

# МОШ. Информатика. Все задачи

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

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  $(a_i + a_j) \bmod p$  максимальна. Здесь  $\bmod$  означает операцию взятия остатка от деления. Если искомым 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 МВ

Код

C++

```

1  #include <iostream>
2  #include <cstdio>
3  #include <vector>
4  #include <string>
5  #include <algorithm>
6  #include <set>
7  #include <map>
8  #include <queue>
9  #include <utility>
10 #include <cassert>
11 #include <numeric>
12 using namespace std;
13
14 #define REQUIRE(cond, message) \
15     do { \
16         if (!(cond)) { \
17             std::cerr << message << std::endl; \
18             assert(false); \
19         } \
20     } while (false)
21
22 #define forn(i, n) for (int i = 0; i < int(n); ++i)
23 #define forl(i, n) for (int i = 1; i <= int(n); ++i)
24 #define forv(i, v) forn(i, v.size())
25 #define pb push_back
26 #define mp make_pair
27 #define all(v) v.begin(), v.end()
28
29 typedef vector<int> vi;
30 typedef long long ll;
31 typedef vector<ll> vl;
32 typedef pair<int, int> pii;
33 typedef vector<string> vs;
34 typedef long double ld;
35 typedef pair<double, double> point;
36
37 double getX(const point& pt) { return pt.first; }
38 double getY(const point& pt) { return pt.second; }
39
40 void solve()
41 {
42     int n, p;
43     cin >> n >> p;
44     assert(n > 1);
45     vector<pii> a(n);
46     forn(i, n) {
47         int jump;
48         cin >> jump;
49         a[i] = mp(jump, i);
50     }
51     sort(all(a));
52     int j = n - 1;
53     int besti = -1, bestj = -1;
54     forn(i, n) {
55         while (i < j && a[i].first + a[j].first >= p) {
56             --j;
57         }
58         if (i < j) {
59             if (besti == -1 ||
60                 a[besti].first + a[bestj].first <
61                 a[i].first + a[j].first) {
62                 besti = i;

```

```
63         bestj = j;
64     }
65 }
66 else {
67     break;
68 }
69 }
70 if (besti == -1 || a[besti].first + a[bestj].first <
71     (a[n-2].first + a[n-1].first) % p) {
72     besti = n - 2;
73     bestj = n - 1;
74 }
75 cout << a[besti].second + 1 << " " << a[bestj].second + 1 <<
endl;
76 }
77
78 int main()
79 {
80     ios_base::sync_with_stdio(false);
81     solve();
82     return 0;
83 }
84
```

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  $N$  ( $1 \leq S \leq 100000$ ,  $1 \leq N \leq \min(100000, S^2)$ ). 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  $S \leq 1\,000$ ,  $N \leq 30$ . Группа оценивается в 30 баллов, баллы начисляются только при прохождении всех тестов группы.
2. Тесты 20–30. В них  $S \leq 1\,000$ ,  $N \leq 1\,000$ . 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
<pre>2 2 2 1 1 2</pre>	<pre>00</pre>
<pre>3 5 1 2 2 3 1 3 2 1 1 1</pre>	<pre>01001</pre>

