

# Отборочный тур МОШ по информатике 2024, 7 класс

15 дек 2023 г., 00:00 — 17 янв 2024 г., 23:59

100 баллов

## Пробка

Глеб оказался в автомобильной пробке. Применяв зоркий глаз, Глеб насчитал, что перед ним стоят  $n$  машин. Также он знает, что зеленый свет горит ровно  $a$  секунд и за каждую секунду зеленого света с пробки успевают уехать ровно  $b$  машин. Красный свет горит ровно  $c$  секунд.

Глеб хочет узнать, сколько еще секунд он будет стоять в пробке, если прямо сейчас загорится зеленый свет.

## Формат входных данных

Первая строка входных данных содержит одно целое число  $n$  ( $1 \leq n \leq 10^9$ ) — количество машин, стоящих перед Глебом.

Вторая строка входных данных содержит три целых числа  $a, b$  и  $c$  ( $1 \leq a, b, c \leq 10^9$ ).

## Формат выходных данных

Выведите единственное число — сколько секунд придется стоять Глебу в пробке.

Обратите внимание, что ответ в этой задаче может быть довольно большим и не помещаться в 32-битные типы данных. Рекомендуется использовать 64-битный тип данных, например `long long` в C++, `long` в Java или `int64` в Pascal.

## Критерии оценивания

В этой задаче каждый тест, кроме тестов из условия, оценивается независимо.

### Примеры

```
5
4 2 3
```

```
3
```

```
11
4 1 3
```

```
18
```

```
2
2 1 5
```

```
8
```

```
2
1 2 5
```

```
7
```

### Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4  #define int long long
5
6  signed main() {
7      int n, a, b, c;
8      cin >> n >> a >> b >> c;
9      int n1 = (n + 1 + b - 1) / b * b;
10     int n2 = n1 / b;
11     int k1 = (n2 - 1) / a * (a + c);
12     cout << k1 + ((n2 - 1) % a + 1);
13 }
14
```

100 баллов

# Уникальный набор

У Антона есть массив  $a$  длины  $n$ , каждый элемент которого имеет целое значение в диапазоне от 1 до  $k$  (включительно). Ему хочется узнать: есть ли в массиве какой-нибудь подотрезок длины  $k$ , который содержит все числа от 1 до  $k$ ?

Если же такой подотрезок есть, требуется вывести позицию его начала (элементы массива нумеруются с 1), иначе требуется вывести  $-1$ . Если вариантов ответов несколько, то требуется вывести минимальный из них.

## Формат входных данных

В первой строке дано два числа  $n$  и  $k$  ( $1 \leq k \leq n \leq 2 \cdot 10^5$ ) — количество элементов массива и число  $k$  соответственно.

Во второй строке даны  $n$  чисел  $a_i$  ( $1 \leq a_i \leq k$ ) — элементы массива  $a$ .

## Формат выходных данных

Выведите ответ на задачу.

## Критерии оценивания

Номер подзадачи	Баллы	Ограничения	Комментарий
0	0	—	Примеры из условия.
1	40	$1 \leq n \leq 1000$	Баллы начисляются, если все тесты этой и предыдущих подзадач пройдены.
2	60	Основные ограничения	Баллы начисляются, если все тесты этой и предыдущих подзадач пройдены.

### Примеры

4 3 1 2 3 1	1
4 3 1 2 2 2	-1
5 4 2 4 2 1 3	2
5 5 2 5 4 2 1	-1
5 2 2 2 2 1 2	3

Ограничения

Время выполнения: 2 секунды

Память: 256 MB

КодC++

