National Taiwan University

2013 NTU Final

A	Ant Colony	8 s
B	Balanced Spanning Tree	10 s
C	Candy Box	8 s
D	Duo Palindr0me	1 s
E	Exiled Angel	2 s
F	Forgotten Sequence	4 s
G	Geodesic	3 s
H	Halin Graph	8 s
I	Iconic Sigil	1 s
J	Journey to the Gold	5 s

January 11, 2014

(This page was intentionally left only almost blank.)

Prologue

Alright, alright, you win!

We're out of ideas this time, no more irritating lousy super-long statemennts. Only super-boring short statments without fun that smells like stink water because in it there's no creativity, no life, no wit, NOTHING.

Happy now? well I sure hope you are happy! Because I'm sorry to tell you after this you'd never get a problemset from us that's as interesting as it's used to be! Not a single one more! What? you say you're even happier?

I hate you!!!!!

Just kidding. We can't wait to challenge you with another action-packed 50 pages problemset. You wait and see!

A. Ant Colony

time limit: 8 second

An ant colony has established their home as an underground network. The underground network takes the form of a tree, and edges between nodes are weighted (where weights represent the length of the edge).

Ants have extremely regular behavior. Everyday, each of them follows their fixed daily schedule and never deviate from it. Specificially, ant i wakes up at time t_i (t_i is the number of millisecond passed from 0:00 AM) on the node s_i, and starts walking to its destination e_i with a fixed speed 1 unit distance per millisecond. Of course, the ants always move in shortest path, which is the unique simple path from its start to destination. Once an ant finishes its tiring journey (reaches its destination, it will always be before midnight), it falls asleep immediately. At which point it will sleep-walk back to its starting point s_i.

So as a recap: each ant has its fixed schedule, which is –

sleep $\rightarrow$ awake at s_i $\rightarrow$ walk to e_i $\rightarrow$ fall asleep and sleepwalk back to s_i.

Due to the never changing living routine, some of the ants may never meet each other. Two ants meet each other if and only if at some point during a day they are both awake and situated at the exact same point. Note that time is continuous and two ants may meet in the middle of an edge. Also the moment an ant wakes up or the moment it falls asleep also count as being "awake", so should another ant approaches at such time, the two ants are considered to have met each other. Sleepwalking, however, does *not* count toward being awake.

Now you wonder, what is size of the largest group of ants, such that every two of them have met each other?

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 20$) indicating the number of test cases.

Each test case starts with two positive integers $N, M (2 \leq N \leq 10000, 0 \leq M \leq 50000$, indicating the number of vertices in the ant nest and the number of ants.

The next $N-1$ lines describe the structure of the nest. Each line contains three integers x_i, y_i, l_i ($1 \leq x_i, y_i \leq N$, $1 \leq l_i \leq 1000$), indicating that room x_i and room y_i are connected by a tunnel with length l_i.

The next M lines describe the motions of ants. Each line contains three integers s_i, e_i, t_i ($1 \leq s_i, e_i \leq N$, $0 \leq t_i \leq 10^6$ and $s_i \neq e_i$), indicating that the i-th ant starts moving at time t_i from node s_i toward e_i.

Output Format

For each test case, output the maximum size of group, such that every pair of ants in the group have met each other.

Sample Input

```
2
4 3
1 2 10
2 3 5
3 4 8
1 4 0
2 4 0
4 1 11
4 3
1 2 10
1 3 10
1 4 10
4 1 0
3 2 0
1 2 10
```

Sample Output

```
2
3
```

B. Balanced Spanning Tree

time limit: 10 seconds

For a weighted connected graph with n vertices, we say a spanning tree of the graph is balanced if and only if the sum of the edge weights of the spanning tree is divisible by $n - 1$.

Given a weighted undirected complete graph, please determine if it has a balanced spanning tree or not, and find a solution if it has one.

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 30$) indicating the number of test cases.

Each test case starts with one integer N ($2 \leq N \leq 100$), indicating the number of vertices. Then N lines follow. The i-th line contains N integers $v_{i,0}, \cdots, v_{i,n-1}$, where $v_{i,j}$ indicates the weight of the edge between vertex i and j. ($0 \leq v_{i,j} \leq 10^9$, $v_{i,j} = v_{j,i}$, $v_{i,i} = 0$)

All vertices are 0-indexed.

