XXIX Maraton Nacional de Programacion Colombia - ACIS / REDIS 2015 ACM ICPC

Problems

(This set contains 10 problems; problem pages numbered from 1 to 20)

General Information
Unless otherwise stated, the following conditions hold for all problems.

Program name

1. Your source file (your solution!) must be called `<codename>.c`, `<codename>.cpp` or `<codename>.java`, as indicated below the problem title.

Input

1. The input must be read from standard input.
2. The input contains several test cases. Each test case is described using a number of lines that depends on the problem.
3. When a line of data contains several values, they are separated by single spaces. No other spaces appear in the input. There are no empty lines.
4. Every line, including the last one, has the usual end-of-line mark.
5. The end of input is indicated by the end of the input stream. There is no extra data after the test cases in the input.

Output

1. The output must be written to standard output.
2. The result of each test case must appear in the output using a number of lines that depends on the problem.
3. When a line of results contains several values, they must be separated by single spaces. No other spaces should appear in the output. There should be no empty lines.
4. Every line, including the last one, must have the usual end-of-line mark.
5. After the output of all test cases, no extra data must be written to the output.
6. To output real numbers, round them to the closest rational with the required number of digits after the decimal point. Ties are resolved rounding to the nearest upper value.

A: **Next Palindromic Numbers**

Source file name: `nextpal.c`, `nextpal.cpp`, *or* `nextpal.java`

Since the moment Mary learned about palindromic numbers, she has been completely obsessed with them. Indeed, she wants to produce a catalog of these numbers as a personal project.

For those of you who do not know, a *palindromic number* is a natural number whose reading (by avoiding leading zeroes) from left to right is the same one as the one from right to left. For instance, 1991 is a palindromic number, but 1492 is not. Note that the only palindromic number that ends with digit 0 is precisely the number 0. Also, a number being palindromic depends on the numerical basis used to denote such a number. Mary is interested only in decimal palindromic numbers, i.e., numbers written in the usual decimal notation.

Of course, Mary's project of building a complete catalog is impossible to deliver since palindromic numbers are infinite. However, she would still be pleased if groups of consecutive palindromic numbers can be produced. In this way, she will be 'closer' in accomplishing the project's goal of a complete printed catalog of palindromic numbers.

Mary has asked for your help. She would like to have a program that, given two positive integer numbers n and d, outputs the first n palindromic numbers greater than d.

Input

The input consists of several test cases. Each test case consists of a line containing two blank-separated integers n and d ($1 \leq n \leq 10^2$, $1 \leq d \leq 10^{60}$). Numbers are given in decimal notation, without leading zeroes.

The input must be read from standard input.

Output

For each test case, output n lines with the first n palindromic numbers greater than d. Output such a list in ascending order, with numbers in decimal notation and without leading zeroes.

The output must be written to standard output.

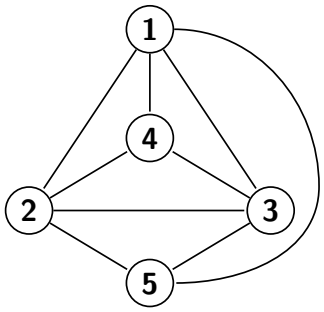
Sample Input	**Sample Output**
3 8 4 175 2 77 1 20003	9 11 22 181 191 202 212 88 99 20102

B: A Subway for Boroughgraph

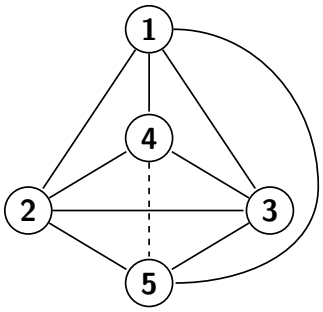
Source file name: `subway.c`, `subway.cpp`, *or* `subway.java`

The mayor of Boroughgraph has been promising its inhabitants a new subway system for quite a few years, and the people are growing impatient. With barely enough money to complete the construction and even less time, he has hired you, a brilliant Boroughgraphian programmer, to help him with this project.

