# KAIST 8th ACM-ICPC Mock Competition 

Sponsored By:

## DEVSISTERS

## DEVSISTERS

NAVER D2
NAVER [2]

## STARTLINK

Page 1 of 21

## Rules

- This contest is KAIST 8th ACM-ICPC Mock Competition.
- This contest is sponsored by DEVSISTERS, NAVER D2, and STARTLINK.
- This contest starts at 14:00 and ends at 19:00, Oct 3rd 2018, Korean Standard Time(GMT +9).
- You can only participate in a team of three students.
- Any use of network is prohibited during the competition, except for submitting source codes and accessing language reference sites. Here are examples of allowed reference sites.

```
- C/C++ : http://en.cppreference.com/w/
- Java : https://docs.oracle.com/javase/8/docs/api/
- Python: https://docs.python.org/
- Kotlin: https://kotlinlang.org/docs/reference/
```

- Each team can bring up to 25 pages of printed, A4-sized reference materials for use during the competition.
- This contest consists of 12 problems.
- Problems are NOT sorted by difficulty.
- Solutions to problems submitted for judging are called runs. Each run is judged as accepted or rejected, and the team is notified of the results.
- Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked first by least total time and, if need be, by the earliest time of submittal of the last accepted run.
- The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submittal of the first accepted run plus 20 penalty minutes for every previously rejected run for that problem. There is no time consumed for a problem that is not solved.
- Memory limit for each problem is fixed as follows:
- C11/C++17: 1024MB
- Java/Kotlin : 1536MB
- PyPy 2/Python 3: 2048MB
- Each problem may have different time limit.


## Problem list

| $\#$ | Problem Name | Time limit <br> (All Languages) |
| :---: | :--- | :---: |
| A | Coloring Roads | 4 seconds |
| B | Dumae | 2 seconds |
| C | Electronic Circuit | 1 second |
| D | Fake Plastic Trees | 1 second |
| E | Fascination Street | 1 second |
| F | Fractions | 1 second |
| G | Game on Plane | 1 second |
| H | Histogram Sequence | 2 seconds |
| I | Repetitive Palindrome | 1 second |
| J | Rising Sun | 1 second |
| K | Utilitarianism | 5 seconds |
| L | Voronoi Diagram Returns | 10 seconds |

## Problem A. Coloring Roads

Input file:
Output file: standard output
Time limit: 4 seconds
In RUN-land, there are $n$ cities numbered 1 to $n$. Some pairs of cities are connected by a bidirectional road. It happens that there are $n-1$ roads in total and that for any two cities, and there is a unique path from one to the other.
The city number 1 is the capital. Initially all roads have no color. Alex, the king of RUN-land asks you to perform the following query $Q$ times.

- $u c m$ : Given a city $u$, a color $c$, and an integer $m$, color all the roads on the unique path from $u$ to the capital in the color $c$. Even if a road already has a color, change its color to $c$. After coloring, compute the number of colors in which exactly $m$ roads are colored.

Given $Q$ queries in total, compute the answer for the second part of each query.

## Input

The first line of the input contains three integers $n, C, Q(1 \leq n, C, Q \leq 200,000)$, separated by a single space, which are the number of cities in RUN-land, the number of possible colors, and the number of queries, respectively. Each of the next $n-1$ lines contains two integers $u, v(1 \leq u, v \leq n)$ meaning that there is a bidirectional road directly connecting the cities numbered $u$ and $v$.
Each of the next $Q$ lines contains a query, which contains 3 integers $u, c, m$ as described in the statement. $(1 \leq u \leq n, 1 \leq c \leq C, 0 \leq m \leq n-1)$

## Output

Print $Q$ lines, one for each query. Each line must contain one integer, the answer to the corresponding query.

