计算思维选题：**[ACM International Collegiate Programming Contest, JUST Collegiate Programming Contest (2018)](https://codeforces.com/gym/101853)**

网址：<https://codeforces.com/gym/101853>

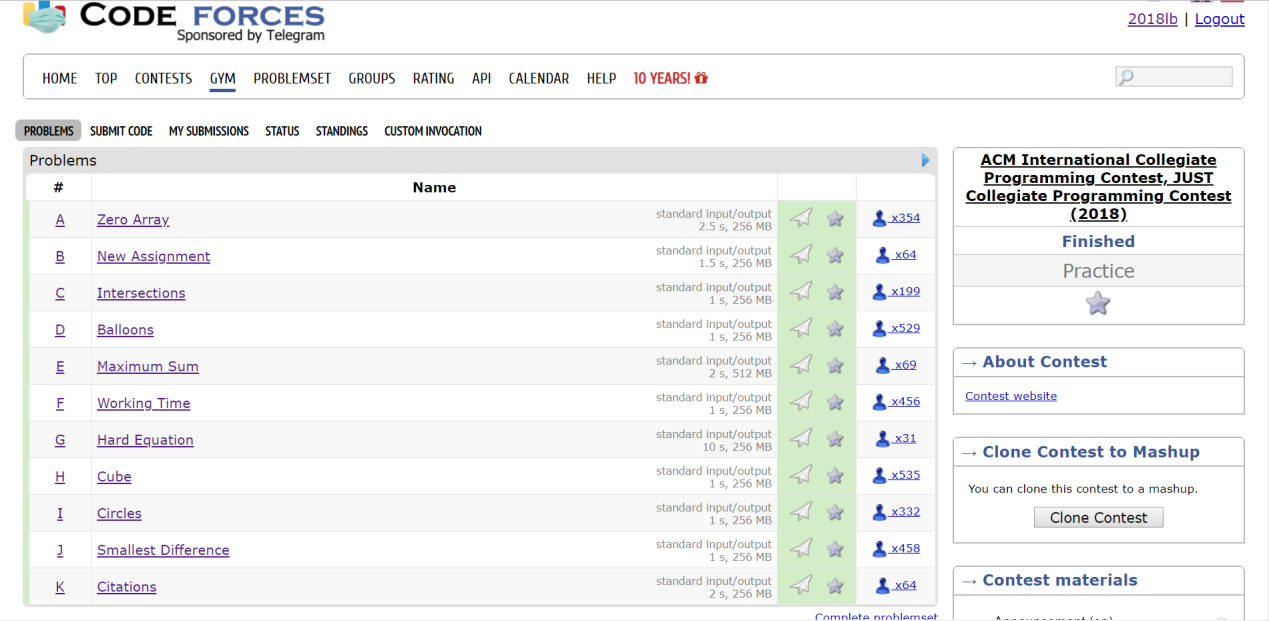
姓名：李斌

学号：20183661148

专业：软件工程

Email：781369524@qq.com

电话：13306091892



题目列表与完成情况

以下每道题通过计算思维问题解析，模式识别，问题形式化表达，抽象等方法进行解答完成。

**代码另附**在目录code的文件下。可执行文件则在目录debug下。

代码测试交由codeforces网站专业数据进行代码运行及测试。

**A. Zero Array**

time limit per test：2.5 seconds

memory limit per test：256 megabytes

input：standard input

output：standard output

You are given an array *a* consisting of *n* elements, and *q* queries. There are two types of queries, as follow:

* "1 p v" – An update query asks to change the value at position *p* in array *a* to *v*.
* "2" – A query asks to print the minimum number of required operations to convert array *a* to a zero array.

A zero array is defined as an array which all its elements are zeros. There is only one allowed operation to convert an array to a zero array. At each operation, you can choose a value *x* and subtract it from all non-zero elements in the array, such that no element will be negative after the operation.

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line of each test case consists of two integers *n* and *q* (1 ≤ *n*, *q* ≤ 105), in which *n* is the size of the array *a*, and *q* is the number of queries.

Then a line follows containing *n* elements *a* 1, *a* 2, ..., *a* *n* (0 ≤ *a* *i* ≤ 109), giving the array *a*.

Then *q* lines follow, each line containing a query in the format described in the problem statement. It is guaranteed that the following constraints hold for the first type of queries: 1 ≤ *p* ≤ *n*, 0 ≤ *v* ≤ 109.

The sum of *n* and *q* overall test cases does not exceed 106 for each.

**Output**

For each query of the second type, print the minimum number of required operations to convert array *a* to a zero array. The queries must be answered in the order given in the input.

**知识点列举（模式识别）：**

数组+哈希（离散化）

**问题分析：**

题目大意就是对于一个数组我们可以对数组某一元素进行修改，在若干次修改后查询经过几次操作，能使数组变成0数组，而能进行的操作是对数组所有非零元素都减去X。

经过思考后我们可以将问题转化成统计一个数组里有多少种不为零且不一样的元素种类，由于数据规模大，于是用map类型平衡树实现哈希离散化。

**代码**

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<map>

#include<iostream>

#include<cstring>

using namespace std;

const int M = 1e5 + 100;

map<int, int> A;

int Num[M];

int main(void)

{

int t;

cin >> t;

while (t--)

{

A.clear();

memset(Num, 0, sizeof(Num));

int n, m, tot = 0;

cin >> n >> m;

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

{

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

if (A[Num[i]]++ == 0 && Num[i]) tot++;

}

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

{

int op;

scanf("%d", &op);

if (op == 2)

printf("%d\n", tot);

else

{

int p, v;

scanf("%d%d", &p, &v);

if (--A[Num[p]] == 0 && Num[p]) tot--;

if (A[v]++ == 0 && v) tot++;

Num[p] = v;

}

}

}

}

在codeforces测试平台下代码运行结果：



**B. New Assignment**

time limit per test：1.5 seconds

memory limit per test：256 megabytes

input：standard input

output：standard output

There is a class consisting of *n* students, in which each one has a number representing his/her personality. The teacher gives the students a new assignment and asks them to solve it in groups so that each group can contain two students at most.

