6134 Different Digits

The inhabitants of Nlogonia are very superstitious. One of their beliefs is that street house numbers that have a repeated digit bring bad luck for the residents. Therefore, they would never live in a house which has a street number like 838 or 1004.

The Queen of Nlogonia ordered a new seaside avenue to be built, and wants to assign to the new houses only numbers without repeated digits, to avoid discomfort among her subjects. You have been appointed by Her Majesty to write a program that, given two integers N and M, determines the maximum number of houses that can be assigned street numbers between N and M, inclusive, that do not have repeated digits.

Input

Each test case is described using one line. The line contains two integers N and M, as described above ($1 \leq N \leq M \leq 5000$).

Output

For each test case output a line with an integer representing the number of street house numbers between N and M, inclusive, with no repeated digits.

Sample Input

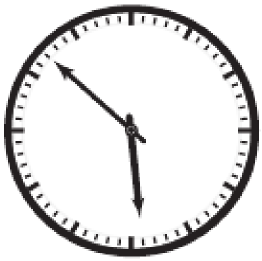
```
87 104
989 1022
22 25
1234 1234
```

Sample Output

```
14
0
3
1
```

6138 Hours and Minutes

Heidi has a discrete analog clock in the shape of a circle, as the one in the figure. Two hands rotate around the center of the circle, indicating hours and minutes. The clock has 60 marks placed around its perimeter, with the distance between consecutive marks being constant.



The minute hand moves from its current mark to the next exactly once every minute. The hour hand moves from its current mark to the next exactly once every 12 minutes, so it advances five marks each hour.

We consider that both hands move discretely and instantly, which means they are always positioned exactly over one of the marks and never in between marks.

At midnight both hands reach simultaneously the top mark, which indicates zero hours and zero minutes. After exactly 12 hours or 720 minutes, both hands reach the same position again, and this process is repeated over and over again. Note that when the minute hand moves, the hour hand may not move; however, when the hour hand moves, the minute hand also moves.

Heidi likes geometry, and she likes to measure the minimum angle between the two hands of the clock at different times of the day. She has been writing some measures down, but after several years and a long list, she noticed that some angles were repeated while some others never appeared. For instance, Heidi's list indicates that both at three o'clock and at nine o'clock the minimum angle between the two hands is 90 degrees, while an angle of 65 degrees does not appear in the list. Heidi decided to check, for any integer number A between 0 and 180, if there exists at least one time of the day such that the minimum angle between the two hands of the clock is exactly A degrees. Help her with a program that answers this question.

Input

Each test case is described using one line. The line contains an integer A representing the angle to be checked ($0 \leq A \leq 180$).

Output

For each test case output a line containing a character. If there exists at least one time of the day such that the minimum angle between the two hands of the clock is exactly A degrees, then write the uppercase letter '`Y`'. Otherwise write the uppercase letter '`N`'.

Sample Input

```
90
65
66
67
128
0
180
```

Sample Output

```
Y
N
Y
N
N
Y
Y
```

7582 Oil

A large part of the world economy depends on oil, which is why research into new methods for finding and extracting oil is still active. Profits of oil companies depend in part on how efficiently they can drill for oil. The International Crude Petroleum Consortium (ICPC) hopes that extensive computer simulations will make it easier to determine how to drill oil wells in the best possible way.

Drilling oil wells optimally is getting harder each day — the newly discovered oil deposits often do not form a single body, but are split into many parts. The ICPC is currently concerned with stratified deposits, as illustrated in Figure G.1.