## Example

|  |  | standard input |  | standard output |
| :--- | :--- | :--- | :--- | :--- |
| 6 | 5 | 5 | 1 |  |
| 1 | 3 | 2 |  |  |
| 2 | 3 | 2 |  |  |
| 1 | 4 | 3 |  |  |
| 6 | 3 | 1 |  |  |
| 5 | 2 |  |  |  |
| 5 | 1 | 3 |  |  |
| 6 | 2 | 2 |  |  |
| 2 | 3 | 1 |  |  |
| 4 | 4 | 1 |  |  |
| 1 | 5 | 0 |  |  |

## Note

The answer for the last query is 1 since color 5 is used in 0 roads.

## Problem B. Dumae

Input file: standard input
Output file: standard output
Time limit: 2 seconds

Do you know Dumae? It is a nickname of the most famous restaurant nearby KAIST, Dumae Charcoal-grilled Barbecue. Because Dumae is a very famous restaurant, lots of KAIST students stand in line even though it has not opened yet. Students wonder how long they have to wait, so they started to guess their order.
There are $N$ students in waiting line and each of them has a distinct student ID from 1 to $N$. Student $i$ (student with student ID $i$ ) guessed that he/she is either $L_{i}$-th, $\left(L_{i}+1\right)$-th, $\cdots,\left(R_{i}-1\right)$-th, or $R_{i}$-th person in the line. (i.e. the number of people standing relatively in front of him/her is in the interval [ $\left.L_{i}-1, R_{i}-1\right]$ ) Also, $M$ claims are made, of which the $i$-th says that student $v_{i}$ can see student $u_{i}$ in the waiting line. It means student $u_{i}$ is relatively in front of student $v_{i}$.
You wonder if all of students' guesses and claims were right. Find an order of waiting line that satisfies all the guesses and claims, or report that such an order does not exist.

## Input

The first line contains two space-separated integers $N, M .(1 \leq N \leq 300,000,0 \leq M \leq 1,000,000)$
In the next $N$ lines, two space-separated integers $L_{i}, R_{i}$ are given. $\left(1 \leq L_{i} \leq R_{i} \leq N\right)$
In the next $M$ lines, two space-separated integers $u_{i}, v_{i}$ are given. $\left(1 \leq u_{i} \leq N, 1 \leq v_{i} \leq N, u_{i} \neq v_{i}\right)$

## Output

If there is no answer that satisfies the condition, print -1 .
Otherwise, print $N$ lines. In the $i$-th line, print the student ID of the $i$-th student from the front.

## Examples

|  | standard input |  |  |
| :--- | :--- | :--- | :--- |
| 3 | 3 | -1 | standard output |
| 1 | 3 |  |  |
| 1 | 3 |  |  |
| 1 | 3 |  | 1 |
| 1 | 2 | 2 |  |
| 2 | 3 | 3 |  |
| 3 | 1 |  |  |
| 3 | 3 |  |  |
| 1 | 3 |  |  |
| 1 | 3 |  |  |
| 1 | 3 |  |  |
| 1 | 2 | 3 |  |
| 1 | 3 |  |  |

## Problem C. Electronic Circuit

Input file: standard input<br>Output file: standard output<br>Time limit: 1 second

Joon is taking General Physics II and he is now studying electronic circuits. An electronic circuit consists of several nodes and undirected wires each connecting two distinct nodes. Moreover, a circuit has two distinctive end nodes; a source node and a sink node, where a voltage is applied (usually it is applied by additional wire with a battery connecting the two nodes, but we will neglect it). Each wire has a resistance, and Joon should know how to calculate the composite resistance of a circuit.
By the way, Joon hates complicated things. So he only cares about circuits that can be made by series and parallel compositions, since they are easy to calculate the composite resistance. He calls them nice circuits; formally, a nice circuit can be defined as follows.

- A circuit with a single wire connecting two end nodes is nice.
- A circuit obtained by merging the sink node of a nice circuit $C_{1}$ and the source node of a nice circuit $C_{2}$ into a single node is nice. The source node and the sink node of the obtained circuit are the source node of $C_{1}$ and the sink node of $C_{2}$, respectively.
- A circuit obtained by merging the two source nodes of nice circuits $C_{1}$ and $C_{2}$ into a single node, and merging the two sink nodes of $C_{1}$ and $C_{2}$ into a single node, is nice. The two end nodes of the obtained circuit are the respective merged end nodes.


