22. HRVATSKA INFORMATIČKA OLIMPIJADA

ZAGREB, 2. TRAVNJA 2016.

# Croatian Olympiad in Informatics

Zagreb, April 2$^{\text{nd}}$ 2016

## Tasks

| Task | Time limit | Memory limit | Score |
|---|---|---|---|
| **Dijamant** | 2 seconds | 512 MiB | 100 |
| **Palinilap** | 1 second | 512 MiB | 100 |
| **Relay** | 2 seconds | 512 MiB | 100 |
| **Torrent** | 2 seconds | 512 MiB | 100 |
| **Total** | | | 400 |

ministarstvo
znanosti
obrazovanja
i sporta

HRVATSKI SAVEZ
INFORMATIČARA

TEH
KUL
TURA
HRVATSKA ZAJEDNICA
TEHNIČKE KULTURE

# Task: Dijamant

We are observing class declarations in an object-oriented programming language similar to C++. Each class declaration is of the form "$K$ : $P_1$ $P_2$ ... $P_k$ ;" where $K$ is the name of the new class being declared, and $P_1, P_2, \ldots, P_k$ the names of classes being *inherited* by class $K$. For example, "`shape : ;`" is a declaration of class "`shape`" that does not inherit any other class, whereas "`square : shape rectangle ;`" is a declaration of class "`square`" that inherits classes "`shape`" and "`rectangle`".

If class $K_1$ inherits class $K_2$, class $K_2$ inherits class $K_3$, and so on, up to class $K_{m-1}$ that inherits class $K_m$, then we say that all classes $K_1, K_2, \ldots, K_{m-1}$ are *derived* from class $K_m$. The rules of the programming language forbid circular definitions, so it is not allowed to have a class derived from itself. In other words, the class hierarchy forms a *directed acyclic graph*. Additionally, it is not allowed for a so-called *diamond* to appear in the class hierarchy. A diamond consists of four different classes $A, B, X, Y$ such that it holds:

- Classes $X$ and $Y$ are derived from $A$.
- Class $B$ is derived from both $X$ and $Y$.
- Neither is class $X$ derived from $Y$, nor is class $Y$ derived from $X$.

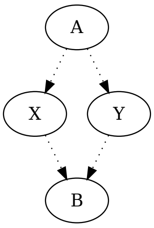


Figure 1: A diamond

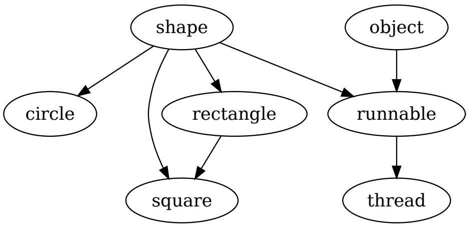


Figure 2: The hierarchy after processing all declarations from the first sample test

You are given a series of class declarations to be processed sequentially, and determine for each one whether it is correctly declared. The correctly declared classes are added to the hierarchy, while the incorrect ones are discarded. Declaration "$K$ : $P_1$ $P_2$ ... $P_k$ ;" is *correctly declared* if the following holds:

1. Class $K$ hasn't been declared yet.
2. All classes $P_1, P_2, \ldots, P_k$ have been previously declared. Notice that this condition ensures that a class can never be derived from itself, or that cycles cannot exist in the class hierarchy.
3. By adding class $K$ that inherits $P_1, P_2, \ldots, P_k$ the class hierarchy remains in order, that is, not a single diamond is formed.

Write a programme that will process the declarations respectively as described above and determine the correctness of each one of them.

## Input

The first line of input contains the integer $n$ – the number of declarations. Each of the following $n$ lines contains a single declaration in the form of "$K$ : $P_1$ $P_2$ ... $P_k$ ;" where $P_1, P_2, \ldots, P_k$ is a series of zero, one or more classes that class $K$ inherits. All class names in a single declaration $K, P_1, P_2, \ldots, P_k$ are mutually different. Each class name is a string of at most 10 lower case letters of the English alphabet. All the elements of a declaration (the class names and characters "`:`" and "`;`") are separated by exactly one space. In each specific declaration, for the number of classes $k$ it holds $0 \le k \le 1\,000$.