Students cannot create groups as they please because the teacher gives the following rules that must be met in order for a group to be valid:

* The group can be composed of one male student, one female student, or male and female students.
* If the number of students in the group is two, these students must share common interests. Two students *i* and *j* share interests if and only if their numbers *a* *i* and *a* *j* share common divisor *d* > 1.
* Since this is a really diverse class, no triple of students share a common interest, therefore all triples *a* *i*, *a* *j*, *a* *k* are co-primes (i.e. *gcd*(*a* *i*, *a* *j*, *a* *k*) ≡ 1).

Your task is to distribute the students into groups such that each student must join exactly one group, and the number of groups is as minimal as possible. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line of each test case contains an integer *n* (1 ≤ *n* ≤ 104), in which *n* is the number of students in the class.

Then a line follows containing *n* integers *a* 1, *a* 2, ..., *a* *n* (1 ≤ *a* *i* ≤ 106), in which *a* *i* if the personality of the *i* *th* student. Then a line follows containing *n* space-separated characters *p* 1, *p* 2, ..., *p* *n* (), in which *p* *i* is "M" if the *i* *th* student is male, and "F" if she is female.

The sum of *n* overall test cases does not exceed 3 × 105.

**Output**

For each test case, print a single line containing the minimum number of groups that can be formed in the class.

**知识点列举（模式识别）：**

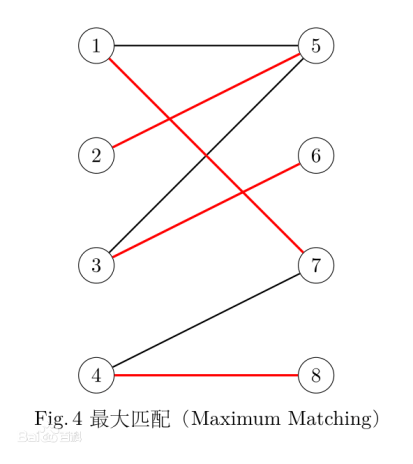
二分图最大匹配+素数筛

**问题分析：**

题目中所描述的问题大意是：一个班要进行分组，如果同学i和同学j要分为一组要满足下列两种条件：1、同学i和同学j是一男一女；2、ai，aj必须有不为一的公约数。默认以分组的同学不能和其他同学分组，如果一个人无法与其他同学分为一组，那么他只能各自为一组，问一个班最少能分几组。

一个班最少能分成几组，那么我们可以转化为能最多配对几组，因为配对的组数越多那么分组的组数就越少。对于分组的匹配，我们先看条件1，既然分组的要求必须是一男一女，那么我们可以把男生看成一个集合，女生看成另一个集合；而条件2我们可以看成两个集合中元素连接的条件，那么这题就是典型的二分图最大匹配问题。

如下图：1234为男生，5678为女生，图中i和j连线就是ai，aj有不为一的公约数，红色的线表示匹配。那么下图最少的分组就是4组。



**代码：**

#include<cmath>

#include<cstdio>

#include<cstring>

#include<iostream>

#include<algorithm>

#include<vector>

using namespace std;

typedef long long LL;

const int N = 2e6 + 7;

const int ME = 1e6 + 7;

const int M = 1e4 + 7;

const int mod = 998244353;

const int INF = 0x3f3f3f3f;

int n, m, top;

int noprime[N], pcnt, p[N / 2];

vector<int> have[N];

int ar[M];

bool is[M];

struct Hopcroft\_Carp {

static const int N = 20000 + 5;

static const int M = 150000 + 5;

static const int oo = 0x3f3f3f3f;

int n1, n2, res, tot;

int ddx[N], ddy[N], mx[N], my[N];

int que[N << 1];

bool vis[N << 1];

int head[N << 1];

int nex[M], u[M], v[M];

void clear() {

tot = -1;

res = 0;

memset(head, -1, sizeof(head));

memset(vis, 0, sizeof(vis));

}

void add(int x, int y) {

u[++tot] = x;v[tot] = y;

nex[tot] = head[x];

head[x] = tot;

}

bool bfs() {

memset(ddx, -1, sizeof(ddx));

memset(ddy, -1, sizeof(ddy));

res = oo;

int rear = 1, tail = 1;

for (int i = 1;i <= n1;++i) {

if (mx[i] == -1) {

que[++tail] = i;ddx[i] = 0;

}

}

while (rear <= tail) {

int x = que[rear];

if (ddx[x] > res)break;

for (int i = head[x];~i;i = nex[i]) {

if (ddy[v[i]] != -1)continue;

ddy[v[i]] = ddx[x] + 1;

if (my[v[i]] == -1)res = ddy[v[i]];

else {

ddx[my[v[i]]] = ddy[v[i]] + 1;

que[++tail] = my[v[i]];

