Problem A. Caocao's Bridges

Description

Caocao was defeated by Zhuge Liang and Zhou Yu in the battle of Chibi. But he wouldn't give up. Caocao's army still was not good at water battles, so he came up with another idea. He built many islands in the Changjiang river, and based on those islands, Caocao's army could easily attack Zhou Yu's troop. Caocao also built bridges connecting islands. If all islands were connected by bridges, Caocao's army could be deployed very conveniently among those islands. Zhou Yu couldn't stand with that, so he wanted to destroy some Caocao's bridges so one or more islands would be seperated from other islands. But Zhou Yu had only one bomb which was left by Zhuge Liang, so he could only destroy one bridge. Zhou Yu must send someone carrying the bomb to destroy the bridge. There might be guards on bridges. The soldier number of the bombing team couldn't be less than the guard number of a bridge, or the mission would fail. Please figure out as least how many soldiers Zhou Yu have to sent to complete the island seperating mission.

Input

There are no more than 12 test cases.

In each test case:

The first line contains two integers, N and M, meaning that there are N islands and M bridges. All the islands are numbered from 1 to N. (2 <= N <= 1000, 0 < M <= N^2)

Next M lines describes M bridges. Each line contains three integers U,V and W, meaning that there is a bridge connecting island U and island V, and there are W guards on that bridge. (U $\neq$ V and 0 <= W <= 10,000)

The input ends with N = 0 and M = 0.

Output

For each test case, print the minimum soldier number Zhou Yu had to send to complete the mission. If Zhou Yu couldn't succeed any way, print -1 instead.

Sample Input

```
3 3
1 2 7
2 3 4
3 1 4
3 2
1 2 7
2 3 4
0 0
```

Sample Output

-1
4

Problem B. Zhuge Liang's mines

Description

In the ancient three kingdom period, Zhuge Liang was the most famous and smartest military leader. His enemy was Shima Yi, who always looked stupid when fighting against Zhuge Liang. But it was Shima Yi who laughed to the end.

Once, Zhuge Liang sent the arrogant Ma Shu to defend Jie Ting, a very important fortress. Because Ma Shu is the son of Zhuge Liang's good friend Ma liang, even Liu Bei, the Ex. king, had warned Zhuge Liang that Ma Shu was always bragging and couldn't be used, Zhuge Liang wouldn't listen. Shima Yi defeated Ma Shu and took Jie Ting. Zhuge Liang had to kill Ma Shu and retreated. To avoid Shima Yi's chasing, Zhuge Liang put some mines on the only road. Zhuge Liang deployed the mines in a Bagua pattern which made the mines very hard to remove. If you try to remove a single mine, no matter what you do ,it will explode. Ma Shu's son betrayed Zhuge Liang , he found Shima Yi, and told Shima Yi the only way to remove the mines: If you remove four mines which form the four vertexes of a square at the same time, the removal will be success. In fact, Shima Yi was not stupid. He removed as many mines as possible. Can you figure out how many mines he removed at that time?

The mine field can be considered as a the Cartesian coordinate system. Every mine had its coordinates. To simplify the problem, please only consider the squares which are parallel to the coordinate axes.

Input

There are no more than 15 test cases.
In each test case:

The first line is an integer N, meaning that there are N mines(0 < N <= 20).

Next N lines describes the coordinates of N mines. Each line contains two integers X and Y, meaning that there is a mine at position (X,Y). (0 <= X,Y <= 100)

The input ends with N = -1.

Output

For each test case ,print the maximum number of mines Shima Yi removed in a line.

Sample Input

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

8
0 0
1 0
2 0
0 1
1 1
2 1
10 1
10 0
-1

Sample Output

0
4

Problem C. The donkey of Gui Zhou

Description

There was no donkey in the province of Gui Zhou, China. A trouble maker shipped one and put it in the forest which could be considered as an N×N grid. The coordinates of the up-left cell is (0,0) , the down-right cell is (N-1,N-1) and the cell below the up-left cell is (1,0)..... A 4×4 grid is shown below:

(0,0)	(0,1)		(0,3)
(1,0)			
(2,0)			
(3,0)			(3,3)

