Reminders

- For all problems, read the input data from standard input and write the results to standard output.

- In general, when there is more than one integer or word on an input line, they will be separated from each other by exactly one space. No input lines will have leading or trailing spaces, and tabs will never appear in any input.

- Compiler options are as follows:

  ```
  g++ -g -O2 -std=gnu++0x -static $*
  gcc -g -O2 -std=gnu99 -static $* -lm
  javac -encoding UTF-8 -sourcepath . -d . $* runjava
  java -client -Xss8m -Xmx1024m $*
  ```
An airline company offers flights out of \( n \) airports, conveniently labeled from 1 to \( n \). The flight time \( t_{ij} \) from airport \( i \) to airport \( j \) is known for every \( i \) and \( j \). It may be the case that \( t_{ij} \neq t_{ji} \), due to things like wind or geography. Upon landing at a given airport, a plane must be inspected before it can be flown again. This inspection time \( p_i \) is dependent only on the airport at which the inspection is taking place and not where the previous flight may have originated.

Given a set of \( m \) flights that the airline company must provide, determine the minimum number of planes that the company needs to purchase. The airline may add unscheduled flights to move the airplanes around if that would reduce the total number of planes needed.

**Input**

The first line of input contains two space-separated integers \( n \) and \( m \) (\( 1 \leq n, m \leq 500 \)). The next line contains \( n \) space-separated integers \( p_1, \ldots, p_n \) (\( 0 \leq p_i \leq 10^6 \)).

Each of the next \( n \) lines contains \( n \) space-separated integers. The \( j \)th integer in line \( i + 2 \) is \( t_{ij} \) (\( 0 \leq t_{ij} \leq 10^6 \)). It is guaranteed that \( t_{ii} = 0 \) for all \( i \). However, it may be the case that \( t_{ij} \neq t_{ji} \) when \( i \neq j \).

Each of the next \( m \) lines contains three space-separated integers, \( s_i, f_i, \) and \( t_i \) (\( 1 \leq s_i, f_i \leq n \), \( s_i \neq f_i \), \( 1 \leq t_i \leq 10^6 \)), indicating that the airline company must provide a flight that flies out from airport \( s_i \) at exactly time \( t_i \), heading directly to airport \( f_i \).

**Output**

Print, on a single line, a single integer indicating the minimum number of planes the airline company must purchase in order to provide the \( m \) requested flights.

### Sample Input

\[
\begin{array}{llll}
2 & 2 \\
1 & 1 \\
0 & 1 \\
1 & 0 \\
1 & 2 & 1 \\
2 & 1 & 1 \\
\end{array}
\]

### Sample Output

\[
2
\]
<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 2</td>
<td>1</td>
</tr>
<tr>
<td>1 1</td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td></td>
</tr>
<tr>
<td>1 2 1</td>
<td></td>
</tr>
<tr>
<td>2 1 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample Input</td>
<td>Sample Output</td>
</tr>
<tr>
<td>5 5</td>
<td>3</td>
</tr>
<tr>
<td>72 54 71 94 23</td>
<td></td>
</tr>
<tr>
<td>0 443 912 226 714</td>
<td></td>
</tr>
<tr>
<td>18 0 776 347 810</td>
<td></td>
</tr>
<tr>
<td>707 60 0 48 923</td>
<td></td>
</tr>
<tr>
<td>933 373 881 0 329</td>
<td></td>
</tr>
<tr>
<td>39 511 151 364 0</td>
<td></td>
</tr>
<tr>
<td>4 2 174</td>
<td></td>
</tr>
<tr>
<td>2 1 583</td>
<td></td>
</tr>
<tr>
<td>4 3 151</td>
<td></td>
</tr>
<tr>
<td>1 4 841</td>
<td></td>
</tr>
<tr>
<td>4 3 993</td>
<td></td>
</tr>
</tbody>
</table>
Problem B — Limit 5 seconds

Magic Multiple

The Elvish races of Middle Earth believed that certain numbers were more significant than others. When using a particular quantity \( n \) of metal to forge a particular sword, they believed that sword would be most powerful if the thickness \( k \) were chosen according to the following rule:

Given a nonnegative integer \( n \), what is the smallest \( k \) such that the decimal representations of the integers in the sequence:

\[ n, \ 2n, \ 3n, \ 4n, \ 5n, \ldots, \ kn \]

contain all ten digits (0 through 9) at least once?