Output Format

For each test case, first output a line containing "Yes" or "No", indicating whether there exists a balanced spanning tree.

If the answer is "Yes", then you should output $n - 1$ more lines to show one such spanning tree. Each of the lines should contain a pair of integers, denoting an edge of the balanced spanning tree. If there are more than one such spanning tree, you may output any one of them.

Sample Input

```
2
4
0 1 1 2
1 0 0 1
1 0 0 1
2 1 1 0
3
0 5 1
5 0 4
1 4 0
```

Sample Output

```
Yes
0 1
0 2
3 1
Yes
0 1
0 2
```

(This page was intentionally left blank.)

C. Candy Box

time limit: 8 seconds

Kelvin loves candies! He has m magic candy boxes with a button and an integer display on each of them. When the button is pressed, the box would spits out varying amount of candies; sometimes Kelvin gets fountain of candies, but sometimes when he pushed a button, only a pitiful one or two candies pop out.

After some experiments, Kelvin figured out the rule of the candy boxes:

There's a sequence of n integers $x_0, \cdots, x_{n-1}$, stored in each box (note that each box has its own sequence). A number called "Tomato Power", varying from time to time, also affects the output of all boxes. It's known that when a box have integer A displayed, and the "Tomato Power" is B, the box would offer $x_{(B-A) \bmod n}$ candies when the button is pressed.

Since the boxes are magical, the integer on them also changes from day to day. At day 0, all boxes displays 0. At the end of each day, the numbers on the i-th box is incremented by R_i.

Kelvin is curious about that, how many total candies would he get, if he pushes all the buttons on day T, where the Tomato Power is P at that time.

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 30$) indicating the number of test cases.

Each test case starts with three integers n, m, q ($n \in \{4, 16, 256, 65536\}$, $1 \leq m, q \leq 10^5$), indicating the sequence length in each box, the number of boxes, and number of queries Kelvin wants to ask.

The following line contains three integers a, b, c, which are used as parameters as depicted below. It is guaranteed that $b(b^2 - a^2)(b^2 - (a-1)^2) \neq 0 \pmod{\text{n+1}}$.

In the following m lines, the i-th line contains three integers R_i, F_i, G_i, describing the i-th box. R_i denotes the number to be added on the i-th box each day. The sequence in the i-th box is generated by the following formula:

$$x_0 = F_i$$
$$x_1 = G_i$$
$$x_{s+2} = \big(2ax_{s+1} + (b^2 - a^2)x_s + c\big) \bmod \text{(n+1)} \qquad s = 2, \cdots, n-1$$

Then, each of the following q lines contains two integers T, P, indicating that Kelvin wants to know the total number of candies he would get when pressing all buttons on day T, with Tomato Power P.

All numbers in input are non-negative, and does not exceed 10^9.

Output Format

For each query, output the total number of candies Kelvin would get, modulo $n + 1$.

Sample Input

```
2
4 2 3
3 1 3
5 1 4
3 1 4
0 0
1 2
3 5
16 1 1
0 2 4
1 1 1
33554433 514514514
```

Sample Output

```
2
4
0
1
```

D. Duo Palindr0me

time limit: 1 seconds

We all know that a string is a palindrome if it reads the same from left to right and from right to left.

Lets define something different, called *palindr0mes*. An integer x is a palindr0me if its decimal representation, after removing trailing 0's (if any), is a palindrome. For example, 51415, 10100, and 123210 are all palindr0mes.

Moreover, lets say a number y is a *duo-palindr0me centroid* if both its neighbors, $y-1$ and $y+1$, are palindr0mes.

Now please find out how many duo-palindr0me centroids are there between L and R (inclusive).

Input Format

The first line of the input file contains an integer T ($1 \le T \le 32000$) indicating the number of test cases.

Each test case has one line, containing two integers, L and R. ($1 \le L \le R \le 10^{18}$)

Output Format

For each test case, output on a single line the number of duo-palindr0me centroids between L and R (inclusive).