### Ограничения

Процессорное время: 1 секунда

Память: 64 МВ

Код

C++

```

1  #include <stdio.h>
2  #include <string.h>
3
4  enum {
5      N = 100000,
6      C = 100001,
7      V = C * 2,
8      M = N * 2 + C * 2
9  };
10
11 int n, m, s, v;
12 int row[N + 1], col[N + 1], number[N + 1];
13 int new_[2][C + 1];
14 int a[M + 1], b[M + 1], p[M + 1], d[M + 1], r[M + 1], color[M + 1];
15 int l[V + 1];
16 char chcol[3] = {'?', '0', '1'};
17
18 int
19 add(int p1, int i1, int p2, int i2)
20 {
21     int v1 = new_[p1][i1], v2 = new_[p2][i2];
22     ++m;
23     a[m] = v1, b[m] = v2;
24     r[m] = m + 1;
25     ++m;
26     a[m] = v2, b[m] = v1;
27     r[m] = m - 1;
28     ++d[v1];
29     ++d[v2];
30     return m - 1;
31 }
32
33 void
34 go(int v, int c)
35 {
36     while (l[v] > 0 && color[l[v]] != 0) l[v] = p[l[v]];
37     if (l[v] == 0) return;
38     int e = l[v];
39     l[v] = p[l[v]];
40     color[e] = c;
41     color[r[e]] = c;
42     go(b[e], 3 - c);
43 }
44
45 int
46 main(void)
47 { int z, i;
48   scanf("%d %d", &s, &n);
49   ++s;
50   for (z = 0; z < 2; ++z) {
51       for (i = 1; i <= s; ++i) {
52           new_[z][i] = z * s + i;
53       }
54   }
55   v = 2 * s;
56   m = 0;
57   memset(d, 0, sizeof(d));
58   for (i = 1; i <= n; ++i) {
59       scanf("%d %d", row + i, col + i);
60       number[i] = add(0, row[i], 1, col[i]);
61   }
62   for (z = 0; z < 2; ++z) {

```

```
63     for (i = 1; i < s; ++i) {
64         if (d[new_z][i] % 2 == 1) add(z, i, 1 - z, s);
65     }
66 }
67 if (d[new_0][s] % 2 == 1) {
68     add(0, s, 1, s);
69 }
70 memset(l, 0, sizeof(l));
71 for (i = 1; i <= m; ++i) {
72     p[i] = l[a[i]];
73     l[a[i]] = i;
74 }
75
76 memset(color, 0, sizeof(color));
77 for (i = 1; i <= v; ++i) {
78     if (l[i] != 0) go(i, 1);
79 }
80
81 for (i = 1; i <= n; ++i) {
82     printf("%c", chcol[color[number[i]]]);
83 }
84 printf("\n");
85 }
86
```

## 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 решил 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 1 4	1
0 1 2	1
1 2 5	1
1 3 6	1
1 1 2	0

### Ограничения

Процессорное время: 3 секунды

Код

C++

```

1  #include <bits/stdc++.h>
2  #define QWE ""
3  // #define mp make_pair
4  #define pb push_back
5  #define sz(a) ((int)a.size())
6  #define cina(a, n) {a.resize(n); for (auto &i_i: a) cin >> i_i;}
7  #define couta(a) {for (auto i_i: a) cout << i_i << ' '; cout <<
   '\n';}
8
9  using namespace std;
10
11 typedef long long ll;
12 typedef long double ld;
13 typedef pair<int, int> par;
14
15 const int INF = 1e9 + 9;
16 const ld EPS = 1e-9;
17 const int MAXN = 1e6;
18
19 int n, m, k;
20 map <par, int> mp;
21 vector <int> a;
22 int cnt = 0;
23
24 void up(int x) {
25     a[x]++;
26     if (a[x] == n - 1) {
27         cnt++;
28     }
29 }
30
31 void dw(int x) {
32     if (a[x] == n - 1) {
33         cnt--;
34     }
35     a[x]--;
36 }
37
38 void solve() {
39     cin >> n >> m >> k;
40     int f = min(2 * m, n - 1);
41     a.resize(n, f);
42     if (f == n - 1) {
43         cnt = n;
44         cout << "0\n";
45     } else {
46         cnt = 0;
47         cout << "1\n";
48     }
49     for (int i = 0; i < k; i++) {
50         int t, q, w;
51         cin >> t >> q >> w;
52         q--, w--;
53         if (q > w) {
54             swap(q, w);
55         }
56         par p = make_pair(q, w);
57         int z = mp[p];
58         if (z == 0) { // friend status did not change
59             if (w - q <= m || q + n - w <= m) { // were friends
60                 before changes and no changes happened
61                 mp[p] = 2;