## Output

You must output $n$ lines. The $i^{th}$ line must contain "`ok`" if the $i^{th}$ declaration is correct, and "`greska`" if it isn't.

## Scoring

| Subtask | Score | Limitations |
|---|---|---|
| 1 | 13 | $1 \le n \le 100$, the correctness can be determined by considering only condition 1. |
| 2 | 14 | $1 \le n \le 100$, the correctness can be determined by considering only conditions 1 and 2. |
| 3 | 29 | $1 \le n \le 100$. |
| 4 | 44 | $101 \le n \le 1\,000$. |

## Sample tests

**input**

```
10
shape : ;
rectangle : shape ;
circle : shape ;
circle : ;
square : shape rectangle ;
runnable : object ;
object : ;
runnable : object shape ;
thread : runnable ;
applet : square thread ;
```

**output**

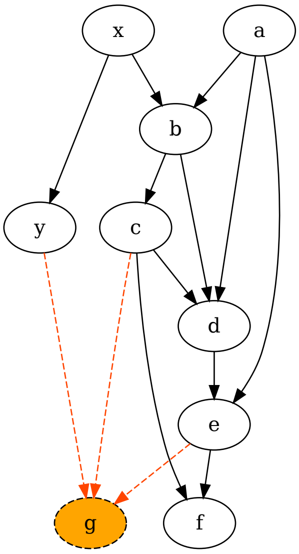
```
ok
ok
ok
greska
ok
greska
ok
ok
ok
greska
```

**input**

```
9
a : ;
x : ;
b : a x ;
c : b ;
d : a b c ;
e : d a ;
f : c e ;
y : x ;
g : c y e ;
```

**output**

```
ok
ok
ok
ok
ok
ok
ok
ok
greska
```



**Clarification of the first example:**

- The fourth declaration is incorrect because class "`circle`" has already been defined in the third row.
- The sixth declaration is incorrect because class "`object`" hasn't been defined yet.
- The eighth declaration is correct because class "`object`" has now been declared, and the sixth declaration was discarded, so class "`runnable`" hasn't been defined yet.
- The tenth declaration is incorrect because otherwise the following diamond forms: "`shape`", "`applet`", "`square`", "`runnable`".

**Clarification of the second example:**

- The tenth declaration is incorrect because otherwise the following diamond forms: "`x`", "`g`", "`y`", "`d`" (and many other).

# Task: Palinilap

A *palindrome* is a word that reads the same backwards as forwards. For example, "`a`", "`abba`" and "`anavolimilovana`" are palindromes A *sample* is a string of one or more lower case letters of the English alphabet, and the weight of a sample is the number of its substrings (words) that are palindromes, counting each word occurrence separately.

More precisely, let $w$ be a sample of length $n$. The word $w_{a,b}$ is obtained by taking all characters from position $a$ to position $b$ in sample $w$. The weight of sample $w$ is defined as the number of different pairs of integers $a$, $b$ ($1 \le a \le b \le n$) such that the word $w_{a,b}$ is a palindrome.

You are given the sample $w$. It can either be left unchanged, or exactly one position can be chosen and the letter on that position arbitrarily changed. Find the maximal possible sample weight that can be obtained as described above.

## Input

The first line of input contains the given sample $w$ – a string of lower case letters of the English alphabet.

## Output

You must output the required maximal possible weight.

## Scoring

Let $n$ be the length of the given sample.

| Subtask | Score | Limitations |
|---|---|---|
| 1 | 17 | $1 \le n \le 100$ |
| 2 | 37 | $101 \le n \le 5\,000$ |
| 3 | 46 | $5001 \le n \le 100\,000$ |

## Sample tests

| | | |
|---|---|---|
| **input** | **input** | **input** |
| `aaaa` | `baccb` | `slavko` |
| **output** | **output** | **output** |
| `10` | `9` | `7` |

**Clarification of the first example:** Each substring from the sample already is a palindrome, so it is best left unchanged.

**Clarification of the second example:** If we change the second letter of the sample to "`c`", we will get the sample "`bcccb`" with a weight of 9.

