Московская олимпиада школьников по информатике 2025-2026 6-7 класс (первый отбор)

19 дек 2025 г., 10:00 — 28 дек 2025 г., 23:59

№ 1

100 баллов

Невероятная находка

Чего только нельзя найти в рюкзаке под Новый год. Вернувшись из школы, Вася обнаружил два прямоугольника — части одной карты сокровищ. Всем известно, что карты сокровищ рисуют прямоугольными, но карты самых ценных сокровищ изображают на квадратном полотне. Васю гложет любопытство, существует ли квадрат, который можно разрезать на два данных прямоугольника, или это всё же был прямоугольник.

Входные данные

На ввод даётся 4 натуральных числа, записанных в отдельных строках (каждое не превосходит 100), где первые два соответствуют ширине и высоте (в любом порядке) первого прямоугольника, а третье и четвёртое число соответствуют ширине и высоте (в любом порядке) второго прямоугольника.

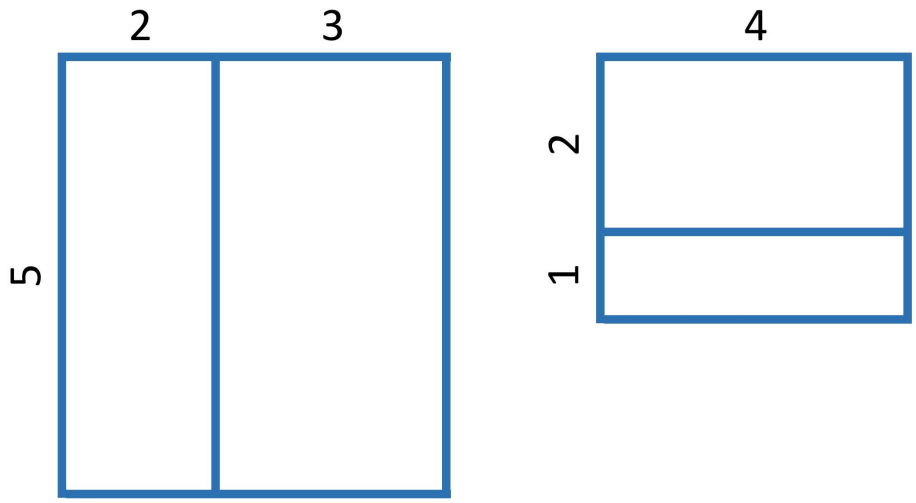
Выходные данные

Выведите **YES** в первой строке, если карта могла быть квадратом, и сторону подходящего квадрата во второй строке. Выведите **NO**, если можно утверждать, что карта точно не могла быть квадратом; во второй строке выведите размеры прямоугольника, который мог быть изначально, стороны можно выводить в любом порядке, разрешается выводить любой подходящий прямоугольник.

Комментарий

Во втором примере могли быть прямоугольники 3×4, 4×3. Любой из них принимается как верный ответ.

Фигуры, получаемые в примерах:



Система оценивания

Все тесты в этой задаче оцениваются независимо. Вы получите баллы за каждый пройденный тест.

Примеры

```
2
5
5
3
```

```
YES
5
```

```
4
2
```

```
1
4
```

```
NO
4 3
```

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код C++

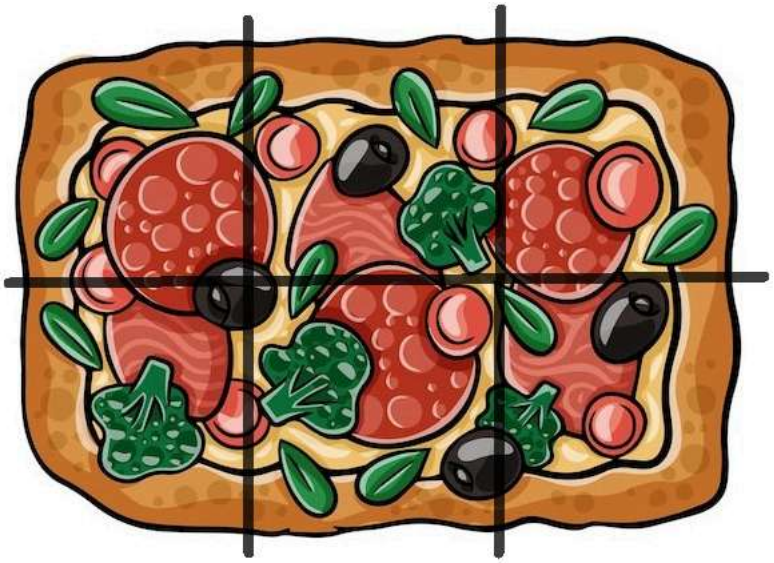
```cpp
#include <bits/stdc++.h>
using namespace std;


int main() {
    int a, b, c, d;
    cin >> a >> b >> c >> d;
    if (b == d) {
        swap(a, b);
        swap(c, d);
    }
    if (a == d)
        swap(c, d);
    if (b == c)
        swap(a, b);
    int p = a * b + c * d;
    // 2 4 2 4
    if ((a * a == p || b * b == p) && (c * c == p || d * d == p))
        cout << "YES\n" << a << '\n';
    else {
        cout << "NO\n";
        if (a == c)
            cout << a << ' ' << b + d << '\n';
        else if (a == d)
            cout << a << ' ' << b + c << '\n';
        else if (b == c)
            cout << b << ' ' << a + d << '\n';
        else if (b == d)
            cout << b << ' ' << a + c << '\n';
    }
}

