МОШ. Информатика. Все задачи

17 дек 2022 г., 00:00 — 17 янв 2023 г., 23:59

№ 1

7.1e-322 баллов

Лучшая пара

Дан массив, состоящий из n элементов. Требуется найти два различных элемента этого массива, сумма значений которых по модулю p максимальна, т.е. максимальным является остаток от деления суммы значений элементов на p.

Формат входных данных

В первой строке входных данных через пробел расположены два числа: n $(1 < n \leq 10^5)$ и p ;$(1 \leq p \leq 10^9)$. Во второй строке расположены n элементов массива a_i $(0 \leq a_i < p)$.

Формат выходных данных

Выведите два числа — номера элементов массива i, j такие, что сумма $(a_i + a_j) mod\ p$ максимальна. Здесь mod означает операцию взятия остатка от деления. Если искомых ответов несколько, то выведите любой из них.

Система оценки

Все тесты оцениваются независимо.

Примеры

Ввод	Вывод
2 10 1 9	1 2
3 5 2 3 2	1 3
9 10 9 8 7 6 5 4 3 2 1	9 2

Ограничения

Процессорное время: 1 секунда

Память: 256 MB

Код C++

```cpp
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <utility>
#include <cassert>
#include <numeric>
using namespace std;

#define REQUIRE(cond, message) \
    do { \
        if (!(cond)) { \
            std::cerr << message << std::endl; \
            assert(false); \
        } \
    } while (false)

#define forn(i, n) for (int i = 0; i < int(n); ++i)
#define for1(i, n) for (int i = 1; i <= int(n); ++i)
#define forv(i, v) forn(i, v.size())
#define pb push_back
#define mp make_pair
#define all(v) v.begin(), v.end()

typedef vector<int> vi;
typedef long long ll;
typedef vector<ll> vl;
typedef pair<int, int> pii;
typedef vector<string> vs;
typedef long double ld;
typedef pair<double, double> point;

double getX(const point& pt) { return pt.first; }
double getY(const point& pt) { return pt.second; }

void solve()
{
    int n, p;
    cin >> n >> p;
    assert(n > 1);
    vector<pii> a(n);
    forn(i, n) {
        int jump;
        cin >> jump;
        a[i] = mp(jump, i);
    }
    sort(all(a));
    int j = n - 1;
    int besti = -1, bestj = -1;
    forn(i, n) {
        while (i < j && a[i].first + a[j].first >= p) {
            --j;
        }
        if (i < j) {
            if (besti == -1 ||
                    a[besti].first + a[bestj].first <
                    a[i].first + a[j].first) {
                besti = i;
```

```
                bestj = j;
            }
        }
        else {
            break;
        }
    }
    if (besti == -1 || a[besti].first + a[bestj].first <
            (a[n-2].first + a[n-1].first) % p) {
        besti = n - 2;
        bestj = n - 1;
    }
    cout << a[besti].second + 1 << " " << a[bestj].second + 1 <<
endl;
}

int main()
{
    ios_base::sync_with_stdio(false);
    solve();
    return 0;
}

```

№ 2

4.74e-322 баллов

Всего поровну

Двумерная таблица состоит из S строк и S столбцов. Часть её клеток заштрихованы, и в них больше ничего нельзя записать. Свободными остались N клеток.

В них необходимо расставить нули и единицы так, чтобы:

- в каждой строке количество нулей и количество единиц различалось бы не более чем на 1;
- в каждом столбце количество нулей и количество единиц также различалось бы не более чем на 1.

Таким образом, вам надо заполнить оставшиеся N клеток с выполнением указанных правил.

Каждая клетка таблицы определяется двумя числами от 1 до S — номером строки и номером столбца. Свободная клетка номер i расположена в a_i-й строке и в столбце номер b_i. Числа a_i и b_i могут принимать любые значения от 1 до S. В частности, может оказаться так, что в какой-нибудь строке не будет ни одной свободной клетки.

Формат входных данных

Сначала вводятся два целых числа S и N ($1 \leqslant S \leqslant 100000$, $1 \leqslant N \leqslant \min(100000, S^2)$). Далее расположены N пар натуральных чисел (a_i, b_i), не превосходящих S. Гарантируется, что все описанные таким образом свободные клетки различные.

Формат выходных данных

Если искомого способа не существует, выведите слово **Impossible**. Иначе выведите единственную строку из N символов **'0'** и **'1'**. Символ на i-й позиции соответствует значению i-й клетки в той же нумерации, в которой они были перечислены во входных данных.

Система оценки

Тесты состоят из четырёх групп.

0. Тесты 1 и 2. Тесты из условия, оцениваются в 0 баллов.
1. Тесты 3—19. В них $S \leqslant 1\,000$, $N \leqslant 30$. Группа оценивается в 30 баллов, баллы начисляются только при прохождении всех тестов группы.
2. Тесты 20—30. В них $S \leqslant 1\,000$, $N \leqslant 1\,000$. Группа оценивается в 30 баллов, баллы начисляются только при прохождении всех тестов этой и предыдущей групп.
3. Тесты 31—34. Полные ограничения. Каждый тест оценивается в 10 баллов. При этом баллы за тесты этой группы ставятся только тогда, когда программа проходит все тесты групп 1 и 2.

Если программа не проходит хотя бы один из тестов, то на остальных тестах она не проверяется.

