数出现次数的同时，统计一下有哪些数出现过，最终输出出现过的数。

```
#include<bits/stdc++.h>//6-1183 去除重复数字 javacn
using namespace std;
int n,x,c;
int a[1010];//统计每个数出现的次数

int main()
{
    cin>>n;
    for(int i = 1;i <= n;i++)
    {
        cin>>x;
        a[x]++;

        if(a[x] == 1) c++; // 如果 x 是首次出现
    }

    cout<<c<<endl;

    for(int i = 0;i <= 1000;i++) // 按从小到大的顺序，输出出现的数
    {
        if(a[i] != 0) cout<<i<<endl;
    }

    return 0;
}
```

数组计数法统计出现过的数对的和：

```
#include<bits/stdc++.h>//7-1334 扑克牌组合 javacn
using namespace std;
/* 数组计数法：1. 循环所有的数对 2. 使用 r 数组来统计哪些数出现过 */
int a[60], r[30];
int n, c = 0; //c: 统计出现过的数有几个
int main()
{
    cin>>n;
    for (int i = 1; i <= n; i++)
    {
        cin>>a[i];
    }
    for (int i = 1; i <= n; i++) // 循环所有的数对
    {
        for (int j = i + 1; j <= n; j++)
        {
            r[a[i]+a[j]]++;
            // 如果 a[i]+a[j] 这个数是第一次出现
            if (r[a[i]+a[j]]==1)
            {
                c++;
            }
        }
    }
    cout<<c<<endl;
    for (int i = 2; i <= 26; i++) // 循环输出出现过的数
    {
        // 任意从 1~13 中找 2 个数相加，和是 2~26 范围的数
        if (r[i] != 0)
        {
            cout<<i<<" ";
        }
    }
    return 0;
}
```

```
#include<bits/stdc++.h>//8-1472 zhengjia
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i, a[1001]={0}, x;
```

```
    cin>>n;
```

```
    for (i=0; i<n; i++)
```

```
    {
```

```
        cin>>x;
```

```
        a[x]++;
```

```
    }
```

```
    for (i=1; i<=1000; i++)
```

```
    {
```

```
        if(a[i]%2==1)
```

```
        {
```

```
            cout<<i<<endl;
```

```
            return 0;
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h> //9-1557 javacn
```

```
using namespace std;
```

```
/*
```

n 个数，数值是  $1 \sim n$ ，求出现次数最多的数

思路：

1. 用长度为 1010 的数组，统计每个数出现的次数
2. 求出现次数最多的数

如果有多个相同的数，取最小的

相当于求：数组的最大数下标

```
*/
```

```
int a[1010], n, x;
```

```
int main()
```

```
{
```

```
    cin >> n >> x;
```

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

```
    {
```

```
        a[x]++;
```

```
        x = (x * 37 + 33031) % n + 1;
```

```
    }
```

```
    // 求 a 数组最大数下标
```

```
    int ma = 1;
```

```
    // 这里的循环范围应该是统计范围
```

```
    // 本题：每个数的范围是  $1 \sim n$ ，因此这里循环  $2 \sim n$ 
```

```
    for (int i = 2; i <= n; i++)
```

```
    {
```

```
        if (a[i] > a[ma])
```

```
        {
```

```
            ma = i;
```

```
        }
```

```
    }
```

```
    cout << ma;
```

```
    return 0;
```

```
}
```

数组计数法求众数，通过排序后归纳最中间的数或者最中间两数的下标，求中位数。

```
#include<bits/stdc++.h>//10-1179 javacn
using namespace std;
// 由于每个数都在 0~100 之间，因此可以用数组计数法，统计每个数出现的次数。
//a 数组存储读入的每个数
//b 数组表示每个读入的数出现的次数
int n, a[10010], b[110], s = 0;
int main()
{
    cin>>n;
    for (int i = 1; i <= n; i++)
    {
        cin>>a[i];
        b[a[i]]++;
        s = s + a[i];
    }

    cout<<fixed<<setprecision(2)<<s * 1.0 / n<<" "; // 平均

    int ma = 0;        // 众数：计数数组的最大数下标
    for (int i = 1; i <= 100; i++)
    {
        if(b[i] > b[ma]) ma = i;
    }

    cout<<ma<<" ";    // 众数

    // 中位数
    sort(a+1, a+n+1);
    // 归纳得知：n 是奇数，中间数下标是 n/2,
    //n 是偶数，中间数下标是 n/2 和 n/2+1
    if(n % 2 == 1)    cout<<fixed<<setprecision(1)<<a[n/2+1]*1.0;
    else              cout<<fixed<<setprecision(1)<<(a[n/2]+a[n/2+1])/2.0;
    return 0;
}
```

数组计数法统计每个人的得票，再求得票最多的人的编号，也就是统计数组的最大数的下标（注意有多个人当选，要求编号最小的，也就是最左侧的最大数下标）。

```
#include<bits/stdc++.h>//11-2005 javacn
using namespace std;
int a[210], n, x, ma, m;
int main()
{
    cin>>n>>m;    //n 人参加竞选， m 张投票
    int i;
    for (i=1; i<=m; i++)
    {
        cin>>x;
        a[x]++; // 数组计数法
    }
    ma=0;
    for (i=1; i<=n; i++)
    {
        if (a[i]>a[ma])
        {
            ma=i;
        }
    }
    cout<<ma;
    return 0;
}
```

数组计数法统计哪些数出现过，再循环找出没有出现过的数。

```
#include<bits/stdc++.h>//12-2029 javacn
using namespace std;
int a[100010], x;
int main()
{
    int n;
    cin>>n;
    for (int i=1; i<=n-2; i++)
    {
        cin>>x;
        a[x]=1;// 标记出现过的数
    }

    for (int i=1; i<=n; i++)
    {
        if(a[i]==0)
            cout<<i<<" ";
    }
    return 0;
}
```

# 连续性元素

```
#include <bits/stdc++.h> //1-1740  javacn
```

```
using namespace std;
```

```
/*
```

思路：将数组排序，统计连续相同的数有几个

1. 遇到每个数都 c++

2. c++ 之后判断连续相同的数是否结束

结束标准：到了最后一个数 || 当前数 != 下一个数

```
*/
```

```
int n, a[1010], c = 0;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    for (int i = 0; i < n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    sort(a, a+n);    // 排序
```

```
    for (int i = 0; i < n; i++)    // 统计连续相同的数出现的次数
```

```
    {
```

```
        c++;    // 统计连续相同的数出现的次数
```

```
        if (i == n - 1 || a[i] != a[i + 1])    // 判断连续相同的数是否结束
```

```
        {
```

```
            cout<<a[i]<<" "<<c<<endl;
```

```
            c = 0;    // 计数器清零
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```

#include <bits/stdc++.h> //2-1886 javacn
using namespace std;
int n, a[110], c; // 统计连续相等的数出现的次数

int main()
{
    cin>>n;
    for(int i = 0; i < n; i++)
    {
        cin>>a[i];
    }

    for(int i = 0; i < n; i++)
    {
        c++; // 连续相等的数至少有一个
            // 如果连续相等的数结束了：两个标记

        if(i == n - 1 || a[i] != a[i+1]) // 到了最后，或者当前数 != 下一个数
        {
            if(c > 1) cout<<a[i]<<" "<<c<<endl; //1个数，不输出
            c = 0; // 计数器清零
        }
    }
    return 0;
}

