ЗАДАЧИ С CODEFORCES

2011-2012 САМАРСКИЙ МЕЖДУНАРОДНЫЙ АЭРОКОСМИЧЕСКИЙ ЛИЦЕЙ, ТРЕНИРОВКА №2

Выполнили: Корниенко П. А.
Канаев С. Д.

ЗАДАЧА E. ИГРА

На столе лежат n стопок игральных карт, по 2 карты в каждой стопке. Нужно угадать такую стопку, что в ней обе карты – черной масти.

Стопка выбрана, и первая вытащенная карта, оказалась черной масти. С какой вероятностью вторая карта из этой стопки также окажется черной масти?

Формат входного файла

В первой строке записано число n ($1 \leq n \leq 1000$) — количество стопок игральных карт.

Далее через пробел записано n строк, где i-ая строка характеризует i-ую стопку. Эти строки состоят ровно из двух символов, каждый из которых описывает карту в стопке и равен “B”, если эта карта черной масти, и “R”, если красной.

Гарантируется, что во входных данных присутствует хотя бы один символ “B”.

Формат выходного файла

Выведите несократимую дробь: вероятность, с которой вторая карта в стопке, выбранной Черным Плащом, окажется черной масти.

Примеры

input.txt	output.txt
1 BB	1/1
2 RR BR	0/1
3 BB RB RR	1/2

КОД ПРОГРАММЫ

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.Scanner;

public class D {

    public static void main(String[] args) throws FileNotFoundException{
        Scanner in = new Scanner(new File("input.txt"));
        PrintWriter out = new PrintWriter(new File("output.txt"));
        int n = in.nextInt();
        int k = 0;
        int l = 0;
        String cards = new String();
        String v[] = new String[n];



        for (int i = 0 ; i < n; i++){
            cards += in.next();
        }
```

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

            v[i] = cards.substring(j, j+2);
            j+=2;
            if (v[i].equals("BB"))
                k+=1;
            if (v[i].equals("RB") || v[i].equals("BR") || v[i].equals("BB"))
                l+=1;

        }

        for ( int i = 2 ; i < 1000 ; i++){
            if (k%i == 0 && l%i == 0){
                k = k/i;
                l = l/i;
            }
        }



        out.print(k);
        out.print('/');
        out.print(l);
        out.close();
    }

}
```

ЗАДАЧА D. РАЗВЕДКА

В некоторых n точках плоскости XOY растёт картофель. Необходимо пройтись вдоль линии, параллельной оси OY. Лучше всего можно разглядеть картофелины , если пройти по такой прямой, что величина $\sum_{i=1}^{n} d_i^2$, где

d_i – расстояние от i-той картофелины до этой прямой принимает наименьшее значение.

Формат входного файла

В первой строке записано число n ($1 \le n \le 100$) — количество картофелин доктора Бушрута.

В следующей строке через пробел записано n чисел $x_1, \ldots, x_n$ ($-100 \le x_i \le 100$), где x_i — абсцисса точки, в которой расположена i-ая картофелина.

Формат выходного файла

Выведите вещественное число x_0 — абсциссу точек, образующих прямую, вдоль которой будет двигаться Черный Плащ. Ваш ответ должен отличаться от верного не более, чем на 10^{-6}.

Примеры

input.txt	output.txt
1 0	0.000000000000
2 0 1	0.500000000000
4 -2 3 -1 4	1.000000000000