```
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int main() {
7      int n, k;
8
9      cin >> n >> k;
10
11     vector<int> a(n);
12
13     for (int i = 0; i < n; i++) {
14         cin >> a[i];
15         a[i]--;
16     }
17
18     vector<int> cnt(k, 0);
19
20     for (int i = 0; i < k; i++) {
21         cnt[a[i]]++;
22     }
23
24     int different = 0;
25
26     for (int i = 0; i < k; i++) {
27         if (cnt[i] > 0) {
28             different++;
29         }
30     }
31
32     if (different == k) {
33         cout << 1;
34         return 0;
35     }
36
37     int ans = -1;
38
39     for (int i = k; i < n; i++) {
40         cnt[a[i - k]]--;
41         if (cnt[a[i - k]] == 0) {
42             different--;
43         }
44
45         cnt[a[i]]++;
46         if (cnt[a[i]] == 1) {
47             different++;
48         }
49
50         if (different == k) {
51             ans = i - k + 2;
52             break;
53         }
54     }
55
56     cout << ans;
57
58     return 0;
59 }
60
```



100 баллов

## Пёстрая дата

По текущей дате определите ближайшую следующую дату, запись которой в виде день, месяц и год состоит из различных цифр.

Учтите, что если день или номер месяца меньше 10, то они записываются без лишних нулей. Например, дата «8 марта 2019 года» запишется так: «8.3.2019».

Гарантируется, что изначальная дата корректна.

## Формат входных данных

В первой строке задано одно целое число  $d$  ( $1 \leq d \leq 31$ ) — день текущей даты.

Во второй строке задано одно целое число  $m$  ( $1 \leq m \leq 12$ ) — месяц текущей даты.

В третьей строке задано одно целое число  $y$  ( $1000 \leq y \leq 10^6$ ) — год текущей даты.

## Формат выходных данных

Выведите одну строку — ближайшую следующую дату, в записи которой все цифры различны.

## Критерии оценивания

Все тесты в этой задаче оцениваются независимо.

## Примечание

Високосный год — такой год, что в нём 366 дней, а именно добавляется 29 февраля. Год является високосным, если он делится на 400 или делится на 4, но не делится на 100.

### Примеры

18 12 2018
4.3.2019

1 2 3456
7.2.3456

### Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++



```

1  #include <fstream>
2  #include <iostream>
3  using namespace std;
4
5  bool leap_year(int y) {
6      return (y % 4 == 0 && y % 100 != 0 || y % 400 == 0);
7  }
8
9  int day_in_month(int m, int y) {
10     if (m == 2)
11         if (leap_year(y))
12             return 29;
13         else
14             return 28;
15     else if (m == 4 || m == 6 || m == 9 || m == 11)
16         return 30;
17     else
18         return 31;
19 }
20
21 void inc(int & d, int & m, int & y) {
22     d += 1;
23     int dm = day_in_month(m, y);
24     if (d > dm) {
25         d = 1;
26         m += 1;
27         if (m > 12) {
28             m = 1;
29             y += 1;
30         }
31     }
32 }
33
34 bool good_data(int d, int m, int y) {
35     int a[10];
36     for (int i = 0; i < 10; i++) a[i] = 0;
37     a[d % 10] += 1;
38     if (d > 9) a[d / 10] += 1;
39     a[m % 10] += 1;
40     if (m > 9) a[m / 10] += 1;
41     while (y > 0) {
42         a[y % 10] += 1;
43         y /= 10;
44     }
45     bool good = true;
46     for (int elem : a)
47         if (elem > 1) {
48             good = false;
49             break;
50         }
51     return good;
52 }
53 int main() {
54     int day, month, year;
55
56     cin >> day >> month >> year;
57     inc(day, month, year);
58     while (!good_data(day, month, year))
59         inc(day, month, year);
60     cout << day << '.' << month << '.' << year;
61
62     return 0;

```



100 баллов

## Странная задача

Есть массив  $a$  длины  $n$  (элементы массива нумеруются с 1). Требуется узнать, сколько существует различных пар  $(i, j)$  таких, что  $a_i + j = a_j + i$  и  $1 \leq i < j \leq n$ .

## Формат входных данных

Первая строка содержит одно целое число  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ) — количество элементов массива  $a$ .

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^6$ ) — элементы массива  $a$ .

## Формат выходных данных

Выведите одно число — ответ на задачу.

## Критерии оценивания

Тесты в этой задаче оцениваются независимо. Решения, верно работающие для  $n \leq 1000$ , наберут не менее 40 баллов.

### Примеры