The donkey lived happily until it saw a tiger far away. The donkey had never seen a tiger ,and the tiger had never seen a donkey. Both of them were frightened and wanted to escape from each other. So they started running fast. Because they were scared, they were running in a way that didn't make any sense. Each step they moved to the next cell in their running direction, but they couldn't get out of the forest. And because they both wanted to go to new places, the donkey would never stepped into a cell which had already been visited by itself, and the tiger acted the same way. Both the donkey and the tiger ran in a random direction at the beginning and they always had the same speed. They would not change their directions until they couldn't run straight ahead any more. If they couldn't go ahead any more ,they changed their directions immediately. When changing direction, the donkey always turned right and the tiger always turned left. If they made a turn and still couldn't go ahead, they would stop running and stayed where they were, without trying to make another turn. Now given their starting positions and directions, please count whether they would meet in a cell.

Input

There are several test cases.

In each test case:
First line is an integer N, meaning that the forest is a N×N grid.

The second line contains three integers R, C and D, meaning that the donkey is in the cell (R,C) when they started running, and it's original direction is D. D can be 0, 1, 2 or 3. 0 means east, 1 means south , 2 means west, and 3 means north.

The third line has the same format and meaning as the second line, but it is for the tiger.

The input ends with N = 0.   (2 <= N <= 1000, 0 <= R, C < N)

Output

For each test case, if the donkey and the tiger would meet in a cell, print the coordinate of the cell where they meet first time.   If they would never meet, print -1 instead.

Sample Input

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

Sample Output

```
-1
1 3
```

Problem D.   Save Labman No.004

Description

Due to the preeminent research conducted by Dr. Kyouma, human beings have a breakthrough in the understanding of time and universe. According to the research, the universe in common sense is not the only one. Multi World Line is running simultaneously. In simplicity, let us use a straight line in three-dimensional coordinate system to indicate a single World Line.

During the research in World Line Alpha, the assistant of Dr. Kyouma, also the Labman No.004, Christina dies. Dr. Kyouma wants to save his assistant. Thus, he has to build a Time Tunnel to jump from World Line Alpha to World Line Beta in which Christina can be saved. More specifically, a Time Tunnel is a line connecting World Line Alpha and World Line Beta. In order to minimizing the risks, Dr. Kyouma wants you, Labman No.003 to build a Time Tunnel with shortest length.

Input

The first line contains an integer T, indicating the number of test cases.

Each case contains only one line with 12 float numbers (x1, y1, z1), (x2, y2, z2), (x3, y3, z3), (x4, y4, z4), correspondingly indicating two points in World Line Alpha and World Line Beta. Note that a World Line is a three-dimensional line with infinite length.

Data satisfy T <= 10000, |x, y, z| <= 10,000.

Output

For each test case, please print two lines.

The first line contains one float number, indicating the length of best Time Tunnel.

The second line contains 6 float numbers (xa, ya, za), (xb, yb, zb), seperated by blank, correspondingly indicating the endpoints of the best Time Tunnel in World Line Alpha and World Line Beta.

All the output float number should be round to 6 digits after decimal point. Test cases guarantee the uniqueness of the best Time Tunnel.

Sample Input

```
1
1 0 1 0 1 1 0 0 0 1 1 1
```

Sample Output

```
0.408248
0.500000 0.500000 1.000000 0.666667 0.666667 0.666667
```

Problem E. Pinball Game 3D

Description

RD is a smart boy and excel in pinball game. However, playing common 2D pinball game for a great number of times results in accumulating tedium.

Recently, RD has found a new type of pinball game, a 3D pinball game. The 3D pinball game space can be regarded as a three dimensional coordinate system containing N balls. A ball can be considered as a point. At the beginning, RD made a shot and hit a ball. The ball hit by RD will move and may hit another ball and the "another ball" may move and hit another another ball, etc. But once a ball hit another ball, it will disappear.

RD is skilled in this kind of game, so he is able to control every ball's moving direction. But there is a limitation: if ball A's coordinate is (x1,y1,z1) and ball B's coordinate is (x2,y2,z2), then A can hit B only if x1 <= x2 and y1 <= y2 and z1 <= z2.

Now, you should help RD to calculate the maximum number of balls that can be hit and the number of different shooting schemes that can achieve that number. Two schemes are different if the sets of hit balls are not the same. The order doesn't matter.