Lord Elrond of Rivendell has commissioned you with the task to develop an algorithm to find the optimal thickness \( (k) \) for any given quantity of metal \( (n) \).

Input

Input will consist of a single integer \( n \) per line. The end of input will be signaled by end of file. The input integer will be between 1 and 200,000,000, inclusive.

Output

The output will consist of a single integer per line, indicating the value of \( k \) needed such that every digit from 0 through 9 is seen at least once.

<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>123456789</td>
<td>3</td>
</tr>
<tr>
<td>3141592</td>
<td>5</td>
</tr>
</tbody>
</table>
In his memoir *So, Anyway...*, comedian John Cleese writes of the class difference between his father (who was “middle-middle-middle-lower-middle class”) and his mother (who was “upper-upper-lower-middle class”). These fine distinctions between classes tend to confuse American readers, so you are to write a program to sort a group of people by their classes to show the true distinctions.

There are three main classes: upper, middle, and lower. Obviously, upper class is the highest, and lower class is the lowest. But there can be distinctions within a class, so upper-upper is a higher class than middle-upper, which is higher than lower-upper. However, all of the upper classes (upper-upper, middle-upper, and lower-upper) are higher than any of the middle classes.

Within a class like middle-upper, there can be further distinctions as well, leading to classes like lower-middle-upper-middle-upper. When comparing classes, once you’ve reached the lowest level of detail, you should assume that all further classes are the equivalent to the middle level of the previous level of detail. So upper class and middle-upper class are equivalent, as are middle-middle-lower-middle-lower-middle and lower-middle.

**Input**

The first line of input contains $n$ ($1 \leq n \leq 1,000$), the number of names to follow. Each of the following $n$ lines contains the name of a person (a sequence of 1 or more lowercase letters ‘z’–’z’), a colon, a space, and then the class of the person. The class of the person will include one or more modifiers and then the word class. The colon, modifiers, and the word class will be separated from each other by single spaces. All modifiers are one of upper, middle, or lower. It is guaranteed that the input is well-formed. Additionally, no two people have the same name. Input lines are no longer than 256 characters.

**Output**

Print the $n$ names, each on a single line, from highest to lowest class. If two people have equivalent classes, they should be listed in alphabetical order by name.
### Sample Input

```
5
mom: upper upper lower middle class
dad: middle middle lower middle class
queenelizabeth: upper upper class
chair: lower lower class
unclebob: middle lower middle class
```

### Sample Output

```
queenelizabeth
mom
dad
unclebob
chair
```

### Sample Input

```
10
rich: lower upper class
mona: upper upper class
dave: middle lower class
charles: middle class
tom: middle class
william: lower middle class
carl: lower class
violet: middle class
frank: lower class
mary: upper class
```

### Sample Output

```
mona
mary
rich
charles
tom
violet
william
carl
dave
frank
```
With our time on Earth coming to an end, Cooper and Amelia have volunteered to undertake what could be the most important mission in human history: travelling beyond this galaxy to discover whether mankind has a future among the stars. Fortunately, astronomers have identified several potentially inhabitable planets and have also discovered that some of these planets have wormholes joining them, which effectively makes the travel distance between these wormhole connected planets zero. For all other planets, the travel distance between them is simply the Euclidean distance between the planets. Given the location of Earth, planets, and wormholes, find the shortest travel distance between any pairs of planets.

Input

• The first line of input is a single integer, $T$ ($1 \leq T \leq 10$) the number of test cases.

• Each test case consists of planets, wormholes, and a set of distance queries.

• The planets list for a test case starts with a single integer, $p$ ($1 \leq p \leq 60$), the number of planets. Following this are $p$ lines, where each line contains a planet name along with the planet’s integer coordinates, i.e. name $x$ $y$ $z$ ($0 \leq x, y, z \leq 2 \cdot 10^6$) The names of the planets will consist only of ASCII letters and numbers, and will always start with an ASCII letter. Planet names are case-sensitive (Earth and earth are distinct planets). The length of a planet name will never be greater than 50 characters. All coordinates are given in parsecs.

• The wormholes list for a test case starts with a single integer, $w$ ($0 \leq w \leq 40$), the number of wormholes, followed by the list of $w$ wormholes. Each wormhole consists of two planet names separated by a space. The first planet name marks the entrance of wormhole, and the second planet name marks the exit from the wormhole. The planets that mark wormholes will be chosen from the list of planets given in the preceding section. Note: you can’t enter a wormhole at its exit.