Примеры

Ввод	Вывод
2 2 2 1 1 2	00
3 5 1 2 2 3 1 3 2 1 1 1	01001

Ограничения

Процессорное время: 1 секунда

Память: 64 MB

Код C++

```c
#include <stdio.h>
#include <string.h>

enum {
    N = 100000,
    C = 100001,
    V = C * 2,
    M = N * 2 + C * 2
};

int n, m, s, v;
int row[N + 1], col[N + 1], number[N + 1];
int new_[2][C + 1];
int a[M + 1], b[M + 1], p[M + 1], d[M + 1], r[M + 1], color[M + 1];
int l[V + 1];
char chcol[3] = {'?', '0', '1'};

int
add(int p1, int i1, int p2, int i2)
{
    int v1 = new_[p1][i1], v2 = new_[p2][i2];
    ++m;
    a[m] = v1, b[m] = v2;
    r[m] = m + 1;
    ++m;
    a[m] = v2, b[m] = v1;
    r[m] = m - 1;
    ++d[v1];
    ++d[v2];
    return m - 1;
}

void
go(int v, int c)
{
    while (l[v] > 0 && color[l[v]] != 0) l[v] = p[l[v]];
    if (l[v] == 0) return;
    int e = l[v];
    l[v] = p[l[v]];
    color[e] = c;
    color[r[e]] = c;
    go(b[e], 3 - c);
}

int
main(void)
{   int z, i;
    scanf("%d %d", &s, &n);
    ++s;
    for (z = 0; z < 2; ++z) {
        for (i = 1; i <= s; ++i) {
            new_[z][i] = z * s + i;
        }
    }
    v = 2 * s;
    m = 0;
    memset(d, 0, sizeof(d));
    for (i = 1; i <= n; ++i) {
        scanf("%d %d", row + i, col + i);
        number[i] = add(0, row[i], 1, col[i]);
    }
    for (z = 0; z < 2; ++z) {
```

```
            for (i = 1; i < s; ++i) {
                if (d[new_[z][i]] % 2 == 1) add(z, i, 1 - z, s);
            }
        }
        if (d[new_[0][s]] % 2 == 1) {
            add(0, s, 1, s);
        }
        memset(l, 0, sizeof(l));
        for (i = 1; i <= m; ++i) {
            p[i] = l[a[i]];
            l[a[i]] = i;
        }

        memset(color, 0, sizeof(color));
        for (i = 1; i <= v; ++i) {
            if (l[i] != 0) go(i, 1);
        }

        for (i = 1; i <= n; ++i) {
            printf("%c", chcol[color[number[i]]]);
        }
        printf("\n");
    }

```

№ 3

100 баллов

Социальные связи

В школе учатся n школьников. Каждый из них в любой момент времени дружит в социальных сетях хотя бы с половиной других учеников этой школы. Естественно, если один ученик дружит со вторым, то и второй дружит с первым. К сожалению, у школьников очень переменчивое настроение, поэтому часто какие-то пары школьников, начинают или перестают дружить. Также в школе проходят разные мероприятия в рамках которых каждой паре учеников школы необходимо периодически взаимодействовать. Но ученик может взаимодействовать только со своим другом по социальной сети. Если же они друзьями не являются, то взаимодействовать они могут только через одного или несколько посредников.

Все учащиеся в базе данных школы пронумерованы числами от 1 до n, причём таким образом, что если они встанут в круг по порядку номеров (при этом первый ученик стоит рядом с n-м), то у каждого ученика ровно m соседей слева и m справа будут его друзьями, а остальные не будут согласно отношениям дружбы на момент 1 сентября.

В течение года у завуча появился список изменений статуса дружбы учащихся. Обработав данные, он решили каждому изменению сопоставить параметр k_i. Где k_i — наименьшее целое число такое, что любые два ученика могут взаимодействовать, используя не более k_i посредников, после i изменений. Необходимо найти значение данного параметра для всех i от 0 до K.

Некоторые записи завуча могут быть неправильными, поэтому если новый статус дружбы пары учеников совпадает с прежним, то ничего в их статусах и не меняется.

Входные данные

В первой строке три целых числа n, m, $K (1 < n \leq 10^6, 0 < m, K \leq 10^6)$, количество учащихся в школе, количество друзей у каждого из них в начале учебного года, делённое на два, и количество записей по изменению статуса дружбы соответственно.

В следующих K строках находятся записи двух видов $0\ x\ y$— учащиеся с номерами x, y перестали быть друзьями в социальной сети (не гарантируется, что они до этого дружили) и $1\ x\ y$— учащиеся с номерами x, y подружились (не гарантируется, что они до этого не дружили) $(1 \leq x, y \leq n)$.

Выходные данные

В $K + 1$ строке необходимо вывести коэффициент k_i для всех i от 0 до K. Если какие то два учащихся не смогут взаимодействовать даже через посредников, то в соответствующей строке необходимо вывести число n.

Система оценки

Все тесты оцениваются независимо.