Figure: Illustration of the definition of nice circuit.
He made a circuit with his wires to calculate the composite resistance, but his friend Pringles screwed up his circuit, so now Joon does not know what the end nodes are. To make things worse, he is not even sure whether the circuit is nice or not.
Joon will give you the circuit. He kindly asks you whether the circuit can be nice by appropriately choosing two end nodes. Be careful that there may be multiple wires connecting two nodes.

## Input

The first line contains two integers, $n$ and $m(2 \leq n \leq 100,000,1 \leq m \leq 300,000)$, where $n$ is the number of nodes and $m$ is the number of wires. All nodes are numbered from 1 to $n$.

In the following $m$ lines, each line contains two integers $u$ and $v(1 \leq u, v \leq n, u \neq v)$, which represents a wire connecting $u$ and $v$. It is guaranteed that every node is attached to at least one wire; otherwise the node does not exist!

## Output

Print Yes if the given circuit can be nice, or No otherwise.

## Examples

|  | standard input |  |
| :--- | :--- | :--- |
| 4 | 6 | Yes |
| 1 | 2 |  |
| 2 | 3 |  |
| 2 | 3 |  |
| 3 | 4 |  |
| 1 | 4 |  |
| 1 | 4 |  |
| 4 | 6 |  |
| 1 | 2 |  |
| 1 | 3 |  |
| 1 | 4 |  |
| 2 | 3 |  |
| 2 | 4 |  |
| 3 | 4 |  |
| 9 | 12 |  |
| 1 | 9 |  |
| 1 | 4 |  |
| 5 | 4 |  |
| 6 | 5 |  |
| 1 | 5 |  |
| 8 | 1 |  |
| 3 | 6 |  |
| 6 | 8 |  |
| 3 | 8 |  |
| 2 | 9 |  |
| 9 | 7 |  |
| 7 | 2 |  |

## Problem D. Fake Plastic Trees

Input file:
Output file: standard output
Time limit: 1 second

Tree is a recursive structure, which is either:

- Empty. Empty tree is denoted as -1 and has a size of 0 .
- Non-empty. Non-empty tree $T$ is denoted as a pair of two trees $\left(T_{1}, T_{2}\right)$, where $T_{1}$ is called left subtree of $T$, and $T_{2}$ is called right subtree of $T$. If $T=(-1,-1)$, then we call such $T$ a leaf. Leaf has a size of 1 , and non-leaf has a size of $\left|T_{1}\right|+\left|T_{2}\right|$, where $\left|T_{1}\right|$ is the size of $T_{1}$, and $\left|T_{2}\right|$ is the size of $T_{2}$.

A non-empty tree $T$ is a Fake Plastic Tree, if the tree is balanced. Formally, Let $T=\left(T_{1}, T_{2}\right)$. If $\left|T_{1}\right|=\left|T_{2}\right|$ or $\left|T_{1}\right|=\left|T_{2}\right|+1$ holds, then $T$ is a Fake Plastic Tree.
In computer science, trees are commonly used as a data structure, and they are stored in a memory. At first, there are no trees in the memory, and only an imaginary null pointer exists (which corresponds to empty tree, -1 ). You can allocate a tree in the memory, by setting $T_{1}$ and $T_{2}$ as either a null pointer or a pointer of an existing tree. Then, the memory is extended by adding $T=\left(T_{1}, T_{2}\right)$ into its structure Note that pointer can be described as a small integer, reducing the need for explicitly storing the whole tree.
Formally, memory $M$ is an inductive structure, which at first contains only empty tree -1 . $(M=\{-1\})$. You can expand the memory with following operation $M \leftarrow M \cup\left\{\left(T_{1}, T_{2}\right)\right\}$, where $T_{1} \in M, T_{2} \in M$. If a tree $T$ is inserted in $i$-th stage, then it has the index $i-1$. For a tree with index $i$, their subtrees can be represented as a pair of integer in range $[-1, i-1]$.
Your task is to construct a memory $M$, which satisfies the following:

- Every tree in $M$ is either empty or Fake Plastic Tree.
- $M$ has at most 125 non-empty trees.
- There exists a tree $T \in M$, where $|T|=N$ holds. $N$ is an integer, and is given as an input.