}

}

rear++;

}

return res != oo;

}

bool dfs(int x) {

for (int i = head[x];~i;i = nex[i]) {

if (vis[v[i]] == 0 && ddy[v[i]] == ddx[x] + 1) {

vis[v[i]] = 1;

if (my[v[i]] != -1 && ddy[v[i]] == res)continue;

if (my[v[i]] == -1 || dfs(my[v[i]])) {

my[v[i]] = x;

mx[x] = v[i];

return true;

}

}

}

return false;

}

int maxMatch() {

int ans = 0;

memset(mx, -1, sizeof(mx));

memset(my, -1, sizeof(my));

while (bfs()) {

memset(vis, 0, sizeof(vis));

for (int i = 1;i <= n1;++i) {

if (mx[i] == -1 && dfs(i))ans++;

}

}

return ans;

}

}Two;

void getprime() {

pcnt = 0;

memset(noprime, 0, sizeof(noprime));

noprime[0] = noprime[1] = 1;

for (int i = 2; i < N; ++i) {

if (!noprime[i])p[pcnt++] = i;

for (int j = 0; j < pcnt && i \* p[j] < N; ++j) {

noprime[i \* p[j]] = 1;

if (i % p[j] == 0)break;

}

}

}

void cal(int t, int id) {

int tmp = (int)sqrt(t\*1.0);

for (int i = 0; i < pcnt && p[i] <= tmp; ++i) {

if (t % p[i] == 0) {

have[p[i]].push\_back(id);

while (t % p[i] == 0) {

t /= p[i];

}

}

if (t == 1)break;

}

if (t > 1)have[t].push\_back(id);

}

int main() {

int tim;

getprime();

scanf("%d", &tim);

while (tim--) {

scanf("%d", &n);

for (int i = 0;i < N - 2;++i)have[i].clear();

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

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

cal(ar[i], i);

}

char s[2];

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

scanf("%s", s);

is[i] = 0;

if (s[0] == 'M')is[i] = 1;

}

Two.clear();

Two.n1 = n; Two.n2 = n;

for (int i = 0;i < N - 2;++i) {

if (have[i].size() <= 1) {

have[i].clear();

continue;

}

int len = have[i].size();

for (int j = 0;j < len;++j) {

for (int k = 0;k < len;++k) {

if (j == k)continue;

if (is[have[i][j]] != is[have[i][k]]) {

Two.add(have[i][j], have[i][k]);

}

}

}

have[i].clear();

}

printf("%d\n", n - Two.maxMatch() / 2);

}

return 0;

}

在codeforces测试平台下代码运行结果：



**C. Intersections**

time limit per test：1 second

memory limit per test：256 megabytes

input：standard input

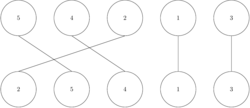
output：standard output

In this problem, you are given two permutations *a* and *b* of *n* numbers, and you need to play a game with them! In this game, you are required to perform the following steps

:

1. For each number *x*, draw a line segment connecting between its positions in the given permutations.
2. Count the number of intersections between the line segments.

For example, let us consider two permutations (5, 4, 2, 1, 3) and (2, 5, 4, 1, 3). The following picture shows the permutations after drawing all line segments. In the picture, the number of intersections between the line segments is 2.



Given the permutations *a* and *b*, your task is to play the game and to count number of intersections between the line segments. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line of each test case contains an integer *n* (1 ≤ *n* ≤ 105), in which *n* is the size of permutations.

Then a line follow containing *n* distinct integers *a* 1, *a* 2, ..., *a* *n* (1 ≤ *a* *i* ≤ *n*), giving the first permutation *a*.

Then a line follow containing *n* distinct integers *b* 1, *b* 2, ..., *b* *n* (1 ≤ *b* *i* ≤ *n*), giving the second permutation *b*.

The sum of *n* overall test cases does not exceed 7 × 105.

**Output**

For each test case, print a single line containing the number of intersections between the line segments.

**知识点列举（模式识别）：**

树状数组求逆序对

**问题分析：**

这问题就是对于两个乱序的队列，将两个队列中相同的数用直线相连，问直线有几个交点。对问题进行分析，发现如果出现交点那么就是出现两个队列的相对逆序的情况。所以这题我们只要统计数列中逆序对的情况就可以了。对于求逆序对，我们可以通过树状数组进行求解。我们把队列一的数一个个插入到树状数组中， 每插入一个数， 统计比队列二中比它出现时间更早的数的个数，则其他数都与这个数构成了逆序对。

**代码：**

#include<cstdio>

#include<algorithm>

#include<cstring>

#include<cstdlib>

#include<iostream>

#define lowbit(x) (x&(-x))

using namespace std;

typedef long long LL;

const int N = 1e5 + 7;

int n, q, tot;

int ar[N], bit[N], br[N];

void add(int x)

{

while (x <= n)

{

bit[x]++;

x += lowbit(x);

}

}

int query(int x)

{

int sum = 0;

while (x)

{

sum += bit[x];

x -= lowbit(x);

}

return sum;

}

int x, y;

int main()

{

int t;

scanf("%d", &t);

while (t--)

{

scanf("%d", &n);

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

{

int x;

scanf("%d", &x);

ar[x] = i;

}

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

{

int x;

scanf("%d", &x);

br[i] = ar[x];

}

long long all = 0;

memset(bit, 0, sizeof(bit));

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

{

int p = br[i];

int tmp = query(p);

all += tmp;

add(p);

}

all = n \* 1LL \* (n - 1) / 2 - all;

printf("%lld\n", all);

}

return 0;

}

在codeforces测试平台下代码运行结果：



**D. Balloons**

time limit per test：1 second

memory limit per test：256 megabytes

inputstandard input

outputstandard output

In many contests, balloons are used to indicate to contestants the general state of the contest. Each time a team solves a problem, a balloon of a specific color is sent to the team and attached on or near their machine. As the contest progresses, the contest floor gradually fills up with a multi-colored display showing how various teams are doing in the contest.