# Task: Relay

A fleet of fishing boats set sail on the open sea from an Adriatic island. The position of each fishing boat is described with a point in the standard coordinate system, whereas the island is described with a *convex polygon*. The boats communicate via radio devices, and the island represents an obstacle for the radio waves. More precisely, if boat $a$ transmits a message, then boat $b$ receives the message if and only if the line segment connecting the positions of $a$ and $b$ does not cross the interior of the island (it is allowed to have the line segment touch the sides and vertices of the island).

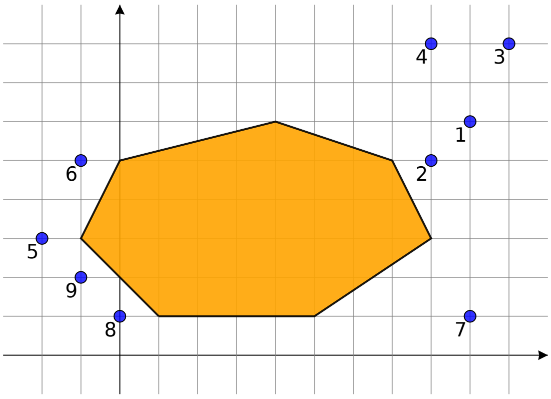


Figure 3: In the first sample test, ships 2, 3, 4 and 7 will receive the original Mayday message, whereas ships 6 and 8 will receive the Relay message.

When ship $a$ gets in trouble, it transmits the so-called *Mayday* message asking for help. All ships that receive the Mayday message immediately send the so-called *Relay* message repeating that ship $a$ needs help. If a ship only receives the Relay message (and not the original Mayday message), then it sends nothing.

You are given the positions of $n$ ships denoted with integers from 1 to $n$ and the location of the island. Ship number 1 has found itself in trouble and sends the Mayday message. Determine the total number of ships that will receive either the original Mayday message or any of the Relay messages.

## Input

The first line of input contains the integer $n$ – the number of ships. The $k^{th}$ of the following $n$ lines contains two integers $x_k$ and $y_k$ $(-10^9 \leq x_k, y_k \leq 10^9)$ – the coordinates of the $k^{th}$ ship. All ships are located on different coordinates, not a single ship is located on a side or inside the polygon.

The following line contains the integer $m$ – the number of vertices of the convex polygon describing the island. The $k^{th}$ of the following $m$ lines contains two integers $x'_k$ and $y'_k$ $(-10^9 \leq x'_k, y'_k \leq 10^9)$ – the coordinates of the $k^{th}$ vertex of the polygon. The polygon's vertices are given in the counter-clockwise direction and form a convex polygon. No two adjacent edges will be parallel.

## Output

You must output the required total number of boats that will receive one of the messages.

## Scoring

| Subtask | Score | Limitations |
|---|---|---|
| 1 | 18 | $1 \le n \le 300, 3 \le m \le 300$ |
| 2 | 19 | $1 \le n \le 3000, 3 \le m \le 3\,000$ |
| 3 | 20 | $1 \le n \le 100\,000, 3 \le m \le 300$ |
| 4 | 43 | $1 \le n \le 100\,000, 3 \le m \le 100\,000$ |

## Sample tests

### input

```
9
9 6
8 5
10 8
8 8
-2 3
-1 5
9 1
0 1
-1 2
7
1 1
5 1
8 3
7 5
4 6
0 5
-1 3
```

### output

```
6
```

### input

```
4
-1 0
-3 -20
6 10
5 10
4
3 0
3 1
0 10
0 -10
```

### output

```
2
```

# Task: Torrent

Mirko works at a data centre and today's task is to copy a file sized 1 GiB to $n$ computers. The computers are denoted with integers from 1 to $n$ and are connected so that they form a so-called *tree*. More precisely, $n - 1$ pairs of computers are directly connected via network cable in a way that there is a unique path between each pair of computers.