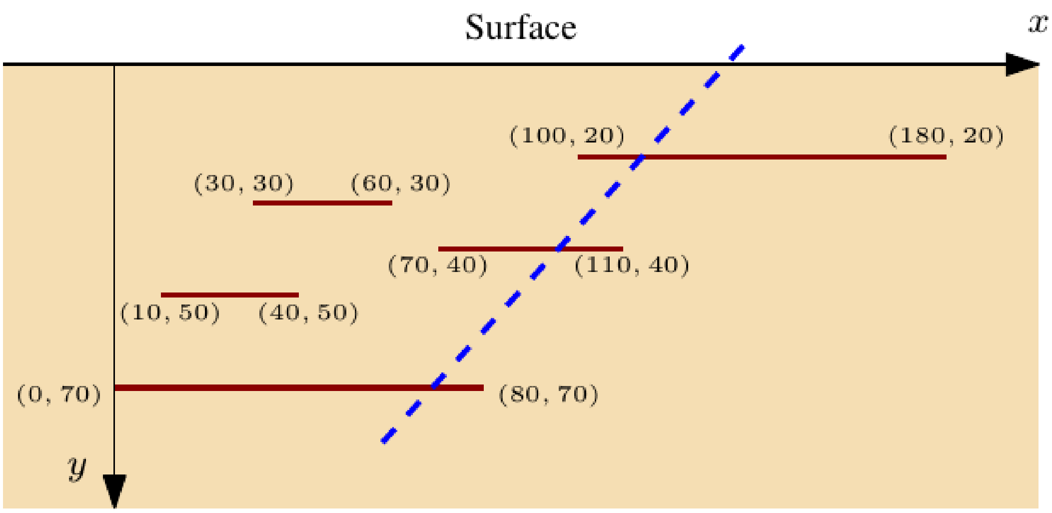


Figure G.1: Oil layers buried in the earth. This figure corresponds to Sample Input 1.

To simplify its analysis, the ICPC considers only the 2-dimensional case, where oil deposits are modeled as horizontal line segments parallel to the earth's surface. The ICPC wants to know how to place a single oil well to extract the maximum amount of oil. The oil well is drilled from the surface along a straight line and can extract oil from all deposits that it intersects on its way down, even if the intersection is at an endpoint of a deposit. One such well is shown as a dashed line in Figure G.1, hitting three deposits. In this simple model the amount of oil contained in a deposit is equal to the width of the deposit. Can you help the ICPC determine the maximum amount of oil that can be extracted by a single well?

Input

The input file contains several test cases, each of them as described below.

The first line of input contains a single integer n ($1 \le n \le 2000$), which is the number of oil deposits. This is followed by n lines, each describing a single deposit. These lines contain three integers x_0, x_1, and y giving the deposit's position as the line segment with endpoints (x_0, y) and (x_1, y). These numbers satisfy $|x_0|, |x_1| \le 10^6$ and $1 \le y \le 10^6$. No two deposits will intersect, not even at a point.

Output

For each test case, display the maximum amount of oil that can be extracted by a single oil well on a line by itself.

Sample Input

```
5
100 180 20
30 60 30
70 110 40
10 40 50
0 80 70
3
50 60 10
-42 -42 20
25 0 10
```

Sample Output

```
200
25
```

6763 Modified LCS

LCS stands for longest common subsequence, and it is a well known problem. A sequence in this problem means a list of integers, and a sequence X is considered a subsequence of another sequence Y, when the sequence X can be obtained by deleting zero or more elements from the sequence Y without changing the order of the remaining elements.

In this problem you are given two sequences and your task is to find the length of the longest sequence which is a subsequence of both the given sequences.

You are not given the sequences themselves. For each sequence you are given three integers N, F and D, where N is the length of the sequence, F is the first element in the sequence. Each element except the first element is greater than the element before it by D.

For example $N = 5$, $F = 3$ and $D = 4$ represents the following sequence: [3, 7, 11, 15, 19].

There will be at least one integer which belongs to both sequences and it is not greater than 1,000,000.

Input

Your program will be tested on one or more test cases. The first line of the input will be a single integer T, the number of test cases ($1 \leq T \leq 100$). Followed by the test cases, each test case is described in one line which contains 6 integers separated by a single space $N1$ $F1$ $D1$ $N2$ $F2$ $D2$ ($1 \leq N1, N2 \leq 10^{18}$) and ($1 \leq F1, D1, F2, D2 \leq 10^9$) representing the length of the first sequence, the first element in the first sequence, the incremental value of the first sequence, the length of the second sequence, the first element in the second sequence and the incremental value of the second sequence, respectively.

Output

For each test case, print a single line which contains a single integer representing the length of the longest common subsequence between the given two sequences.

Sample Input

```
3
5 3 4 15 3 1
10 2 2 7 3 3
100 1 1 100 1 2
```

Sample Output

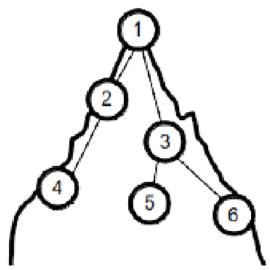
```
4
3
50
```

6829 Intrepid climber

Who would guess? You climbed the highest mountain of your city. You are so excited about it that you need to tell it to all your friends, and you decided to start with those that are trying to be exactly where you are at this precise moment.