Примеры

Ввод	Вывод
6 2 5 1 1 4 0 1 2 1 2 5 1 3 6 1 1 2	1 1 1 1 1 0

Ограничения

Процессорное время: 3 секунды

Память: 256 MB

Код C++

```cpp
#include <bits/stdc++.h>
#define QWE ""
// #define mp make_pair
#define pb push_back
#define sz(a) ((int)a.size())
#define cina(a, n) {a.resize(n); for (auto &i_i: a) cin >> i_i;}
#define couta(a) {for (auto i_i: a) cout << i_i << ' '; cout << '\n';}

using namespace std;

typedef long long ll;
typedef long double ld;
typedef pair<int, int> par;

const int INF = 1e9 + 9;
const ld EPS = 1e-9;
const int MAXN = 1e6;

int n, m, k;
map <par, int> mp;
vector <int> a;
int cnt = 0;

void up(int x) {
    a[x]++;
    if (a[x] == n - 1) {
        cnt++;
    }
}

void dw(int x) {
    if (a[x] == n - 1) {
        cnt--;
    }
    a[x]--;
}

void solve() {
    cin >> n >> m >> k;
        int f = min(2 * m, n - 1);
        a.resize(n, f);
        if (f == n - 1) {
        cnt = n;
        cout << "0\n";
        } else {
            cnt = 0;
            cout << "1\n";
        }
        for (int i = 0; i < k; i++) {
        int t, q, w;
        cin >> t >> q >> w;
        q--, w--;
        if (q > w) {
            swap(q, w);
        }
        par p = make_pair(q, w);
        int z = mp[p];
        if (z == 0) {  // friend status did not change
            if (w - q <= m || q  + n - w <= m) {  // were friends before changes and no changes happened
                mp[p] = 2;
```

```
                z = 2;
            } else {  // were not friends before changes and no changes happened
                mp[p] = 1;
                z = 1;
            }
        }
        if (z == 1) { // giants are now not friends
            if (t == 1) {  // check if they want to become friends
                mp[p] = 2;
                up(q);
                up(w);
            }
        } else if (z == 2) {  // giants are now friends
            if (t == 0) {  // check if they want to stop being friends
                mp[p] = 1;
                dw(q);
                dw(w);
            }
        }
        // couta(a)cout << cnt << ' ';
        if (cnt == n) {
            cout << "0\n";
        } else {
            cout << "1\n";
        }
        }
}

void solveCountInVertex() {
    cin >> n >> m >> k;
        int f = min(2 * m, n - 1);
        a.resize(n, f);
        if (f == n - 1) {
        cnt = n;
        cout << "0\n";
        } else {
            cnt = 0;
            cout << "1\n";
        }
        for (int i = 0; i < k; i++) {
        int t, q, w;
        cin >> t >> q >> w;
        q--, w--;
        if (q > w) {
            swap(q, w);
        }
        if (t == 1) {  // check if they want to become friends
            up(q);
            up(w);
        } else if (t == 0) {  // check if they want to stop being friends
            dw(q);
            dw(w);
        }
        // couta(a);
        if (cnt == n) {
            cout << "0\n";
        } else {
            cout << "1\n";
        }
```

```
        }
}

void solveCountTotal() {
    cin >> n >> m >> k;
        int f = min(2 * m, n - 1);
        ll cnt = 0;
        if (f == n - 1) {
        cnt = (ll)n * (n - 1) / 2;
        cout << "0\n";
        } else {
            cnt = (ll)n * m;
            cout << "1\n";
        }
        for (int i = 0; i < k; i++) {
        int t, q, w;
        cin >> t >> q >> w;
        q--, w--;
        if (q > w) {
            swap(q, w);
        }
        if (t == 1) {  // check if they want to become friends
            cnt++;
        } else if (t == 0) {  // check if they want to stop being
friends
            cnt--;
        }
        // couta(a);
        if (cnt == (ll)n * (n - 1) / 2) {
            cout << "0\n";
        } else {
            cout << "1\n";
        }
        }
}

void validateTest(string s = "s") {
    cin >> n >> m >> k;
    if (s[0] == 't') {
        assert(n > 0);
        assert(k >= 0);
        assert(m * 2 >= n / 2);
        assert(n <= MAXN);
        assert(m <= MAXN);
        assert(k <= MAXN);
    }
        int f = min(2 * m, n - 1);
        a.resize(n, f);
        if (f == n - 1) {
        cnt = n;
        if (s == "s") {
            cout << "1\n";
        }
        } else {
            cnt = 0;
            if (s == "s") {
            cout << "2\n";
        }
        }
        for (int i = 0; i < k; i++) {
        int t, q, w;
        cin >> t >> q >> w;
```

```
        if (s[0] == 't') {
            assert(t == 0 || t == 1);
            assert(q != w);
            assert(1 <= q && q <= n);
            assert(1 <= w && w <= n);
        }
        q--, w--;
        if (q > w) {
            swap(q, w);
        }
        par p = make_pair(q, w);
        int z = mp[p];
        if (z == 0) {  // friend status did not change
            if (w - q <= m || q  + n - w <= m) {  // were friends before changes and no changes happened
                mp[p] = 2;
                z = 2;
            } else {  // were not friends before changes and no changes happened
                mp[p] = 1;
                z = 1;
            }
        }
        if (z == 1) { // giants are now not friends
            if (s == "t+") {
                assert(t == 1);
            }
            if (t == 1) {  // check if they want to become friends
                mp[p] = 2;
                up(q);
                up(w);
            }
        } else if (z == 2) {  // giants are now friends
            if (s == "t+") {
                assert(t == 0);
            }
            if (t == 0) {  // check if they want to stop being friends
                mp[p] = 1;
                dw(q);
                dw(w);
            }
        }
        if (s == "s") {
            if (cnt == n) {
                cout << "1\n";
            } else {
                cout << "2\n";
            }
        }
        }
}

const int N = 10000;
bitset <N> b[N];
bitset <N> used;


par bfs(int v, bool us) {
    queue <par> q;
    q.push(make_pair(0, v));
    used[v] = us;
```

```
    for (int i = 0; i < n - 1; i++) {
        par z = q.front();
        q.pop();
        for (int i = 0; i < n; i++) {
            if (b[z.second][i] == 0 || used[i] == us) {
                continue;
            }
            q.push(make_pair(z.first + 1, i));
            used[i] = us;
        }
    }
    return q.front();
}

int getAns() {
    par ans = make_pair(INT_MIN, INT_MIN);
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            used[j] = 0;
        }
        par p = bfs(i, 1);
        if (p.first > ans.first) {
            ans = p;
        }
    }
    return max(0, ans.first - 1);
}

void solveTL() {
    cin >> n >> m >> k;
    for (int i = 0; i < n; i++) {
        for (int j = 1; j <= m; j++) {
            int z = (i + j) % n;
            b[i][z] = 1;
            b[z][i] = 1;
        }
    }
    cout << getAns() << '\n';
    for (int i = 0; i < k; i++) {
        int t, q, w;
        cin >> t >> q >> w;
        q--, w--;
        b[q][w] = t;
        b[w][q] = t;
        cout << getAns() << '\n';
    }
}

void write1() {
    cin >> n >> m >> k;
    for (int i = 0; i <= k; i++) {
        cout << "1\n";
    }
}

int main()
{
        ios_base::sync_with_stdio(0);
        cout << fixed << setprecision(9);
        cin.tie(0);
        cout.tie(0);
        #ifdef MAKS
```

```
302         {
303                 freopen("input.txt", "r", stdin);
304                 freopen("output.txt", "w", stdout);
305         }
306         #else
307         {
308             /*string s;
309             freopen("input.txt", "r", stdin);
310             cin >> s;
311             freopen(s.c_str(), "w", stdout);
312                 #ifdef QW
313                 {
314                         freopen(QWE".in", "r", stdin);
315                         freopen(QWE".out", "w", stdout);
316                 }
317                 #endif//QW*/
318                 /*freopen("input.txt", "r", stdin);
319                 freopen("output.txt", "w", stdout);*/
320         }
321         #endif//QP
322
323         // solveCountTotal();
324         // solveCountInVertex();
325         solve();
326         // validateTest("t");
327         // solveTL();
328         // write1();
329
330     return 0;
331 }
332
```