```

№ 2

100 баллов

Римская пицца

Дима и Лёша заказали огромную прямоугольную римскую пиццу. Дима ужасно проголодался и хочет поскорее приступить к еде. Единственный способ для него поскорее добраться до своей порции и начать радоваться жизни — это разрезать пиццу на как можно большее количество частей, используя доступное количество разрезов. Дима может делать только прямые горизонтальные и вертикальные разрезы, никак не перемещая образовавшиеся куски.



Три разреза, дающие шесть частей.

Ваша задача, зная количество разрезов N, которое Дима может сделать, найти максимальное количество частей, на которое можно порезать пиццу.

Входные данные

Первая строка входных данных содержит единственное число N ($1 \le N \le 100$) — количество разрезов, которое может сделать Дима.

Выходные данные

Выведите наибольшее количество частей, на которое Дима может разрезать римскую пиццу.

Система оценивания

Все тесты в этой задаче оцениваются независимо. Вы получите баллы за каждый пройденный тест.

Примеры

3	
6	

Ограничения

Время выполнения: 1 секунда
Память: 256 MB

Код C++

```cpp
#include <bits/stdc++.h>

using namespace std;


int main() {
    int n;
    cin >> n;
    cout << (n/2 + 1) * ((n+1)/2+1);

}

```

№ 3

100 баллов

Хижина в лесу

Коля решил стать лесником. Лес представляет собой квадрат 10×10 клеток. В лесу отмечены клетки, в которых растут деревья. Коля хочет построить хижину. Требуется определить максимальную длину хижины, которая может поместиться в свободных от деревьев клетках. Хижина представляет собой прямоугольник ширины 1 и располагается горизонтально или вертикально. (Гарантируется, что на поле есть хотя бы одна свободная клетка.)

Входные данные

Вводятся 10 строк по 10 чисел в каждой, числа разделены пробелами. Число 1 означает, что в соответствующей клетке есть дерево, число 0 — что клетка свободна.

Выходные данные

Требуется вывести одно число от 1 до 10 — максимальную возможную длину хижины.

Система оценивания

Все тесты в этой задаче оцениваются независимо. Вы получите баллы за каждый пройденный тест.

Примеры

```
0 0 0 0 1 0 0 0 0 0
0 1 0 0 0 0 0 0 1 0
0 0 1 0 0 0 0 1 0 0
0 0 0 1 0 0 1 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 1 0 0 1 0 0 0
0 0 1 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0 1 0
1 0 0 0 0 0 0 0 0 0
```

```
10
```

```
1 0 0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0 1 0
0 0 1 0 0 0 0 1 0 0
0 0 0 1 0 0 1 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 1 0 0 1 0 0 0
0 0 1 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0 1 0
1 0 0 0 0 0 0 0 0 1
```

```
8
```

Ограничения

Время выполнения: 2 секунды

Память: 256 MB

Код C++

```cpp

#include <bits/stdc++.h>
using namespace std;

int main()
{
    int a[10][10] = {0};
    for(int i=0; i<10; ++i)
        for(int j=0; j<10; ++j)
            cin>>a[i][j];
    for(int k=10; k>=1; --k){
        bool f = 0;
        for(int i=0; i<10; ++i)
            for(int j=0; j<10; ++j) {
                if (i + k <= 10) {
                    bool f1 = 1;
                    for (int ii = i; ii < i + k; ++ii)
                        if (a[ii][j])
                            f1 = 0;
                    f |= f1;
                }

                if (j + k <= 10) {
                    bool f2 = 1;
                    for (int jj = j; jj < j + k; ++jj)
                        if (a[i][jj])
                            f2 = 0;
                    f |= f2;
                }
            }
        if(f){
            cout<<k;
            return 0;
        }
    }
    return 0;
}
```

№ 4

100 баллов

Полтора кружка

Саша настолько сильно хочет учиться программировать, что записался сразу в два кружка. В каждом кружке учащимся задано N задач, причём Саша уже обучался в третьем кружке и имеет N решений задач, которые уже использовал для сдачи таких же задач в двух новых кружках.