```

```

#include <bits/stdc++.h> //3-1887 连胜王 yexiang
using namespace std;
int a[101];
int main()
{
    int n, i, c = 1;
    int ma; // 打擂台的方式获得连胜的次数
    int mai; // 记录擂台上球队编号
    cin >> n;
    for (i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    mai = a[0]; // 默认是第一个队 // 防止只有 1 个队伍
    ma = 1; // 连胜 1 次

    for (i = 1; i < n; i++)
    {
        if (a[i] == a[i-1])
        {
            c++; // 代表连续胜利
            if (c > ma) // 打擂台看看是否超过
            {
                ma = c;
                mai = a[i];
            }
        }
        else
        {
            c = 1; // 结束连胜
        }
    }
    cout << mai;
    return 0;
}

```

```

#include <bits/stdc++.h> //4-2173 连续的最长偶数序列 javacn
using namespace std;
int n, a[110];
int c=0, ma=0; //c 记录当前连续偶数的个数, ma 记录最大的个数
int main()
{
    cin>>n;

    for(int i = 0; i < n; i++)        cin>>a[i];        // 循环记录下输入的 n 个整数

    for(int i=0; i<n; i++)        // 遍历整个数组
    {

        if(a[i]%2==0)            // 如果是偶数计数器加一
        {
            c++;
            // 判断如果当前是数组中最后一个数或者下一个数是奇数
            // 则说明连续偶数的序列结束, 此时的计数器 c 的值为当前连续偶数序列的长度
            if(i==n-1 || a[i+1]%2!=0)
            {

                if(c>ma)            ma=c;            // 用 c 与当前记录的最大个数 ma 做比较

                c=0;            //c 清 0, 为下次计数做准备
            }
        }
    }

    cout<<ma;

    return 0;
}

```

```
#include<bits/stdc++.h>//5-1461 温度统计员 zzb
using namespace std;
int main()
{
    long long n, a[1000000], i, j, t=1, ans=0;//n 是 10^6, a 数组是 10^9 所以用 long
    long。
    cin>>n;
    for (i=0; i<n; i++)
    {
        cin>>a[i];
    }
    for (j=0; j<n; j++)
    {
        if(a[j+1]>a[j])
        {
            t++; // 如果比前一个大, t++。
            if(t>ans)
                ans=t; // 把最大的连续上升天数放在 ans 里。
        }
        else
            t=1; // 否则 t 又变成 1。
    }
    cout<<ans;
    return 0;
}
```

使用连续性元素统计的方法统计连续相同的数出现的次数，然后根据题意计分。  
再使用数组下标计数法，统计出每个队伍的得分；最后统计得分最多的队伍的得分值。

```
#include <iostream> //6-1559    javacn
using namespace std;
int x[110], cnt[110];    //x: 存储每场比赛的胜队 //cnt: 存储每个队伍总得分
int n, c = 0;
int main()
{
    cin >> n >> x[1];    // 第一场胜利的队伍
    for (int i = 2; i <= n; i++)
    {
        x[i] = ((x[i-1]*3703+1047)%n)+1;
    }
    for (int i = 1; i <= n; i++)    // 算分: 知道每个队伍是第几场连胜
    {
        c++;
        if (c <= 3) cnt[x[i]] += c;    // 算分
        else cnt[x[i]] += 3;

        if (i == n || x[i] != x[i+1])    // 判断连胜结束
        {
            c = 0;
        }
    }
    int ma = 0;    // 求 cnt 数组的最大数
    for (int i = 1; i <= n; i++)
    {
        if (cnt[i] > ma)
        {
            ma = cnt[i];
        }
    }
    cout << ma;
    return 0;
}
```

```

#include <bits/stdc++.h> //7-1540 小 X 数字母 javacn
using namespace std;
int n, c=0, mx=0;
string s; // 输入的字符串
int main()
{
    cin>>n;
    cin>>s;

    for (int i=0; i<s.size(); i++) // 遍历这个字符串 s
    {

        if (s[i]=='A') // 如果字符是 A 则计数器 +1
        {
            c++;
        }
        // 如果当前是最后一个字符 或者 下一个字符不是 A
        // 则计数结束, 计数器清 0, 记录下当前是否是最大连续数
        if (i==s.size()-1 || s[i+1] != 'A')
        {
            if (c>mx)
            {
                mx = c;
            }
            c=0;
        }
    }

    cout<<mx;
    return 0;
}

```

```

#include<iostream>//8-2002 投篮 13202828973
#include<algorithm>
#include<cmath>
using namespace std;
char a[1000000];
int b[1000000];
int main()
{
    int n;
    cin>>n;
    for (int i=1;i<=n;i++)
    {
        cin>>a[i];
    }
    int sum=0,k=0;
    /*
    8
    VVTTXW
    3+1 -1 +2=5
    */
    for (int i=1;i<=n;i++)
    {
        if(a[i]=='V') k++; sum++;

        if(i==n || a[i+1]!='V')
        {
            if(k>=3) sum=sum+k-2;
            k=0;
        }

        if(a[i]=='X') sum--;
    }
    cout<<sum;
    return 0;
}

```

使用连续元素统计的思想，统计出连续非素数的数量，本题可以先筛素数来提升效率，筛素数的写法，同学可以自行尝试，这里提供的解法可以过，不过素数存在重复的判断。

```
#include <bits/stdc++.h> //9-1587 javacn
using namespace std;
bool prime(int n) // 定义函数判断素数
{
    if(n <= 1) return false;
    for(int i = 2; i <= sqrt(n); i++)
    {
        if(n % i == 0) return false;
    }
    return true;
}

int n, ma, c = 0;
int main()
{
    cin >> n;

    for(int i = 1; i <= n; i++) // 统计连续非素数的最长长度
    {
        if(!prime(i))
        {
            c++;

            if(i == n || prime(i+1)) // 如果到了最后一个，或者下一个是素数
            {
                ma = max(ma, c);
                c = 0;
            }
        }
    }
    cout << ma;
    return 0;
}
```

# 数组综合

思路:

1. 从每个数开始求连续 4 个数的和。2. 注意: 从倒数第 3 个数开始没有连续的 4 个数, 要加上前面的数。可以采用环形数组的处理方法: 下标 % 数组长度

连续的四个数的下标:  $a[i], a[(i+1)\%n], a[(i+2)\%n], a[(i+3)\%n]$

比如:

$n=8$

下标:

0 1 2 3 4 5 6 7  $8\%8=0$   $9\%8=1\dots$

```
#include <bits/stdc++.h> //1-1163 相加之和最大, 并给出它们的起始位置 javacn
```

```
using namespace std;
```

```
//ma 代表最大的和, r 代表最大和的起始位置
```

```
int a[30], n, s, ma = INT_MIN, r;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    for(int i = 0; i < n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
// 从每个数的位置开始求连续 4 个数的和
```

```
for(int i = 0; i < n; i++)
```

```
{
```

```
    s = a[i]+a[(i+1)%n]+a[(i+2)%n]+a[(i+3)%n];
```

```
    if(s > ma)
```

```
    {
```

```
        ma = s;
```

```
        r = i + 1;
```

```
    }
```

```
}
```

```
cout<<ma<<endl<<r;
```

```
return 0;
```

```
}
```

循环第 1 个数的可能范围，由于第 2 个数是第 1 个数的 2 倍，第 3 个数是第 1 个数的 3 倍，因此第 1 个数的范围最小是 100，最大是 333。