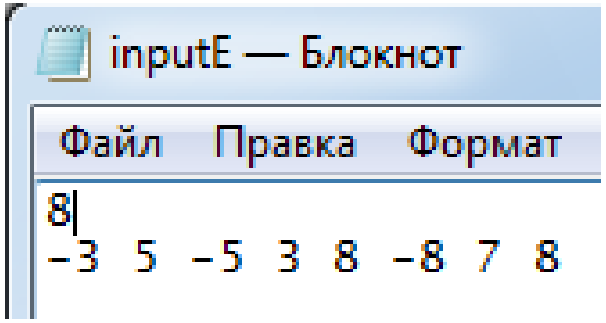
Note

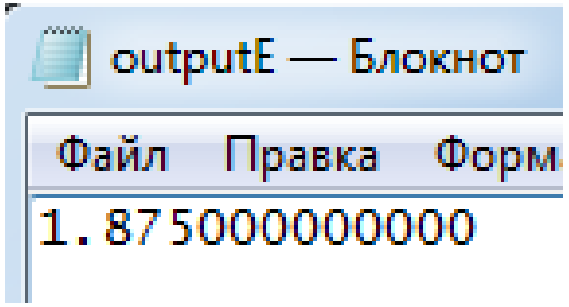
Другими словами, надо определить такое число x_0, что Черный Плащ будет двигаться вдоль прямой $x = x_0$.

КОД ПРОГРАММЫ

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.Locale;
import java.util.Formatter;
import java.util.Scanner;


public class D {

    public static void main(String[] args) throws FileNotFoundException {
        Scanner in = new Scanner (new File("input.txt"));
        PrintWriter out = new PrintWriter (new File("output.txt"));
  int n = in.nextInt();
  int tochki [] = new int[n];
  double k=0;

  for (int i = 0; i < tochki.length; i++){
      tochki[i] = in.nextInt();
      k+=tochki[i];
        }
     k = k / n;
```