The mayor, of course, wants the new subway to favor mostly those whom he thinks will vote for him, so he has already determined the subway plan. The only problem is that Boroughgraph was built over a series of swamps, so the subway tunnels can be drilled only at a certain fixed depth below ground without risking collapse. Since there is no leftover money to invest in overpasses or ground-level lines, two subway lines can never intersect (except possibly at their endpoints), but their paths need not be straight.



The subway can be built



The subway cannot be built

This is where you come in. Given a specification of a subway network, can you write a program that tells the mayor whether it is possible to build it in Boroughgraph?

Input

The input consists of several subway network specifications. Each specification begins with a line containing a single integer N indicating the number of subway stations ($2 \leq N \leq 64$), which are numbered from 1 to N. Then follow N lines indicating the layout of the subway system: line i contains exactly d_i blank-separated integers from 1 to N (excluding i), indicating the stations to which station i should be connected ($1 \leq d_i \leq N-1$). The subway network is bidirectional, so if station i appears in station j's line, then it is guaranteed that station j will appear in station i's line.

The input must be read from standard input.

Output

For each specification, print a line with the character '`Y`' if the subway can be built in Boroughgraph, or with the character '`N`', otherwise.

The output must be written to standard output.

Sample Input	**Sample Output**
5 2 3 4 5 1 3 4 5 1 2 4 5 1 2 3 1 2 3 5 2 3 4 5 1 3 4 5 1 2 4 5 1 2 3 5 1 2 3 4	Y N

C: **Sub-expression Counting**

Source file name: `counting.c`, `counting.cpp`, *or* `counting.java`

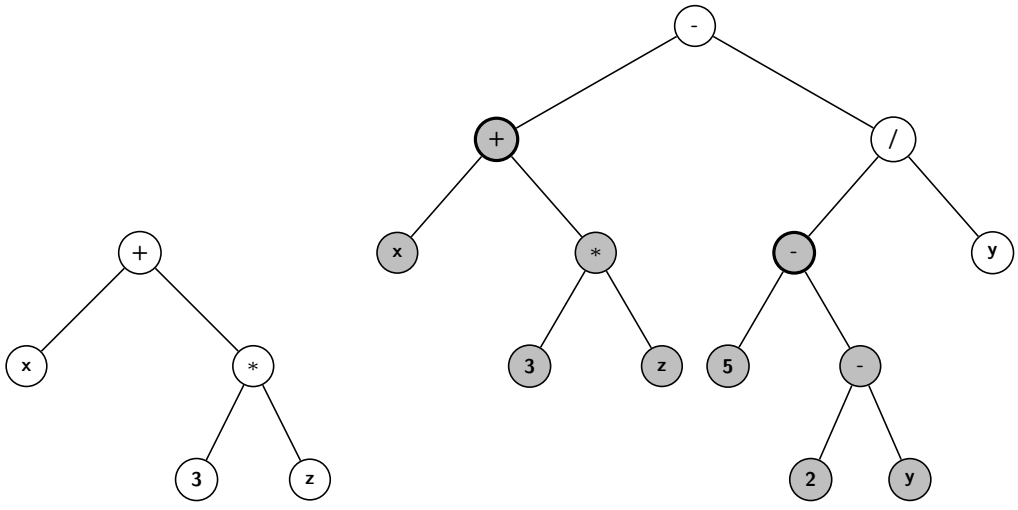
Suppose we want to search arithmetic expressions for sub-expressions of certain shape. We are considering only fully parenthesized expressions with binary operators, numerical constants, and variables, as defined in the following BNF-like notation:

$$
\begin{aligned}
\langle expr \rangle &::= \langle var \rangle \mid \langle num \rangle \mid \texttt{(}\langle expr \rangle \langle binop \rangle \langle expr \rangle\texttt{)} \\
\langle var \rangle &::= \texttt{a} \mid \texttt{b} \mid \cdots \mid \texttt{z} \\
\langle num \rangle &::= \langle digit \rangle \mid \langle digit \rangle \langle num \rangle \\
\langle digit \rangle &::= \texttt{0} \mid \texttt{1} \mid \cdots \mid \texttt{9} \\
\langle binop \rangle &::= \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/}
\end{aligned}
$$