Consider a contest of n problems in which each problem is attached to a specific balloon color. You will be able to see the balloon color of a problem in the contest floor if at least one team solved it during the contest.

You are given the number of accepted solutions on each problem. Your task is to count how many different colors you will be able to see. Can you?

**Input**

The first line contains an integer T (1 ≤ T ≤ 1000), in which T is the number of test cases.

The first line of each test case contains an integer n (1 ≤ n ≤ 20), in which n is the number of problems in the contest.

Then a line follows containing n integers p 1, p 2, ..., p k (0 ≤ p i ≤ 100), in which p i is the number of accepted solutions on the i th problem.

**Output**

For each test case, print a single line containing the number different colors will you be able to see in the contest's floor.4

**知识点列举（模式识别）：**

数组+桶排

**问题分析：**

题目告诉我们在ACM比赛中每道题的解决队伍数，问我们能在比赛场上中能看见的气球颜色总数。很明显，某一题在有队伍解决后我们才能在场上看见其对应颜色的气球。那么我们可以把这道题抽象成对于一组队列，求队列中非零元素的个数。这么抽象的话就这么做就十分明了了。

**代码：**

#include<cstdio>

#include<cstring>

#include<iostream>

using namespace std;

int main(void)

{

int t;

cin >> t;

while (t--)

{

int n, ans = 0, num;

cin >> n;

while (n--)

{

cin >> num;

if (num) ans++;

}

cout << ans <<endl;

}

}

在codeforces测试平台下代码运行结果：



E. Maximum Sum

time limit per test：2 seconds

memory limit per test：512 megabytes

input：standard input

output：standard output

You are given a grid consisting of *n* rows each of which is dived into *n* columns. The rows are numbered from 1 to *n* from top to bottom, and the columns are numbered from 1 to *n* from left to right. Each cell is identified by a pair ( *x*, *y*), which means that the cell is located in the row *x* and column *y*. All cells in the grid contain positive integers.

Your task is to choose a subset of the grid's cells, such that their summation is as maximal as possible, and there are no two adjacent cells in that subset. Two cells are considered adjacent if they are horizontal, vertical, or diagonal neighbors.

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line contains an integer *n* (1 ≤ *n* ≤ 16), in which *n* is the number of rows and columns in the grid.

Then *n* lines follow, each line contains *n* integers, giving the grid. All values in the grid are between 1 and 1000 (inclusive).

**Output**

For each test case, print a single line containing the maximum sum of a subset of the grid's cells. The chosen subset must not contain any adjacent cells.

**知识点列举（模式识别）：**

动态规划+状态压缩

**问题分析：**

题意比较简单，就是对于一个n\*n的矩阵，选中若干个数字，使得选中的数字总和最大，且选中的数字中没有两个数字是相邻的，如果两个数字是水平，垂直或对角线，则认为它们相邻。

一看到n的范围就想到状压dp，这和铺瓷砖很类似，就是一层一层的转移状态。如果直接暴力枚举状态的话复杂度是O(n∗22n) ，肯定要超时的，遇到这种问题，就要想想如果只考虑单行，那么合法状态有多少种，直接提取出单行合法的进行转移，可以预处理出发现只有2000种，那么复杂度就降为O(n\*4\*106)。同时还可以预处理出每行每种状态的权值和这样转移就只需要O(1)的时间。

**代码：**

#include<cstdio>

#include<iostream>

#include<cstring>

#include<algorithm>

using namespace std;

typedef long long LL;

const int N = 10 + 8;

const int ME = 1 << 16;

int n, m;

int ok[ME + 1];

int dp[N][ME + 1];

int ar[N][N];

inline bool adj(int x) { return (x&(x << 1)) || (x&(x >> 1)); }

inline int get\_num(int t, int s)

{

int sum = 0;

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

{

if (s&(1 << i))sum += ar[t][i];

}

return sum;

}

int main()

{

int t;

scanf("%d", &t);

while (t--)

{

scanf("%d", &n);

memset(dp, 0, sizeof(dp));

memset(ok, 0, sizeof(ok));

int aa = 1 << n, tot = 0;

for (int i = 0; i < aa; ++i)

{

if (adj(i) == 0)ok[tot++] = i;

}

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

{

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

{

scanf("%d", &ar[i][j]);

}

}

for (int \_i = 0, i; \_i < tot; ++\_i)

{

i = ok[\_i];

dp[0][\_i] = get\_num(0, i);

}

for (int \_i = 0, i, j; \_i < tot; ++\_i)

{

i = ok[\_i];

for (int \_j = 0; \_j < tot; ++\_j) {

j = ok[\_j];

if ((i&j) || (i&(j << 1)) || (i&(j >> 1)))continue;

dp[1][\_i] = max(dp[0][\_j] + get\_num(1, i), dp[1][\_i]);