## Input

The first line contains a single integer $T$, the number of test cases. ( $1 \leq T \leq 2,000$ )
In the next $T$ lines, a single integer $N$ is given, which indicates the number of leaves your tree should contain. $\left(1 \leq N \leq 10^{18}\right)$

## Output

For each case, you should print $V+2$ lines, where $V$ is the number of non-empty trees in $M$. $(1 \leq V \leq 125)$.
In the first line, you should print single integer $V$.
In the next $V$ lines, you should print two space-separated integer $L_{i}, R_{i}$, which indicates the index of left subtree and right subtree for a tree with index $i .\left(-1 \leq L_{i}, R_{i} \leq i-1\right)$.
In the $(V+2)$-th line, you should print $P$, the index of the tree which contains $N$ nodes. $(0 \leq P \leq V-1)$. It is guaranteed that an answer always exists under the given condition.

## Example

| standard input | standard output |
| :---: | :---: |
| 4 | 1 |
| 1 | -1 -1 |
| 2 | 0 |
| 3 | 3 |
| 4 | -1 -1 |
|  | -1 -1 |
|  | 01 |
|  | 2 |
|  | 3 |
|  | -1 -1 |
|  | 00 |
|  | 10 |
|  | 2 |
|  | 5 |
|  | -1 -1 |
|  | 00 |
|  | 00 |
|  | 21 |
|  | 12 |
|  | 3 |

## Note



Figure: Illustration of output for $N=4$ in the example.

## KAIST 8th ACM-ICPC Mock Competition

RUN@KAIST, Sponsor: DEVSISTERS, NAVER D2, STARTLINK, October 3rd, 2018

## Problem E. Fascination Street

## Input file: standard input <br> Output file: standard output <br> Time limit: 1 second

A street named Fascination Street is divided into $N$ equal length of blocks. For each block $i(1 \leq i \leq N)$, it has block $i-1$ in its left side if $i>1$, and block $i+1$ in its right side if $i<N$.
Unlike its name, the street is infamous to be a dark and eerie place in the night. To solve this, Robert decided to install the streetlight for some of the blocks. The cost of installing a streetlight for $i$-th block is $W_{i}$, and the total cost is the sum of each installation cost. After installing, every block should either have a streetlight, or have a streetlight in it's left or right block.
Robert also has some tricks to reduce the cost. Before installing the streetlight, Robert selects two distinct blocks $i$ and $j$, and exchange their position. After the operation, the cost of installation is exchanged. In other words, the operation simply swaps the value of $W_{i}$ and $W_{j}$. This operation have no cost, but Robert can only perform it at most $K$ times.
Now, given the array $W$ and the maximum possible number of operations $K$, you should find the minimum cost of lighting the whole street.

## Input

The first line contains two space-separated integers $N, K . N$ is the number of blocks, and $K$ is the maximum possible number of operations. ( $1 \leq N \leq 250000,0 \leq K \leq 9$ )
The second line contains $N$ space-separated integers $W_{1}, W_{2}, \cdots, W_{N}$, where $W_{i}$ is the cost of installing a streetlight for $i$-th block. $\left(0 \leq W_{i} \leq 10^{9}\right)$

## Output

Print a single integer which contains the minimum cost of lighting the whole street.