For example, consider the arithmetic expressions α and β defined as follows:

$$
\begin{aligned}
&\alpha : (x + (3 * z)) \\
&\beta : ((x + (3 * z)) - ((5 - (2 - y))/y)).
\end{aligned}
$$

The syntax tree associated to each one of these arithmetic expressions is shown below:



We want to report *all* nodes v in β such that the sub-tree rooted at v is structurally identical to α, ignoring all labels in the nodes. In this case, there are 2 such nodes because: *(i)* expression α is a sub-expression of β and *(ii)* sub-expression $(5 - (2 - y))$ of β has the same tree structure as α. The corresponding sub-trees have been shaded in the syntax tree of β depicted above.

Your task is to write an efficient computer program that, given inputs α and β, computes the number of nodes v in β such that the sub-tree rooted at v is structurally identical to α.

Input

The input consists of several test cases. Each test case consists of two lines: the first line describes the expression α and the second one the expression β. You can assume that $1 \leq |\alpha| \leq 400000$ and $1 \leq |\beta| \leq 400000$, and that these expressions do not contain any blanks.

The input must be read from standard input.

Output

For each test case, output the number of nodes v in β such that the sub-tree rooted at v is structurally identical to α.

The output must be written to standard output.

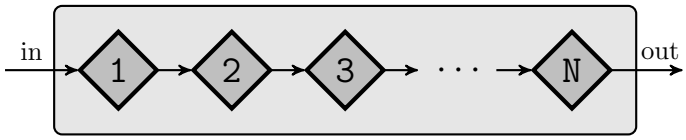
Sample Input	**Sample Output**

`1978`
`((x+0)+z)`
`(x+(3*z))`
`((x+(3*z))-((5-(2-y))/y))`
	`3`
	`2`

D: **Designing an Electronic Device**

Source file name: `device.c`, `device.cpp`, *or* `device.java`

Reviewers of Electronic Designs ∼ *International Service* (REDIS) is an organization that invests in electronic designs. Some time ago, REDIS invented an electronic device, called *Asynchronous Communication Internal Switch* (ACIS), that is built connecting some components in series. In order to ensure the proper function of ACIS, it is required that all components work correctly. For that purpose, inside each component can be installed a fixed number of regulators, decreasing the probability of failure of ACIS.



Components connected in series on ACIS.

For each component in ACIS, REDIS defines the maximum number of regulators that can be installed inside it. Also, for each component and for each possible number of regulators that can be installed inside that component, REDIS specifies the cost of installing that number of regulators inside the component, and the probability of failure of the component if it were installed that number of regulators inside it. For each component, it is guaranteed that the cost is strictly increasing on the number of regulators, and that the probability of failure is strictly decreasing on the number of regulators.

The Financial Manager of REDIS has hired you to design ACIS, minimizing its probability of failure. Given a specific budget, you must decide how many regulators you should install inside each component, in order to minimize the probability of failure of ACIS, ensuring that the total cost of your investment does not exceed the given budget.

Input

The input consists of several test cases, each one describing a distinct electronic device. The first line of a test case contains two blank-separated integers N and K indicating, respectively, the number of components connected on ACIS ($1 \leq N \leq 8$), and the budget you have to invest ($0 \leq K \leq 1000$). The second line of a test case contains exactly N blank-separated integers $M_1, M_2, \ldots, M_N$, where M_i denotes the maximum number of regulators that can be installed inside the i-th component ($1 \leq M_i \leq 16$), for each $1 \leq i \leq N$. Then N lines follow, each one describing a component: line i contains exactly $3 \cdot M_i$ blank-separated integers $\alpha_{i1}, \beta_{i1}, \gamma_{i1}, \alpha_{i2}, \beta_{i2}, \gamma_{i2}, \ldots, \alpha_{iM_i}, \beta_{iM_i}, \gamma_{iM_i}$, where γ_{im} is the cost of installing m regulators inside the i-th component ($1 \leq \gamma_{im} \leq 1000$, for each $1 \leq m \leq M_i$), and α_{im}/β_{im} is the probability of failure of the i-th component if it were installed m regulators inside it ($0 \leq \alpha_{im} < \beta_{im} \leq 100$, for each $1 \leq m \leq M_i$).