}

}

for (int t = 2, i, j; t < n; ++t)

{

for (int \_i = 0; \_i < tot; ++\_i)

{

i = ok[\_i];

for (int \_j = 0; \_j < tot; ++\_j)

{

j = ok[\_j];

if ((i&j) || (i&(j << 1)) || (i&(j >> 1)))continue;

dp[t][\_i] = max(dp[t - 1][\_j] + get\_num(t, i), dp[t][\_i]);

}

}

}

int ans = 0;

for (int \_i = 0; \_i < tot; ++\_i)

{

ans = max(ans, dp[n - 1][\_i]);

}

printf("%d\n", ans);

}

return 0;

}

在codeforces测试平台下代码运行结果：



**F. Working Time**

time limit per test：1 second

memory limit per test：256 megabytes

input standard：input

output standard：output

Working time is the period of time that a person spends at paid labor.

You are given the working record of an employee containing the starting and ending time of work in n days. Your task is to determine if that employee worked sufficiently during these n days. An employee is considered to be worked sufficiently if the total number of working hours during the n days is at least m hours.

**Input**

The first line contains an integer T (1 ≤ T ≤ 300), in which T is the number of test cases.

The first line of each test case contains two integers n and m (1 ≤ n ≤ 100) (0 ≤ m ≤ 2400), in which n is the number of working days, and m is the minimum number of required working hours during these n days.

Then n lines follow, each line contains two strings s and e, in which s is the starting time of work, and e is the ending time of work. Both times are given in the format hh: mm, where hh represents the hours and mm represents the minutes. It is guaranteed that the ending time is not less than the starting time, and both times will be in one day. Times are given in 24-hour time notation.

The sum of n overall test cases does not exceed 2 × 104.

**Output**

For each test case, print "YES" if the employee is considered to be worked sufficiently. Otherwise, print "NO".

**知识点列举（模式识别）：**

数据模拟

**问题分析：**

这题题意相对简单，就是给你一个人n天的工作开始时间和结束时间，问这个人这n天的工作总时间是否有超过要求的时间。这题的做法就照着题目的意思，但有一个小技巧，我们可以把所有小时全部转化为分钟，之后计算结束后再转化为小时，这样可以避免很多我们需要考虑的麻烦。

**代码：**

#include<cstdio>

#include<cstring>

#include<iostream>

using namespace std;

int main(void)

{

int t;

cin >> t;

while (t--)

{

int n, tot\_hour = 0, tot\_minute = 0, num, tim;

cin >> n >> tim;

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

{

int start\_hour, start\_minute, end\_hour, end\_minute;

scanf("%d:%d %d:%d", &start\_hour, &start\_minute, &end\_hour, &end\_minute);

//cin >> start\_hour >> start\_minute >> end\_hour >> end\_minute;

if (end\_minute >= start\_minute) tot\_minute += end\_minute - start\_minute;

else tot\_minute += 60 + end\_minute - start\_minute, end\_hour = end\_hour - 1;

tot\_hour += end\_hour - start\_hour;

}

tot\_hour += tot\_minute / 60;

if (tot\_hour >= tim) printf("YES\n");

else printf("NO\n");

}

}

在codeforces测试平台下代码运行结果：



**G. Hard Equation**

time limit per test：10 seconds

memory limit per test：256 megabytes

input：standard input

output：standard output

Consider the following equation

https://espresso.codeforces.com/4ebbbbf8665baa8cba7179128d458a4c87584bcc.png

Given *a*, *b* and *m*, your task is to find a value *x* that satisfy the equation for the given values. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 500), in which *T* is the number of test cases.

Each test case consists of a line containing three integers *a*, *b* and *m* (0 ≤ *a*, *b* < *m* ≤ 109).

**Output**

For each test case, print a single line containing an integer *x* (0 ≤ *x* ≤ 1017) that satisfy the equation https://espresso.codeforces.com/4ebbbbf8665baa8cba7179128d458a4c87584bcc.png, for the given *a*, *b*and *m*.

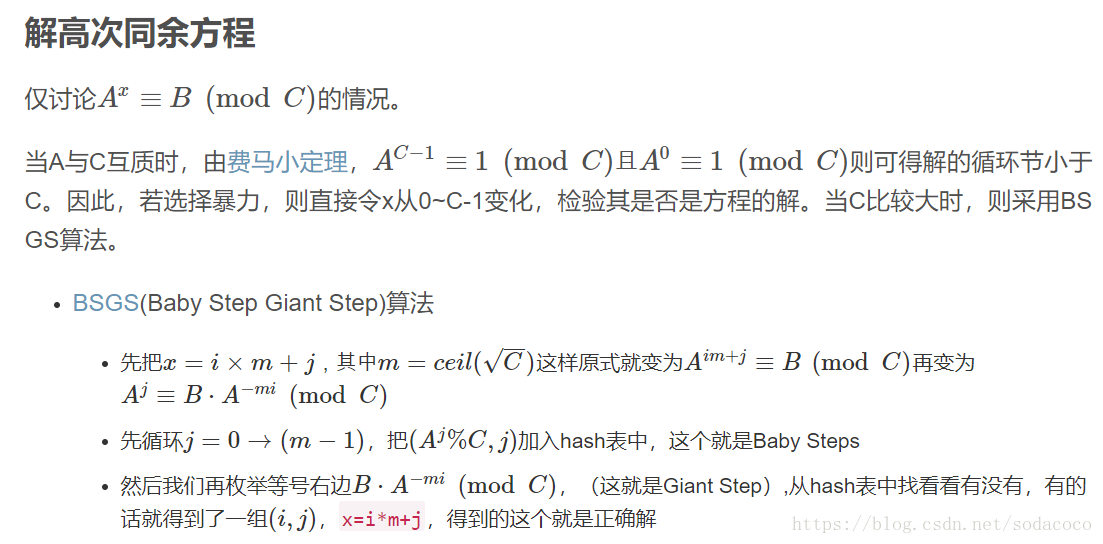
If there are multiple solutions, print any of them. It is guaranteed that an answer always exist for the given input.