```
StringBuilder sb = new StringBuilder();
Formatter formatter = new Formatter (sb, Locale.US);
formatter.format("%12.12f", k);
System.out.println(formatter);
out.print(formatter);
out.close();
        }
    }
```





ЗАДАЧА B. ЛОВУШКА

В каждой точке с цифрой находится банк. В первый день был ограблен банк, расположенный в точке с координатами (0,1), во второй – банк, расположенный в точке с координатами (1,1), в третий - в точке (1,2), затем (0,2), (-1,2),(-1,1),(-2,1),(-2,2)…

$$
\begin{array}{ccccccccc}
\dots & -\frac{3}{1} & -\frac{2}{1} & -\frac{1}{1} & \frac{0}{1} & \frac{1}{1} & \frac{2}{1} & \frac{3}{1} & \dots \\
\dots & -\frac{3}{2} & -\frac{2}{2} & -\frac{1}{2} & \frac{0}{2} & \frac{1}{2} & \frac{2}{2} & \frac{3}{2} & \dots \\
\dots & -\frac{3}{3} & -\frac{2}{3} & -\frac{1}{3} & \frac{0}{3} & \frac{1}{3} & \frac{2}{3} & \frac{3}{3} & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{array}
$$

Формат входного файла

В первой строке записано число t ($1 \le t \le 1000$) — количество тестов.

В каждой из следующих t строк записано число k ($1 \le k \le 1000$) — номер дня, в который Черный Плащ хочет поймать Антиплаща.

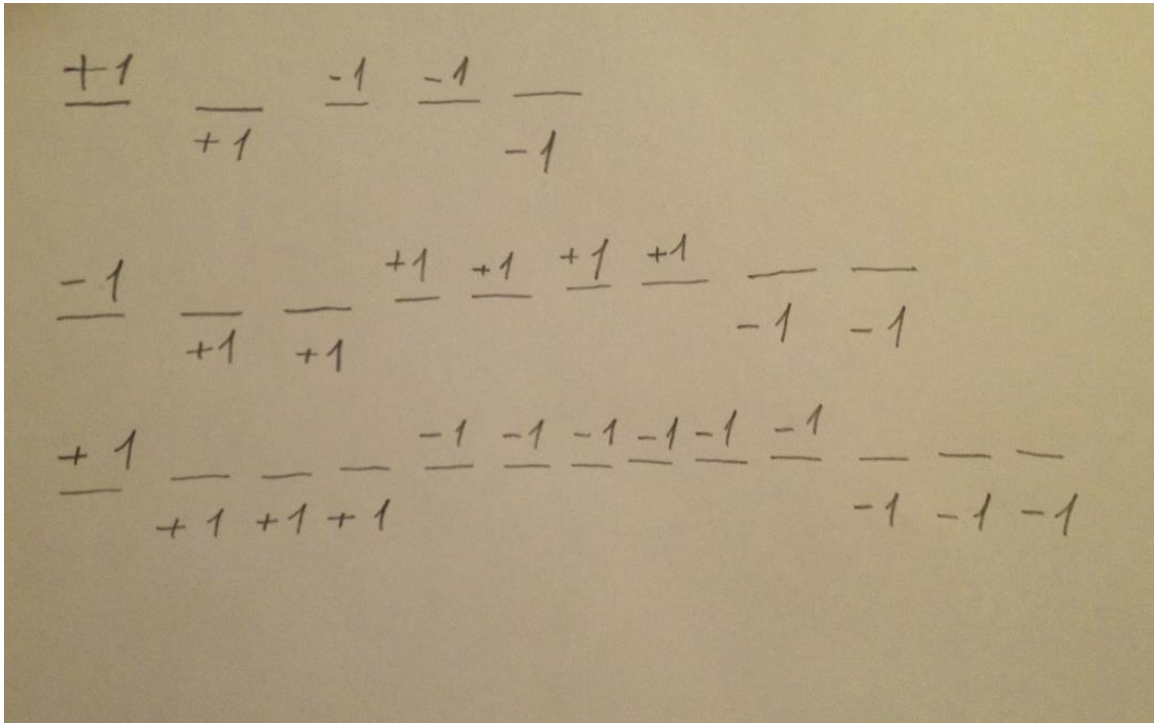
Формат выходного файла

Выведите t строк. В каждой строке должна быть записана рациональная дробь, характеризующая координаты банка, в том виде, в каком она была записана на оставленной Антиплащом бумажке.

Примеры

input.txt	output.txt
8 1 2 3 4 5 6 7 8	0/1 1/1 1/2 0/2 -1/2 -1/1 -2/1 -2/2

$$
\begin{array}{ccccccccc}
\cdots & -\frac{3}{1} & -\frac{2}{1} & -\frac{1}{1} & \frac{0}{1} & \frac{1}{1} & \frac{2}{1} & \frac{3}{1} & \cdots \\
\cdots & -\frac{3}{2} & -\frac{2}{2} & -\frac{1}{2} & \frac{0}{2} & \frac{1}{2} & \frac{2}{2} & \frac{3}{2} & \cdots \\
\cdots & -\frac{3}{3} & -\frac{2}{3} & -\frac{1}{3} & \frac{0}{3} & \frac{1}{3} & \frac{2}{3} & \frac{3}{3} & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots
\end{array}
$$

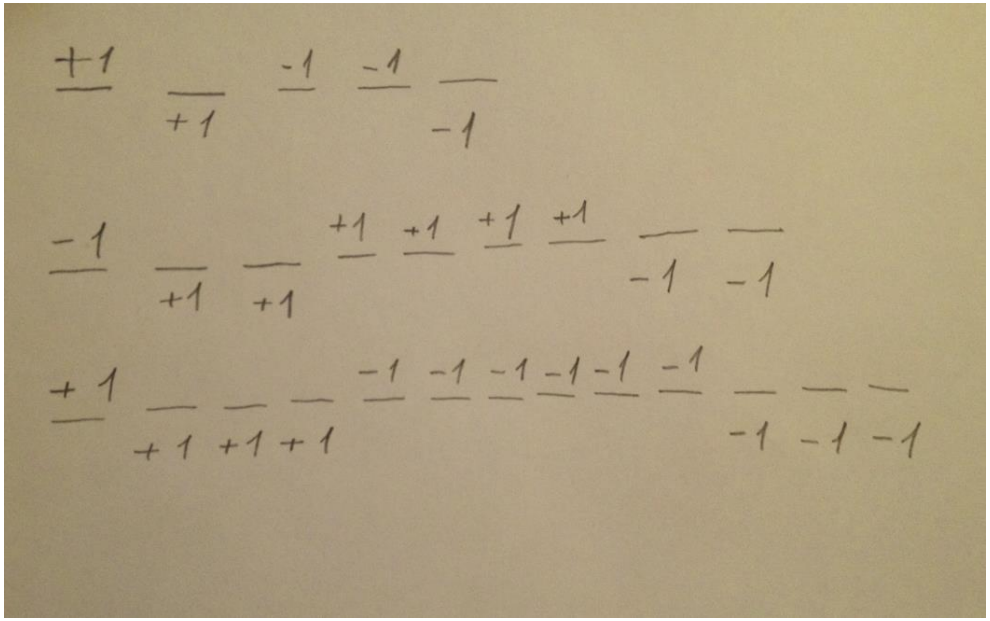


КОД ПРОГРАММЫ

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.Scanner;

public class B {

    public static void main(String[] args) throws FileNotFoundException{

        Scanner in = new Scanner(new File("input.txt"));
        PrintWriter out = new PrintWriter(new File("output.txt"));

        int t = in.nextInt();
        int k = in.nextInt();


        int a = 0;
        int b = 1;
        int z = 1;
        int y = 2;
        int x = 1;
        int n = 0;

        out.println(a+"/"+b);
        n += 1;
        if (n == t) {
            out.close();
            return;
        }
```