You can assume, for each component i, that:

- The cost of installing regulators inside that component, is strictly increasing on the number of regulators (i.e., $0 < \gamma_{i1} < \gamma_{i2} < \cdots < \gamma_{iM_i}$).

- The probability of failure of that component if it were installed regulators inside it, is strictly decreasing on the number of regulators (i.e., $1 > \alpha_{i1}/\beta_{i1} > \alpha_{i2}/\beta_{i2} > \cdots > \alpha_{iM_i}/\beta_{iM_i}$).
- The cost of installing 0 regulators inside that component is 0.
- The probability of failure of that component is 1 if it were installed 0 regulators inside it.

The input must be read from standard input.

Output

For each test case, print a line of the form '`a/b`', where a, b are integers such that $0 \leq a \leq b$, $b > 0$, $gcd(a, b) = 1$, and the fraction a/b is the minimum probability of failure of ACIS with the constraints given. Note that, if such probability is 0, you must print '`0/1`'.

The output must be written to standard output.

Sample Input	**Sample Output**
1 50 1 0 1 100 1 100 1 0 1 100 2 300 2 3 2 3 10 1 10 100 9 10 100 1 2 200 1 5 250 2 100 2 3 2 3 10 1 10 100 9 10 100 1 2 200 1 5 250	1/1 0/1 11/20 1/1

E: Earthquake Disaster

Source file name: `earthquake.c`, `earthquake.cpp`, *or* `earthquake.java`

An earthquake with a magnitude of 8.0 on the Richter scale has just occurred in Seismicity. Houses collapsed, many people are homeless and food becomes scarce. Now, it is time for the rest of the country to help. The country has N cities, numbered from 1 to N, being the city N Seismicity.

Each city has a maximum number of tons of basic goods (water, food, blankets, clothing, medicine) it can donate, and these basic goods will be transported to Seismicity in truck containers through the national road network. Each road is bidirectional and connects two cities. Additionally, if government decides to use a road to move trucks along it, it must pay a compensation to the road neighbors because of contamination and noise, and no more than a fixed number of tons can be transported over it.

Your task is to compute the maximum number of tons of basic goods that can be taken from other cities to Seismicity and the minimum amount of money to be paid to road neighbors to achieve it.

Input

The input consists of several test cases. The first line of each test case contains two blank-separated integers N and M, where N is the number of cities ($2 \leq N \leq 20$), and M is the number of roads ($1 \leq M \leq 500$). Then $N-1$ lines follow, each one containing an integer t_i indicating the maximum number of tons of basic goods that the city i can donate ($0 \leq t_i \leq 50$, for each $1 \leq i \leq N-1$). Each of the next M lines contains four blank-separated integers a, b, w and c, indicating that there is a road between cities a and b over which no more than w tons can be transported at a cost of c per ton ($1 \leq a < b \leq N$, $1 \leq w \leq 50$, $1 \leq c \leq 50$). Note that all roads are bidirectional, and that there may be several roads between any two fixed cities.

The input must be read from standard input.

Output

For each test case, print the maximum number of tons of basic goods that can be taken from other cities to Seismicity, followed by the minimum amount of money to be paid to road neighbors to achieve it.

The output must be written to standard output.

Sample Input	**Sample Output**
2 1 8 1 2 3 2 2 3 8 1 2 3 2 1 2 2 4 1 2 2 3 2 3 8 1 2 5 2 1 2 4 4 1 2 4 3 3 1 4 5 1 2 3 2 4 4 10 15 8 1 2 10 5 1 4 10 8 2 4 30 10 3 4 6 4	3 6 7 20 8 19 0 0 31 254

F: **Farmer Jane**

Source file name: `farmer.c`, `farmer.cpp`, *or* `farmer.java`

Farmer Jane owns farming land in a city where property boundaries are defined by trees. The city council has determined that in order to avoid conflicts among neighbors, it will guarantee that in every property in the city, it should be possible to go from every point in the property, to any other point, using a straight path without leaving its boundaries.