Input

The first line contains one integer T indicating the number of cases.
In each case, the first line contains one integer N indicating the number of balls.
The next N lines each contains three non-negative integer (x, y, z), indicating the coordinate of a ball.
The data satisfies T <= 3, N <= 10^5, 0 <= x, y, z <= 2^{30}, no two balls have the same coordinate in one case.

Output

Print two integers for each case in a line, indicating the maximum number of balls that can be hit and the number of different shooting schemes. As the number of schemes can be quite large, you should output this number mod 2^{30}.

Sample Input

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

0 2 2
3 3 3

Sample Output

2 1
3 2

Note

In the first case, RD can shoot the second ball at first and hit the third ball indirectly.
In the second case, RD can shoot the second or the third ball initially and hit the fourth ball as well as the fifth ball. Two schemes are both the best.

Problem F. Sparrow

Description

Sparrow is a kind of game. Four players (1,2,3,4) sits around a table. They take turns to act. Player i+1 is next to player i(i = 1,2,3) and player 1 is next to player 4

There are two categories of tiles in this game:
1) Simple tiles:
Simple tiles have 27 kinds: A1,A2...A9, B1,B2...B9 and C1,C2...C9. Each kind have 4 copies.

2) Honor tiles:
Honor tiles have seven kinds: E, S, W, N, Z, F, 0(zero) . Each of them also have 4 copies.

Several tiles can form a good set. There are 4 kinds of good set:

1) Pong is a set of three identical tiles.
2) Kong is a set of four identical tiles.
3) Chow is a set of three continuous SIMPLE tiles which have the same leading letter, such as A2,A3 and A4, or C7,C8 and C9. A1,B2 and A3 don't form a Chow because their leading letters are not the same.
4) Eye is a set of two identical tiles.

A tile can't belong to more than one good set.

A tile can be in one of three state:
1) On the wall.
2) Belongs to one player and faces down.
3) Belongs to one player and faces up.

At the beginning of the game, some tiles are on the wall, some tiles belong to players and face down. Then they play round by round. They draw tiles from the wall, discard some tiles, make some tiles face up, until a player declares that he/she holds a winning hand, or there is no tiles on the wall.

One player holds a winning hand, if and only if he can split his face-down tiles into zero, one or several sets of Pong/Chow, plus one set of Eye. A winning hand could be:

{E, E, E, N, N, N, 0, 0}, or {C1, C2, C3, B5, B5}

The one who declares a winning hand wins the game.

In each round, mostly , the round owner draw a tile from the wall and put it face-down. If the round owner finds out that the he/she has a winning hand, he/she will declare it and win the game. If no so, the round owner choose one (we call it T)of his face-down tiles to discard. Then comes the "declare phase". The other 3 players looks into their FACE-DOWN tiles, and may do 3 kinds of declarations below:

1) D0: declare a winning hand

If T can form a winning hand after one player owns it, this player can own T and declare a winning hand.

2) D1: declare a Pong.

If T forms a Pong set after one player owns it, then that player owns T, declares this Pong set, and changes the 3 tiles of the declared Pong set into face-up.

3)D2: declare a Kong.
Similar to D1, that player owns T , declare this Kong set, , and changes the 4 tiles of the declared Kong set into face-up.

In "declare phase", first they will ask each other whether someone want to declare a D0. If everyone give up, go on asking for D1. If no one declare a D1, go on asking D2.

If multiple declares happen at the same time, the player with the smallest id get T.

A declare will end a round. If one player declares, he gets T and becomes the owner of the next round. However, after D1 the new round owner does not draw a tile from the wall, he/she just discards a tile . After D2, the new round owner draws a tile then discards a tile.

If no one declare in the "declare phase", the player next to the round owner becomes the new round owner and draw a tile from the wall.

At any time, if someone gets a winning hand, the game ends. If one needs to draw a tile from the wall, but the wall has no more tiles, then the game ends with no winner.

One day, Alice and Bob suddenly want to play sparrow. But sparrow needs four players, and they could not find anyone in a moment. So they decide to play a two-man sparrow, that is, Alice will act as player 1 and player 3, and Bob will act as player 2 and player 4. Both of them control two roles at the same time.

Alice and Bob are very very clever children. And they both have a super gift: perspective. They can see any tile, even if it's face-down on the wall, or face-down owned by the other player!