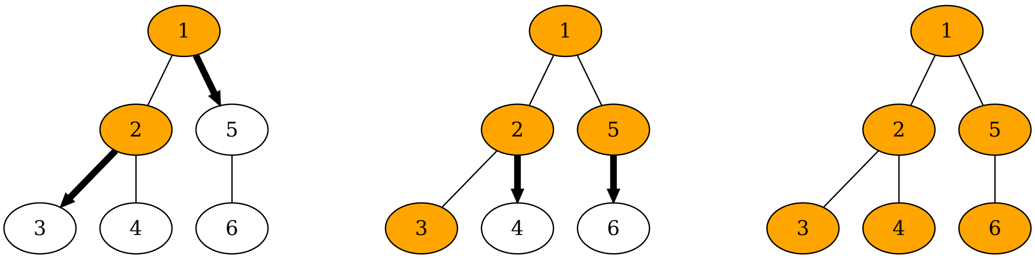


Figure 4: In the first sample test, it takes two minutes for the file to be copied to all computers.

Initially, Mirko manually placed the file on two different computers – computer $a$ and computer $b$ and is now writing commands that will copy the file to all other computers. The file can be copied from computer $x$ to computer $y$ only if the two computers are directly connected, and the copying process takes exactly one minute. At any moment, each individual computer can take part in at most one copying process, but it is allowed to have the file being copied between arbitrarily many different pairs of computers at the same time. Therefore, when the copying process ends from computer $x$ to computer $y$, it is possible in the next minute to copy the file from computer $x$ to computer $w$ and from computer $y$ to computer $z$.

Determine the minimal amount of time it takes for the file to be copied to all computers.

## Input

The first line of input contains the integer $n$ and two different integers $a$ and $b$ ($1 \leq a, b \leq n$) – the number of computers and the labels of the computers already containing the file. Each of the following $n - 1$ lines contains two different integers $x$ and $y$ ($1 \leq x, y \leq n$) – the labels of the computers directly connected via network cable. The computer network forms a tree, as described in the task.

## Input

You must output the required minimal amount of time in minutes.

## Scoring

| Subtask | Score | Limitations |
|---|---|---|
| 1 | 31 | $2 \leq n \leq 1\,000$ |
| 2 | 69 | $1\,000 \leq n \leq 300\,000$ |

## Sample tests

**input**

```
6 2 1
1 2
2 3
2 4
1 5
5 6
```

**output**

```
2
```

**input**

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

**output**

```
4
```

**input**

```
17 1 2
1 3
1 4
4 6
6 7
3 8
3 9
3 10
1 13
13 5
13 11
13 12
13 14
14 15
15 16
15 17
14 2
```

**output**

```
5
```

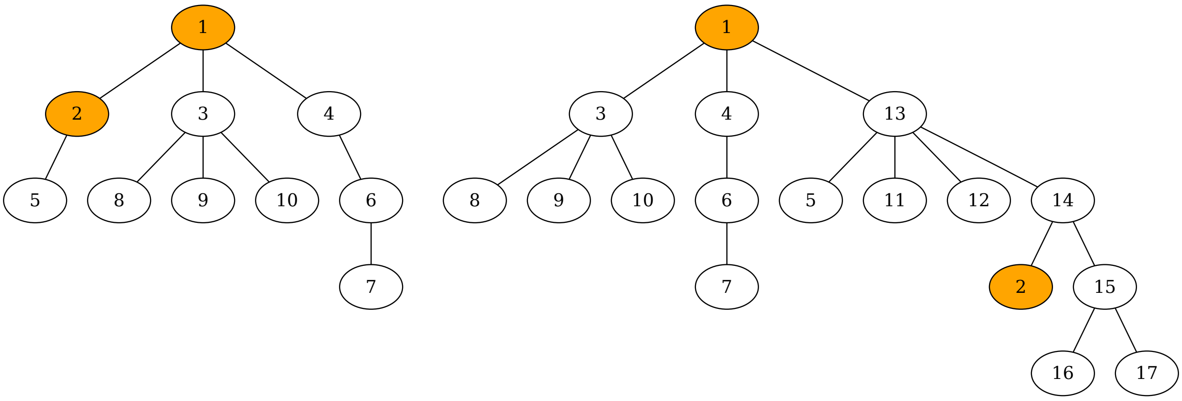


Figure 5: Illustrations of the second and third sample test