As a property owner, Farmer Jane must install an irrigation system to keep the trees delimiting her property in great condition. She has decided to install a single water sprinkler in some place of her property to water all the trees, but she wants to minimize the costs of doing so. The cost of watering a single tree, is given by the square of the distance from the tree to the sprinkler, times the amount of water needed by the tree. This is, for a tree located in position (x_i, y_i) that needs w_i units of water, and a sprinkler located in position (x, y), the cost of watering the tree is given by $w_i \cdot ((x_i - x)^2 + (y_i - y)^2)$. The cost of watering N trees is given by $\sum_{i=1}^{N} w_i \cdot ((x_i - x)^2 + (y_i - y)^2)$.

Given the number N of trees delimiting the property, and values x_i, y_i, w_i for each tree, your task is to find the minimum cost of installing the sprinkler. It is possible to install a sprinkler in a point (x, y) where x and y are real numbers. Also, it is possible to install the sprinkler in the same position of a tree and in the boundaries of the property.

Input

The input consists of several test cases. The first line of a test case contains a single integer N indicating the number of trees delimiting the property ($3 \leq N \leq 100$). Then follow N lines: line i contains exactly 3 blank-separated integers x_i, y_i, and w_i, where (x_i, y_i) is the position of the i-th tree ($0 \leq x_i \leq 1000$, $0 \leq y_i \leq 1000$), and w_i is the number of units of water needed by the i-th tree ($1 \leq w_i \leq 10$). You may assume that there are not two trees located in the same position, and that the area of the property is a non-empty convex polygon.

The input must be read from standard input.

Output

For each test case, print a single line with a number indicating the minimum cost of installing the sprinkler in the property. The answer should be formatted and approximated to three decimal places. The floating point delimiter must be '.' (i.e., the dot). The rounding applies towards the *nearest neighbor* unless both neighbors are equidistant, in which case the result is rounded up (e.g., 78.3712 is rounded to 78.371; 78.5766 is rounded to 78.577; 78.3745 is rounded to 78.375, etc.).

The output must be written to standard output.

Sample Input	**Sample Output**
4 1 0 2 5 4 2 1 4 3 7 0 6 3 2 1 5 2 5 1 5 3 3	148.308 34.889

G: Peanoland contacting Gaussland

Source file name: `gauss.c`, `gauss.cpp`, *or* `gauss.java`

When scientists in Peanoland discovered that the remote Gaussland planet was inhabited they rejoiced at knowing that they were not alone in the universe. Likewise, there were widespread celebrations in Gaussland when they discovered life in planet Peanoland. Joy did not last long as both planets realized how difficult it was for them to communicate.

After many experiments they discovered that the best way to communicate was through basic light impulses and each planet independently developed its own ACIS table (think of it like an ASCII table but for interplanetary communications). Unfortunately, the ACIS table was not working and scientists simply could not figure out what they had done wrong.

Enter the brilliant Dr. Albert C. Munchhausen whom, through a far bit of trial and error, noticed that communication problems were due to the usage of two different numerical systems. In Peanoland, they were using natural numbers $\mathbb{N}$ in their ACIS table. Gausslanites were instead using Gaussian integers, i.e. the set $\mathbb{Z}[i] = \{a + b{\cdot}i \mid a, b \in \mathbb{Z}\}$, where $i^2{=}{-}1$. Consider $\mathbb{Z}[i]$ as subset of complex numbers:

$$\mathbb{Z}[i] \subset \mathbb{C}$$

''The communication problem is solved'', announced Dr. Munchhausen to his fellow Peanolanders, ''because every number that we use in Peanoland has a corresponding number in Gaussland and we can use binary representations both to communicate and perform the conversion. Let me show you how ...''. Those were the last words of Dr. Munchhausen who was hit by a meteorite and never recovered.