```
4
1 2 3 4
```

```
6
```

```
3
1 17 9
```

```
0
```

```
4
2 3 9 10
```

```
2
```

### Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  #define ll long long
6
7  int main() {
8      ll n;
9      cin >> n;
10     map<ll, ll> mp;
11     for (int i = 0; i < n; ++i) {
12         ll x;
13         cin >> x;
14         mp[x - i]++;
15     }
16     ll ans = 0;
17     for (auto i : mp) {
18         ans += (i.second * (i.second - 1)) / 2;
19     }
20     cout << ans;
21 }
22
```

100 баллов

## Закрась по правилам

Дан клетчатый прямоугольник из  $N$  строк по  $M$  клеток, каждая из которых или уже закрашена, или еще не закрашена. Если в каком-либо квадрате размером  $2 \times 2$  три клетки уже закрашены, то можно закрасить и четвёртую клетку.

Оцените, сколько клеток могут в итоге оказаться закрашенными.

## Формат входных данных

Первая строка входных данных содержит два целых числа  $N$  и  $M$  ( $1 \leq N, M \leq 500$ ) — количество строк и столбцов в прямоугольнике. Следующие  $N$  строк по  $M$  символов описывают клетки прямоугольника.

Символ «.» соответствует незакрашенной клетке, а «#» — закрашенной клетке. Строки нумеруются от 1 до  $N$ , столбцы — от 1 до  $M$ .

## Формат выходных данных

Выведите одно число — максимальное количество клеток, которые могут оказаться закрашенными.

## Критерии оценивания

Каждый тест оценивается независимо. Решения, правильно работающие для ограничений  $1 \leq N, M \leq 50$ , будут набирать не менее 60 баллов.

### Примеры

```
2 2
##
#.
```

4

```
3 4
#...
#...
###.
```

9

```
3 5
...##
#....
#.#..
```

5

### Ограничения

Время выполнения: 2 секунды

Память: 256 MB

Код

C++

```

1  #include <iostream>
2  #include <vector>
3  #include <cstdio>
4
5  using namespace std;
6
7  int table[1000][1000], n, m;
8  vector < pair < int, int> > st;
9
10 int sumSquare(int i, int j)
11 {
12     return table[i][j] + table[i + 1][j] + table[i][j + 1] +
13     table[i + 1][j + 1];
14 }
15 void pushSquare(int i, int j)
16 {
17     if ((0 <= i) && (i < n - 1) && (0 <= j) && (j < m - 1))
18         if (sumSquare(i, j) == 3)
19             st.push_back(make_pair(i, j));
20 }
21 void pushNeighbors(int i, int j)
22 {
23     pushSquare(i - 1, j - 1);
24     pushSquare(i, j - 1);
25     pushSquare(i - 1, j);
26     pushSquare(i, j);
27 }
28 void updTile(int i, int j)
29 {
30     if ((0 <= i) && (i < n) && (0 <= j) && (j < m))
31         if (!table[i][j])
32             table[i][j] = 1;
33             pushNeighbors(i, j);
34 }
35
36 void updSquare(int i, int j)
37 {
38     updTile(i, j);
39     updTile(i + 1, j);
40     updTile(i, j + 1);
41     updTile(i + 1, j + 1);
42 }
43
44 int main()
45 {
46     // freopen("f.in", "r", stdin);
47     //     freopen("f.out", "w", stdout);
48     cin >> n >> m;
49     char c;
50     for (int i = 0; i < n; i++)
51     {
52         for (int j = 0; j < m; j++)
53         {
54             cin >> c;
55             if (c == '#')
56                 table[i][j] = 1;
57         }
58     }
59     for (int i = 0; i < n - 1; i++)
60         for (int j = 0; j < m - 1; j++)
61             if (sumSquare(i, j) == 3)

```

```
62         st.push_back(make_pair(i, j));
63     while (!st.empty())
64     {
65         pair < int, int > p = st.back();
66         st.pop_back();
67         if (sumSquare(p.first, p.second))
68             updSquare(p.first, p.second);
69     }
70     int answer = 0;
71     for (int i = 0; i < n; i++)
72         for (int j = 0; j < m; j++)
73             answer += table[i][j];
74     cout << answer << endl;
75     return 0;
76 }
77
```

100 баллов

## Гиревой спорт

В школьном тренажерном зале оказалось  $n$  различных гирь, разложенных в некотором порядке так, что они образовали массив  $a$ , где значение  $a_i$  равно удовольствию, которое можно получить при поднятии гири с номером  $i$  (удовольствие также может быть отрицательным, например, если гиря задела самолюбие).