• The queries list for a test case starts with a single integer, $q$ ($1 \leq q \leq 20$), the number of queries. Each query consists of two planet names separated by a space. Both planets will have been listed in the planet list.
Output

For each test case, output a line, “Case i:”, the number of the i-th test case. Then, for each query in that test case, output a line that states “The distance from planet1 to planet2 is d parsecs.”, where the planets are the names from the query and d is the shortest possible travel distance between the two planets. Round d to the nearest integer.

<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
</table>
| 3
4
Earth 0 0 0
Proxima 5 0 0
Barnards 5 5 0
Sirius 0 5 0
2
Earth Barnards
Barnards Sirius
6
Earth Proxima
Earth Barnards
Earth Sirius
Proxima Earth
Barnards Earth
Sirius Earth
3
z1 0 0 0
z2 10 10 10
z3 10 0 0
1
z1 z2
3
z2 z1
z1 z2
z1 z3
2
Mars 12345 98765 87654
Jupiter 45678 65432 11111
0
1
Mars Jupiter | Case 1:
The distance from Earth to Proxima is 5 parsecs.
The distance from Earth to Barnards is 0 parsecs.
The distance from Earth to Sirius is 0 parsecs.
The distance from Proxima to Earth is 5 parsecs.
The distance from Barnards to Earth is 5 parsecs.
The distance from Sirius to Earth is 5 parsecs.
Case 2:
The distance from z2 to z1 is 17 parsecs.
The distance from z1 to z2 is 0 parsecs.
The distance from z1 to z3 is 10 parsecs.
Case 3:
The distance from Mars to Jupiter is 89894 parsecs.
Problem J — limit 10 seconds

Temple Build

The Dwarves of Middle Earth are renowned for their delving and smithy ability, but they are also master builders. During the time of the dragons, the dwarves found that above ground the buildings that were most resistant to attack were truncated square pyramids (a square pyramid that does not go all the way up to a point, but instead has a flat square on top).

The dwarves knew what the ideal building shape should be based on the height they wanted and the size of the square base at the top and bottom. They typically had three different sizes of cubic bricks with which to work. Their goal was to maximize the volume of such a building based on the following rules:

The building is constructed of layers; each layer is a single square of bricks of a single size. No part of any brick may extend out from the ideal shape, either to the sides or at the top. The resulting structure will have jagged sides and may be shorter than the ideal shape, but it must fit completely within the ideal design. The picture at the right is a vertical cross section of one such tower.

There is no limit on how many bricks of each type can be used.

Input

Each line of input will contain six entries, each separated by a single space. The entries represent the ideal temple height, the size of the square base at the bottom, the size of the square base at the top (all three as non-negative integers less than or equal to one million), then three sizes of cubic bricks (all three as non-negative integers less than or equal to ten thousand). Input is terminated upon reaching end of file.

Output

For each line of input, output the maximum possible volume based on the given rules, one output per line.

<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>500000 800000 300000 6931 11315 5000</td>
<td>160293750000000000</td>
</tr>
</tbody>
</table>

2012 Pacific Northwest Region Programming Contest
Problem J: Justified Jungle

Time limit: 6 s
Memory limit: 512 MiB

As you probably know, a tree is a graph consisting of \( n \) nodes and \( n - 1 \) undirected edges in which any two nodes are connected by exactly one path. A forest is a graph consisting of one or more trees. In other words, a graph is a forest if every connected component is a tree. A forest is justified if all connected components have the same number of nodes.

Given a tree \( G \) consisting of \( n \) nodes, find all positive integers \( k \) such that a justified forest can be obtained by erasing exactly \( k \) edges from \( G \). Note that erasing an edge never erases any nodes. In particular when we erase all \( n - 1 \) edges from \( G \), we obtain a justified forest consisting of \( n \) one-node components.

**Input**

The first line contains an integer \( n \) (\( 2 \leq n \leq 1\,000\,000 \)) — the number of nodes in \( G \). The \( k \)-th of the following \( n - 1 \) lines contains two different integers \( a_k \) and \( b_k \) (\( 1 \leq a_k, b_k \leq n \)) — the endpoints of the \( k \)-th edge.

**Output**

The first line should contain all wanted integers \( k \), in increasing order.

**Example**

```
input
8
1 2
2 3
1 4
4 5
6 7
8 3
7 3

output
1 3 7
```

Figures depict justified forests obtained by erasing 1, 3 and 7 edges from the tree in the example input.