Desperately, scientists studied his notebooks and found the following note:

> If $p \in \mathbb{N}$ and $g \in \mathbb{Z}[i]$, then p and g are equivalent if and only if, there exist $n \in \mathbb{N}$, $b_0 \in \{0, 1\}$, $b_1 \in \{0, 1\}$, ..., and $b_n \in \{0, 1\}$, such that
>
> $$p = \sum_{k=0}^{n} b_k{\cdot}2^k \qquad \text{and} \qquad g = \sum_{k=0}^{n} b_k{\cdot}(i{-}1)^k$$

Another note said:

Eureka!
$$292 = 20 - 6{\cdot}i$$

Your task, to honor the memory of Dr. Munchhausen, is to build the translator that he intended to build in order to convert from the numbers used in Peanoland to the numbers used in Gaussland.

Input

The input consists of several test cases. Each test case consists of a line with a single natural number p in the Peanoland system ($0{\leq}p{<}10^9$).

The input must be read from standard input.

Output

For each test case output two blank-separated integers a and b, where $g = a + b \cdot i$ is the number in the Gaussian system that corresponds to the natural number p in the Peanoland system.

The output must be written to standard output.

Sample Input	**Sample Output**
0 1 2 3 4 292 999999999	0 0 1 0 -1 1 0 1 0 -2 20 -6 5290 -5347

H: **Texting with Grandma**

Source file name: `texting.c`, `texting.cpp`, *or* `texting.java`

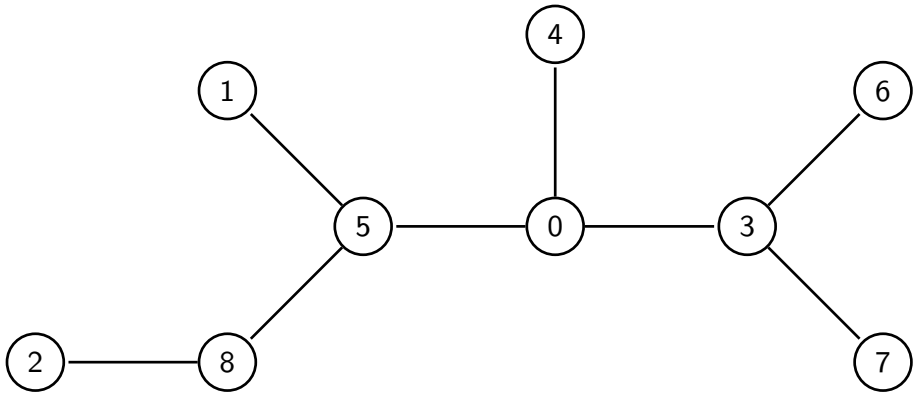
Grandma has bought a state-of-the-art shinny new mobile phone. She is specially excited because she will text with Eloi even if he is away from home. ‘‘Grandma is sugar and spice, and everything nice, specially when she is texting ... she is a living message sent from a time we did not live.’’ Eloi, always set in the old ways, is worried he will not be able to text with Grandma because his mobile phone is so old that he is hardly able to play ‘snakes’ on it. As a matter of fact, his mobile has been discontinued for a couple of years already.

The new hype in the texting scene is to text trees because research in computer science have found a way to ‘pack’ any tree. By ‘tree’ we mean a labeled undirected graph in which there is exactly one path between any pair of vertices. Grandma cannot wait to send Eloi messages packed with tons of electronic trees of many sorts drawn with all her love.

For a tree T with set of n vertices $V=\{0,\ldots,n-1\}$, where $n\geq 2$, the *tree packing algorithm* implemented in Grandma’s mobile on input T builds a sequence t of exactly $n-2$ vertices as follows:

```
let t the empty sequence
for each k = 1, ..., n-2
  let L be the set of leaves of T
  let m be the minimum element of L
  append to t the unique vertex adjacent to m
  remove m from T
```

As an example, consider the following ‘oak’ drawn by Grandma, with set of vertices $\{0,1,\ldots,8\}$:



One can check that the oak drawn by Grandma is packed into the sequence $5,8,0,3,3,0,5$ by the tree packing algorithm.