(Now, you know why they can find no one playing sparrow with them ^o^)

The game goes very exciting!(oh it makes nonsense, since they know what tiles the other holds, and they are both clever enough to find the other's strategy) And finally, it comes to such a situation that: after player 4 discards a tile, no one declare and player 1 (controled by Alice) becomes the round owner. Alice is ready to draw a tile.

There are only N tiles on the wall. All four players each have ten face-down tiles.

Decide whether Alice or Bob could win the game, if both of them act the optimal way.

Input

First line is an integer Q: the number of input data set.
Then comes Q input set, each with 5 lines. The first 4 lines each describe ten tiles of player 1..4.

The fifth line of each dataset describes the tiles on the wall. First the integer N, then comes N tiles in order.

There may be extra blank lines and spaces in the input file. Q<=20, and N<=4.

Output

If Alice could win the game, print "Alice win";
If Bob could win the game, print "Bob win";
Otherwise, print "tie".

Sample Input

```
3
A1 A1 A1 A2 A2 A2 A3 A3 A3 A8
B1 B1 B1 B2 B2 B2 B3 B3 B3 B4
B4 B5 B5 B5 B6 B6 B6 B8 B8 B8
Z  Z  Z  F  F  F  S  S  S  N
2  A5 B4
A1 A1 A1 A2 A2 A2 A3 A3 A3 B8
B1 B1 B1 B2 B2 B2 B3 B3 B3 B4
B4 B5 B5 B5 B6 B6 B6 B8 B8 B8
Z  Z  Z  F  F  F  S  S  S  N
2  A5 B4
A1 A1 A1 A2 A2 A2 A3 A3 A3 B8
B1 B1 B1 B2 B2 B2 B3 B3 B3 B4
B4 B5 B5 B5 B6 B6 B6 B8 B8 B8
Z  Z  Z  F  F  F  S  S  S  N
1  Z
```

Sample Output

```
Bob win
Alice win
tie
```

Problem G. Starloop System

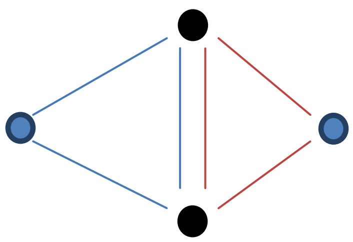
Description

At the end of the 200013th year of the Galaxy era, the war between Carbon-based lives and Silicon civilization finally comes to its end with the Civil Union born from the ruins. The shadow fades away, and the new-born Union is opening a new page.

Although the war ended, the affect of the war is far from over. Now the council is busy fixing the transportation system. Before the war, all the stars were connected with artificial wormholes which were destroyed during the war. At the same time, natural wormholes are breaking down with the growing traffic. A new traffic system is on the schedule.

As two civilizations combine, the technology integrates. Scientists find a new traffic system called the Starloop System.

This system is made up of several starloops. People build a starway to connect two stars. A startloop is a closed path with no repetitions of stars or starways allowed, other than the repetition of the starting and ending star. And a starloop contains at least two starts and two starways. A startloop's cost is the sum of the length of all the starways in it. Length of a starway connecting two stars is floor(x), which x is the euclidean distance between two stars. You can build more than one starway between any two stars, but one starway can only belongs to one starloop.



As the picture above shows, there are two starloops. One is blue and the other one is brown.

As a starloop is set up, each star on the starloop will get a unit of star-energy. So the two blue stars can get one unit of star-energy, and at the same time the black two stars can get two units because they both belong to two starloops. When a star earns a certain number of energy units, the transporter on that star will be activated. One can easily travel between any two stars whose transporter is activated.

Now the council wants to know the minimal cost to build a starloop system on all the stars . In other words, every star's transporter should be activated

Input

There are multiple test cases.
For each test case:

There is a line with one integer n which is the number of stars.

The following n lines each describes a star by four integers xi, yi, zi and wi, defined as the spatial coordinate and the number of energy units the star needs to activate the transporter. Please NOTE that getting more than wi energy units will put the star in a dangerous situation, so it is not allowed.
The input ends with n = 0.

$$1 \le n \le 100$$
$$|xi|, |yi|, |zi| \le 200$$
$$wi \le 50$$

Output

For each test case, output one line that contains an integer equals to the minimal cost you can get. If there is no solution, just output -1;

Sample Input

3
0 0 2 1
0 2 0 1
2 0 0 1
3
0 0 2 2
0 2 0 1
2 0 0 1
3
0 0 2 3
0 2 0 1
2 0 0 1
0

Sample Output

6
8
-1

Problem H. Two Rabbits

Description

Long long ago, there lived two rabbits Tom and Jerry in the forest. On a sunny afternoon, they planned to play a game with some stones. There were n stones on the ground and they were arranged as a clockwise ring. That is to say, the first stone was adjacent to the second stone and the n-th stone, and the second stone is adjacent to the first stone and the third stone, and so on. The weight of the i-th stone is a_i.