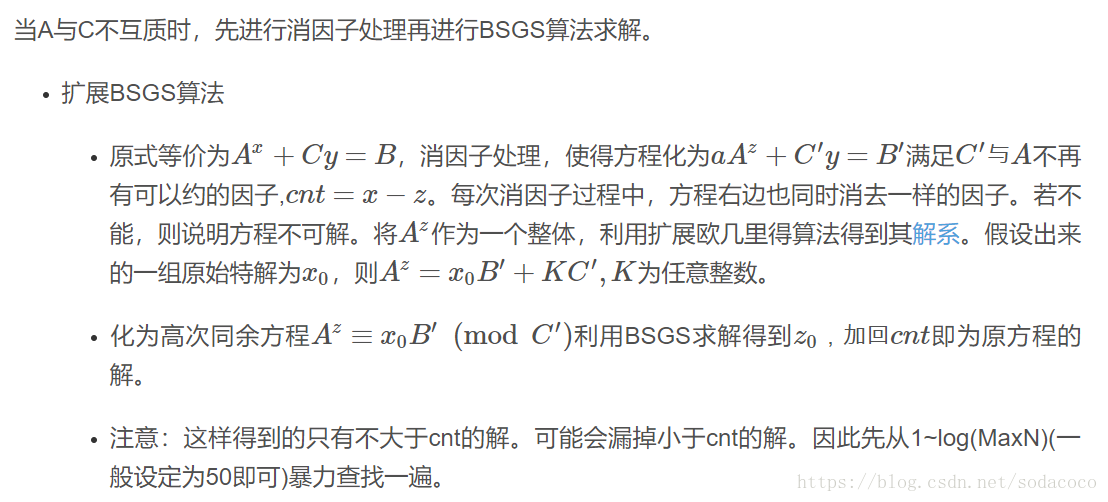
**知识点列举（模式识别）：**

离散对数 + 拓展BSGS

**问题分析：**

经典数学题，对于方程ax ≡ b(mod m)，给我们a，b和m求x。这题很难，超出了我的知识范围，借助网络的力量我才勉强能大概理解怎么解答，好像还是一道EXBSGS的模板题。学习BSGS。





#include<cstdio>

#include<iostream>

#include<cstring>

#include<algorithm>

#include<map>

#include<cmath>

#define ll long long

using namespace std;

inline ll gcd(ll a, ll b) {

return (!b) ? a : gcd(b, a % b);

}

ll work(ll a, ll b, ll mod)

{

a = a % mod; b = b % mod;

if (b == 1) return 0;

ll cnt = 0;

ll t = 1;

for (ll g = gcd(a, mod); g != 1; g = gcd(a, mod))

{

if (b%g) return -1;

mod /= g, b /= g;

t = t \* a / g % mod;

cnt++;

if (t == b) return cnt;

}

map<ll, ll> mp;

int m = ceil(sqrt(1.0 \* mod));

ll e = 1;

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

{

mp[e\*b%mod] = i;

e = e \* a % mod;

}

ll nw = t;

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

{

nw = e \* nw % mod;

if (mp.count(nw))

{

return i \* m - mp[nw] + cnt;

}

}

return -1;

}

int main()

{

int t, a, b, m;

scanf("%d", &t);

while (t--)

{

scanf("%d %d %d", &a, &b, &m);

ll x = work(a, b, m);

if (x == -1);

else printf("%lld\n", x);

}

return 0;

}

在codeforces测试平台下代码运行结果：



**H. Cube**

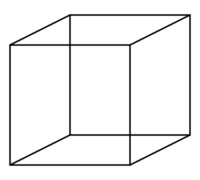
time limit per test：1 second

memory limit per test：256 megabytes

input：standard input

output：standard output

In geometry, a cube is a three-dimensional solid object bounded by six square faces, with three meeting at each vertex. The image bellow is an example of a cube.



You are given the surface area of a cube, and your task is to find the length of that cube's edge. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 1000), in which *T* is the number of test cases.

Each test case consists of a line containing an integer *a* (1 ≤ *a* ≤ 6 × 106), giving the surface area of a cube.

**Output**

For each test case, print a single line containing the length of the given cube's edge.