Eloi will be getting texts from Grandma soon and he is faced with the following problem: given a sequence t of integer numbers, he would like to have an app for his old mobile for detecting if t can be the output of the tree packing algorithm for some input tree T with set of vertices $V=\{0,\ldots,m-1\}$ for some positive integer m. If this is the case, he would like to ‘unpack’ t into its associated tree T, so that he can appreciate Grandma’s texting love. Since Eloi’s mobile

is old (as in 'very old'), this app needs to be very efficient.

Input

The input consists of several test cases. Each test case consists of a single line describing a non-empty sequence t of blank-separated integers between 0 and 100000, both limits inclusive. You can assume that $1 \leq |t| \leq 100000$.

The input must be read from standard input.

Output

For sequence t described in each test case:

- if t can not be the output of the tree packing algorithm, then output '`impossible`';
- if t is the output of the tree packing algorithm for some input tree T with set of vertices V, output $|V|+1$ lines: the first line contains $|V|$ and the next $1 \leq i \leq |V|$ lines each lists in ascending order the set of vertices directly associated to vertex $i-1$ in T (output a single blank between consecutive numbers).

Print a line with a single asterisk ('`*`') between consecutive test cases.

The output must be written to standard output.

Sample Input	**Sample Output**

`5 8 0 3 3 0 5`
`1 7 1 2`
	`9`
	`3 4 5`
	`5`
	`8`
	`0 6 7`
	`0`
	`0 1 8`
	`3`
	`3`
	`2 5`
	`*`
	`impossible`

I: Interstellar Travel

Source file name: `interstellar.c`, `interstellar.cpp`, *or* `interstellar.java`

The *Agency for Cross-Constellation and Interstellar Space Travel* (ACIS) is ready to offer its clients space travel among several planets across the universe.

ACIS offers a list of flight options consisting of an origin planet, a destination planet, a cost, and a duration. One of the ''killer'' features ACIS will offer to its clients is that of being able to plan a trip between two planets under the constraint of a maximum number of stops. That is, given a natural number n, ACIS would like to offer each client the cheapest possible trip from an origin planet to a destination planet with at most n stops. Since interstellar in-flight sleep is not pleasant, it is also important to minimize the amount of time spent in a trip.

Can you help ACIS in finding an efficient algorithm for such a task?

Input

The input consists of several test cases. Each test case begins with a line with three blank-separated integers p, f, and q ($1 \leq p \leq 300$, $0 \leq f \leq 5000$, and $0 \leq q \leq 1000$), indicating the number of planets, flights, and queries, respectively. The next p lines each contains a planet name s ($1 \leq |s| \leq 30$). The next f lines each contains two planet names and two integers s_o, s_d, c, and t (separated by a blank), denoting that there is a direct flight from s_o to s_d costing c dollars ($0 \leq c \leq 10^5$) with a duration of t units of time ($0 \leq t \leq 10^5$). The next line contains a planet name s_i indicating the initial planet for the trip. The next q lines each contains a query with a destination planet name s_f for the trip and a natural number n, both separated by a blank ($0 \leq n \leq 300$). You can assume that planet names consist only of alphabetic characters, and that s_o, s_d, s_i, and s_f are in the list of p planet names.

The input must be read from standard input.

Output

For each query s_i, s_f, n output two blank-separated integers indicating the minimum cost and the corresponding minimum travel time for this cost of an interstellar trip from s_i to s_f with at most n stops. If this is not possible, then print two blank-separated asterisks (‘`*`’).

Print a line with a single period (‘`.`’) between consecutive test cases.

The output must be written to standard output.