第 1 个数有了，第 2 个和第 3 个数就有了，将 3 个数组合成一个 9 位数，判断这个 9 位数各个位分别用到了数字 1~9，那么这 3 个数就是符合条件的数。

```
#include<bits/stdc++.h> //2-1467 等比数 javacn
using namespace std;
int main()
{
    int x, y, z;
    // 第 1 个 3 位数
    for (int i = 100; i <= 333; i++)
    {
        x = 2 * i; // 第 2 个
        y = 3 * i; // 第 3 个
        z = i * 1000000 + x * 1000 + y; // 9 位数
        int a[10] = {0}; // 如果 z 用到了 1~9 的每个数
        while (z != 0)
        {
            a[z % 10]++;
            z = z / 10;
        }

        bool f = true; // 判断 1~9 是否出现了一次 // 假设都出现了一次
        for (int j = 1; j <= 9; j++)
        {
            if (a[j] != 1)
            {
                f = false;
                break;
            }
        }
        if (f == true) { cout << i << " " << x << " " << y << endl; }
    }
    return 0;
}
```

依次判断  $x \sim y$  之间的每个数是否符合条件，其中判断一个整数是否各个位不同，可以采用短除法拆出这个数的各个位，并用数组计数法统计每一位出现的次数。

```
#include <bits/stdc++.h> //3-1118-1 既生瑜，何生亮! 推荐学习 javacn
```

```
using namespace std;
```

```
bool fun(int n) // 判断某个数是否是 7 位数，且各个位不相等
```

```
{  
    if(n < 1000000 || n > 9999999) return false;  
    // 判断 n 各个位互不相等  
    // 数组计数法，统计 n 的各个位出现的次数  
    int a[20] = {0};  
    while(n != 0)  
    {  
        a[n % 10]++;  
        n = n / 10;  
    }  
    // 判断是否有某个数出现了不止 1 次  
    for(int i = 0; i <= 9; i++)  
    {  
        if(a[i] > 1) return false;  
    }  
    return true;  
}
```

```
int main()  
{  
    int x, y;  
    cin >> x >> y;  
    for(int i = x; i <= y; i++)  
    {  
        if(fun(i*i))  
            cout << i << endl;  
    }  
    return 0;  
}
```

```
#include <bits/stdc++.h>//3-1118-2
```

仅供参考

zhengjia

```
using namespace std;
int a[9999];
int main()
{
    int g, s, b, q, w, sw, bw;
    int x, y, i;
    scanf("%d%d", &x, &y);
    for (int n=x; n<=y; n++)
    {
        i=n*n;
        g=i%10;
        s=i/10%10;
        b=i/100%10;
        q=i/1000%10;
        w=i/10000%10;
        sw=i/100000%10;
        bw=i/1000000%10;
        if (g!=s&&g!=b&&g!=q&&g!=w&&g!=sw&&g!
        =bw&&s!=b&&s!=q&&s!=w&&s!=sw&&s!=bw&&b!
        =q&&b!=w&&b!=sw&&b!=bw&&q!=w&&q!=sw&&q!
        =bw&&w!=sw&&w!=bw&&sw!=bw)
        {
            if (n>=1000&&n<=9999)
            {
                printf("%d\n", n);
            }
        }
    }
    return 0;
}
```

```
#include <bits/stdc++.h>//3-1118-3 仅供参考 leirui
```

```
using namespace std;
```

```
int main() {  
    int x, y;  
    cin>>x>>y;  
    for (int i=x; i<=y; i++)  
    {  
        int t = i*i;  
        int a[105]={0};  
        int index = 1;  
        int x = t;  
        while (x!=0)  
        {  
            a[index] = x%10;  
            index++;  
            x/=10;  
        }  
        index--;  
        int flag = 1;  
        for (int j=1; j<=index; j++)  
        {  
            for (int k=j+1; k<=index; k++)  
            {  
                if (a[j]==a[k])  
                {  
                    flag = 0;  
                    break;  
                }  
            }  
            if (flag ==0 ) { break; }  
        }  
        if (flag ==1) { cout<<i<<"\n"; }  
    }  
    return 0;  
}
```

求从每个数开始连续 m 个数的和，打擂台求最大，再求平均。

```
#include <bits/stdc++.h> //4-1165-1 下载电影 推荐学习 javacn
```

```
using namespace std;
```

```
int n, m;
```

```
int a[110];
```

```
int main()
```

```
{
```

```
    cin >> n >> m;
```

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

```
    {
```

```
        cin >> a[i];
```

```
    }
```

```
    // 求从每个数开始的连续 m 个数的和
```

```
    // 最多循环到倒数第 m 个数才能保证有 m 个数
```

```
    int s;
```

```
    int ma = 0; // 最大的和
```

```
    for (int i = 1; i <= n - m + 1; i++)
```

```
    {
```

```
        // 求连续 m 个数的和
```

```
        s = 0;
```

```
        for (int j = i; j <= i + m - 1; j++)
```

```
        {
```

```
            s = s + a[j];
```

```
        }
```

```
        if (s > ma)
```

```
        {
```

```
            ma = s;
```

```
        }
```

```
    }
```

```
    cout << fixed << setprecision(2) << ma * 1.0 / m;
```

```
    return 0;
```

```
}
```

## 一维数组解法

`#include <iostream>`//4-1165-2 仅供参考 zzy123

`using namespace std;`

`float pddown(int *data, int n, int m)`

`{//m=3 n=6`

`float flag=0;`

`for (int i=0; i<=n-m; i++)`

`{`

`float s=0;`

`for (int j=i; j<i+m; j++)`

`{`

`s+=data[j];`

`}`

`if(flag<s) flag=s;`

`}`

`return flag;`

`}`

`int main()`

`{`

`int a;`

`cin>>a;`

`int data[a];`

`int n;`

`cin>>n;`

`for (int i=0; i<a; i++) cin>>data[i];`

`float s=pddown(data, a, n);`

`printf("%. 2f", s/n);`

`return 0;`

`}`

#include<bits/stdc++.h>//4-1165-3 仅供参考 hasome

```
using namespace std;
int n, m, a[110], i, s, ma=INT_MIN;
int main()
{
    cin>>n>>m;
    for (i=1; i<=n; i++)
    {
        cin>>a[i];
        s=s+a[i];
        // 加 m 个数后，连续和减去最开始的数，这样只需要单循环。
        if (i>=m)
            s=s-a[i-m];
            ma=max(ma, s);
    }
    printf("%.2f", ma*1.0/m);
    return 0;
}
```

#include<bits/stdc++.h>//5-1171-1 数字交换 推荐学习 hasome

```
using namespace std;
int main()
{
    int a[30], n, x1, x2, y1, y2, i, j;
    cin>>n;
    for (i=1; i<=n; i++) cin>>a[i];
    cin>>x1>>x2;
    cin>>y1>>y2;
    for (i=x1, j=y1; i<=x2, j<=y2; i++, j++)
    {
        swap(a[i], a[j]);
    }
    for (i=1; i<=n; i++) cout<<a[i]<<" ";
    return 0;
}
```

将  $s_1 \sim s_2$ ,  $e_1 \sim e_2$ , 这两段的数字, 按顺序交换:

```
#include<bits/stdc++.h> //5-1171-2 数字交换 javacn
```

```
using namespace std;
```

```
int a[30];
```

```
int n;
```

```
int s1, s2, e1, e2, t;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

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

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    cin>>s1>>s2>>e1>>e2;    // 读入 2 段
```

```
    // 将  $s_1 \sim s_2$  和  $e_1 \sim e_2$  之间的数, 实现真正的交换
```

```
    /*
```

```
    i=1 交换: a[s1] 和 a[e1]
```

```
    i=2 交换: a[s1+1] 和 a[e1+1]
```

```
    i=3 交换: a[s1+2] 和 a[e1+2]
```

```
    归纳可知:
```

```
    i 交换: a[s1+i-1] 和 a[e1+i-1]
```

```
    */
```

```
    for (int i = 1; i <= s2-s1+1; i++)
```

```
    {
```

```
        t = a[s1+i-1];
```

```
        a[s1+i-1] = a[e1+i-1];
```

```
        a[e1+i-1] = t;
```

```
    }
```