Спортсмен хочет выбрать некоторый подотрезок гирь и поднять их. Удовольствие, которое он получит, будет равно сумме удовольствий гирь на этом подотрезке. Все было бы просто, но при поднятии каждой  $k$ -й гири из выбранного диапазона спортсмен теряет  $x$  единиц удовольствия.

Например, если  $k = 2$ , а спортсмен поднял 5 гирь, то он получит удовольствие равное сумме удовольствий от поднятия выбранных гирь минус  $2 \cdot x$ .

Найдите такой подотрезок, чтобы удовольствие, которое получит спортсмен, будет максимально, и выведите величину этого удовольствия.

Можно не поднимать ни одной гири, если это будет оптимально.

## Формат входных данных

Первая строка содержит три целых числа  $n$ ,  $k$  и  $x$  ( $1 \leq n \leq 2 \cdot 10^5, 1 \leq k \leq 5, 1 \leq x \leq 10^4$ ) — количество элементов массива, а также числа  $k$  и  $x$ .

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $-10^4 \leq a_i \leq 10^4$ ) "--- элементы массива  $a$ .

## Формат выходных данных

Выведите одно число — максимальное удовольствие, которое можно получить при описанных правилах.

## Пояснения к примерам

В первом примере оптимально взять отрезок с третьего по шестой элемент, тогда спортсмен получит  $14 - 2 + 9 + 4 = 25$  единиц удовольствия от гирь и потеряет 8 единиц удовольствия от третей гири.

Во втором примере отпимально не брать ни одну гирю.

В третьем примере можно взять либо отрезок с первого по второй элемент, либо взять только четрвертый элемент.

## Критерии оценивания

Все тесты в этой задаче оцениваются независимо.