## Examples

| standard input |
| :---: |
| $\begin{array}{lllll} 5 & 0 & & & \\ 1 & 3 & 10 & 3 & 1 \end{array}$ |
| standard output |
| 4 |
| standard input |
| $\begin{array}{lllll} \hline 5 & 1 & & & \\ 1 & 3 & 10 & 3 & 1 \end{array}$ |
| standard output |
| 2 |
| standard input |
| $\begin{array}{llllllllll} 12 & 0 \\ 317 & 448 & 258 & 208 & 284 & 248 & 315 & 367 & 562 & 500 \end{array} 426390$ |
| standard output |
| 1525 |
| standard input |
| $\begin{array}{llllllllll} 12 & 2 \\ 317 & 448 & 258 & 208 & 284 & 248 & 315 & 367 & 562 & 500 \end{array} 426390$ |
| standard output |
| 1107 |

## Problem F. Fractions

Input file: standard input<br>Output file: standard output<br>Time limit: 1 second

About 44 days are left before College Scholastic Ability Test is held. This exam aims to measure students' achievement of National Curriculum standards and scholastic ability required for college education. (http://www.kice.re.kr/sub/info.do?m=0205\&s=english)
One of the subjects covered by this test is Mathematics, which consists of 21 multiple choice questions and 9 short-answer questions. The answer of each short-answer question is guaranteed to be a unique positive integer below 1000, as you can see from the answer sheet below.


However, the organizers might want to give students short-answer questions with non-integer answers, such as $2 \sqrt{3}$ or $\frac{5}{3}$. Usually, the workaround is to write the answer in a canonical form, and then sum up all the integers inside that form and ask students to write that number instead.
In particular, when the answer is a positive rational number $\frac{a}{b}$, the organizers usually ask students to reduce it and sum up the numerator and the denominator of the reduced fraction. For example, when the answer is $\frac{18}{10}$, the student should reduce it to $\frac{9}{5}$ and write the final answer as $9+5=14$.

(단, $p$ 와 $q$ 는 서로소인 자연수이다.)

However, when the answer is $\frac{521}{500}$, the reduced fraction is also $\frac{521}{500}$, so the student should write the final answer as $521+500=1021$. But this shouldn't happen, since all the answers for the short-answer questions are below 1000. To avoid this situation, the organizers should make sure that after reducing the fraction, the sum of the numerator and the denominator shouldn't exceed 999. Let's call such fractions as Suneung Fractions. For example, $\frac{1996}{2}$ and $\frac{18}{10}$ are Suneung fractions, while $\frac{1998}{2}$ and $\frac{521}{500}$ are not.

## KAIST 8th ACM-ICPC Mock Competition

Suppose that, this year, one of the organizers wrote a problem, and the answer to that problem is $\frac{x}{y}$. Since the problem is not finalized yet, the only thing we know is $A \leq x \leq B$ and $C \leq y \leq D$ holds, for given $A, B, C, D$. The organizers want to know, among all the pairs $(x, y)$, how many of $\frac{x}{y}$ is a Suneung fraction. Write a program that counts this number.

## Input

The first and only line contains four space-separated integers $A, B, C$ and $D\left(1 \leq A \leq B \leq 10^{12}\right.$, $1 \leq C \leq D \leq 10^{12}$ )

## Output

Print the number of integral pairs $(x, y)(A \leq x \leq B, C \leq y \leq D)$, where $\frac{x}{y}$ is a Suneung fraction.

## Examples

| 5 standard input | standard output |
| :--- | :--- |
| 5863201820182019 | 16 |

## Problem G. Game on Plane

## Input file: standard input <br> Output file: standard output <br> Time limit: 1 second

You are given $N$ points on a plane. These points are precisely the set of vertices of some regular $N$-gon. Koosaga, an extreme villain, is challenging you with a game using these points. You and Koosaga alternatively take turns, and in each turn, the player

1. chooses two of the given points, then
2. draws the line segment connecting the two chosen points.

Also, the newly drawn line segment must not intersect with any of the previously drawn line segments in the interior. It is possible for two segments to meet at their endpoints. If at any point of the game, there exists a convex polygon consisting of the drawn line segments, the game ends and the last player who made the move wins.
Given the integer $N$, Koosaga is letting you decide who will move first. Your task is decide whether you need to move first or the second so that you can win regardless of Koosaga's moves.

## Input

The input consists of many test cases. The first line contains an integer $T(1 \leq T \leq 5,000)$, the number of test cases. Each of the following $T$ test cases is consisted of one line containing the integer $N$ ( $3 \leq N \leq 5,000$ ).