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

```
    {
```

```
        cout<<a[i]<<" ";
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//6-1173 求子序列的个数 hasome
using namespace std;
int i, c=1, n, a[20];
int main()
{
    cin>>n;
    for (i=1; i<=n; i++)
    {
        cin>>a[i];
    }
    // 如果 a[i] 是转折元素, 则计数器加 1
    for (i=2; i<n; i++)
    {
        if((a[i]>a[i+1] && a[i]>a[i-1]) || (a[i]<a[i+1] && a[i]<a[i-1]))
            c++;
    }
    cout<<c;
    return 0;
}
```

```
#include<bits/stdc++.h>>//7-1215-1    Fish 学数学    liuzhong
```

```
using namespace std;
```

```
int a[200], n, m=0;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        for (int j=1; j<=n-i-1; j++)
```

```
        {
```

```
            if(a[i]>a[i+j])
```

```
            {
```

```
                m++;
```

```
            }
```

```
            else
```

```
            {
```

```
                break;
```

```
            }
```

```
        }
```

```
    }
```

```
    cout<<m<<endl;
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h>//7-1215-2 hasome
```

```
using namespace std;
```

```
int a[210], i, j, n, c;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    for (i=1;i<=n;i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    for (i=n;i>=1;i--)
```

```
    {
```

```
        j=i;
```

```
        while(j<=n)
```

```
        {
```

```
            if(a[i]>a[j])
```

```
            {
```

```
                c++;
```

```
            }
```

```
            j++;
```

```
        }
```

```
    }
```

```
    cout<<c<<endl;
```

```
    return 0;
```

```
}
```

由于 m 和 n 的最大值是 100，按最大值算，能组合出的邮资的最大值  $=100*3+100*5=800$   
用长度为 810 的数组来记录不同邮资的出现

```
#include <bits/stdc++.h> //8-1252 邮票组合 javacn
```

```
using namespace std;
```

```
int m, n, a[810], c = 0; //c 表示不同方案的数量
```

```
int main()
```

```
{
```

```
    cin >> m >> n;
```

```
    int x;
```

```
    // 组合所有的可能
```

```
    //3 分的可能范围
```

```
    for (int i = 0; i <= m; i++)
```

```
    {
```

```
        //5 分的数量范围
```

```
        for (int j = 0; j <= n; j++)
```

```
        {
```

```
            x = 3 * i + 5 * j;
```

```
            if (x != 0)
```

```
            {
```

```
                a[x]++;
```

```
                if (a[x] == 1) c++; // 统计不同方案的数量
```

```
            }
```

```
        }
```

```
    }
```

```
    // 循环可能的邮资范围
```

```
    for (int i = 1; i <= 800; i++)
```

```
    {
```

```
        if (a[i] != 0)
```

```
            cout << i << " ";
```

```
    }
```

```
    cout << endl;
```

```
    cout << c;
```

```
    return 0;
```

```
}
```

思路：通过观察规律，如果有 7 个数，是 10 20 30 40 50 60 70

排队的结果：10 30 50 70 60 40 20

第 1 个数 ~  $n/2+1$  个数，分别是原来数组按从小到大排序的：第 1 3 5 7... 个数

倒数第 1 个数 ~ 倒数  $n/2$  个数值，分别是原来数组的的第 2 4 6... 个数

```
#include <bits/stdc++.h> //9-1283 合唱队形 javacn
using namespace std;
int a[60], n, b[60]; //b 数组存储结果
int main()
{
    cin >> n;
    for (int i = 1; i <= n; i++) cin >> a[i];

    sort(a+1, a+n+1);
    // 赋值前半
    for (int i = 1; i <= n / 2 + 1; i++) b[i] = a[i*2-1];

    int k = 1; //k 用来标记 a 数组下标
    // 赋值后半
    for (int i = n; i > n / 2 + 1; i--)
    {
        b[i] = a[k*2];
        k++;
    }
    // 输出
    for (int i = 1; i <= n; i++)
    {
        cout << b[i] << " ";
    }
    return 0;
}
```

```
#include <bits/stdc++.h> //10-1319 立定跳远成绩求解 fidel
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int x;
```

```
    double sum=0, num=0;
```

```
    for (int i=1; i<=99999999; i++)
```

```
    {
```

```
        cin >> x;
```

```
        if (x==0)
```

```
        {
```

```
            break;
```

```
        }
```

```
        sum+=x;
```

```
        num++;
```

```
    }
```

```
    cout << fixed << setprecision(1);
```

```
    cout << sum/num << " ";
```

```
    sum=0;
```

```
    num=0;
```

```
    for (int i=1; i<=99999999; i++)
```

```
    {
```

```
        cin >> x;
```

```
        if (x==0)
```

```
        {
```

```
            break;
```

```
        }
```

```
        sum+=x;
```

```
        num++;
```

```
    }
```

```
    cout << fixed << setprecision(1);
```

```
    cout << sum/num;
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//11-1326-1 需要安排几位师傅加工零件 liuchunhui001
using namespace std;           // 推荐学习
int n, m, a[105], sum, ct;//sum 代表最少的师傅加工几件, ct 代表最少需要几位师傅
int main()
{
    cin>>m>>n;
    for (int i=1; i<=n; i++)
        cin>>a[i];
    sort(a+1, a+n+1, greater<int>()); // 师傅的工作效率要大, 所以采取从大到小排序
    for (int i=1; i<=n; i++)
    {
        if (sum<m)
            ct++, sum+=a[i]; // 累加件数
        else
        {
            cout<<ct;
            return 0; // 代表师傅够, 不够就不输出
        }
    }
    cout<<"NO"; // 如果在 for 那里没 return 0; 的话, 这里就会输出 NO
    return 0;
}
```

由于题目要求的是最少需要多少师傅来加工零件，因此我们优先挑选加工能力强的师傅来加工零件。

```
#include <bits/stdc++.h> //11-1326-2 javacn
using namespace std;
bool cmp(int a, int b) // 辅助降序排序的函数
{
    if(a > b) { return true; }
    else { return false; }
}
int a[110], i, s, n, m;
int main()
{
    cin >> m >> n;
    // 读入 n 个师傅的加工能力
    for(i = 1; i <= n; i++) { cin >> a[i]; }
    // 对 n 个师傅的加工能力进行排序
    sort(a+1, a+n+1, cmp);
    // 逐个求和
    for(i = 1; i <= n; i++)
    {
        s = s + a[i];
        // 人数是否足够
        if(s >= m)
        {
            cout << i;
            break;
        }
    }
    // 如果所有师傅都来加工人数也不够
    if(s < m)
    {
        cout << "NO";
    }
    return 0;
}
```

```
#include <bits/stdc++.h>//12-1328 柱状图 fideI2021
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n;
```

```
    int a[1023];
```

```
    cin >> n;
```

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

```
    {
```

```
        cin >> a[i];
```

```
    }
```