№ 4

7.1e-322 баллов

Игра в массив

Так как у вас не всегда есть возможность играть в игры даже на телефоне, игры вы можете придумывать и сами. В том числе с выписанными на бумаге произвольными числами.

Пусть задана последовательность a, состоящая из n натуральных чисел. Игрок может сделать несколько ходов. За один ход игрок может выбрать некоторый элемент последовательности (обозначим выбранный элемент a_k) и удалить его, при этом из последовательности также удаляются все элементы, равные $a_k + 1$ и $a_k - 1$, а другие элементы равные a_k — не удаляются. Описанный ход приносит игроку a_k очков.

Определите, какое максимальное количество очков можно набрать для конкретной последовательности.

Формат входных данных

В первой строке задано целое число n $(1 \le n \le 10^5)$— количество элементов последовательности.

Во второй строке записаны n целых чисел $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^5)$— элементы последовательности.

Формат выходных данных

Выведите целое число— максимальное количество очков, которые может набрать Максим.

Замечание

Рассмотрим третий тестовый пример.

В этом примере предлагаем такую последовательность действий. Первоначально надо выбрать любой элемент, равный 2. В этом случае из последовательности исчезнут все элементы, равные 1 и 3, и мы заработаем 2 балла. Последовательность станет равна $[2, 2, 2, 2]$.

Далее сделаем еще 4 хода, на каждом ходу выберем любой элемент, равный 2. Итого заработаем 10 очков.

Больше 10 очков в этой игре мы заработать не сможем.

Примеры

Ввод	Вывод
2 1 2	2
3 1 2 3	4
9 1 2 1 3 2 2 2 2 3	10

Ограничения

Процессорное время: 1 секунда

Память: 256 MB

Код C++

```cpp
// #pragma comment(linker, "/stack:200000000")
// #pragma GCC optimize("Ofast,no-stack-protector")
// #pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,avx2,tune=native")
// #pragma GCC optimize("unroll-loops")

#include <bits/stdc++.h>

#ifdef PERVEEVM_LOCAL
    #define debug(x) std::cerr << (#x) << ":\t" << (x) << std::endl
#else
    #define debug(x) 238;
#endif

#define fastIO std::ios_base::sync_with_stdio(false); std::cin.tie(0); std::cout.tie(0)
#define NAME "File"

using ll = long long;
using ld = long double;

#ifdef PERVEEVM_LOCAL
    std::mt19937 rnd(238);
#else
    std::mt19937 rnd(std::chrono::high_resolution_clock::now().time_since_epoch().count());
#endif

template<typename T>
bool smin(T& a, const T& b) {
    if (b < a) {
        a = b;
        return true;
    }
    return false;
}

template<typename T>
bool smax(T& a, const T& b) {
    if (a < b) {
        a = b;
        return true;
    }
    return false;
}

const double PI = atan2(0.0, -1.0);
const int INF = 0x3f3f3f3f;
const ll LINF = (ll)2e18;
const int N = 100100;

int cnt[N];
ll dp[N];

void run() {
    int n;
    scanf("%d", &n);

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