**It is guaranteed that all answers are integer numbers. Do not print any floating-point values.**

**知识点列举（模式识别）：**

几何数学

**问题分析：**

这题算是最简单的一道题，题目给出正方体的表面积，问我们正方体边的长度。设正方体表面积为X，由于正方体有6个面，则正方体边长易得为X/6的开根号。

**代码：**

#include<cstdio>

#include<cstring>

#include<iostream>

#include<cmath>

using namespace std;

int main(void)

{

int t;

cin >> t;

while (t--)

{

int n, surface\_area;

cin >> surface\_area;

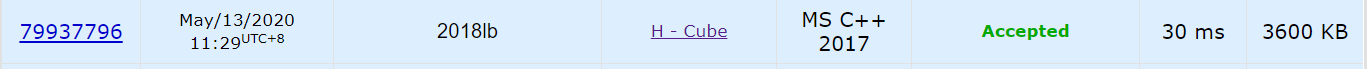
int edge = (int)sqrt(surface\_area/6);

printf("%d\n", edge);

}

}

在codeforces测试平台下代码运行结果：



**I. Circles**

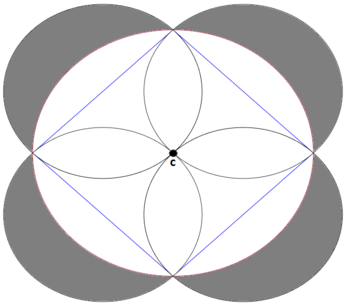
time limit per test：1 second

memory limit per test：256 megabytes

input：standard input

output：standard output

Let *d* be the diameter of a circle with center *c* = ( *a*, *b*). A square has been drawn inside that circle such that its vertices lies at the circle's circumference. Four circles have been drawn on the square's sides such that the diameter of each circle is the side of the square, as shown in the figure below.



Your task is to calculate the shaded area in the figure for a given *d*. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 105), in which *T* is the number of test cases.

Each test case consists of a line containing an three integers *a*, *b*, and *d* ( - 109 ≤ *a*, *b* ≤ 109) (1 ≤ *d* ≤ 109), giving the center and the diameter of a circle.

**Output**

For each test case, print a single line containing shaded area.

Your answer will be considered correct if its absolute or relative error does not exceed 10- 6.

**知识点列举（模式识别）：**

几何数学

**问题分析：**

一道几何数学题目，给我们C点坐标（a，b）和圆的直径*d*，求图中阴影面积。首先读题发现阴影面积与C点坐标无关，然后对图进行几何分析。图中正方形边长为*d*，则其面积为；图中中心白色圆直径为*d*，其面积为；图中正方形外四个半圆的直径都为*d*，其总面积为2 \* = 。

最后求得阴影面积为：正方形外四个半圆的面积减去中心白色圆面积与正方形面积的差 = - （ - ）最后得出阴影面积就为。

**代码：**

#include<cstdio>

#include<cstring>

#include<iostream>

#include<cmath>

using namespace std;

int main(void)

{

int t;

cin >> t;

while (t--)

{

long long a,b,d;

cin >> a >> b >> d;

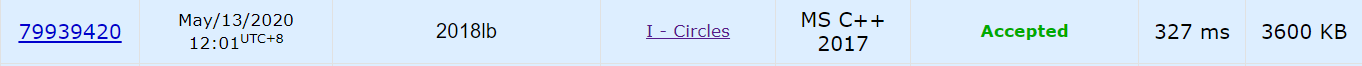
double ans = d/sqrt(2);

printf("%lf\n", ans\*ans);

}

}

在codeforces测试平台下代码运行结果：



**J. Smallest Difference**

time limit per test：1 second

memory limit per test：256 megabytes

input：standard input

output：standard output

You are given an array *a* consists of *n* elements, find the maximum number of elements you can select from the array such that the absolute difference between any two of the chosen elements is  ≤ 1.

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line of each test case consist of an integer *n* (2 ≤ *n* ≤ 104), in which *n* is size of the array *a*

The a line follow containing *n* elements *a* 1, *a* 2, ..., *a* *n* (1 ≤ *a* *i* ≤ 104), giving the array *a*.

**Output**

For each test case, print a single line containing the maximum number of elements you can select from the array such that the absolute difference between any two of the chosen elements is  ≤ 1.

**知识点列举（模式识别）：**

数字统计、数据操作

**问题分析：**

题目大意就是求从数组中可选择的最大元素数，以使任意两个选定元素之间的绝对差为≤1。解决方法就是用数组统计每个元素的个数，枚举一遍寻找最大值记录后最后输出。

**代码：**

#include<cstdio>

#include<iostream>

#include<algorithm>

#include<cstring>

using namespace std;

const int M = 1e4+100;

int Sum[M];

int main(void)

{

int t;

cin >> t;

while(t--)

{

memset(Sum, 0, sizeof(Sum));

int n, ans = 0, Maxx = 0;

cin >> n;

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

{

int a;

cin >> a;

Maxx = max(a, Maxx);

Sum[a]++;

}

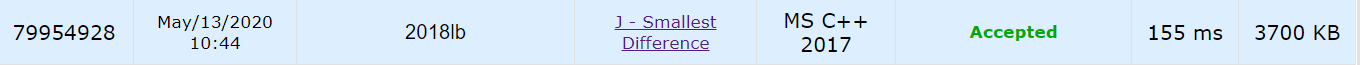
for(int i = 0; i < Maxx; i++) ans = max(ans, Sum[i]+Sum[i+1]);

cout << ans << endl;

}

}

在codeforces测试平台下代码运行结果：



**K. Citations**

time limit per test:2 seconds

memory limit per test:256 megabytes

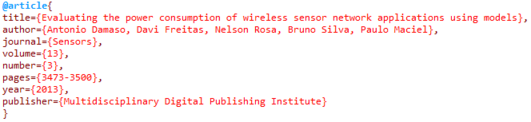
input:standard input

output:standard output

A citation is a reference to a published or unpublished source. This source can be a book or an article. For example, if you are writing a new article discusses the Advanced Encryption Standard (AES) you may add a reference to the AES Proposal: Rijndael, which contains the detailed description of the algorithm from the authors. Each article or book may contain multiple citations that are provided after the end of the content.