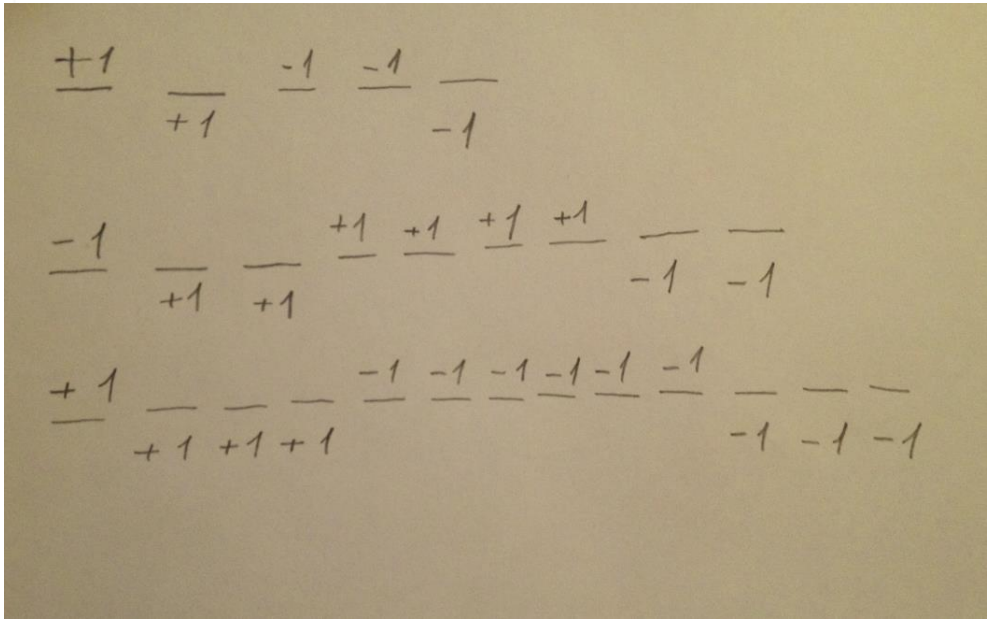
```
for(int i = 0 ; i < t ; i++){

        a += 1;
        out.println(a+"/"+b);
        n += 1;
        if (n == t) {
            out.close();
            return;
        }
        for (int j = 0 ; j < z; j++){
        b += 1;
        out.println(a+"/"+b);
        n += 1;
        if (n == t) {
            out.close();
            return;
        }
        }
        z += 1;
        for (int j = 0 ; j < y ; j++){
            a -=1;
            out.println(a+"/"+b);
            n += 1;
            if (n == t) {
                out.close();
                return;
            }
        }
        y+=2;
```

```
for (int j = 0; j < x ; j++){
    b -=1;
    out.println(a+"/"+b);
    n += 1;
    if (n == t) {
        out.close();
        return;
    }
}
x+=1;
```



```
a -= 1;
out.println(a+"/"+b);
n += 1;
if (n == t) {
    out.close();
    return;
}
for (int j = 0 ; j < z; j++){
b += 1;
out.println(a+"/"+b);
n += 1;
if (n == t) {
    out.close();
    return;
}
}
z += 1;
for (int j = 0 ; j < y ; j++){
    a +=1;
    out.println(a+"/"+b);
    n += 1;
    if (n == t) {
        out.close();
        return;
    }
    }
y+=2;
```

```
    for (int j = 0; j < x ; j++){
        b -=1;
        out.println(a+"/"+b);
        n += 1;
        if (n == t) {
            out.close();
            return;
        }
    }

    x+=1;


}
```