## Output

For each test case, print one line containing the string First if you need to move first or Second if you need to move second so that you can win regardless of Koosaga's moves.

## Example

| standard input | standard output |
| :--- | :--- |
| 2 | First |
| 3 | Second |
| 5 |  |

## Problem H. Histogram Sequence

Input file:
Output file:
standard input
Time limit:
standard output
2 seconds

A histogram is a polygon made by aligning $N$ adjacent rectangles that share a common base line. Each rectangle is called a bar. The $i$-th bar from the left has width 1 and height $H_{i}$.


Figure: This picture depicts a case when $N=9$ and $H=[7,4,3,5,4,2,5,1,2]$.

One day, you wanted to find the area of the largest rectangle contained in the given histogram. What you did was to make a list of integers $A$ by the following procedure:

- For each $1 \leq i \leq j \leq N$, calculate the largest area of the rectangle contained in the histogram, where the rectangle's base line coincides with the base line of the $i, i+1, \cdots, j-1, j$-th bar. Add the area to the list $A$.


Figure: This picture depicts a case when $i=3$ and $j=5$. The area is 9 .
The length of the list $A$ is exactly $\frac{N(N+1)}{2}$ since you chose each pair $(i, j)$ exactly once. To make your life easier, you sorted the list $A$ in non-decreasing order. Now, to find the largest area of the rectangle contained in the histogram, you just need to read the last element of $A, A_{N(N+1) / 2}$.
However, you are not satisfied with this at all, so I decided to let you compute some part of the list $A$. You have to write a program that, given two indices $L$ and $R(L \leq R)$, calculate the values $A_{L \cdots R}$, i.e. $A_{L}, A_{L+1}, \cdots, A_{R-1}, A_{R}$.

## Input

The first line of the input contains an integer $N(1 \leq N \leq 300,000)$ which is the number of bars in the histogram.

The next line contains $N$ space-separated positive integers $H_{1}, H_{2}, \cdots, H_{N}\left(1 \leq H_{i} \leq 10^{9}\right)$, where $H_{i}$ is the height of the $i$-th bar.
The last line contains two integers $L$ and $R\left(1 \leq L \leq R \leq \frac{N(N+1)}{2}, R-L+1 \leq 300,000\right)$.

## Output

Print $R-L+1$ integers. The $j$-th $(1 \leq j \leq R-L+1)$ of them should be the $(L+j-1)$-th element of the list $A$, i.e. $A_{L+j-1}$.

## Example

| standard input | standard output |
| :---: | :---: |
| 9 | 12121415 |
| 743542512 |  |
| 4245 |  |

## Problem I. Repetitive Palindrome

Input file:
Output file:
Time limit:
standard input
standard output
1 second

You are given a string $s$ consisting of lowercase alphabets, and an integer $k$.
Make a new string $t$ by concatenating $k$ copies of $s$. Determine whether $t$ is a palindrome, e.g. is the same backward as forward.

## Input

The first line contains a string $s$ consisting of lowercase alphabets. $(1 \leq|s| \leq 250,000)$
The second line contains an integer $k .\left(1 \leq k \leq 10^{18}\right)$

## Output

If $t$ is a palindrome, print YES. If not, print NO.

## Examples

| abc standard input | standard output |
| :--- | :--- |
| abc <br> 3 | NO |
| abba |  |
| 1 | YES |

## Problem J. Rising Sun

Input file: standard input<br>Output file: standard output<br>Time limit: 1 second

Joon has a midterm exam tomorrow, but he hasn't studied anything. So he decided to stay up all night to study. He promised himself that he will not stop studying before the sun rises.