```

```

61         z = 2;
62     } else { // were not friends before changes and no
changes happened
63         mp[p] = 1;
64         z = 1;
65     }
66 }
67 if (z == 1) { // giants are now not friends
68     if (t == 1) { // check if they want to become friends
69         mp[p] = 2;
70         up(q);
71         up(w);
72     }
73 } else if (z == 2) { // giants are now friends
74     if (t == 0) { // check if they want to stop being
friends
75         mp[p] = 1;
76         dw(q);
77         dw(w);
78     }
79 }
80 // cout<<a>cout << cnt << ' ';
81 if (cnt == n) {
82     cout << "0\n";
83 } else {
84     cout << "1\n";
85 }
86 }
87 }
88
89 void solveCountInVertex() {
90     cin >> n >> m >> k;
91     int f = min(2 * m, n - 1);
92     a.resize(n, f);
93     if (f == n - 1) {
94         cnt = n;
95         cout << "0\n";
96     } else {
97         cnt = 0;
98         cout << "1\n";
99     }
100     for (int i = 0; i < k; i++) {
101         int t, q, w;
102         cin >> t >> q >> w;
103         q--, w--;
104         if (q > w) {
105             swap(q, w);
106         }
107         if (t == 1) { // check if they want to become friends
108             up(q);
109             up(w);
110         } else if (t == 0) { // check if they want to stop being
friends
111             dw(q);
112             dw(w);
113         }
114         // cout<<a>;
115         if (cnt == n) {
116             cout << "0\n";
117         } else {
118             cout << "1\n";
119         }

```

```

120     }
121 }
122
123 void solveCountTotal() {
124     cin >> n >> m >> k;
125     int f = min(2 * m, n - 1);
126     ll cnt = 0;
127     if (f == n - 1) {
128         cnt = (ll)n * (n - 1) / 2;
129         cout << "0\n";
130     } else {
131         cnt = (ll)n * m;
132         cout << "1\n";
133     }
134     for (int i = 0; i < k; i++) {
135         int t, q, w;
136         cin >> t >> q >> w;
137         q--, w--;
138         if (q > w) {
139             swap(q, w);
140         }
141         if (t == 1) { // check if they want to become friends
142             cnt++;
143         } else if (t == 0) { // check if they want to stop being
friends
144             cnt--;
145         }
146         // couta(a);
147         if (cnt == (ll)n * (n - 1) / 2) {
148             cout << "0\n";
149         } else {
150             cout << "1\n";
151         }
152     }
153 }
154
155 void validateTest(string s = "s") {
156     cin >> n >> m >> k;
157     if (s[0] == 't') {
158         assert(n > 0);
159         assert(k >= 0);
160         assert(m * 2 >= n / 2);
161         assert(n <= MAXN);
162         assert(m <= MAXN);
163         assert(k <= MAXN);
164     }
165     int f = min(2 * m, n - 1);
166     a.resize(n, f);
167     if (f == n - 1) {
168         cnt = n;
169         if (s == "s") {
170             cout << "1\n";
171         }
172     } else {
173         cnt = 0;
174         if (s == "s") {
175             cout << "2\n";
176         }
177     }
178     for (int i = 0; i < k; i++) {
179         int t, q, w;
180         cin >> t >> q >> w;

```

```

181         if (s[0] == 't') {
182             assert(t == 0 || t == 1);
183             assert(q != w);
184             assert(1 <= q && q <= n);
185             assert(1 <= w && w <= n);
186         }
187         q--, w--;
188         if (q > w) {
189             swap(q, w);
190         }
191         par p = make_pair(q, w);
192         int z = mp[p];
193         if (z == 0) { // friend status did not change
194             if (w - q <= m || q + n - w <= m) { // were friends
before changes and no changes happened
195                 mp[p] = 2;
196                 z = 2;
197             } else { // were not friends before changes and no
changes happened
198                 mp[p] = 1;
199                 z = 1;
200             }
201         }
202         if (z == 1) { // giants are now not friends
203             if (s == "t+") {
204                 assert(t == 1);
205             }
206             if (t == 1) { // check if they want to become friends
207                 mp[p] = 2;
208                 up(q);
209                 up(w);
210             }
211         } else if (z == 2) { // giants are now friends
212             if (s == "t+") {
213                 assert(t == 0);
214             }
215             if (t == 0) { // check if they want to stop being
friends
216                 mp[p] = 1;
217                 dw(q);
218                 dw(w);
219             }
220         }
221         if (s == "s") {
222             if (cnt == n) {
223                 cout << "1\n";
224             } else {
225                 cout << "2\n";
226             }
227         }
228     }
229 }
230
231 const int N = 10000;
232 bitset <N> b[N];
233 bitset <N> used;
234
235
236 par bfs(int v, bool us) {
237     queue <par> q;
238     q.push(make_pair(0, v));
239     used[v] = us;