The mountain has N landmarks, and one of them is the top of the mountain, where you are now. Each of your friends that is climbing the mountain is in some other landmark, and you want to visit all of them. There are tracks that connect pairs of landmarks in such a way that there exists exactly one route (that is, a sequence of consecutive tracks) that goes down from the top of the mountain to each other landmark. To visit two friends in two different landmarks, you may have to go down some tracks, climb others, and go down others again. Going down the mountain is "easy", so you do not consume energy when you go down through the tracks. But each time you climb a track, you consume a certain amount of energy. After visiting all your friends, you can just sit and rest.

For example, consider the mountain in the picture below, which has $N = 6$ landmarks. If your friends are in landmarks 5 and 2, you can visit both if you follow the sequence of landmarks $1 \downarrow 2 \uparrow 1 \downarrow 3 \downarrow 5$, where $a \downarrow b$ means that you go down a track from landmark a to landmark b, and $a \uparrow b$ means that you climb a track from landmark a to landmark b. Another possible sequence is $1 \downarrow 3 \downarrow 5 \uparrow 3 \uparrow 1 \downarrow 2$.



Given the tracks between the landmarks, the energy required to climb them, and the landmarks where your friends are, compute the minimum total amount of energy required to visit all your friends from the top of the mountain.

Input

The input contains several test cases; each test case is formatted as follows. The first line contains two integers N and F ($1 \leq F < N \leq 10^5$), representing respectively the number of landmarks and the number of your friends that are climbing the mountain. Landmarks are identified with distinct integers from 1 to N, being 1 the top of the mountain, where you initially are. Each of the next $N-1$ lines describes a different track with three integers A, B and C, indicating that there is a track from A to B that goes down and requires an amount C of energy to be climbed ($1 \leq A \leq N$, $2 \leq B \leq N$, $A \neq B$ and $1 \leq C \leq 100$). The next line contains F different integers L_1, L_2, ..., L_F ($2 \leq L_i \leq N$ for $i = 1, 2, \ldots, F$) representing the landmarks where your friends are. You may assume that the tracks between landmarks are such that there exists exactly one route that goes down from the top of the mountain to each other landmark.

Output

For each test case in the input, output a line with an integer representing the minimum total amount of energy required to visit all your friends starting from the top of the mountain.

Sample Input

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

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

Sample Output

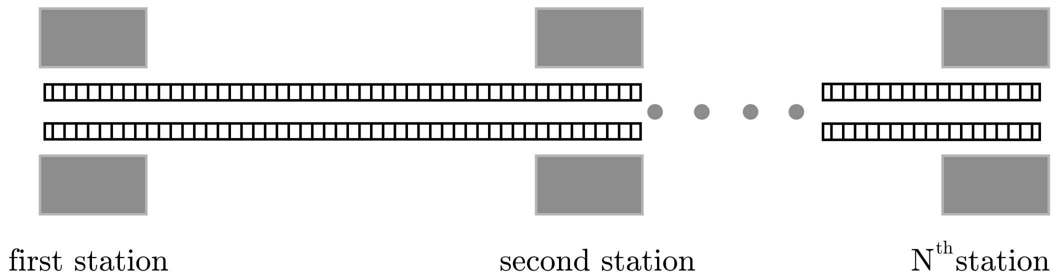
```
2
2
0
```

2728 A Spy in the Metro

Secret agent Maria was sent to Algorithms City to carry out an especially dangerous mission. After several thrilling events we find her in the first station of Algorithms City Metro, examining the time table. The Algorithms City Metro consists of a single line with trains running both ways, so its time table is not complicated.

Maria has an appointment with a local spy at the last station of Algorithms City Metro. Maria knows that a powerful organization is after her. She also knows that while waiting at a station, she is at great risk of being caught. To hide in a running train is much safer, so she decides to stay in running trains as much as possible, even if this means traveling backward and forward. Maria needs to know a schedule with minimal waiting time at the stations that gets her to the last station in time for her appointment. You must write a program that finds the total waiting time in a best schedule for Maria.