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

```
    {
```

```
        cout << i << ":";
```

```
        for (int j=1;j<=a[i];j++)
```

```
        {
```

```
            cout << "*";
```

```
        }
```

```
        cout << endl;
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include <cstdio>//13-1333-1 兴趣班的排班 推荐学习 13202828973
#include <algorithm>// 本题建议自己写答案，写公倍数函数，连续公倍数得出答案
#include <iostream>
using namespace std;
int arr[1000000];
int main()
{
    int n, sum;
    cin>>n;
    for (int i=1; i<=n; i++)
    { //3
        cin>>arr[i]; //3 2 4
    }
    for (int i=1; i<=n; i++)
    { //3
        sum = arr[1]; //3
        if(n == 1)
        {
            printf("%d\n", sum);
            continue;
        }
        for (int i=2; i<=n; i++)
        {
            sum = sum/___gcd(sum, arr[i])*arr[i];
        }
    }
    cout<<sum+1;
    return 0;
}
```

```
#include <bits/stdc++.h>//13-1333-2 fidel2021
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, a[10], d=1;
```

```
    cin >> n;
```

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

```
    {
```

```
        cin >> a[i];
```

```
    }
```

```
    for (int i=1; i<=999999; i++)
```

```
    {
```

```
        d=1;
```

```
        for (int j=1; j<=n; j++)
```

```
        {
```

```
            if (i%a[j]!=0)
```

```
            {
```

```
                d=0;
```

```
            }
```

```
        }
```

```
        if (d==1)
```

```
        {
```

```
            cout << i+1;
```

```
            break;
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h>//14-1400 补发礼物? 13202828973
```

```
using namespace std;
```

```
int a[1000000], b[1000000];
```

```
bool cmp(int x, int y)
```

```
{
```

```
    return x>y;
```

```
}
```

```
int main()
```

```
{
```

```
    int n, k=0;
```

```
    cin>>n;
```

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

```
    {
```

```
        cin>>a[i];
```

```
    }
```

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

```
    {
```

```
        if(a[i]<10 || a[i]%4!=0)
```

```
        {
```

```
            for (int j=0;; j++)
```

```
            {
```

```
                if((a[i]+j)%4==0&&(a[i]+j)>=10)
```

```
                {
```

```
                    a[i]=a[i]+j;
```

```
                    break;
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
    sort(a+1, a+n+1, cmp);
```

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

```
        cout<<a[i]<<" ";
```

```
    return 0;
```

```
}
```

```

#include <bits/stdc++.h>//15-1456 淘淘捡西瓜 javacn
using namespace std;
/*
思路：从最轻的西瓜开始向后尝试 贪心算法
*/
int n, x, a[110], s = 0;//s 代表总共装了多重的西瓜

int main()
{
    cin>>n>>x;
    for (int i = 1;i <= n;i++)
    {
        cin>>a[i];
    }

    sort(a+1, a+n+1); // 排序

    // 从最轻的开始尝试
    int c = 0;// 装了多少个
    for (int i = 1;i <= n;i++)
    {
        // 背包已经装了 s 斤西瓜，如果再加这个西瓜的重量，总和 <=x
        if(s + a[i] <= x)
        {
            c++;
            s = s + a[i];
        }
        else
        {
            break;
        }
    }
    cout<<c;
    return 0;
}

```

用数组来标记哪些点有树，哪些点没有树；先将数组所有值标记为 1（有树），再进行 m 次挖树操作，每次将下标  $x \sim y$  之间所有点标记为 0，最后再统计有多少个 1，就表示还剩多少棵树。

```
#include<bits/stdc++.h>//17-1470  javacn
using namespace std;
int a[10010], l, m, c;
int main()
{
    cin>>l>>m;
    // 将数组所有值都标记为 1
    for (int i = 0; i <= l; i++)
    {
        a[i] = 1;
    }

    int x, y;
    for (int i = 1; i <= m; i++)    //m 次挖树操作
    {
        cin>>x>>y;
        // 将  $x \sim y$  之间的值标记为 0
        for (int j = x; j <= y; j++)
        {
            a[j] = 0;
        }
    }

    // 统计长度为 l 的马路上，剩余的树有几棵
    for (int i = 0; i <= l; i++)
    {
        if(a[i] == 1) c++;
    }

    cout<<c;
    return 0;
}
```

```
#include<bits/stdc++.h>//18-1471 并集与交集 javacn
```

```
using namespace std;
int a[1010], b[1010];
int c[2010], k1 = 0; // 并集
int d[1010], k2 = 0; // 交集
int m, n;
bool f;
int main()
{
    cin>>m>>n;
    for (int i = 0; i < m; i++)
    {
        cin>>a[i];
        c[k1] = a[i]; // 第 1 个数组直接存入并集 c
        k1++;
    }
    for (int i = 0; i < n; i++)
    {
        cin>>b[i]; // 判断 b[i] 在 a 数组是否出现
        f = false; // 假设没有出现
        for (int j = 0; j < m; j++)
        {
            if(a[j] == b[i]) { f = true; break; }
        }
        // 如果 b[i] 在 a 数组没有出现, 说明是并集
        if(f == false) { c[k1] = b[i]; k1++; }
        else { d[k2] = b[i]; k2++; } // 交集
    }
    sort(c, c+k1);
    sort(d, d+k2);
    for (int i = 0; i < k1; i++) cout<<c[i]<<" ";
    cout<<endl;
    for (int i = 0; i < k2; i++) cout<<d[i]<<" ";
    return 0;
}
```

注意：本题 ab 数组本身不重复。

1. a 数组的每个数都可以直接存入并集 c 数组。2. b 数组的每个数，要判断在 a 数组不存在，才能存入并集 c 数组，否则，存入交集数组。

不含自定义函数的解法

`#include <iostream>`//19-1535-1 小 X 与数字 (ten) [推荐学习](#) `javacn`

```
using namespace std;
```

```
long long n, k = 1, a[20];
```

```
long long t, c;
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    while(1)
```

```
    {
```

```
        t = n * k;
```

```
        while(t!=0)
```

```
        {
```

```
            a[t%10]++;
```

```
            t /= 10;
```

```
        }
```

```
        // 统计 1~9 出现了几种
```

```
        c = 0;
```

```
        for(int i = 1; i <= 9; i++)
```

```
        {
```

```
            if(a[i]!=0) c++;
```

```
        }
```

```
        if(c == 9) break;
```

```
        k++;
```

```
    }
```

```
    cout<<k*n;
```

```
    return 0;
```

```
}
```

从 n 开始，用短除法将  $n$   $2*n$   $3*n...$  拆位，用数组计数法统计出现过的数有哪些，直到 1~9 都出现。

```
#include<bits/stdc++.h>//19-1535-2 推荐学习 javacn
```

```
using namespace std;
```

```
int a[20];// 统计 1~9 哪些数出现过
```

```
long long n, k = 1, t;
```

```
bool fun(long long x)// 计算到当前为止是否 1~9 都出现了
```

```
{  
    while(x != 0)// 将 n*k 的各个位拆出来，统计到目前位置是否包含 1~9  
    {  
        a[x%10]++;  
        x=x/10;  
    }  
    for (int i = 1; i <= 9; i++)  
    {  
        if(a[i] == 0) return false;  
    }  
    return true;  
}
```

```
}
```

```
int main()
```

```
{
```

```
    cin>>n;
```

```
    while(true)
```

```
    {
```

```
        t = n * k;
```

```
        k++;
```

```
        if(fun(t)==true)// 一直计算到 1~9 都出现
```

```
        {
```

```
            cout<<t;
```

```
            break;
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```

include <bits/stdc++.h>//19-1535-3 仅供参考 liuzhong
using namespace std;
int Xx(long long int n) {
    bool M[9] = {false, false, false, false, false, false, false, false, false};
    int k = 1;
    long long int N;
    bool flag = true;
    while(flag) {
        N = n * k;
        while(N > 0) {
            switch(N%10) {
                case 1: M[0] = true;break;    case 2: M[1] = true;break;
                case 3: M[2] = true;break;    case 4: M[3] = true;break;
                case 5: M[4] = true;break;    case 6: M[5] = true;break;
                case 7: M[6] = true;break;    case 8: M[7] = true;break;
                case 9: M[8] = true;break;
                default: break;
            }
            N /= 10;
        }
        Flag = true;
        for(int i = 0; i < 9; i++) {
            if(M[i] == false) {    Flag = false;    break;    }
        }
        if(Flag)    {    break;    }
        else    {    k++;    }
    }
    return n*k;
}
int main() {
    long long int n;
    cin >> n;
    cout << Xx(n);
    return 0;
}

```

原来数组的第:  $i \rightarrow 1 \sim n/2$ , 会到新数组的第: 1 3 5 7 9...