Sample Input

```
2 3 1
Earth
Mars
Earth Mars 2 3
Earth Mars 4 1
Earth Earth 3 2
Earth
Mars 0
3 3 5
Tatooine
Endor
Geonosis
Tatooine Endor 300 15
Endor Geonosis 10 78
Geonosis Tatooine 1 1
Endor
Endor 0
Geonosis 0
Geonosis 4
Tatooine 0
Tatooine 1
5 5 8
Earth
Kaishin
Namek
Vegeta
NewNamek
Earth Kaishin 10 10
Kaishin Namek 10 5
Kaishin Vegeta 15 30
Earth Vegeta 25 50
NewNamek Earth 100 1
Earth
Kaishin 0
Kaishin 1
Kaishin 2
Namek 0
Namek 1
Vegeta 0
Vegeta 1
NewNamek 5
```

Sample Output

```
2 3
.
0 0
10 78
10 78
* *
11 79
.
10 10
10 10
10 10
* *
20 15
25 50
25 40
* *
```

J: **Voting Duels**

Source file name: `voting.c`, `voting.cpp`, *or* `voting.java`

In order to choose their president, the congressmen of X must follow strict election bylaws. If there are n candidates, the election must be accomplished by means of rounds, each facing two of the candidates in a *voting duel*. The loser of a round is eliminated and the winner remains in the competition. If there is a tie, both candidates remain (although the corresponding duel will not be repeated). At the end, every pair of remaining candidates should have had their duel and, since several candidates could be finally tied, in this case they all are elected presidents.

Congressmen are grouped in disjoint blocs of different sizes. Each bloc has its particular opinion and its members sort the candidates according to their particular preferences. If a round must decide among candidates a and b, each bloc gives everyone of its votes for the candidate they more like in its preference list.

To illustrate the process, suppose there are four candidates, a, b, c and d and that congressmen are grouped as follows (for each bloc, size in brackets and preferences in descending order):

- Bloc 1: [17] $c\ a\ d\ b$
- Bloc 2: [32] $a\ b\ d\ c$
- Bloc 3: [34] $d\ b\ c\ a$
- Bloc 4: [17] $b\ a\ c\ d$

Denoting by x:y a round to vote x against y, the result of an election accomplished by rounds a:b, b:d and c:d would have d as winner. To see that, note that

- a:b results in 17 (Bloc 1) plus 32 (Bloc 2) votes for a, and 34 (Bloc 3) plus 17 votes for b (Bloc 4), i.e, b wins (49 against 51);
- b:d results in 32 (Bloc 2) plus 17 (Bloc 4) votes for b, and 17 (Bloc 1) plus 34 votes for d (Bloc 3), i.e, d wins (49 against 51);
- c:d results in 17 (Bloc 1) plus 17 (Bloc 4) votes for c, and 32 (Bloc 2) plus 34 votes for d (Bloc 3), i.e, d wins (34 against 66).

But it should be clear that if the rounds order were b:d, c:d, a:d the winner would be a.

Some congressmen have observed this situation and want to analyze it more deeply. They want to know whether, for a particular bloc grouping with given preference lists is it possible that one specific candidate could be the winner if an appropriate rounds order is chosen. Could you help them?

Input

In order to identify candidates in the input, when in a case there are n candidates to be considered, each one of them is denoted with one of the first n lower case English alphabet letters.

The input consists of several test cases. The first line of a case contains two integer numbers n ($1 \leq n \leq 26$) and b ($1 \leq b \leq 100$), and one English alphabet letter, indicating, respectively, the number of candidates, the number of blocs, and the identifier that represents the designated candidate (for whom it should be decided if he/she can be president through an appropriate voting order).

Then there are b lines, so that the i-th of these lines indicates the size and the preferences of the i-th bloc, with an integer number s indicating the size of the bloc ($1 \leq s \leq 50$), followed by a word that is a permutation of the first n letters of the English alphabet (lower case). This last word defines the preference voting order for the i-th bloc: leftmost identifiers in this word represent candidates that are preferred to right most ones.

You may assume that the identifier read in the first line is one of the first n English alphabet letters.

Data in lines is left justified and separated by one blank.

The input must be read from standard input.

Output

For each test case, print one line with exactly one letter '`Y`' or '`N`', indicating if there is or there is not a way to define the order of the voting duels so that the designated candidate (that one in the first input line) is elected president (eventually, not alone).

The output must be written to standard output.

Sample Input	**Sample Output**

`4 4 c`
`17 cadb`
`32 abdc`
`34 dbca`
`17 bacd`
`3 2 a`
`12 cba`
`10 bca`
	`Y`
	`N`