注意题目要求的输出格式，有时候隐含表示数字一定要求几位“%05d”

size是长度属性

\_\_int64 a;//定义一个64位整数，scanf(“%I64d”,&a);printf(“%I64d”,a);cin>>a;

\_\_int64最大能表示16位十六进制数；scanf(“%I64x”,&a); printf(“%I64x”,a);//小写的x表示转化为小写的十六进制abcdef,大写的X表示转化为大写的ABCDEF.

0x表示16进制，f在16进制中就表示十进制数15，这样0xffffff就表示一个16进制的数fffffff，例如：  
      0xf就表示16进制的f，也就是10进制的15.  
      0xff就表示16进制的ff，也就是10进制的（15\*16+15）。  
length是数组长度的属性；

reverse（s.begin(),s.end()）;//数组反转

s[i]-‘0’就是用ASCII码转换为数字

str.insert(1,s);//在原str字符串下标为1的字符前插入字符串s

str.insert(1,“s”); //在原str字符串下标为1的字符前插入字符s

str.erase(1,2);//删除str字符串下标位1开始的2个字符

toupper()将小写字母全部变为大写字母（一次只能一位）a=‘A’+32；

tolower()将大写字母全部变为小写字母（头文件#include <cctype>）

printf(“%s”,s.c\_str());用printf输出字符串s时注意c\_str()

string中find()返回值是字母在母串中的下标位置。失败-1;

string a=s.substr(0,3)将s的第0个字符开始的三个字符转换为a的字符串；

getline(cin,s)//可以读入带有空格的字符串，回车结束。第一次用前有getchar();//这是为了把上次输入中末尾的换行符吸收掉  
stringstream l(s)//将s作为一个输入流输入到一个定义为l的字符流。

set<string> mySet;

string sentence="you are my friend";

stringstream ss(sentence);

string word;

while(ss>>word) //把字符流输入给字符串word，以空格区分开

mySet.insert(word);//先把you插入到mySet中，再把are插入到mySet中，依次插入，最后mySet中有四个字符串；mySet.size()等于4；

#include<cmath>(使用时要加上头文件)：

使用sqrt(x)来求参数x的平方根 abs()取绝对值

round()用来对double类型的变量四舍五入取整

**pow(x,y)//一般要求为double型的，就等于x的y次方**

**sqrt(a,2);//对a开根号，2表示开二次根号**

**continue;//在需要的地方临时结束循环的当前轮回，然后直接进入下一个轮回**

s.substr(pos,n)//string，拷贝包含s中从pos开始的n位字符

s.substr(1)//从下标为1开始拷贝，直到结束(下标为0的不拷贝)

strlen(a);//用来求数组a中字符长度，但字符长度不能等于数组给定长度；

int len=0;while(str[len++] != '\0'); // 数组str的长度为len(最后别忘了len减1)

int len = strlen(str);//strlen()只能用来求char类型的数组中元素的个数

将字符串对象转换为数字 int 的一种有效方法是使用**stoi () 函数**

将字符串对象转换为数字 double 的一种有效方法是使用**stod () 函数**

s+=to\_string(1)//在字符串s后插入1；to\_string(c)//将c所代表的东西转化为字符串

# auto在声明变量时可根据变量初始值的数据类型自动为该变量选择与之匹配的数据类型

while(scanf("%d",&n) != EOF)相当于while(scanf("%d",&n) != EOF)，或"~scanf("%d",&n)"，或"scanf("%d",&n) == 1 "或**while((cin >> a) != 0)//**当没有输入时程序结束

while(scanf("%d",&n),n)或 **while (cin>>n,n)//**输入一个数，这数不为0时进入循环，为0时跳出循环。while(scanf("%d %d",&n,&m),n||m)//n和m同时为0程序结束。while(scanf("%d %d",&n,&m)!=EOF)没有输入n,m时结束

unordered\_map与map的区别是第一个是无序的，mp.first会得到Map中key的有效值，

mp.second会得到Map中value的有效值

set<int> s;//使用s.insert()插入，且插入的元素不允许重复，若是重复则不进行插入，使用s.erase(t)删除元素t。

**vector<int> v;//**使用v.push\_back()插入，使用v.erase(v.begin()+k)删除第k+1个元素，v.begin()指向第一个元素，v.end()指向最后一个元素的下一个地址。

vector<int> v(n,1);//是定义一个v数组，大小为n，全部都设为1.