原来数组的第:  $n/2+i$  张牌, 回到新数组的第: 2 4 6 8 10...

`#include<bits/stdc++.h> //20-1555-1 洗牌 推荐学习 javacn 本题比较难, 建议后期学`

```
using namespace std;
```

```
int a[1010], b[1010], n, k, c;
```

```
int main()
```

```
{
```

```
    cin>>n>>k>>c;
```

```
    for (int i = 1; i <= n; i++) // 为第 i 张牌赋值
```

```
    {
```

```
        a[i] = i;
```

```
    }
```

```
    for (int i = 1; i <= k; i++) // 模拟洗牌的过程 // 洗牌次数
```

```
    {
```

```
        for (int j = 1; j <= n / 2; j++) // 循环一半, 将洗牌后的结果, 存入 b 数组
```

```
        {
```

```
            b[2*j-1] = a[j];
```

```
            b[2*j] = a[n/2+j];
```

```
        }
```

```
        for (int j = 1; j <= n; j++) // 将 n 张牌, 倒回 a 数组
```

```
        {
```

```
            a[j] = b[j];
```

```
        }
```

```
    }
```

```
    cout<<a[c];
```

```
    return 0;
```

```
}
```

```

#include<iostream>//20-1555-2 推荐学习 13202828973
#include<algorithm>
#include<cmath>
using namespace std;
long long a[1000000],b[1000000],c[1000000];
int main() {
    int n,p,q;
    cin >> n >> p>> q;
    for (int i=1;i<=n;i++) { a[i]=i; } // 总牌张数 n
    for (int j=1;j<=p;j++)// 洗牌 p 次
    {
        int x=1;
        for (int i=1;i<=n/2;i++)// 前半堆
        {
            b[x]=a[i];
            x+=2;
        }
        int y=2;
        for (int z=n/2+1;z<=n;z++)// 后半堆
        {
            b[y]=a[z];
            y+=2;
        }
        for (int i=1;i<=n;i++)
        {
            a[i]=b[i];
        }
    }
    for (int i=1;i<=n;i++)
    {
        if(i==q)
            cout<<b[i];
    }
    return 0;
}

```

## 基础洗牌

```
#include <bits/stdc++.h> //20-1555-3 gengjun
using namespace std;
int n, k, p, a[1010], t[1010], x;

int main()
{
    cin >> n >> k >> p;
    for (int i = 1; i <= n; i++) { a[i] = i; }
    // 洗牌 k 次
    for (int i = 1; i <= k; i++)
    {
        // 前半部分, 从后往前依次插入奇数位
        x = n - 1;
        for (int j = n / 2; j >= 1; j--)
        {
            t[x] = a[j];
            x -= 2;
        }
        // 后半部分, 从前往后依次插入偶数位
        x = 2;
        for (int j = n / 2 + 1; j <= n; j++)
        {
            t[x] = a[j];
            x += 2;
        }
        // 将洗好的牌, 更新到 a 数组中
        for (int j = 1; j <= n; j++)
        {
            a[j] = t[j];
        }
    }
    cout << a[p];
    return 0;
}
```

## 前后端分奇偶下标插入

例如  $n = 6$ ，初始时牌堆中牌的编号为 1, 2, 3, 4, 5, 6。首次洗牌时，会将牌分成 1, 2, 3 和 4, 5, 6 两堆，交叉插入后的结果为 1, 4, 2, 5, 3, 6。再次洗牌，会将牌分成 1, 4, 2 和 5, 3, 6 两堆。交叉插入后得到 1, 5, 4, 3, 2, 6。可见规律是每次洗牌将数组分为前后两部分，然后按照下标奇偶插入成一个新数组。

```
#include<iostream>//20-1555-4 anselxu
#include&ltcstring>
using namespace std;
int a[1010],b[1010];
int main()
{
    int n,k,x;
    cin>>n>>k>>x;
    for(int i=1;i<=n;i++) a[i]=i;
    while(k)
    {
        memcpy(b,a,sizeof(a));//复制数组a到b
        for(int i=1;i<=n/2;i++)
        {
            a[i*2]=b[n/2+i];//后半段插入偶数下标
            a[i*2-1]=b[i];//前半段插入奇数下标
        }
        k--;//计数器
    }
    cout<<a[x];
    return 0;
}
```

二维数组求解:

#include <bits/stdc++.h> //21-1561-1 买木头 javacn 本题比较难, 建议后期学

using namespace std;

int main() {

int n, m, l\_1, s\_1, i, j;

int a[10010][2]; // 记录 n 个商家每个商家的木头的长度和数量

int r; // 最长木头长度

cin >> n >> m >> l\_1 >> s\_1;

a[1][1] = l\_1; // 第一个商家木头长度和数量

a[1][2] = s\_1;

int max = a[1][1]; // 最长木头的长度

for (i = 2; i <= n; i++)

{

a[i][1] = ((a[i - 1][1] \* 37011 + 10193) % 10000) + 1; // 木头长度

a[i][2] = ((a[i - 1][2] \* 73011 + 24793) % 100) + 1; // 木头数量

if (a[i][1] > max) { max = a[i][1]; }

}

int c; // 在每个长度下, 各个供应商能供多少根木头

for (i = max; i >= 1; i--)

{

c = 0;

for (j = 1; j <= n; j++) // 循环每个供应商

{

c += a[j][1] / i \* a[j][2];

}

if (c >= m) // 如果根数够

{

r = i;

break;

}

}

cout << r << endl;

return 0;

}

二分求解:

```
#include <bits/stdc++.h> //21-1561-2 javacn
using namespace std;
const int N = 10010;
int len[N], cnt[N];
int n, m;
bool check(int mid) // 检验切割长度为 mid, 能否切出 m 根以上的木头
{
    int c = 0;
    for (int i = 1; i <= n; i++)
    {
        c += len[i]/mid*cnt[i];
    }
    return c >= m;
}
int main()
{
    cin >> n >> m >> len[1] >> cnt[1];
    int l = 1, r = len[1], mid;

    for (int i = 2; i <= n; i++) // 递推出每个供货商的木头长度和数量
    {
        len[i] = ((len[i-1]*37011+10193)%10000)+1;
        cnt[i] = ((cnt[i-1]*73011+24793)%100)+1;
        r = max(len[i], r);
    }
    while (l <= r) // 二分可能的长度
    {
        mid = l + r >> 1;
        if (check(mid)) l = mid + 1; // 如果长度为 mid 能切出 >=m 个木头 // 放长
        else r = mid - 1;
    }
    cout << l-1;
    return 0;
}
```

```

#include<bits/stdc++.h>//22-1563-1 植树 javacn
using namespace std;
long long a[110]; // 存储每个点的种树时间
long long s; // 种树时间的和
long long x, y, z;
int m, n;
int main()
{
    cin>>m>>n;
    long long i, j;
    // 初始化
    for (i = 0; i <= m; i++) a[i] = 16;
    // 读入 n 种不同的土壤
    for (i = 1; i <= n; i++)
    {
        cin>>x>>y>>z; // 从 x 到 y 之间的种树时间是 z
        for (j = x; j <= y; j++)
        {
            a[j] = z;
        }
    }
    // 计算 5 的倍数的点的种树时间的和
    for (int i = 0; i <= m; i++)
    {
        if (i % 5 == 0)
        {
            s = s + a[i];
        }
    }
    cout<<s * 2; // 道路两旁都植树，所以乘以 2
    return 0;
}