input — ...

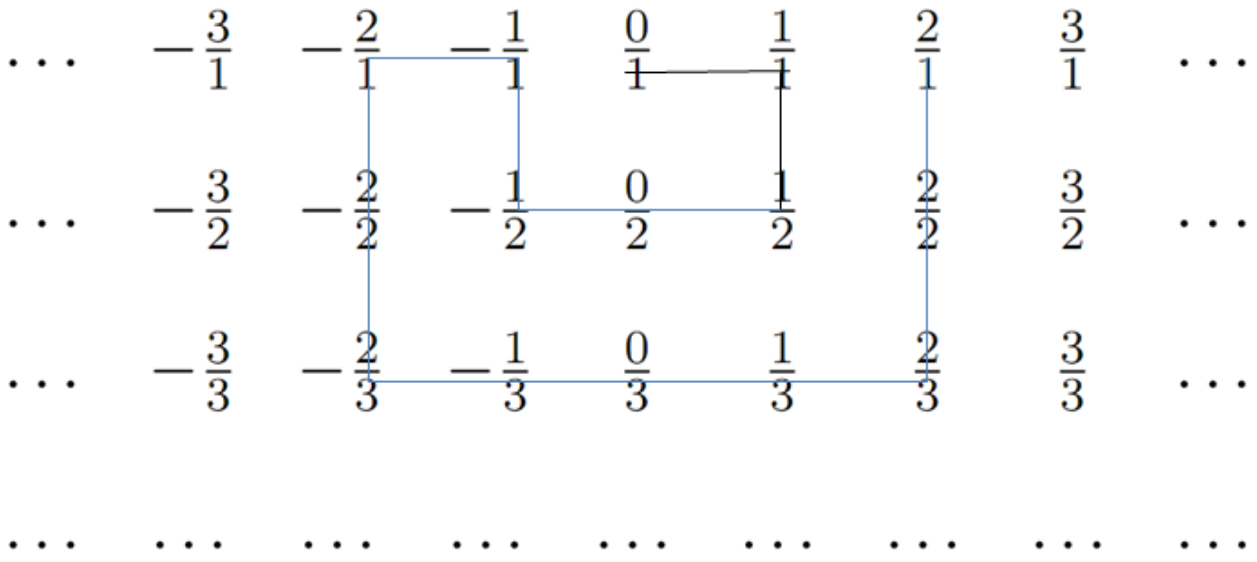
Файл Правка Формат Вид Справка

```
12
1
2
3
4
5
6
7
8
9
10
11
12
```

output...

Файл Правка Формат Вид Справка

```
0/1
1/1
1/2
0/2
-1/2
-1/1
-2/1
-2/2
-2/3
-1/3
0/3
1/3
```



ЗАДАЧА C. ПОБЕГ

Формат входного файла

В первой строке находится число n ($1 \le n \le 100000$) — размер массива.