Joon's house is in some mountains. For convenience, let's say that Joon is living in a 2 -dimensional coordinate system. The mountains are in the region $y \geq 0$, starting at $x=a_{0}$, and the boundary of them consists of $2 n$ line segments parallel to either $y=x$ or $y=-x$.
More precisely, the boundary of the mountains can be described by $(2 n-1)$ additional integers, where the $i$ th number $a_{i}$ of them is the $x$-coordinate of the $i$ th cusp of the mountains. The boundary line starts at ( $a_{0}, 0$ ), proceeds parallel to $y=x$ until its $x$-coordinate reaches $a_{1}$, then proceeds parallel to $y=-x$ until its $x$-coordinate reaches $a_{2}$, and so on. After the last step, the line proceeds parallel to $y=-x$ until it meets the $x$-axis.
The interior of the mountains is the region below the boundary and above the $x$-axis. Thus, the interior and the boundary are disjoint.
At some point between $x=a_{0}$ and $x=a_{2 n-1}$, there is Joon's house on the boundary of the mountains. The size of his house is neglectable compared to the mountains.
Currently, the sun is at the origin and it rises vertically ( $+y$ direction), 1 per minute. Joon can see the sun if the interior of the mountains does not intersect the straight line segment connecting Joon's house and the sun. Joon is completely exhausted and wants to know when can he stop studying. But as you may expect, he is out of his mind, so he cannot do such difficult math. Help him!

## Input

The first line of the input contains an integer $n(1 \leq n \leq 1,000)$.
The next line contains $2 n$ integers, the $i$ th of which is the integer $a_{i-1}\left(1 \leq a_{0}<\cdots<a_{2 n-1} \leq 10^{6}\right)$.
The last line contains an integer $x$, the $x$-coordinate of Joon's house ( $a_{0} \leq x \leq a_{2 n-1}$ ).
It is guaranteed that the boundary of the mountains is in the region $y \geq 0$.

## Output

Print exactly one integer $m$, the smallest integer such that Joon can see the sun after $m$ minutes.

## Examples

| standard input | standard output |
| :---: | :---: |
| $\begin{array}{llll} 2 & & & \\ 1 & 4 & 6 & 7 \\ 7 & & \end{array}$ | 5 |
| $\begin{array}{llll} 2 & & & \\ 3 & 4 & 5 & 7 \\ 7 & & \end{array}$ | 0 |
| $\begin{array}{\|llllll} \hline 3 & & & & & \\ 4 & 9 & 12 & 13 & 14 & 16 \\ 15 & & & & \\ \hline \end{array}$ | 8 |

## Note



Figure: Illustration of the first example.

## KAIST 8th ACM-ICPC Mock Competition

RUN@KAIST, Sponsor: DEVSISTERS, NAVER D2, STARTLINK, October 3rd, 2018

## Problem K. Utilitarianism

## Input file: standard input <br> Output file: standard output <br> Time limit: 5 seconds

In RUN-land, there are $n$ cities numbered 1 to $n$. Some pairs of cities are connected by a bidirectional road. It happens that there are $n-1$ roads in total and that for any two cities, there is a unique path from one to the other. Also, each road is assigned an integer called the value.
Today, to honor the $k$ co-founders of RUN-land, Alex, the king of RUN-land, has decided to choose $k$ different roads and give one road to each of the $k$ co-founders. To prevent unnecessary conflicts, there should be no city that is connected to more than one chosen roads.
In this process, Alex, the king of RUN-land, does not care about who gets what road. Instead, Alex, the king of RUN-land, is only interested in the sum of the values of the $k$ chosen roads. Your task is to choose the roads to maximize this sum.

## Input

The first line contains two integers $n$ and $k(2 \leq n \leq 250,000,1 \leq k \leq n-1)$, which are the number of cities in RUN-land, and the number of roads to choose. Each of the next $n-1$ lines contains three integers $u, v, c(1 \leq u, v \leq n,-1,000,000 \leq c \leq 1,000,000)$, which means that the city $u$ and the city $v$ are directly connected with a bidirectional road with value $c$.

## Output

If there is no way to choose $k$ roads to satisfy the conditions, print Impossible. Otherwise, print one integer, the maximum sum of the values of the $k$ chosen roads.