vector<vector<int>> ve;//定义一个vector类型的二维数组，可以用ve.push\_back(v)往二维数组ve中插入一维数组来插入数据

set<int> s;

s.insert(1);

s.insert(3);

s.insert(2);

set<int>::iterator it;

for(it = s.begin();it != s.end();it++) cout<<\*it<<endl;//iterator迭代器，可以输出1，2，3，(使用set容器插入时会自动从小到大排序)因为该迭代器指向的是 set 容器存储的某个元素，而不是键值对，因此通过 \*it 可以直接获取该迭代器指向的元素的值。

map<int,vector<int>>mp;//把一个int型数映射到vector数组中，即二维数组mp[a][b]

用mp[a].push\_back(i)插入映射的二维数组。

map函数map<char,int> mp; mp[‘c’]=1；//把字符c标记为1

While（cin>>value）//while语句的条件执行输入运算符>>，它负责将数据从cin读到value并返回cin（输出运算符<<和输入运算符>>的调用结果都是返回其左侧运算对象）。

bool cmp(int a,int b){

return a>b; //从大到小排序

}

int gcd(int a,int b){ //求a与b的最大公约数  
 if(b==0) return a;  
 return gcd(b,a%b);  
}

sort(v.begin(),v.end(),cmp); //调用cmp函数将数组v排序

bool check(int m){

for(int i=2;i\*i<=m;i++)

if(m%i==0)

return false; //判断素数

return true;

}

int a;**while（cin >> a)**表示输入整形数据正常时，循环，只要输入的数据不是整型数就会结束循环，如：字母、符号等字符。

printf("%.2f",m);保留两位小数输出

02的意思是如果输出的整型数不足两位，左侧用0补齐。例如：  
int m=1;  
printf("%02d",m);  
1只有一位，左侧加0补齐，输出结果就是01  
如果输出的整型数不少于两位，则不用补。例如：  
int m=100;  
printf("%02d",m);  
输出结果是100

int find(int x) {

if (x == fa[x]) return x;

return fa[x] = find(fa[x]); //并查集的find()函数

}

for (int i = 0; i <= n; i++) fa[i] = i; //初始化并查集

while (n--) {

cin >> a >> b;

int fa1 = find(a);

int fa2 = find(b);

if (fa1 != fa2) fa[fa1] = fa2; //并查集的合并

}

for(int i=0;i<n;i++)//对数组a进行从大到小的冒泡排序

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

if(a[j]>a[j + 1])

swap(a[j],a[j+1]);

for(int i=0;i<(sum-sum%k);i=i+k)

reverse(list+i,list+i+k);//每k个一反转

for(int j=0;j+i<n;j=j+i) sort(s.begin()+j,s.begin()+i+j);

if(n%i) sort(s.begin()+n/i\*i,s.begin()+n);//对数组s进行i路归并排序

scanf("%d%\*d%d",&a,&b,&c);

printf("a=%d，b=%d，c=%d",a,b,c);

输入：12 34 56

输出：a=12，b=56，c=0

相当于：忽略掉第二个输入34(它本身)

scanf("%.\*d%d%d",&a,&b,&c);

printf("a=%d，b=%d，c=%d",a,b,c);

输入：12 34 56

输出：a=12，b=56，c=0

相当于：忽略掉第二个输入34(它的下一个)

所以a和第一个输入匹配，b和第三个输入匹配，

中间那个输入被忽略了，所以c也就没有匹配上。  
unsigned int是32位无符号整数

memset(数组名a,值0,sizeof(数组名a));//将数组a的元素全部赋值为0

strcat(字符数组1，字符数组2);//把字符数组2接到字符数字1后面

const int n=1e5+10;//即把n定义为一个定值10的5次方+10即为100010

string s = “test”;

for(char c : s) //一次输出字符串s的各个字符，同理其他的也可以

cout << c;

vector<string> ss(n);

for(string s :: ss) //一次输出vector数组ss的各个字符串

cout << s << endl;