В следующей строке через пробел записано n чисел $a_1, \ldots, a_n$ ($1 \le a_i \le 1000$) — элементы массива.

В следующей строке записано число q — количество вопросов системы безопасности.

В каждой из следующей q строк записано по 2 числа L и R ($1 \le L \le R \le n$) — левая и правая граница отрезка очередного вопроса системы.

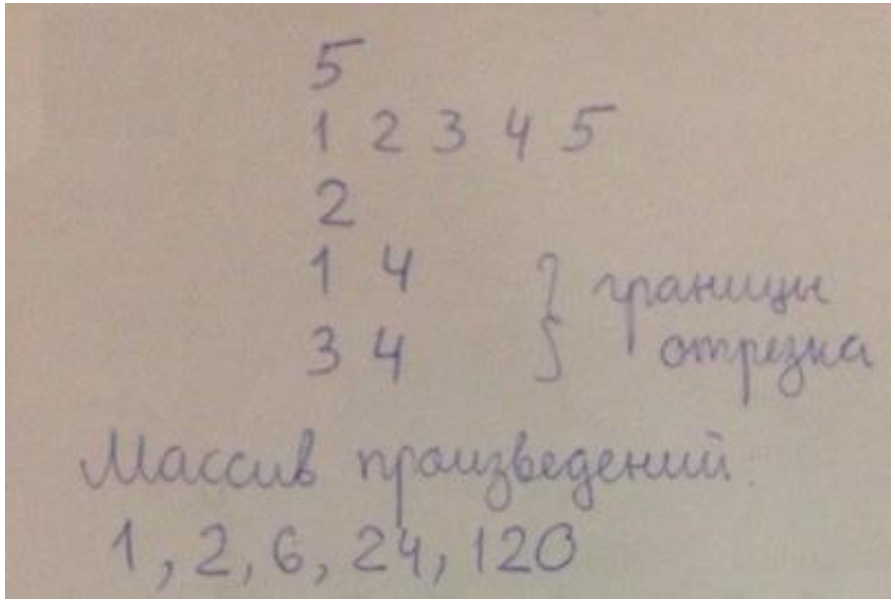
Формат выходного файла

Выведите q строк — ответы на вопросы системы безопасности.

В каждой строке должно содержаться единственное число — произведение всех элементов массива a на отрезке $[L, R]$, взятое по модулю 10007.

Примеры

input.txt	output.txt
5 1 2 3 4 5 4 1 2 2 3 3 4 4 5	2 6 12 20
6 1 4 2 8 5 7 6 1 6 2 6 3 6 4 6 5 6 6 6	2240 2240 560 280 35 7
2 101 101 1 1 2	194



1 запрос [1;4]: Для этого достаточно взять число из массива произведений с инд. [4], т.е. ответ: (24)

2 запрос [3;4]
Берём число с инд. [4], но необх. учесть, что эл-ты с индексами [1] и [2] нам не нужны, => 24 / 2 = (12).
Деление на эл-т массива произведений с инд. [2].