```

1. 正常情况下种一棵树需要 16 分钟

2. 道路每隔 5 米种一棵树

3. 种树的时间有时会和正常情况不一样

4. 道路起始位置为 0 (下标为 0 有树的, 下标为 m 也是有树的)

5. 道路的两边都要植树

思路:

1. 将所有的点值都设置为 16

2. 将读入的道路范围, 设置为读入的种树时间

3. 将 5 的倍数的点, 求和 \* 2

```
#include<bits/stdc++.h>//22-1563-2  remedy1314
```

```
using namespace std;
```

```
int n,m,l,s,k,r; int a[10010];
```

```
int main()
```

```
{
```

```
    cin>>n>>m;
```

```
    for (int i=0;i<=n;i++)
```

```
        a[i]=16;
```

```
    while (m--)
```

```
    {
```

```
        cin>>l>>r>>k;
```

```
        for (int i=l;i<=r;i++)
```

```
            a[i]=k;
```

```
    }
```

```
    for (int i=0;i<=n;i+=5)
```

```
        s+=a[i];
```

```
    cout<<s*2;
```

```
    return 0;
```

```
}
```

#include<bits/stdc++.h>//23-1564-1 洗牌2 推荐学习 13202828973 本题难, 后期学

using namespace std;

long long a[1000000], b[1000000], c[1000000];

int main() {

int n, m;

cin>>n>>m;

for (int i=1; i<=n; i++) { cin>>a[i]; } // 总牌张数

int f, r, t;

for (int i=1; i<=m; i++)

{

cin>>f;

if (f==0)

{

int x=1; // 弹牌1次

for (int i=1; i<=n/2; i++) // 前半堆

{

b[x]=a[i];

x+=2;

}

int y=2;

for (int z=n/2+1; z<=n; z++) // 后半堆

{

b[y]=a[z];

y+=2;

}

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

{

a[i]=b[i];

}

}

```
else
{
    cin>>r>>t;
    // 切牌 1 次
    int v=1;
    for (int i=r; i<=t; i++)
    {
        b[v]=a[i];
        v++;
    }
    for (int i=1; i<=r-1; i++)
    {
        b[v]=a[i];
        v++;
    }
    for (int i=t+1; i<=n; i++)
    {
        b[v]=a[i];
        v++;
    }
    for (int i=1; i<=n; i++)
    {
        a[i]=b[i];
    }
}
for (int i=1; i<=n; i++)
{
    cout<<b[i]<<" ";
}

return 0;
}
```

#include<bits/stdc++.h>//23-1564-2 洗牌 2 推荐学习 javacn 本题比较难, 建议后期学

```
using namespace std;
```

```
/*
```

归纳规律:

洗牌:

原来  $1 \sim n/2$  位置之间的数, 会存入新数组的  $i*2+1$  位置

原来  $n/2+1 \sim n$  位置之间的数, 会存入新数组的  $i*2$  位置

切牌:

将指定区间内的元素放到数组的开始位置

```
*/
```

```
int a[100], b[100];
```

```
int n, m;
```

```
// 将 b 数组的内容拷贝回 a 数组
```

```
void fun()
```

```
{  
    for (int i = 1; i <= n; i++) { a[i] = b[i]; }  
}
```

```
int main()
```

```
{  
    cin >> n >> m;  
    for (int i = 1; i <= n; i++) { cin >> a[i]; }  
    // m 次操作
```

```
    int order, x, y;
```

```
    int k; // 代表 b 数组长度
```

```
    while (m--)
```

```
    {
```

```
        cin >> order;
```

```
        // 切牌
```

```
        if (order == 1)
```

```
        {
```

```
            cin >> x >> y;
```

```
            k = 0;
```

```
            // 将  $x \sim y$  之间的拷贝
```

```
            // 将  $x \sim y$  之间的拷贝
```

```

// 将 x~y 之间的拷贝
for (int i = x; i <= y; i++)
{
    k++;
    b[k] = a[i];
}
// 将剩余的拷贝
for (int i = 1; i <= n; i++)
{
    if (i < x || i > y)
    {
        k++;
        b[k] = a[i];
    }
}
// 将 b 数组拷贝回去
fun();
}
else
{
    for (int i = 1; i <= n / 2; i++)
    {
        b[i*2-1] = a[i]; // a 前半
        b[i*2] = a[i+n/2]; // a 后半
    }
    fun();
}
}
for (int i = 1; i <= n; i++)
{
    cout << a[i];
    if (i != n) cout << " "; // 去除行末空格
}
return 0;
}

```

```

#include <bits/stdc++.h>//25-1806  dragoncatter
using namespace std;
int main()
{
    int n, fdc=0, fdb=0;
    double dc, db;
    cin>>n;
    double a[n+10];
    for (int i=1; i<=n; i++)
    {
        cin>>a[i];
    }
    dc=a[1]-a[2];// 等差的值
    db=a[1]/a[2];// 等比的值
    for (int i=2; i<n; i++)
    {
        if(a[i]-a[i+1]!=dc)
        {
            fdc=1; // 不是等差就标记
        }
        if(a[i]/a[i+1]!=db)
        {
            fdb=1; // 不是等比就标记
        }
    }
    if(fdc==0) { cout<<"dengcha"; }
    else if(fdb==0) { cout<<"dengbi"; }
    else { cout<<"no"; }
    return 0;
}

```

```
#include <bits/stdc++.h> //26-1807 zhengjia
```

```
using namespace std;
```

```
int a[10010]; int n,m;
```

```
int main()
```

```
{
```

```
    bool x=0,y=0;
```

```
    scanf("%d",&n);
```

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

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

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

```
        if(a[i]>a[i+1])
```

```
            x=1;
```

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

```
        if(a[i]<a[i+1])
```

```
            y=1;
```

```
    if(x&& y)
```

```
    {
```

```
        printf("No");
```

```
        return 0;
```

```
    }
```

```
    printf("Good");
```

```
    return 0;
```

```
}
```

```
#include<bits/stdc++.h>//27-1816 fidel2021
```

```
using namespace std;
```

```
int a[11], n, p;
```

```
int main()
```

```
{  
    cin >> n;  
    for (int i=1;i<=n;i++)  
    {  
        cin >> a[i];  
    }  
    for (int i=3;i<=n;i++)  
    {  
        if(a[i]!=a[i-2]+a[i-1])  
        {  
            p=i;  
            break;  
        }  
    }  
    cout << p << endl;  
    cout << 1 << " " << 1 << " ";  
    for (int i=3;i<=n;i++)  
    {  
        a[i]=a[i-2]+a[i-1];  
        cout << a[i] << " ";  
    }  
    return 0;  
}
```

```

#include <bits/stdc++.h> //28-1852-1 javacn
using namespace std;
/*
c 交集: a 有 b 也有
d 并集: a 和 b 合并去重复
e 余集: a 有 b 没有
*/
int a[1010], b[1010], c[1010], d[2010], e[1010];
//n 代表 a 有几个数, m 代表 b 有几个数
//k1~k3 代表: c d e 数组的下标
int n, m, k1, k2, k3;
bool have(int arr[], int n, int x) // 长度为 n 的 arr 数组中是否有元素 x
{
    for(int i = 1; i <= n; i++)
    {
        if(arr[i] == x) return true;
    }
    return false;
}
void print(int arr[], int len) // 输出
{
    for(int i = 1; i <= len; i++)
    {
        cout<<arr[i]<<" ";
    }
    cout<<endl;
}
int main()
{
    cin>>n;
    for(int i = 1; i <= n; i++)
    {
        cin>>a[i];
        //a 数组每个数都是并集的一部分
    }
}