The Algorithms City Metro system has N stations, consecutively numbered from 1 to N. Trains move in both directions: from the first station to the last station and from the last station back to the first station. The time required for a train to travel between two consecutive stations is fixed since all trains move at the same speed. Trains make a very short stop at each station, which you can ignore for simplicity. Since she is a very fast agent, Maria can always change trains at a station even if the trains involved stop in that station at the same time.



Input

The input file contains several test cases. Each test case consists of seven lines with information as follows.

Line 1. The integer N ($2 \le N \le 50$), which is the number of stations.

Line 2. The integer T ($0 \le T \le 200$), which is the time of the appointment.

Line 3. $N - 1$ integers: $t_1, t_2, \ldots, t_{N-1}$ ($1 \le t_i \le 20$), representing the travel times for the trains between two consecutive stations: t_1 represents the travel time between the first two stations, t_2 the time between the second and the third station, and so on.

Line 4. The integer $M1$ ($1 \le M1 \le 50$), representing the number of trains departing from the first station.

Line 5. $M1$ integers: $d_1, d_2, \ldots, d_{M1}$ ($0 \le d_i \le 250$ and $d_i < d_{i+1}$), representing the times at which trains depart from the first station.

Line 6. The integer $M2$ ($1 \leq M2 \leq 50$), representing the number of trains departing from the N-th station.

Line 7. $M2$ integers: $e_1, e_2, \ldots, e_{M2}$ ($0 \leq e_i \leq 250$ and $e_i < e_{i+1}$) representing the times at which trains depart from the N-th station.

The last case is followed by a line containing a single zero.

Output

For each test case, print a line containing the case number (starting with 1) and an integer representing the total waiting time in the stations for a best schedule, or the word '`impossible`' in case Maria is unable to make the appointment. Use the format of the sample output.

Sample Input

```
4
55
5 10 15
4
0 5 10 20
4
0 5 10 15
4
18
1 2 3
5
0 3 6 10 12
6
0 3 5 7 12 15
2
30
20
1
20
7
1 3 5 7 11 13 17
0
```

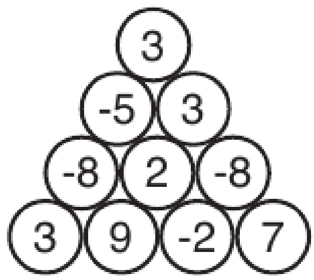
Sample Output

```
Case Number 1: 5
Case Number 2: 0
Case Number 3: impossible
```

ACM-ICPC Live Archive

5790 Ball Stacking

The XYZ TV channel is developing a new game show, where a contestant has to make some choices in order to get a prize. The game consists of a triangular stack of balls, each of them having an integer value, as the following example shows.



The contestant must choose which balls he is going to take and his prize is the sum of the values of those balls. However, the contestant can take any given ball only if he also takes the balls directly on top of it. This may require taking additional balls using the same rule. Notice that the contestant may choose not to take any ball, in which case the prize is zero.

The TV show director is concerned about the maximum prize a contestant can make for a given stack. Since he is your boss and he does not know how to answer this question, he assigned this task to you.

Input

Each test case is described using several lines. The first line contains an integer N representing the number of rows of the stack ($1 \leq N \leq 1000$). The i-th of the next N lines contains i integers B_{ij} ($-10^5 \leq B_{ij} \leq 10^5$ for $1 \leq j \leq i \leq N$); the number B_{ij} is the value of the j-th ball in the i-th row of the stack (the first row is the topmost one, and within each row the first ball if the leftmost one).

The last test case is followed by a line containing one zero.

Output

For each test case output a line with an integer representing the maximum prize a contestant can make from the stack.

Sample Input

```
4
3
-5 3
-8 2 -8
3 9 -2 7
2
-2
1 -10
3
1
-5 3
6 -4 1
0
```

Sample Output

```
7
0
```

6

7893 Hotel Rewards

You are planning to spend your holidays touring Europe, staying each night in a different city for N consecutive nights. You have already chosen the hotel you want to stay in for each city, so you know the price P_i of the room you'll be staying at during the i-th night of your holidays, for $i = 1, \ldots, N$.