КОД ПРОГРАММЫ

```java
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.math.BigInteger;
import java.util.StringTokenizer;

public class zadacha_C
{

    FastScanner in;
    PrintWriter out;

 public void run()
 {
     try
     {
         in = new FastScanner(new File("input.txt"));
         out = new PrintWriter(new File("output.txt"));

         solve();

         out.close();
     }
     catch (IOException e)
     {
         e.printStackTrace();
     }
 }
```

```java
    class FastScanner
    {
        BufferedReader  br;
        StringTokenizer st;

        FastScanner(File f)
        {
            try
            {
                br = new BufferedReader(new FileReader(f));
            }
            catch (FileNotFoundException e)
            {
                e.printStackTrace();
            }
        }

      String next()
      {
          while (st == null || !st.hasMoreTokens())
          {
              try
              {
                  st = new StringTokenizer(br.readLine());
              }
              catch (IOException e)
              {
                  e.printStackTrace();
              }
          }
          return st.nextToken();
      }

      int nextInt()
      {
          return Integer.parseInt(next());
      }
  }

  public static void main(String[] arg)
  {
      new zadacha_C().run();
  }
}
```

```
    public void solve() throws IOException
    {
        int n = in.nextInt();
        int a[] = new int[n];
        int L, R;

        for (int i = 0; i < n; i++)
        {
            a[i] = in.nextInt();
        }
        long Masproizv[] = new long[n];
        int mod = 10007;
        Masproizv[0] = a[0];
        for (int i = 1; i < n; i++)
        {
            Masproizv[i] = Masproizv[i - 1] * a[i] % mod;
        }
        int q = in.nextInt();
        for (int i = 0; i < q; i++)
        {
            L = in.nextInt() - 1;
            R = in.nextInt() - 1;
            long k = Masproizv[R];
            if (L > 0)
            {
long gran = BigInteger.valueOf(Masproizv[L - 1]).modInverse(BigInteger.valueOf(mod)).longValue();

                k = k * gran % mod;
            }
            out.println(k);
        }

        out.close();
    }
```

ЗАДАЧА A. ТРУБОПРОВОД

Формат входного файла

В первой строке записаны 2 целых числа n и m ($1 \le n \le 1000, 0 \le m \le 1000$). m — это количество труб в трубопроводе, а n — количество соединений между ними.

Каждая из следующих m строк описывает очередную трубу. В ней содержатся 2 целых числа a и b ($1 \le a, b \le n$, $a \ne b$) — номера стыков, которые соединяет данная труба. Каждая труба описана ровно один раз. Каждая пара стыков соединена не более, чем одной трубой.

В последней строке находится число v ($1 \le v \le n$) — соединение, в котором Черный Плащ дважды видел Ликвигада.

Формат выходного файла

Если Черный Плащ заснул и только поэтому видел Ликвигада дважды на стыке v, выведите единственное число -1.

Иначе в первой строке выведите число k — количество труб, которое понадобилось преодолеть Ликвигаду между двумя проходами через соединение v.

Затем во второй строке выведите k чисел — номера соединений между трубами, пройденных Ликвигадом за это время. Первым из этих чисел должно быть v, а последним — номер соединения, из которого Ликвигад вернулся в v.

Если существует несколько ответов, выведите любой.

Примеры

input.txt	output.txt
3 3 1 2 2 3 3 1 1	3 1 2 3
5 3 1 2 2 3 2 4 3	-1

КОД ПРОГРАММЫ

```
import java.util.*;
import java.io.*;

public class NMA
{

    static ArrayList<Integer> way;
    static int[][] x;
    static boolean[] used;
    static PrintWriter out;

    public static void main(String[] args) throws IOException
    {

        Scanner in = new Scanner(new File("input.txt"));
        out = new PrintWriter(new File("output.txt"));

        int n = in.nextInt();
        int m = in.nextInt();

        x = new int[n][n];
        used = new boolean[n];
        way = new ArrayList<Integer>();

        for ( int i = 0 ; i < m ; i++ )
        {
            int a = in.nextInt()-1;
            int b = in.nextInt()-1;

            x[a][b] = x[b][a] = 1;
        }

        int v = in.nextInt() - 1;

        dfs(v);

        out.print(-1);
        out.close();


    }
```

```
static void dfs(int v)
{
    used[v] = true;
    way.add(v);

    if (way.size() > 2 && x[v][way.get(0)] == 1)
    {

        out.println( way.size() );

        for(Integer cur : way)
            out.print( (cur+1)+" " );

        out.close();
        System.exit(0);
    }

    for ( int i=0 ; i < x[v].length ; i++)
    {
        if(!used[i] && x[v][i]== 1)
            dfs(i);

    }

    way.remove(way.size()-1);
}
```