```
58          scanf("%d", &cur);
59          ++cnt[cur];
60      }
61
62      dp[1] = cnt[1];
63      for (int i = 2; i < N; ++i) {
64          dp[i] = dp[i - 1];
65          smax(dp[i], dp[i - 2] + 1ll * i * cnt[i]);
66      }
67
68      printf("%lld\n", dp[N - 1]);
69  }
70
71  int main(void) {
72      // freopen(NAME".in", "r", stdin);
73      // freopen(NAME".out", "w", stdout);
74
75      auto start = std::chrono::high_resolution_clock::now();
76      run();
77      auto end = std::chrono::high_resolution_clock::now();
78
79      #ifdef PERVEEVM_LOCAL
80          std::cerr << "Execution time: "
81                    <<
    std::chrono::duration_cast<std::chrono::milliseconds>(end -
    start).count()
82                    << " ms" << std::endl;
83      #endif
84
85      return 0;
86  }
87
```

№ 5

2.37e-322 баллов

Урок физкультуры

На уроке физкультуры первоклассники построились в шеренгу. После объяснения правил выполнения строевых команд последовала команда «налево». При ее исполнении некоторые школьники повернулись налево, а некоторые— направо, перепутав направление поворота. Ученики, которые оказались лицом к лицу со своим соседом, подумали, что совершили ошибку (хотя ошибку совершил кто-то один из школьников пары). Чтобы её исправить, каждый из них быстро повернулся на 180°. Если описанная ситуация затем опять повторялась, то есть какие-то рядом стоящие школьники оказывались лицом друг к другу, то они снова поворачивались на 180°. Эта процедура продолжалась, пока в шеренге оставалась хотя бы одна пара стоящих лицом друг к другу учащихся.

Вам нужно составить программу, которая по расположению школьников сразу после исполнения команды «налево» вычисляет количество пар учащихся, совершивших впоследствии развороты на 180° в соответствии с вышеописанной процедурой.

Формат входных данных

Входные данные состоят из двух строк. В первой строке записано число $N (2 \le N \le 10^5)$— количество школьников в шеренге. Во второй строке содержится последовательность из N символов, каждый из которых может быть либо символом '<', либо символом '>' (символ '<' означает ученика по команде повернувшегося налево, символ '>'—ученика, повернувшегося направо).

Формат выходных данных

Выходные данные должны содержать либо одно число—количество развернувшихся пар, либо число -1, если процесс бесконечен.

Система оценки

Все тесты в этой задаче оцениваются независимо.

Замечание

Расположение школьников	Количество пар, которые должны развернуться	Комментарии
>><<><	2	Расположение школьников сразу после исполнения команды «налево»
><><<>	2	Расположение школьников после первого этапа разворотов
<><><>	2	Расположение школьников после второго этапа разворотов
<<><>>	1	Расположение школьников после третьего этапа разворотов
<<<>>>	Общее количество развернувшихся пар — 7	Конечное расположение школьников

Примеры

Ввод	Вывод
6 >><<><	7

Ограничения

Процессорное время: 1 секунда

Память: 64 MB

Код Python 3

```
n = int(input())
s = input()
ans = 0
counter = list(s).count('<')
for c in s:
    if c == '>':
        ans += counter
    else:
        counter -= 1
print(ans)

```

№ 6

2.37e-322 баллов

Время полёта

Время вылета самолёта из города A по местному времени составляет h_1 часов m_1 минут, а время прилёта в город B по местному времени города B составляет h_2 часов m_2 минут. В обратный рейс из города B он вылетает в h_3 часов m_3 минут по местному времени города B, возможно, уже в другие сутки и прилетает в город A в h_4 часов m_4 минут по местному времени города A. При этом полёт в обе стороны продолжается одно и то же время. Сколько длится полет в одну сторону? Ответ нужно вывести в часах и минутах, округлив его при необходимости до целого числа минут в большую сторону.

Входные данные

В каждой из четырех строк в формате hh:mm записаны времена вылета и прилёта в том порядке, в котором они перечислены в условии; $0 \le h_j < 24, 0 \le m_j < 60$.

Выходные данные

Выведите время полёта в том же формате hh:mm. Если правильрных ответов несколько, выведите минимальный.

Система оценки

Тесты в данной задаче оцениваются независимо.

Примеры

Ввод	Вывод
08:00 10:00 12:00 18:00	04:00

Ограничения

Процессорное время: 2 секунды

Память: 64 MB

Код C++

```cpp
#include <iostream>
#include <vector>
#include <sstream>
#include <string>
#include <cmath>

using namespace std;

#define ll int

bool check(ll mid, ll n) {
        ll ost = n - mid;
        return (ost / 2) + 1 < mid;
}

int main() {
        ll timeA1, timeB1, timeA2, timeB2;
        ll a, b;
        scanf("%d:%d", &a, &b);
        timeA1 = a * 60 + b;
        scanf("%d:%d", &a, &b);
        timeB1 = a * 60 + b;
        scanf("%d:%d", &a, &b);
        timeB2 = a * 60 + b;
        scanf("%d:%d", &a, &b);
        timeA2 = a * 60 + b;
        ll w1 = timeA2 - timeA1;
        ll w2 = timeB2 - timeB1;
        ll x = (timeB1+timeA2-timeA1-timeB2 + (24 * 60)) % (24 * 60);
        if (x == 0) {
                x = 24 * 60;
        }
        ll ans = (x + 1) / 2;
        printf("%02d:%02d\n", (ans / 60), (ans - (ans / 60 * 60)));
        return 0;
}
```

№ 7

2.37e-322 баллов

Сумма подряд идущих

Формулировка этой задачи очень проста: сколько существует различных непрерывных последовательностей подряд идущих натуральных чисел, которые в сумме дают заданное число n?

Входные данные

На вход подается n, не превосходящее 10^{12}.

Выходные данные

Выведите одно натуральное число - ответ на задачу.

Система оценки

Баллы за каждый тест начисляются независимо.

Пояснение к примеру

В примере нужно найти количество непрерывных подпоследовательностей подряд идущих натуральных чисел, дающих в сумме число 15. Таких подпоследовательности 4:

1. $1 + 2 + 3 + 4 + 5 = 15$
2. $4 + 5 + 6 = 15$
3. $7 + 8 = 15$
4. $15 = 15$

Примеры

Ввод	Вывод
15	4

Ограничения

Процессорное время: 2 секунды

Память: 64 MB

Код Python 3