One of the most exhausting things is to prepare the format of the citations because each journal has different requirements for the citations section. In this problem, you will help all the researchers in the world by providing an automated method to prepare the citations for them.

Usually, the citations are placed in a file named BibTeX file. This file consists of a list of citations, such that each citation is represented as follow:



Each entry in the BibTeX file is consisting of 8 sections, which are: author, title, journal, year, volume, number, pages, and publisher.

To generate a human-readable format of the citations, the *BibTeX* file is compiled and the information it contains will be used to show the human-readable format of the citations in following structure:

author. title. journal. year;volume(number):pages.

Thus, the above citation will be shown as follow:

An. D, Da. F, Ne. R, Br. S, Pa. M. Evaluating the power consumption of wireless sensor network applications using models. Sensors. 2013;13(3):3473-3500.

Notice that all the information provided in the BibTeX file was used as it is given in the file except for the author section. Each author name will be converted so that the first two letters of author’s first name and the first letter of the author’s last name will be taken and concatenated with a dot and space in between (i.e. ". "), as shown in the example above.

You are given a BibTeX file contains *n* citations, and your task is to convert them to the human-readable format. Can you?

**Input**

The first line contains an integer *T* (1 ≤ *T* ≤ 100), in which *T* is the number of test cases.

The first line of each test case contains an integer *n* (1 ≤ *n* ≤ 1000), in which *n* is the number of citations in the BibTeX file. Then *n*citations follow, giving a BibTeX file. The format of each citation will be as the picture shown in the statement. Please check the sample input for more clarification.

It is guaranteed that no section will be empty and no white spaces will exist outside the curly braces of each section (i.e. {}), and each section will be on a separate line. Also, volume, number, and year sections will contain only integer values. Moreover, the format of the pages section is always "x-y", in which *x* and *y* are the starting and ending pages, respectively. Both *x* and *y* are integer values. Finally, each author's name consists of two space-separated names; the first and the last names. Both names are non-empty strings consisting of English letters such that the first letter of each name is uppercase and the remaining letters are lowercase. The length of author's first name is at least 2 letters. The author's section may contain multiple authors, these authors will be separated by a comma and space (i.e. ", ").

Sections inside each citation are not given in a specific order. The length of each section’s line is at most 120.

**Output**

For each citation, print a single line containing the human-readable format. Each citation must be printed in one line only and without any extra spaces.

**知识点列举（模式识别）：**

字符串处理

**问题分析：**

题目大意就是给你图书信息，让你处理后输出处理后结果。可以抽象成字符串处理，就是简单的模拟操作，对应题目要求对于每一行进行判断后输出即可。

**代码：**

#include<cstdio>

#include<algorithm>

#include<cstring>

#include<iostream>

#include <stdio.h>

using namespace std;

typedef long long ll;

char author[120];

char title[120];

char journal[120];

char year[120];

char volume[120];

char number[120];

char pages[120];

int ai = 0, ti = 0, ji = 0, yi = 0, vi = 0, ni = 0, pi = 0;

int main()

{

int t;

cin >> t;

while (t--)

{

int n;

scanf("%d", &n);

getchar();

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

{

char x[150];

for (int j = 0; j < 10; j++)

{

gets\_s(x);

int len = strlen(x);

int flag = 0;

if (x[0] == 'a')

{

for (int k = 0; k < len; k++)

{

if (!flag&&x[k] >= 'A'&&x[k] <= 'Z')

{

author[ai++] = x[k];

author[ai++] = x[++k];

author[ai++] = '.';

author[ai++] = ' ';

flag = 1;

}

else if (flag&&x[k] >= 'A'&&x[k] <= 'Z')

{

author[ai++] = x[k];

flag = 0;

author[ai++] = ',';

author[ai++] = ' ';

}

}

author[ai - 1] = '\0';

author[ai - 2] = '.';

}

else if (x[0] == 't')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

title[ti++] = x[k];

}

}

title[ti++] = '.';

}

else if (x[0] == 'j')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

journal[ji++] = x[k];

}

}

journal[ji++] = '.';

}

else if (x[0] == 'y')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

year[yi++] = x[k];

}

}

year[yi++] = ';';

}

else if (x[0] == 'v')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

volume[vi++] = x[k];

}

}

}

else if (x[0] == 'n')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

number[ni++] = x[k];

}

}

}

else if (x[0] == 'p'&&x[1] == 'a')

{

for (int k = 0; k < len; k++)

{

if (x[k] == '}')

break;

if (x[k] == '{')

{

flag = 1;

}

else if (flag)

{

pages[pi++] = x[k];

}

}

}

}

cout << author << " " << title << " " << journal << " " << year <<

volume << '(' << number << "):" << pages << '.' << endl;

ai = 0, ti = 0, ji = 0, yi = 0, vi = 0, ni = 0, pi = 0;

memset(author, 0, sizeof(author));

memset(title, 0, sizeof(title));

memset(journal, 0, sizeof(journal));

memset(year, 0, sizeof(year));

memset(volume, 0, sizeof(volume));

memset(number, 0, sizeof(number));

memset(pages, 0, sizeof(pages));

}

}

return 0;

}

在codeforces测试平台下代码运行结果：