## Examples

|  |  | standard input |  |
| :--- | :--- | :--- | :--- |
| 5 | 1 |  | 10 |
| 1 | 2 | 2 | standard output |
| 2 | 3 | 3 |  |
| 2 | 4 | 10 |  |
| 4 | 5 | 6 |  |
| 5 | 2 | 9 |  |
| 1 | 2 | 2 |  |
| 2 | 3 | 3 |  |
| 2 | 4 | 10 |  |
| 4 | 5 | 6 |  |
| 5 | 3 |  |  |
| 1 | 2 | 2 |  |
| 2 | 3 | 3 |  |
| 2 | 4 | 10 | 5 |

## KAIST 8th ACM-ICPC Mock Competition

 RUN@KAIST, Sponsor: DEVSISTERS, NAVER D2, STARTLINK, October 3rd, 2018
## Problem L. Voronoi Diagram Returns

Input file:
Output file:
Time limit:
Memory limit:
standard input
standard output
10 seconds
1024 megabytes


Figure: Voronoi Diagram of size 4.
In the 2-dimensional Cartesian coordinate system, we define the Voronoi Diagram of a non-empty set of points $S$, as a diagram that divides the plane by the criteria "which point in the set $S$ is closest in this location?". More precisely, the Voronoi diagram of a given non-empty point set $\left\{P_{1}, P_{2}, \cdots, P_{n}\right\}$ is a collection of regions: A point $K$ is included in region $i$ if and only if $d\left(P_{i}, K\right) \leq d\left(P_{j}, K\right)$ holds for all $1 \leq j \leq n$, where $d(X, Y)$ denotes the Euclidean distance between point $X$ and $Y$.
For example, in the picture above, every location over the plane is colored by the closest point with such location. The points which belongs to a single region is colored by a light color indicating a region, and the points which belongs to more than one region forms lines and points colored black.
There is an algorithm which computes the Voronoi Diagram in $\mathcal{O}(n \log (n))$, but it is infamous to be very complicated and hard. In fact, we are lenient problem setters, so we set $n \leq 2000$, which means you can solve this task with slower Voronoi Diagram algorithms!

In this task, you should solve the point query problem in Voronoi diagram: in the Voronoi diagram constructed with the set of points $\left\{P_{1}, P_{2}, \cdots, P_{n}\right\}$, you should determine which region(s) the point belongs to. More precisely, you will be given $q$ queries of points. For each query point, you should determine the following:

- If it's not included in any region, output NONE.
- If it's included in exactly one region, output REGION $X$, where $X$ is the index of such region.
- If it's included in exactly two regions, output LINE $X Y$, where $X$ and $Y(X<Y)$ are the indices of such two regions.
- If it's included in three or more regions, output POINT.


## Input

In the first line, the number of points consisting Voronoi diagram $n$, and the number of queries $q$ are given. ( $3 \leq n \leq 2,000,1 \leq q \leq 250,000$ )
In the $i$ th line of next $n$ lines, two integers indicating $x$ and $y$ co-ordinate of $P_{i}$ are given. These are the points consisting the Voronoi diagram. All $n$ points are distinct. $(|x|,|y| \leq 10,000)$
In the $j$ th line of next $q$ lines, two integers indicating $x$ and $y$ co-ordinate of $Q_{j}$ are given. For each point $Q_{j}$, you should determine in which region(s) the given point belongs to. ( $|x|,|y| \leq 10,000$ )

## Output

Output consists of $q$ lines. In the $j$ th line, you should print one of following:

- If $Q_{j}$ is not included in any region, output NONE.
- If $Q_{j}$ is included in exactly one region, output REGION X , where X is the index of such region.
- If $Q_{j}$ is included in exactly two regions, output LINE X Y , where X and $\mathrm{Y}(\mathrm{X}<\mathrm{Y})$ are the indices of such two regions.
- If $Q_{j}$ is included in three or more regions, output POINT.


## Example

|  | standard input | standard output |  |
| :--- | :--- | :--- | :--- |
| 43 |  | LINE 1 2 |  |
| -5 | 0 | POINT |  |
| 0 | 5 | REGION 3 |  |
| 3 | 4 |  |  |
| $1-6$ |  |  |  |
| -2 | 2 |  |  |
| 0 | 0 |  |  |
| 2 | 2 |  |  |

## Note

Example is illustrated as a diagram above.