```python
n = int(input())
k = 1
count = 0
while k * (k + 1) // 2 <= n:
    if k % 2:
        if n % k == 0:
            #print(k, n // k - k // 2, n // k + k // 2)
            count += 1
    else:
        if n % k == k // 2:
            #print(k, n // k - k // 2 + 1, n // k + k // 2)
            count += 1
    k += 1
print(count)
```

№ 8

2.37e-322 баллов

Перепутанные арифметические прогрессии

Напомним, что арифметическая прогрессия - это числовая последовательность, каждый член которой, начиная со второго, равен предыдущему, сложенному с постоянным для этой последовательности числом d, называемым разностью прогрессии. Два ученика выбрали свои начальные данные — два натуральных числа: первый член последовательности a_1 и разность d, и приготовили карточки с n первыми элементами своей арифметической прогрессии. Известно, что у них оказались различными как a_1, так и d.

Неизвестно кто виноват в произошедшем, но все карточки перемешались. Ребята так расстроились, что абсолютно забыли свои начальные данные! Помогите им!

Формат входных данных

Первая строка входных данных содержит одно натуральное число n ($5 \le n \le 100000$). Во второй строке находятся $2 \cdot n$ натуральных чисел a_i, записанных в порядке неубывания — элементы двух объединенных последовательностей ($1 \le a_i \le 10^9$). Гарантируется, что входные данные соответствуют условию задачи.

Формат выходных данных

Выведите в порядке возрастания n натуральных чисел — элементы арифметической прогрессии, которая начинается с первого элемента объединенной последовательности.

Система оценивания

Баллы за каждый тест начисляются независимо.

Примеры

Ввод	Вывод
6 3 5 8 11 13 17 18 23 23 28 29 35	3 8 13 18 23 28

Ограничения

Процессорное время: 2 секунды

Память: 64 MB

Код Python 3

```python
def check(A, B):
    if len(A) != n or len(B) != n:
        return False
    d1 = A[1] - A[0]
    d2 = B[1] - B[0]
    flag_1 = True
    for i in range(2, len(A)):
        if A[i] - A[i - 1] != d1:
            flag_1 = False
    flag_2 = True
    for i in range(2, len(B)):
        if B[i] - B[i - 1] != d2:
            flag_2 = False
    if flag_1 and flag_2 and d1 != d2:
        return True
    return False

n = int(input())
L = list(map(int, input().split()))

a1 = L[0]
# 1
d1 = L[1] - L[0]
A = [L[0], L[1]]
next_a = L[1] + d1
B = []
for i in range(2, len(L)):
    if L[i] == next_a and len(A) < n:
        A.append(L[i])
        next_a += d1
    else:
        B.append(L[i])
if check(A, B):
    f1 = True
    ans1 = A
else:
    f1 = False
# 2
d1 = L[2] - L[0]
A = [L[0], L[2]]
next_a = L[2] + d1
B = [L[1]]
for i in range(3, len(L)):
    if L[i] == next_a and len(A) < n:
        A.append(L[i])
        next_a += d1
    else:
        B.append(L[i])
if check(A, B) and L[1] != L[2]:
    f2 = True
    ans2 = A
else:
    f2 = False
# 3
d2 = L[2] - L[1]
B = [L[1], L[2]]
next_b = L[2] + d2
A = [L[0]]
for i in range(3, len(L)):
    if L[i] == next_b and len(B) < n:
        B.append(L[i])
        next_b += d2
```

```
        else:
            A.append(L[i])
    if check(A, B):
        f3 = True
        ans3 = A
    else:
        f3 = False
    if f1 == True and f2 == False and f3 == False:
        print(' '.join(str(i) for i in ans1))
    elif f1 == False and f2 == True and f3 == False:
        print(' '.join(str(i) for i in ans2))
    elif f1 == False and f2 == False and f3 == True:
        print(' '.join(str(i) for i in ans3))
    elif f1 == False and f2 == True and f3 == True and L[2] == L[3]:
        print(' '.join(str(i) for i in ans3))
    else:
        print('Ambiguity')
    #print(f1, f2, f3)
```

№ 9

7.1e-322 баллов

Разложение на множители

Факторизацией называется разложение произвольного натурального числа n в произведение целых положительных чисел, больших 1. Два разложения числа n, которые отличались лишь порядком сомножителей, будем считать одинаковыми. Например, число 12 имеет четыре различные факторизации: 12, $6 \cdot 2$, $4 \cdot 3$ и $3 \cdot 2 \cdot 2$. Так как всех факторизаций может быть много, будем выделять такие, у которых наибольший множитель не превосходит заданного числа m.

Вам необходимо подсчитать количество факторизаций данного числа n с наибольшим множителем, не превосходящим m.

Формат входных данных

В единственной строке записаны два целых числа n и m ($1 < n \leq 10^{11}$, $1 < m \leq n$).

Формат выходных данных

Выведите одно целое число — искомое количество факторизаций.

Система оценивания

Баллы за каждый тест начисляются независимо.

Примеры

Ввод	Вывод
4 2	1
12 12	4
11 10	0

Ограничения

Процессорное время: 3 секунды

Память: 256 MB

Код Python 3