```

```

240     for (int i = 0; i < n - 1; i++) {
241         par z = q.front();
242         q.pop();
243         for (int i = 0; i < n; i++) {
244             if (b[z.second][i] == 0 || used[i] == us) {
245                 continue;
246             }
247             q.push(make_pair(z.first + 1, i));
248             used[i] = us;
249         }
250     }
251     return q.front();
252 }
253
254 int getAns() {
255     par ans = make_pair(INT_MIN, INT_MIN);
256     for (int i = 0; i < n; i++) {
257         for (int j = 0; j < n; j++) {
258             used[j] = 0;
259         }
260         par p = bfs(i, 1);
261         if (p.first > ans.first) {
262             ans = p;
263         }
264     }
265     return max(0, ans.first - 1);
266 }
267
268 void solveTL() {
269     cin >> n >> m >> k;
270     for (int i = 0; i < n; i++) {
271         for (int j = 1; j <= m; j++) {
272             int z = (i + j) % n;
273             b[i][z] = 1;
274             b[z][i] = 1;
275         }
276     }
277     cout << getAns() << '\n';
278     for (int i = 0; i < k; i++) {
279         int t, q, w;
280         cin >> t >> q >> w;
281         q--, w--;
282         b[q][w] = t;
283         b[w][q] = t;
284         cout << getAns() << '\n';
285     }
286 }
287
288 void write1() {
289     cin >> n >> m >> k;
290     for (int i = 0; i <= k; i++) {
291         cout << "1\n";
292     }
293 }
294
295 int main()
296 {
297     ios_base::sync_with_stdio(0);
298     cout << fixed << setprecision(9);
299     cin.tie(0);
300     cout.tie(0);
301     #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
```

## 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  $n$  ( $1 \leq n \leq 10^5$ ) — количество элементов последовательности.

Во второй строке записаны  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^5$ ) — 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 МВ

```

1 // #pragma comment(linker, "/stack:2000000000")
2 // #pragma GCC optimize("Ofast,no-stack-protector")
3 // #pragma GCC
  target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,avx2,tune=nativ
  e")
4 // #pragma GCC optimize("unroll-loops")
5
6 #include <bits/stdc++.h>
7
8 #ifdef PERVEEVM_LOCAL
9     #define debug(x) std::cerr << (#x) << ":\t" << (x) << std::endl
10 #else
11     #define debug(x) 238;
12 #endif
13
14 #define fastIO std::ios_base::sync_with_stdio(false);
  std::cin.tie(0); std::cout.tie(0)
15 #define NAME "File"
16
17 using ll = long long;
18 using ld = long double;
19
20 #ifdef PERVEEVM_LOCAL
21     std::mt19937 rnd(238);
22 #else
23     std::mt19937
  rnd(std::chrono::high_resolution_clock::now().time_since_epoch().co
  unt());
24 #endif
25
26 template<typename T>
27 bool smin(T& a, const T& b) {
28     if (b < a) {
29         a = b;
30         return true;
31     }
32     return false;
33 }
34
35 template<typename T>
36 bool smax(T& a, const T& b) {
37     if (a < b) {
38         a = b;
39         return true;
40     }
41     return false;
42 }
43
44 const double PI = atan2(0.0, -1.0);
45 const int INF = 0x3f3f3f3f;
46 const ll LINF = (ll)2e18;
47 const int N = 100100;
48
49 int cnt[N];
50 ll dp[N];
51
52 void run() {
53     int n;
54     scanf("%d", &n);
55
56     for (int i = 0; i < n; ++i) {
57         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 PERVEEVN_LOCAL
80         std::cerr << "Execution time: "
81                 <<
82         std::chrono::duration_cast<std::chrono::milliseconds>(end -
83         start).count()
84                 << " ms" << std::endl;
85     #endif
86     return 0;
87 }