Sample Input

```
3
18 23
21 21
50 55
```

Sample Output

```
1
1
0
```

(This page was intentionally left blank.)

E. Exiled Angel

time limit: 2 second

Angels are agents of high heaven that fight against the corrupte force of the hell. Among them, five most powerful beings are elected as Archangels. They are Auriel, Archangel of Hope, Imperius, Archangel of Valor, Itherael, Archangel of Fate, Malthael, Archangel of Wisdom, and Tyrael, Archangel of Justice.

During an failed attempt to put down the resurrected evil, Mephisto, Archangel Malthael decided to use his own body as a cage, and sealed the demon Mephisto within his own flesh. The demon was contained alright, but the dark voice inside Malthael never stop wailing ever since. It was only a matter of time before Malthael went crazy, and was exiled from the heaven. He was banished to ever walk the realm of sanctuary, so he became a lone soul that wanders the land of Caldeum.

The dark force within Malthael constantly battles against his own will, and the two conscious alternatively gain control of Malthael's body. Malthael knows that there is still hope. There are sacred shrines that could cleanse his corrupted soul everytime he visits them. Should he be able to visit the sacred shrines infinite number of times, he would gain salvation eventually.

Formally, the map of Caldeum could be viewed as a *directed* graph, where nodes represent specific locales, and uni-directional arcs represent existence of road to travel from one place to another. Sacred shrines are situated in some of the locales.

Malthael's own will and the dark will within gains control alternatively. Every time a side gain control, it would drive Malthael's body to move to a neighboring location of its choice. If some move is possible at all, one must be chosen; otherwise (if no move is possible), Malthael is trapped at the same location forever. Initially Malthael is in control of his own body. He wants to know whether it is possible for him to guarantee that he would visit sacred shrines inifinite number of times? Note that if Malthael gets stucked at a point with sacred shrine, it is also considered that he would be able to visit it an infinite number of times.

Of course, things are not so trivial as the adverserial dark force within will do whatever it can to stop him, misdirecting Malthael from shrines whenever possible. Help Malthael figure out if there's a plan for him to gain redemption is he starts from a particular location.

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 100$) indicating the number of test cases.

Each test case starts with two integers N, K ($0 \leq K \leq N \leq 5000$), where N is the number of locations marked on the map of Caldeum, and K is the number of loactions that have a sacred shrine. The following line has K integers, indicating the indices of the vertices that have a shrine on it. After that is an integer M ($0 \leq M \leq 5000$), denoting the number of directed arcs on the map. M lines then follow, each containing two integers x_i and y_i, representing an arc from x_i to y_i.

Nodes are 1-indexed.

Output Format

For each test case, output in a line the list of nodes that if Malthael starts on, would guarantee a plan to redemption.

Nodes should be output as an increasing sequence. Please follow the sample output for the specific format.

Sample Input

```
3
3 1
1
3
1 2
2 3
3 1
4 2
1 2
2
2 3
2 4
3 1
1
3
1 2
2 1
2 3
```

Sample Output

```
[1,2,3]
[1]
[2]
```

F. Forgotten Sequence

time limit: 4 second

You once knew this magic sequence, which is a permutation of 1 to n, but you no longer remember what it is.

Fortunately, you wrote down some clues of the sequence, and now you are going to reconstruct it from the clues.

Assume the sequence is $a_1, a_2, \cdots, a_n$. Your clues are the inversion count of every length-k subsequence, i.e. $a_i, \cdots, a_{i+k-1}$ for every i. Note that the inversion count of subsequence $a_i, \cdots, a_{i+k-1}$ is defined as

$$\sum_{p=i}^{i+k-1} \sum_{q=p+1}^{i+k-1} [\![a_p > a_q]\!]$$

where $[\![\cdot]\!]$ is the boolean indicator function which is 1 if the argument is true, and 0 otherwise.

Input Format

The first line of the input file contains an integer T $(1 \le T \le 100)$ indicating the number of test cases.

Each test case starts with two integers n and k $(2 \le n \le 10^5,\ 2 \le k \le 5,\ k \le n)$. The following line contains $n-k+1$ integers, where the i-th integer denotes the inversion count of the subsequence of length k starting at a_i.