```python
D = dict()
f = []
def Fact(n, m):
    if m == 1 and n > 1:  return 0
    if n == 1:  return 1

    if (n, m) in D:
        return D[(n, m)]
    s = 0
    for el in f:
        if el > m: break
        if n % el == 0:
            s += Fact(n // el, min(el, n//el))
    D[(n, m)] = s
    return s


n, m = map(int, input().split())

i = 1
while i*i < n:
    if n % i == 0:
        f.append(i)
        f.append(n // i)
    i += 1
if i * i == n:
    f.append(i)
f.sort()
f = f[1:]
print(Fact(n, m))

```

№ 10

4.74e-322 баллов

Восстанови порядок

В очереди стояли n человек, у каждого из которых на футболке сзади написан свой собственный номер от 1 до n. На следующий день каждый помнил за кем именно он стоял, но не помнил состояние очереди в целом.

Вам необходимо по этой информации восстановить порядок стоявших в очереди людей.

Формат входных данных

В первой строке записано одно целое число n — количество людей в очереди ($2 \le n \le 5 \cdot 10^5$). Следующие $n - 1$ строк содержат по два разделённых пробелом целых числа a и b — номера на футболках стоявших рядом друг с другом людей, где a — номер на футболке человека, стоявшего перед человеком в футболке с номером b ($1 \le a, b \le n$). Порядок этих пар произвольный.

Формат выходных данных

В единственной строке запишите через пробел n целых чисел — номера на футболках фанатов в обратном порядке очереди.

Если возможных порядков несколько, то выведите любой из них.

Система оценки

Баллы за каждый тест начисляются независимо.

Примеры

Ввод	Вывод
3 3 2 1 3	1 3 2
5 4 1 3 4 1 2 5 3	5 3 4 1 2

Ограничения

Процессорное время: 2 секунды

Память: 256 MB

Код Python 3

```python
def main():
    from sys import stdin, stdout

    n = int(stdin.readline())

    is_first = [True] * (n + 1)
    next = [-1] * (n + 1)

    for i in range(n - 1):
        a, b = list(map(int, input().split()))
        next[a] = b
        is_first[b] = False

    cur = -1
    for i in range(1, n + 1):
        if is_first[i]:
            cur = i

    st = [0] * n
    for i in range(n):
        st[i] = cur
        cur = next[cur]

    stdout.write(" ".join(map(str, st)))


main()

```

№ 11

4.74e-322 баллов

Январская снежинка

Никто не ждал, но в Москву опять пришла зима, и выпал снег. На сегодняшнем уроке весь класс рисует зимний лес. К сожалению, с передачей художественных образов изобразительными методами дела у Тимофея обстоят из рук вон плохо. Но хоть что-то нарисовать нужно, поэтому Тимофей рисует снежинку по клеточкам.

Каждая снежинка имеет восемь лучей, исходящих из общего центра (вообще-то, у обычной снежинки шесть лучиков, но Иван относит себя к авангардистам и отвергает традиционные формы), причем длины лучиков могут быть разными. Испортив несколько листов бумаги, юный художник решил поручить низменную работу по отрисовке своей возвышенной творческой идеи компьютеру. Через пару минут результат был готов.

Докажите Ивану, что ничего особо трудного в этой работе нет — напишите программу, формирующую изображение снежинки с заданными длинами лучей.

Формат входных данных

Единственная строка входных данных содержит восемь натуральных чисел x_i $(1 \leqslant x_i \leqslant 20)$, записанных через пробел — длины лучей снежинки в порядке обхода по часовой стрелке. Первое число соответствует направлению вверх. Обратите внимание, что центральная клетка тоже является частью луча.

Формат выходных данных

Выведите изображение заданной снежинки. Лучи отрисовываются с помощью символов '#' (ASCII код 35) на фоне символов '.' (ASCII код 46). Программа не должна выводить строк или столбцов, состоящих только из символов фона.

Система оценки

Баллы за каждый тест начисляются независимо.

Решения, верно работающие при лучиках одинаковой длины, получат не менее 25 баллов.

Примеры

Ввод	Вывод
1 2 3 4 5 6 7 8	#......... .#........ ..#....... ...#...... ....#..... .....#.... ......#.#.. .#########. ......###.. .....#.#.#. ....#..#..# ...#...#... ..#.......
5 5 5 5 5 5 5 5	#...#...# .#..#..#. ..#.#.#.. ...###...

	######### ...###... ..#.#.#.. .#..#..#. #...#...#

Ограничения

Процессорное время: 2 секунды

Память: 64 MB

Код C++

```cpp
#include <iostream>
#include <cmath>
#include <string>
#include <vector>

using namespace std;

int main()
{
    vector <int> L(8);
    for (int i = 0; i < 8; ++i) cin >> L[i];
    int n = max(max(L[0], L[1]), L[7]) + max(max(L[3], L[4]), L[5]) - 1;
    int m = max(max(L[1], L[2]), L[3]) + max(max(L[5], L[6]), L[7]) - 1;
    char M[n][m];
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < m; ++j)
         M[i][j] = '.';

int center_x = max(max(L[0], L[1]), L[7]) - 1;
int center_y = max(max(L[5], L[6]), L[7]) - 1;
int C[8][2] = {{-1, 0}, {-1, 1}, {0, 1}, {1, 1}, {1, 0}, {1, -1}, {0, -1}, {-1, -1}};
for (int i = 0; i < 8; ++i) {
    int x = center_x;
    int y = center_y;
    for (int j = 0; j < L[i]; ++j) {