```

## 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  $180^\circ$ . Если описанная ситуация затем опять повторялась, то есть какие-то рядом стоящие школьники оказывались лицом друг к другу, то они снова поворачивались на  $180^\circ$ . 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  $180^\circ$  в соответствии с вышеописанной процедурой.

## Формат входных данных

Входные данные состоят из двух строк. В первой строке записано число  $N (2 \leq N \leq 10^5)$  — 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 МВ

Код

Python 3

```
1  n = int(input())
2  s = input()
3  ans = 0
4  counter = list(s).count('<')
5  for c in s:
6      if c == '>':
7          ans += counter
8      else:
9          counter -= 1
10 print(ans)
11
```

## Время полёта

Время вылета самолёта из города А по местному времени составляет  $h_1$  часов  $m_1$  минут, а время прилёта в город В по местному времени города В составляет  $h_2$  часов  $m_2$  минут. В обратный рейс из города В он вылетает в  $h_3$  часов  $m_3$  минут по местному времени города В, возможно, уже в другие сутки и прилетает в город А в  $h_4$  часов  $m_4$  минут по местному времени города А. 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;  $0 \leq h_j < 24, 0 \leq m_j < 60$ .

## Выходные данные

Выведите время полёта в том же формате hh:mm. Если правильных 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 МВ

Код

C++

```

1  #include <iostream>
2  #include <vector>
3  #include <sstream>
4  #include <string>
5  #include <cmath>
6
7  using namespace std;
8
9  #define ll int
10
11 bool check(ll mid, ll n) {
12     ll ost = n - mid;
13     return (ost / 2) + 1 < mid;
14 }
15
16 int main() {
17     ll timeA1, timeB1, timeA2, timeB2;
18     ll a, b;
19     scanf("%d:%d", &a, &b);
20     timeA1 = a * 60 + b;
21     scanf("%d:%d", &a, &b);
22     timeB1 = a * 60 + b;
23     scanf("%d:%d", &a, &b);
24     timeB2 = a * 60 + b;
25     scanf("%d:%d", &a, &b);
26     timeA2 = a * 60 + b;
27     ll w1 = timeA2 - timeA1;
28     ll w2 = timeB2 - timeB1;
29     ll x = (timeB1+timeA2-timeA1-timeB2 + (24 * 60)) % (24 *
30 60);
31     if (x == 0) {
32         x = 24 * 60;
33     }
34     ll ans = (x + 1) / 2;
35     printf("%02d:%02d\n", (ans / 60), (ans - (ans / 60 * 60)));
36     return 0;
37 }

```

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 МВ

Код

Python 3

```
1 n = int(input())
2 k = 1
3 count = 0
4 while k * (k + 1) // 2 <= n:
5     if k % 2:
6         if n % k == 0:
7             #print(k, n // k - k // 2, n // k + k // 2)
8             count += 1
9     else:
10        if n % k == k // 2:
11            #print(k, n // k - k // 2 + 1, n // k + k // 2)
12            count += 1
13        k += 1
14 print(count)
```

## 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  $n$  ( $5 \leq n \leq 100000$ ). Во второй строке находятся  $2 \cdot n$  натуральных чисел  $a_i$ , записанных в порядке неубывания — элементы двух объединенных последовательностей ( $1 \leq a_i \leq 10^9$ ). 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 МВ