### Примеры

6	3	8			
1	-3	14	-2	9	4

17

5	5	5		
-1	-2	-3	-4	-10

0

4	2	5	
6	6	-9	7

7

### Ограничения

Время выполнения: 2 секунды



## Код

C++

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  #define ll long long
6  #define ld long double
7
8  ld INF = (ld) 1e20;
9  int iINF = (int) 1e20;
10 ll mod = (ll) 1e9 + 7;
11
12 int main() {
13     ll n, y, x;
14     cin >> n >> y >> x;
15     vector<ll> a(n);
16     ll maxim[n], sum[n], dp[n];
17     for (int i = 0; i < n; ++i) {
18         cin >> a[i];
19         maxim[i] = 0;
20         sum[i] = 0;
21         dp[i] = 0;
22     }
23     for (int i = 0; i < n; ++i) {
24         ll now = 0;
25         for (int j = 0; j < y - 1; ++j) {
26             if (i + j >= n)
27                 break;
28             now += a[i + j];
29             if (now > maxim[i]) {
30                 maxim[i] = now;
31             }
32         }
33         if (i + y - 1 < n) {
34             now += a[i + y - 1];
35         }
36         sum[i] = now;
37     }
38     ll ans = 0;
39     for (int i = n - 1; i >= 0; --i) {
40         dp[i] = max(dp[i], maxim[i]);
41         if (i + y < n) {
42             dp[i] = max(dp[i], sum[i] + dp[i + y] - x);
43         }
44         ans = max(ans, dp[i]);
45     }
46     cout << max(ans, (ll) 0);
47 }
48
```

## Каждой твари по паре

Коллекционировать животных можно не только на Ноевом ковчеге, но и на дереве. А дерево, как известно из теории графов, является связным ациклическим графом. Иными словами, дерево состоит из  $n$  вершин и  $n - 1$  ребра между ними. Каждое ребро соединяет некоторые две вершины дерева. Из любой вершины дерева можно добраться по ребрам до любой другой. Также дерево не содержит циклов, т. е. нельзя построить путь по ребрам из некоторой вершины и вернуться в нее, не пройдя по одному и тому же ребру дважды.

Для расположения животных на нашем дереве сделано  $q$  гнезд, где  $q$  — четное число. Животным каждого вида предназначены два гнезда.

После расселения по дереву каждая пара одного вида может перемещаться по ребрам дерева между своими гнездами. Они не должны мешать другим видам, т. е. чтобы никакие два пути не имели общего ребра. При этом допускается пересечение путей в вершине. Вам необходимо разместить  $\frac{q}{2}$  видов животных по гнездам таким образом, чтобы пути животных одного вида удовлетворяли этому условию.

## Формат входных данных

В первой строке входных данных находится два целых числа  $n$  и  $q$  ( $2 \leq n \leq 2 \cdot 10^5, 2 \leq q \leq n, q$  четно) — общее количество вершин в дереве и количество вершин, в которых расположены гнезда.

Во второй строке входных данных находятся  $q$  различных целых чисел  $m_i$  ( $1 \leq m_i \leq n$ ) — номера вершин с гнездами.

В каждой из следующих  $n - 1$  строк входных данных находится два целых  $u_i$  и  $v_i$  ( $1 \leq u_i, v_i \leq n, u_i \neq v_i$ ). Это означает, что между вершинами  $u_i$  и  $v_i$  в дереве проведено ребро.

Гарантируется, что ребра во входных данных образуют дерево.

## Формат выходных данных

Выходные данные должны содержать  $\frac{q}{2}$  строк.

В каждой строке должно содержаться два целых числа  $x_i$  и  $y_i$  ( $1 \leq x_i, y_i \leq n$ ) — номера вершин, в которых будут располагаться гнезда  $i$ -го вида животных.

Все числа в выходных данных должны быть различны и должны представлять собой номера вершин с гнёздами.

Если разместить животных требуемым образом невозможно, выведите число  $-1$ .

## Критерии оценивания

Баллы начисляются за каждый тест в отдельности, но ряд тестов имеют специфическую структуру:

- $u_i = i, v_i = i + 1$
- $u_i = 1, v_i = i + 1$
- $n \leq 3000, q \leq 6$
- $q = n$ .
- Нет дополнительных ограничений.

### Примеры

```
9 6
1 3 2 5 8 6
3 1
3 2
3 4
3 5
```

3 6  
4 7  
7 8  
1 9

3 2  
8 5  
1 6

Ограничения

Время выполнения: 1 секунда  
Память: 256 МВ

Код

C++

```
1  #include <vector>
2  #include <array>
3  #include <iostream>
4  using namespace std;
5  const int N = 400100;
6  vector<int> g[N];
7  vector<array<int,2>> pairs;
8  int n, q, paired[N];
9
10 void make_paired(int v, int p) {
11     for (int u : g[v]) {
12         if (u == p) continue;
13
14         make_paired(u, v);
15
16         if (paired[u] >= 0) {
17             if (paired[v] >= 0) {
18                 pairs.push_back({paired[v], paired[u]});
19                 paired[v] = -1;
20                 paired[u] = -1;
21             } else {
22                 paired[v] = paired[u];
23                 paired[u] = -1;
24             }
25         }
26     }
27 }
28
29 int main() {
30     ios_base::sync_with_stdio(0); cin.tie(0);
31
32     cin >> n >> q;
33     fill(paired, paired + n, -1);
34     for (int i = 0; i < q; i++) {
35         int v; cin >> v;
36         v--;
37         paired[v] = v;
38     }
39     for (int i = 1; i < n; i++) {
40         int x, y; cin >> x >> y;
41         x--; y--;
42         g[x].push_back(y);
43         g[y].push_back(x);
44     }
45
46     make_paired(0, -1);
47
48     for (auto pair : pairs) {
49         cout << pair[0] + 1 << " " << pair[1] + 1 << '\n';
50     }
51
52     return 0;
53 }
54
```