        M[x][y] = '#';
        x += C[i][0];
        y += C[i][1];
    }
}
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j)
         cout << M[i][j];
    cout << endl;
    }
    return 0;
}

```

№ 12

4.74e-322 баллов

Магия чисел

Стёпа верит в магию чисел и считает своим лучшим другом число 34, а злейшим врагом — число 43. Сегодня Стёпе для годового отчета нужно составить из набора цифр одно число. Стёпа руководствуется следующими правилами:

1. В составе числа не может быть комбинаций рядом стоящих цифр «43»;
2. Комбинаций «34» в составе числа, наоборот, должно быть как можно больше;
3. Получившееся число должно быть максимально возможным.

Поскольку Стёпа очень занят, работу поручили Вам.

Формат входных данных

В единственной строке входных данных через пробел записаны десять чисел — количества цифр каждого вида в порядке от 0 до 9 в наборе. Гарантируется, что в наборе существует хотя бы одна цифра больше нуля.

Формат выходных данных

Выведите одно натуральное число - ответ на задачу. Гарантируется, что длина выводимого числа не превысит 250.

Система оценки

Баллы за каждый тест начисляются независимо.

Примеры

Ввод	Вывод
0 0 0 2 1 0 0 0 0 0	334
1 0 0 3 4 0 1 0 0 0	346344034

Ограничения

Процессорное время: 2 секунды

Память: 64 MB

Код C++

```cpp
#include <iostream>
#include <string>
#include <cmath>

using namespace std;

int L[10];
int main(){
    for (int i = 0; i < 10; ++i) cin >> L[i];
    string digits = "";
    string ans;
    for (int i = 9; i >= 0; --i)
    if (i != 3 and i != 4)
        for (int j = 0; j < L[i]; ++j)
        digits += to_string(i);


    int count_34 = min(L[3], L[4]);

    if (L[3] == 0 or L[4] == 0) {
       ans = "";
    for (int i = 9; i >= 0; --i)
        for (int j = 0; j < L[i]; ++j)
           ans += to_string(i);
    L[3] = 0;
    L[4] = 0;
   }
  else if (count_34 < digits.size() + 1) {
    // this is the case when the number of digits-splitter is more than needed
    ans = "";
    int i = 0;
    while (i < digits.size() and digits[i] > '2')
        i += 1;

    while (i <= digits.size() and L[3] > 0 and L[4] > 0) {
        digits = digits.insert(i, "34");

        L[3] -= 1;
        L[4] -= 1;
        i += 3;
    }
    while (L[3] > 0 and L[4] > 0) {
        i = digits.find("34");
        i -= 1;
        digits = digits.insert(i, "34");
        L[3] -= 1;
        L[4] -= 1;
    }
    ans = digits;

  }
else
    // this is the case when the number of digits-splitter is fewer than needed or equal
    if (digits.size() > 0) {
        ans = "34";

        for (int j = 0; j < digits.size(); ++j)
            ans = ans + digits[j] + "34";
        L[3] -= digits.size() + 1;
        L[4] -= digits.size() + 1;
```

```
61
62      }
63      else {
64          ans = "";
65          for (int j = 0; j < L[3]; ++j) ans += "3";
66          for (int j = 0; j < L[4]; ++j) ans += "4";
67          L[3] = 0;
68          L[4] = 0;
69      }
70  // now let's place the rest of the fours.
71  if (L[4]){
72
73      if (ans.find("2") >= 0 and ans.find("2") < ans.size()) {
74          string s4 = "";
75          for (int j = 0; j < L[4]; ++j) s4 += "4";
76          ans.insert(ans.find("2"),s4);
77      }
78      else if (ans.find("1") >= 0 and ans.find("1") < ans.size()) {
79          string s4 = "";
80          for (int j = 0; j < L[4]; ++j) s4 += "4";
81          ans.insert(ans.find("1"),s4);
82      }
83      else if (ans.find("0") >= 0 and ans.find("0") < ans.size()) {
84          string s4 = "";
85
86          for (int j = 0; j < L[4]; ++j) s4 += "4";
87
88          ans.insert(ans.find("0"),s4);
89      }
90      else {
91              string s4 = "";
92          for (int j = 0; j < L[4]; ++j) s4 += "4";
93          ans += s4;
94      }
95  }
96
97  // now let's place the rest of the threes.
98
99  if (L[3])
100     if (ans.back() != '4') {
101         string s3 = "";
102         for (int j = 0; j < L[3]; ++j) s3 += "3";
103         int j;
104         for (j = ans.size() - 1; j >=0; --j)
105             if (ans[j] == '4') break;
106         ans = ans.insert(j+2, s3);
107         }
108       else {
109         int j;
110         for (j = ans.size() - 1; j >=0; --j)
111             if (ans[j] == '4') break;
112         int k = j;
113         while (ans[k] == '4')
114             k -= 1;
115         string s3 = "";
116         for (int j = 0; j < L[3]; ++j) s3 += "3";
117         ans = ans.insert(k, s3);
118       }
119     cout << ans << endl;
120     return 0;
121
```

```
122   }
123
```