Код

Python 3

```

1 def check(A, B):
2     if len(A) != n or len(B) != n:
3         return False
4     d1 = A[1] - A[0]
5     d2 = B[1] - B[0]
6     flag_1 = True
7     for i in range(2, len(A)):
8         if A[i] - A[i - 1] != d1:
9             flag_1 = False
10    flag_2 = True
11    for i in range(2, len(B)):
12        if B[i] - B[i - 1] != d2:
13            flag_2 = False
14    if flag_1 and flag_2 and d1 != d2:
15        return True
16    return False
17
18 n = int(input())
19 L = list(map(int, input().split()))
20
21 a1 = L[0]
22 # 1
23 d1 = L[1] - L[0]
24 A = [L[0], L[1]]
25 next_a = L[1] + d1
26 B = []
27 for i in range(2, len(L)):
28     if L[i] == next_a and len(A) < n:
29         A.append(L[i])
30         next_a += d1
31     else:
32         B.append(L[i])
33 if check(A, B):
34     f1 = True
35     ans1 = A
36 else:
37     f1 = False
38 # 2
39 d1 = L[2] - L[0]
40 A = [L[0], L[2]]
41 next_a = L[2] + d1
42 B = [L[1]]
43 for i in range(3, len(L)):
44     if L[i] == next_a and len(A) < n:
45         A.append(L[i])
46         next_a += d1
47     else:
48         B.append(L[i])
49 if check(A, B) and L[1] != L[2]:
50     f2 = True
51     ans2 = A
52 else:
53     f2 = False
54 # 3
55 d2 = L[2] - L[1]
56 B = [L[1], L[2]]
57 next_b = L[2] + d2
58 A = [L[0]]
59 for i in range(3, len(L)):
60     if L[i] == next_b and len(B) < n:
61         B.append(L[i])
62         next_b += d2

```

```
63     else:
64         A.append(L[i])
65     if check(A, B):
66         f3 = True
67         ans3 = A
68     else:
69         f3 = False
70     if f1 == True and f2 == False and f3 == False:
71         print(' '.join(str(i) for i in ans1))
72     elif f1 == False and f2 == True and f3 == False:
73         print(' '.join(str(i) for i in ans2))
74     elif f1 == False and f2 == False and f3 == True:
75         print(' '.join(str(i) for i in ans3))
76     elif f1 == False and f2 == True and f3 == True and L[2] == L[3]:
77         print(' '.join(str(i) for i in ans3))
78     else:
79         print('Ambiguity')
80     #print(f1, f2, f3)
```

## 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 МВ

Код

Python 3

```
1 D = dict()
2 f = []
3 def Fact(n, m):
4     if m == 1 and n > 1: return 0
5     if n == 1: return 1
6
7     if (n, m) in D:
8         return D[(n, m)]
9     s = 0
10    for el in f:
11        if el > m: break
12        if n % el == 0:
13            s += Fact(n // el, min(el, n//el))
14    D[(n, m)] = s
15    return s
16
17
18 n, m = map(int, input().split())
19
20 i = 1
21 while i*i < n:
22     if n % i == 0:
23         f.append(i)
24         f.append(n // i)
25     i += 1
26 if i * i == n:
27     f.append(i)
28 f.sort()
29 f = f[1:]
30 print(Fact(n, m))
31
```

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 ( $2 \leq n \leq 5 \cdot 10^5$ ). 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  $b$  ( $1 \leq a, b \leq n$ ). 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 МВ

Код

Python 3

```
1 def main():
2     from sys import stdin, stdout
3
4     n = int(stdin.readline())
5
6     is_first = [True] * (n + 1)
7     next = [-1] * (n + 1)
8
9     for i in range(n - 1):
10        a, b = list(map(int, input().split()))
11        next[a] = b
12        is_first[b] = False
13
14    cur = -1
15    for i in range(1, n + 1):
16        if is_first[i]:
17            cur = i
18
19    st = [0] * n
20    for i in range(n):
21        st[i] = cur
22        cur = next[cur]
23
24    stdout.write(" ".join(map(str, st)))
25
26
27 main()
28
```

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  $x_i$  ( $1 \leq x_i \leq 20$ ), 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, получают не менее 25 баллов.