```

```
//a 数组每个数都是并集的一部分
```

```
    k2++;
```

```
    d[k2] = a[i];
```

```
}
```

```
cin>>m;
```

```
for (int i = 1; i <= m; i++)
```

```
{
```

```
    cin>>b[i];
```

```
    if (have(a, n, b[i])) // 交集: b[i] 如果在 a 中出现了, 存入 c 的数组
```

```
    {
```

```
        k1++;
```

```
        c[k1] = b[i];
```

```
    }
```

```
    else // b[i] 在 a 中没有出现, 存入并集
```

```
    {
```

```
        k2++;
```

```
        d[k2] = b[i];
```

```
    }
```

```
}
```

```
sort(c+1, c+k1+1);
```

```
print(c, k1); // 输出交集
```

```
sort(d+1, d+k2+1);
```

```
print(d, k2); // 并集
```

```
for (int i = 1; i <= n; i++) // 求余集: a 有 b 没有
```

```
{
```

```
    if (have(b, m, a[i]) == false)
```

```
    {
```

```
        k3++;
```

```
        e[k3] = a[i];
```

```
    }
```

```
}
```

```
sort(e+1, e+k3+1); // 余集
```

```
print(e, k3);
```

```
return 0;
```

```
}
```

```

#include<bits/stdc++.h>//28-1852-2   tc_ljp6
using namespace std;
int a[1010]={0};
int b[1010]={0};
int c[2020]={0}; // 并集
int main()
{
    int n,m,k,p;
    cin>>n;
    for (int i=0;i<n;i++)
    {
        cin>>k;
        a[i]=k;
        c[i]=k;
    }
    cin>>m;
    for (int i=0;i<m;i++)
    {
        cin>>p;
        b[i]=p;
        c[i+n]=p;
    }
    sort(a, a+n);
    sort(b, b+m);
    sort(c, c+n+m);
    for (int i=0;i<n;i++) // 交集 A B 都有的
    {
        for (int j=0;j<m;j++)
        {
            if (a[i]==b[j])
            {
                cout<<a[i]<<" ";
            }
        }
    }
}

```

```
cout<<endl;
```

```
for (int i=0;i<n+m;i++)// 并集 A 和 B 全部 去掉重复的
```

```
{  
    if(c[i]!=c[i+1]&&i!=n+m-1)  
    {  
        cout<<c[i]<<" ";  
    }  
    if(i==n+m-1)  
    {  
        cout<<c[i]<<endl;  
    }  
}
```

```
for (int i=0;i<n;i++) // A 有 B 没有
```

```
{  
    int q=0;// 此处可以修改，代码可以写的更简洁高效  
    for (int j=0;j<m;j++)  
    {  
        if(a[i]==b[j])  
        {  
            q++;  
        }  
    }  
    if(q==0)  
    {  
        cout<<a[i]<<" ";  
    }  
}  
return 0;  
}
```

#include<bits/stdc++.h>//29-1858-1 数组查找及替换 推荐学习 marsshu

```
using namespace std;
```

```
int a[110];
```

```
int main()
```

```
{
```

```
    // 定义与输入
```

```
    int n, b;
```

```
    cin>>n>>b;
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    // 排序
```

```
    sort(a, a+n);
```

```
    // 找出不可被模除的数
```

```
    for (int i=0; i<n; i++)
```

```
    {
```

```
        if(a[i]%b!=0)
```

```
        {
```

```
            // 在字符编码 A~Z 中输出字母，否则输出数字
```

```
            if(a[i]>='A' && a[i]<='Z')
```

```
            {
```

```
                cout<<char(a[i])<<" ";
```

```
            }
```

```
            else
```

```
            {
```

```
                cout<<a[i]<<" ";
```

```
            }
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```
#include <bits/stdc++.h>//29-1858-2 推荐学习 javacn
```

```
using namespace std;
```

```
int a[110];
```

```
int c[110];
```

```
int main()
```

```
{
```

```
    int n, b, m=0;
```

```
    cin>>n>>b;
```

```
    for (int i=0;i<n;i++)
```

```
    {
```

```
        cin>>a[i];
```

```
    }
```

```
    for (int i=0;i<n;i++)//从a数组中寻找b的倍数，如果不是b的倍数存入c数组
```

```
    {
```

```
        if(a[i]%b!=0)
```

```
        {
```

```
            c[m] = a[i];
```

```
            m++;
```

```
        }
```

```
    }
```

```
    sort(c, c+m); //默认从小到大排序
```

```
    for (int i=0;i<m;i++)
```

```
    {
```

```
        if(c[i]>=65&& c[i]<=90)
```

```
        {
```

```
            cout<<char(c[i])<<" ";
```

```
        }
```

```
        else
```

```
        {
```

```
            cout<<c[i]<<" ";
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

```

#include <iostream>//29-1858-3 liuchunhui001
#include <stdlib.h>
#include <string>
#include <algorithm>
#include <vector>
using namespace std;
int main() {
    int n ,b;
    cin >> n >> b;
    int *p = (int *)malloc((n + 1) * 4);
    int *p1 =(int *)malloc((n + 1) * 4);
    int t = 0;
    for (int i = 1; i <= n; i++) {      cin >> *(p + i);  }
    int j = 1;
    for (int i = 1; i <= n; i++) {
        if (*(p + i) % b != 0) {      *(p1 + j) = *(p + i);  j++;  t++;  }
    }
    int s;
    for (int i = 1;i <= t - 1;i ++){
        for (int k = 1;k <= t - i;k++){
            if (*(p1 + k) > *(p1 + k + 1)) {
                s = *(p1 + k + 1);
                *(p1 + k + 1) = *(p1 + k);
                *(p1 + k) = s;
            }
        }
    }
    for (int i = 1; i <= t; i++){
        if (*(p1 + i) >= 'A' && *(p1 + i) <= 'Z') {
            cout << (char) *(p1 + i) << ' ';
        }
        else {      cout << *(p1 + i) << ' ';      }
    }
    return 0;
}

```

```
#include <bits/stdc++.h>//31-1965 奖学金 fidel2021
```

```
using namespace std;
```

```
int n, a[105];
```

```
int main()
```

```
{  
    cin >> n;  
    for (int i=1; i<=n; i++)  
    {  
        cin >> a[i];  
    }  
    sort (a+1, a+n+1);  
    for (int i=n; i>=1; i--)  
    {  
        cout << a[i] << " ";  
        if(i>n-2)  
        {  
            cout << 500 << endl;  
        }  
        else if(i>n-6)  
        {  
            cout << 300 << endl;  
        }  
        else if(i>n-10)  
        {  
            cout << 100 << endl;  
        }  
        else  
        {  
            cout << 0 << endl;  
        }  
    }  
    return 0;  
}
```