The rabbits jumped from one stone to another. Tom always jumped clockwise, and Jerry always jumped anticlockwise.

At the beginning, the rabbits both choose a stone and stand on it. Then at each turn, Tom should choose a stone which have not been stepped by itself and then jumped to it, and Jerry should do the same thing as Tom, but the jumping direction is anti-clockwise.

For some unknown reason, at any time , the weight of the two stones on which the two rabbits stood should be equal. Besides, any rabbit couldn't jump over a stone which have been stepped by itself. In other words, if the Tom had stood on the second stone, it cannot jump from the first stone to the third stone or from the n-the stone to the 4-th stone.

Please note that during the whole process, it was OK for the two rabbits to stand on a same stone at the same time.

Now they want to find out the maximum turns they can play if they follow the optimal strategy.

Input

The input contains at most 20 test cases.
For each test cases, the first line contains a integer n denoting the number of stones.
The next line contains n integers separated by space, and the i-th integer a_i denotes the weight of the i-th stone.(1 <= n <= 1000, 1 <= a_i <= 1000)
The input ends with n = 0.

Output

For each test case, print a integer denoting the maximum turns.

Sample Input

1
1
4
1 1 2 1
6

2 1 1 2 1 3
0

Sample Input

1
4
5

Note

For the second case, the path of the Tom is 1, 2, 3, 4, and the path of Jerry is 1, 4, 3, 2.
For the third case, the path of Tom is 1,2,3,4,5 and the path of Jerry is 4,3,2,1,5.

Problem I. Mophues

Description

As we know, any positive integer C (C >= 2) can be written as the multiply of some prime numbers:

$C = p_1 \times p_2 \times p_3 \times \dots \times p_k$

which p_1, p_2 ... p_k are all prime numbers.For example, if C = 24, then:

$24 = 2 \times 2 \times 2 \times 3$

here, $p_1 = p_2 = p_3 = 2$, $p_4 = 3$, $k = 4$

Given two integers P and C. if k<=P(k is the number of C's prime factors), we call C a lucky number of P.

Now, XXX needs to count the number of pairs (a, b), which 1<=a<=n , 1<=b<=m, and gcd(a,b) is a lucky number of a given P ("gcd" means "greatest common divisor").

Please note that we define 1 as lucky number of any non-negative integers because 1 has no prime factor.

Input

The first line of input is an integer Q meaning that there are Q test cases.
Then Q lines follow, each line is a test case and each test case contains three non-negative numbers: n, m and P (n, m, P <= 5×10^5. Q <=5000).

Output

For each test case, print the number of pairs (a, b), which 1<=a<=n , 1<=b<=m, and gcd(a,b) is a lucky number of P.

Sample Input

```
2
10 10 0
10 10 1
```

Sample Output

```
63
93
```

Problem J. Mex

Description

Mex is a function on a set of integers, which is universally used for impartial game theorem. For a non-negative integer set S, mex(S) is defined as the least non-negative integer which is not appeared in S. Now our problem is about mex function on a sequence.

Consider a sequence of non-negative integers {ai}, we define mex(L,R) as the least non-negative integer which is not appeared in the continuous subsequence from aL to aR, inclusive. Now we want to calculate the sum of mex(L,R) for all 1 <= L <= R <= n.

Input

The input contains at most 20 test cases.
For each test case, the first line contains one integer n, denoting the length of sequence.
The next line contains n non-integers separated by space, denoting the sequence.
(1 <= n <= 200000, 0 <= ai <= 10^9)
The input ends with n = 0.

Output

For each test case, output one line containing a integer denoting the answer.

Sample Input

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

Sample Output

```
5
24
```

Note

For the first test case:
mex(1,1)=1, mex(1,2)=2, mex(1,3)=2, mex(2,2)=0, mex(2,3)=0,mex(3,3)=0.
1 + 2 + 2 + 0 +0 +0 = 5.