### Примеры

Ввод	Вывод
1 2 3 4 5 6 7 8	<pre>#..... .#..... ..#..... ...#..... ....#..... .....#..... .....#.#.. #####. .....###. .....#.#.#. ....#.#.#. ...#.#.#. ..#.#.#. .#.....</pre>
5 5 5 5 5 5 5 5	<pre>#...#...# .#...#...# ..#.#.#.. ...###...</pre>

```
#####  
...###...  
..#.#.#..  
.#...#...#.  
#...#...#
```

### Ограничения

Процессорное время: 2 секунды

Память: 64 МВ

### Код

C++

```
1  #include <iostream>  
2  #include <cmath>  
3  #include <string>  
4  #include <vector>  
5  
6  using namespace std;  
7  
8  int main()  
9  {  
10     vector <int> L(8);  
11     for (int i = 0; i < 8; ++i) cin >> L[i];  
12     int n = max(max(L[0], L[1]), L[7]) + max(max(L[3], L[4]), L[5])  
- 1;  
13     int m = max(max(L[1], L[2]), L[3]) + max(max(L[5], L[6]), L[7])  
- 1;  
14     char M[n][m];  
15     for (int i = 0; i < n; ++i)  
16         for (int j = 0; j < m; ++j)  
17             M[i][j] = '.';  
18  
19     int center_x = max(max(L[0], L[1]), L[7]) - 1;  
20     int center_y = max(max(L[5], L[6]), L[7]) - 1;  
21     int C[8][2] = {{-1, 0}, {-1, 1}, {0, 1}, {1, 1}, {1, 0}, {1, -1},  
{0, -1}, {-1, -1}};  
22     for (int i = 0; i < 8; ++i) {  
23         int x = center_x;  
24         int y = center_y;  
25         for (int j = 0; j < L[i]; ++j) {  
26  
27             M[x][y] = '#';  
28             x += C[i][0];  
29             y += C[i][1];  
30         }  
31     }  
32     for (int i = 0; i < n; ++i) {  
33         for (int j = 0; j < m; ++j)  
34             cout << M[i][j];  
35     cout << endl;  
36     }  
37     return 0;  
38 }  
39
```

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 МВ

Код

C++

```

1  #include <iostream>
2  #include <string>
3  #include <cmath>
4
5  using namespace std;
6
7  int L[10];
8  int main(){
9      for (int i = 0; i < 10; ++i) cin >> L[i];
10     string digits = "";
11     string ans;
12     for (int i = 9; i >= 0; --i)
13         if (i != 3 and i != 4)
14             for (int j = 0; j < L[i]; ++j)
15                 digits += to_string(i);
16
17
18     int count_34 = min(L[3], L[4]);
19
20     if (L[3] == 0 or L[4] == 0) {
21         ans = "";
22         for (int i = 9; i >= 0; --i)
23             for (int j = 0; j < L[i]; ++j)
24                 ans += to_string(i);
25         L[3] = 0;
26         L[4] = 0;
27     }
28     else if (count_34 < digits.size() + 1) {
29         // this is the case when the number of digits-splitter is more
30         // than needed
31         ans = "";
32         int i = 0;
33         while (i < digits.size() and digits[i] > '2')
34             i += 1;
35
36         while (i <= digits.size() and L[3] > 0 and L[4] > 0) {
37             digits = digits.insert(i, "34");
38
39             L[3] -= 1;
40             L[4] -= 1;
41             i += 3;
42         }
43         while (L[3] > 0 and L[4] > 0) {
44             i = digits.find("34");
45             i -= 1;
46             digits = digits.insert(i, "34");
47             L[3] -= 1;
48             L[4] -= 1;
49         }
50         ans = digits;
51     }
52     else
53         // this is the case when the number of digits-splitter is fewer
54         // than needed or equal
55         if (digits.size() > 0) {
56             ans = "34";
57
58             for (int j = 0; j < digits.size(); ++j)
59                 ans = ans + digits[j] + "34";
60             L[3] -= digits.size() + 1;
61             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

```