Из-за нехватки времени Саша может решить не более K задач, а его преподаватели согласны совершенно безвозмездно подарить по одной печеньке Саше за каждую решённую задачу из своего кружка. Саша быстро сообразил, что ему выгодно в первую очередь решать задачи, заданные в обоих кружках. В таком случае он получает две печеньки за одну решённую задачу. Подскажите Саше, какое максимальное количество печенек он может получить.

Обратите внимание, что за старые решённые задачи Саша печеньки не получает, так как отправил их на первой же минуте кружков, и преподаватели не считают эти решения полезными.

Входные данные

В первой строке записано два целых числа N и K ($1 \le N \le 100, 1 \le K \le 100$). Во второй строке записаны N целых чисел - номера задач, решённых Сашей ранее. Третья строка содержит номера задач, заданных на первом кружке в том же формате, а четвёртая - номера задач, заданных на втором кружке. Каждая из этих трёх строк содержит последовательность различных целых чисел от 1 до 1000.

Выходные данные

Выведите единственное число — количество печенек, которое получит Саша, действуя оптимальным способом.

Система оценивания

Все тесты в этой задаче оцениваются независимо. Вы получите баллы за каждый пройденный тест.

Примеры

```
2 1
103 107
103 108
107 108
```

```
2
```

```
2 100
13 17
15 18
18 17
```

```
3
```

```
3 2
2 3 1
5 3 4
7 5 2
```

```
3
```

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код C++

```cpp
#include <bits/stdc++.h>

using namespace std;


int main() {
    int n, k;
    cin >> n >> k;
    vector<int> a(n), b(n), c(n), cnt(1001, 0);
    for (int i = 0; i < n; ++i) {
        cin >> a[i];
    }
    for (int i = 0; i < n; ++i) {
        cin >> b[i];
        cnt[b[i]]++;
    }
    for (int i = 0; i < n; ++i) {
        cin >> c[i];
        cnt[c[i]]++;
    }
    for (int i = 0; i < n; ++i) {
        cnt[a[i]] = 0;
    }
    std::sort(cnt.rbegin(), cnt.rend());
    int ans = 0;
    for (int i = 0; i < k; ++i)
        ans += cnt[i];
    cout << ans;
}

```

№ 5

100 баллов

Очень правильное число

Правильными суммами называют суммы нескольких подряд идущих натуральных чисел. Например, суммы 7 + 8 и 4 + 5 + 6 — правильные, а сумма 3 + 5 + 7 — нет, хотя результат суммирования во всех случаях равен 15. (Сумма из одного слагаемого 15 тоже считается правильной.) Правильностью целого положительного числа будем называть количество представлений этого числа в виде правильных сумм. Например, правильность числа 15 равна 4, поскольку 15 представляется в виде правильных сумм четырьмя способами: $15 = 7 + 8 = 4 + 5 + 6 = 1 + 2 + 3 + 4 + 5$.

Из двух целых чисел более правильным считается то, у которого больше представлений в виде правильных сумм. При равенстве количеств таких представлений предпочтение в правильности отдаётся меньшему из них. Например, у чисел 15 и 30 правильность одинаковая (равна 4), однако более правильным считается число 15.

Вам необходимо составить программу, которая в заданном наборе целых положительных чисел находит самое правильное число и определяет его правильность.

Входные данные

В первой строке записано целое n - количество чисел в наборе ($2 \le n \le 10^3$). Во второй строке содержится n целых положительных чисел a_i, каждое из которых не превосходит $2 \cdot 10^9$.

Выходные данные

Выведите два целых числа — самое правильное из всех чисел набора и значение его правильности.

Система оценивания

Все тесты в этой задаче оцениваются независимо. Вы получите баллы за каждый пройденный тест.

Примеры

```
2
15 30
```

```
15 4
```

```
3
20 30 20
```

```
30 4
```

Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код C++

```cpp
#include <iostream>
using namespace std;

long long primary(long long m)
{
    int k = 2 + m % 2;
    while((m % k != 0) && (k * k < m))
        k += 2;
    if (k * k > m) return m; else return k;
}

int main()
{
    int n;
    cin >> n;
    pair < int,int > beautiful(1,1);
    // Самое красивое число и его красота пока равны 1
    for (int i = 0; i < n; ++i) {
        long long a, aa;
        cin >> a;
        aa = a;

        int k = 0;
        long long d[50];
        while (aa != 1) {
            d[k] = primary(aa);
            aa = aa / d[k];
            if (d[k] > 2) k++;
        }
        d[k] = -1;

        int beauty = 1;
        int count = 1;
        for (int i = 0; i < k; i++)  {
            if (d[i] == d[i+1]) count++;
            else {
                beauty *= count + 1;
                count = 1;
            }
        }

        if (beauty == beautiful.second)
            if (beautiful.first > a) beautiful.first = a;
        if (beauty > beautiful.second) {
            beautiful.first = a;
            beautiful.second = beauty;
        }
    }
    cout << beautiful.first << " " << beautiful.second << endl;
}
```