栈

s.push(e); //从栈顶将元素入栈，即是插入

s.pop(); //将栈顶元素出栈，即是删除

s.top(); //读取栈顶元素，从此栈顶元素依然存在

s.empty(); //判断队列是否为空，空返回true

string str = “abcdefg”;

str.pop\_back(); //将字符串最后一个字符删除

ListNode\* dummy = new ListNode(0, head); //头指针前加一个虚拟节点

str = “12345”;

char ch; // at(i),i就是想要获取的字符的下标，函数返回值为指定的字符

ch = str.at(0);

cout<<ch; //ch=='1';

map<string, vector<string>> mp;

vector<vector<string>> ans;

for(auto it = mp.begin(); it != mp.end(); ++it)

ans.emplace\_back(it->second);

emplace\_back()和push\_back()一样的是都是在容器末尾添加一个新的元素进去，但是emplace\_back()在效率上相比较于push\_back()有了一定的提升。

第一行声明一个图mp，string是这个图的key，vector<string>是其value；

it 表示的是图mp的第一整个元素；

it->first 表示的是这个元素的key的值；

it->second 表示的是这个元素的value的值。

n&1就是——判断n是否为奇数

vector<vector<int>> table(size1, vector<int>(size2, 0)); //创建一个size1 \* size2的二维数组，初始值赋值为0；

v.insert(v.begin()+4, elem);//在指定位置,例如在第五个元素前插入一个elem元素

int getlength(ListNode\* head) { //获取链表长度

int length = 0;

while(head) {

length++;

head = head->next;}

return length;

}

arr.front();返回的是第一个元素的引用。 //arr是定义的一个数组

arr.back();返回的的是最后一个元素的引用。

vector的成员assign方法，负责分配新的内容至vector中，以代替现有的内容并相应的修改其size。

str.find(i) != str.end() //说明找到了

用find这个函数，去找str这个序列中的i元素，如果序列中所找的这个元素不存在，就会返回end().

int maxnum = \*max\_element(nums.begin(), nums.end()); //查找数组nums中最大值

int position = max\_element(nums.begin(), nums.end()) - nums.begin();//查找数组nums中最大值的下标。

copy(s.begin(), s.end(), nums.begin());//将数组s中的元素复制给数组nums

^ //异或运算符 两个位相同为0，相异为1

set\_name.count(element) 统计set容器中element元素出现次数

isdigit( ) 函数可以用来判断字符是否为数字。当判断的字符是数字时，函数返回1~9的非零值，当判断的字符不是数字时，函数返回0.

priority\_queue<pair<int, int>> pq; 声明了一个优先队列 pq，其中存储的元素是 pair<int, int> 类型的对象。每个 pair<int, int> 对象都包含两个整数值，作为队列中的一个元素。

优先队列是一种特殊的队列，它的元素按照一定的优先级顺序进行排序和访问。默认情况下，优先队列是按照元素的第一个值（即 pair 中的第一个 int）进行排序，且排序顺序为从大到小（大顶堆），再插入时自动排序。

插入方法：

pq.push(pair<int, int>(1，2));// 插入一个 pair<int, int>，值为 (1, 2)

pq.push(make\_pair(10, 1)); // 插入一个 pair<int, int>，值为 (10, 1)

pq.push({5, 2}); // 插入一个 pair<int, int>，值为 (5, 2)

pair<int, int> topElement = pq.top(); // 获取队首元素，但不删除

int a = topElement.first, b = topElement.second;//a为元素第一个值。

pq.pop(); // 弹出队首元素

priority\_queue<node> q;//创建一个node类型的优先队列

struct node {

int val, x, y;

bool operator < (const node& other)const{

return val > other.val;

}

};// 结构体中重载的 < 运算符是用来定义结构体对象之间的比较规则.return val > other.val;：表示如果当前对象的 val 大于另一个对象 other 的 val，则认为当前对象小于 other 对象；换句话说，即按照 val 的降序进行比较。在p.push\_back();时自动按照val值降序排序。

static bool cmp() //注意力扣中排序想要自定义要这样写