The input is guaranteed to be valid, that is, there is at least one solution.

Output Format

For each test case, output n integers on a single line separated by a whitespace, representing the sequence $a_1, \cdots, a_n$. If there are more than one solution, output any of them.

Sample Input

```
1
3 2
0 1
```

Sample Output

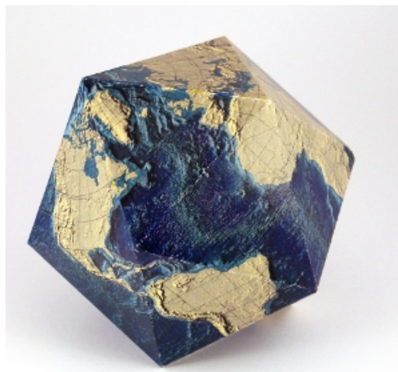
```
1 3 2
```

(This page was intentionally left blank.)

G. Geodesic

time limit: 3 seconds

The planet you live on is a convex polyhedron. Given two points on the surface, find out the distance of the shortest route between these two points on the surface. Notice that the route can not leave the surface of the planet.



The polyhedron is composed of triangle faces, and no two triangles lie on the same plane. The distance between any two vertices of the polyhedron is larger than 1.

Input Format

The first line of the input file contains an integer T ($1 \le T \le 50$) indicating the number of test cases.

Each test case starts with two integers N, M ($4 \le N \le 20$, $4 \le M \le 30$), indicating the number of vertices and the number of faces.

Each of the next N lines contains three real numbers x_i, y_i, z_i ($-10 \le x_i, y_i, z_i \le 10$), indicating the coordinate of the i-th vertex, None of these real numbers has more than 2 digits after the decimal point.

Each of the next M lines contains three integers a_i, b_i, c_i ($1 \le a_i, b_i, c_i \le N$), indicating the indices of the vertices of the i-th triangle face. a_i, b_i and c_i are pairwise distinct. Besides, a_i, b_i, c_i will be in clockwise order if you view this triangle from the inside of the polyhedron.

The next line contains two integers a, b ($1 \le a, b \le M$), indicating two points p_a, p_b, that p_a is the centroid of the a-th triangle face and p_b is the centroid of the b-th triangle face. You have to measure the distance of the shortest route between p_a and p_b.

Output Format

For each test case, output a single real number which is the minimum distance. Absolute or relative error less then 10^{-6} is allowed.

Sample Input

```
2
4 4
0.0 0.0 0.0
0.0 0.0 2.0
0.0 2.0 0.0
2.0 0.0 0.0
1 2 3
3 4 1
2 1 4
2 4 3
1 2
6 8
0 -1 0
1 0 0
0 1 0
-1 0 0
0 0 -1
0 0 1
1 2 6
2 3 6
3 4 6
4 1 6
2 1 5
3 2 5
4 3 5
1 4 5
5 3
```

Sample Output

```
1.33333333
2.16024690
```

H. Halin Graph

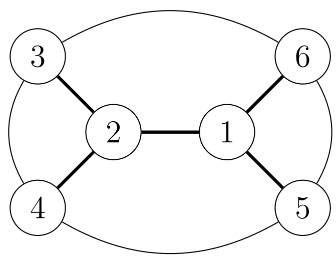
time limit: 8 seconds

Given a undirected weighted 3-regular Halin graph, find out the length of the minimum Hamiltonian cycle.

What? You say you've no idea what a Halin graph is? Consider a planar embedding of a tree. All leaves could thus be listed in clockwise order. Link leaves one by one following the order until you finish the whole loop, and you get a Halin graph!

What? You say you don't know 3-regular either? Well, a graph is said to be k-regular if every vertex has the same degree k, so each vertex in a 3-regular graph is connected by exactly 3 edges.

Here is an example of 3-regular Halin graph:



What? Did you ask what a Hamiltonian cycle is? It is a cycle on a graph that visits each vertex exactly once.

What? You say you don't know what a graph is? Maybe you should consider crying and confessing to your algorithm teacher.

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 30$) indicating the number of testcases.

Each test case starts with two integers N, M ($4 \leq N \leq 100000$, $0 \leq M \leq 150000$), indicating the number of the vertices and edges in the graph. The indices of vertices are in range $[1, N]$.

Then M lines follow, and each line contains 3 integers u_i, v_i, w_i, indicating a bidirected edge (u_i, v_i) with weight w_i ($1 \leq w_i \leq 1000$) in the graph.

Output Format

For each test case, output one line containing a number, which is the weight of the minimum weight Hamiltonian cycle.

If there's no Hamiltonian cycle exist, output -1.

Sample Input

```
2
4 6
1 2 1
2 3 1
3 4 1
4 1 1
1 3 2
2 4 2
6 9
1 2 1
2 3 1
2 4 1
1 5 1
1 6 1
3 4 3
4 5 3
5 6 3
6 3 3
```

Sample Output

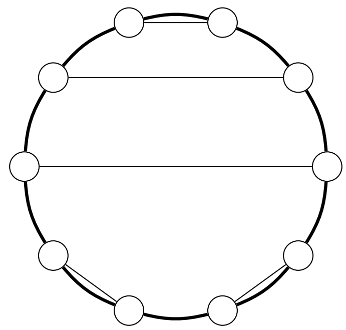
```
4
10
```

I. Iconic Sigil

time limit: 1 second

Seanwu is a mighty warrior who has conquered countless cities. Poems and songs of him are sung by the children, statues are carved in his figure, and young girls dreamt of him as the most perfect prince that could ever happen.

However to Seanwu, no legend of a warrior is complete without a perfect sigil on his shield. Therefore he decided to start pondering about what his shield sigil would be.



Formally, Seanwu's shield could be seen as a circle, and he has already marked $2N$ points on the perimeter. A valid design consists of all points on the perimeters being pairwisely connected by straight lines, where each point are connected to exactly one other point. Furthermore, to maintain a sense of elegance, two links must not intersect.

Seanwu wonders how many valid designs are there? Note that since the shield is a perfect circle, and the points on the perimeters are equidistant, two designs are considered identical if after rotation they look the same.

Input Format

The first line of the input file contains an integer T ($1 \leq T \leq 100$) indicating the number of test cases.

Each test case consists of a single integer N on a line, meaning there are a total of $2N$ points around Seanwu's shield. ($1 \leq N \leq 100000$)

Output Format

For each testcase, output a single integer representing the number of distinct valid designs. Please output the answer modulo $10^9 + 7$.

Sample Input

```
6
2
3
5
8
13
514
```

Sample Output

```
1
2
6
95
28640
525492395
```

J. Journey to the Gold

time limit: 5 seconds

You are on a grid of $W \times H$ cells. Some of the cells are *gold* cells, and others are regular. A gold cell provides you one unit of gold whenever you enter it, while a regular cell provides you nothing.

You currently stand on a regular cell. Each second you have to move to one of 4 neighboring cells (up, down, left, or right), but you cannot go out of the grid. You cannot stay at the same cell for any consecutive two seconds, but you may enter any cell multiple times. Each time you enter a gold cell, you will receive a unit of gold.

Your goal is to collect t units of gold. How fast can you do that?

Input Format

The first line of the input file contains an integer T ($1 \le T \le 60$) indicating the number of test cases.

Each test case starts with a line of three integers W, H, N ($1 \le W, H \le 1000$, $1 \le N \le 100000$, $N < W \cdot H$), indicating the width and height of the grid, and the number of gold cells. Then N lines follow, each containing two intergers x_i, y_i ($1 \le x_i \le W$, $1 \le y_i \le H$), indicating that the cell (x_i, y_i) are a gold cell.

The following line containins an integer Q ($Q \le 10000$), denoting the number of queries. Each of the following Q lines contains three integers, x_q, y_q, t, denoting the starting cell (x_q, y_q) and the amount of gold needed. ($1 \le x_q \le W$, $1 \le y_q \le H$, $t \le 10^9$) The starting cell is always a regular cell.

Output Format

For each query, output one line containing of a single integer representing the minimum time (in seconds) you need to collect t units of gold.

Sample Input

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

Sample Output

```
8
1
```

(This page was intentionally left blank.)