You will book your accommodation through a website that has a very convenient rewards program, which works as follows. After staying for a night in a hotel you booked through this website you are awarded one point, and at any time you can exchange K of these points in your account for a free night in any hotel (which will however not give you another point).

For example, consider the case with $N = 6$ and $K = 2$ where the prices for the rooms are $P_1 = 10$, $P_2 = 3$, $P_3 = 12$, $P_4 = 15$, $P_5 = 12$ and $P_6 = 18$. After paying for the first four nights you would have four points in your account, which you could exchange to stay for free the remaining two nights, paying a total of $P_1 + P_2 + P_3 + P_4 = 40$ for your accommodation. However, if after the first three nights you use two of the three points you earned to stay the fourth night for free, then you can pay for the fifth night and use the final two points to get the sixth one for free. In this case, the total cost of your accommodation is $P_1 + P_2 + P_3 + P_5 = 37$, so this option is actually more convenient.

You want to make a program to find out what the minimum possible cost for your holidays' accommodation is. You can safely assume that all hotels you want to stay always will have a room available for you, and that the order of the cities you are going to visit cannot be altered.

Input

The input file contains several test cases, each of them as described below.

The first line of input contains two integers N and K, representing the total number of nights your holidays will last, and the number of points you need in order to get a free night ($1 \leq N, K \leq 10^5$). The second line contains N integers $P_1, P_2, \ldots, P_N$, representing the price of the rooms you will be staying at during your holidays ($1 \leq P_i \leq 10^4$ for $i = 1, 2, \ldots, N$).

Output

For each test case, output a line with one integer representing the minimum cost of your accommodation for all of your holidays.

Sample Input

```
6 2
10 3 12 15 12 18
6 1
10 3 12 15 12 18
5 5
1 2 3 4 5
```

Sample Output

```
37
25
15
```

Problem A

Catch the Plane

Time limit: 10 seconds

Your plane to the ICPC Finals departs in a short time, and the only way to get to the airport is by bus. Unfortunately, some of the bus drivers are considering going on strike, so you do not know whether you can get to the airport on time. Your goal is to plan your journey in such a way as to maximize the probability of catching your plane.

You have a detailed map of the city, which includes all the bus stations. You are at station 0 and the airport is at station 1. You also have a complete schedule of when each bus leaves its start station and arrives at its destination station. Additionally, for each bus you know the probability that it is actually going to run as scheduled, as opposed to its driver going on strike and taking the bus out of service. Assume all these events are independent. That is, the probability of a given bus running as planned does not change if you know whether any of the other buses run as planned.

If you arrive *before* the departure time of a bus, you can transfer to that bus. But if you arrive exactly at the departure time, you will not have enough time to get on the bus. You cannot verify ahead of time whether a given bus will run as planned – you will find out only when you try to get on the bus. So if two or more buses leave a station at the same time, you can try to get on only one of them.

Bus Schedule

Start Station	Destination Station	Departure Time	Arrival Time
0	1	0 (20%)	900
0	2	100	500
2	1	500	700
2	1	501 (10%)	701
0	3	200 (50%)	400
3	1	500 (10%)	800
3	0	550 (90%)	650
0	1	700 (10%)	900

Figure A.1: Bus schedule corresponding to Sample Input 1.

Consider the bus schedule shown in Figure A.1. It lists the start and destination stations of several bus routes along with the departure and arrival times. You have written next to some of these the probability that that route will run. Bus routes with no probability written next to them have a 100% chance of running. You can try catching the first listed bus. If it runs, it will take you straight to the airport, and you can stop worrying. If it does not, things get more tricky. You could get on the second listed bus to station 2. It is certain to leave, but you would be too late to catch the third listed bus which otherwise would have delivered you to the airport on time. The fourth listed bus – which you can catch – has only a 0.1 probability of actually running. That is a bad bet, so it is better to stay at station 0 and wait for the fifth listed bus. If you catch it, you can try to get onto the sixth listed bus to the airport; if that does not run, you still have the chance of returning to station 0 and catching the last listed bus straight to the airport.

Input

The first line of input contains two integers m ($1 \le m \le 10^6$) and n ($2 \le n \le 10^6$), denoting the number of buses and the number of stations in the city. The next line contains one integer k ($1 \le k \le 10^{18}$), denoting the time by which you must arrive at the airport.

Each of the next m lines describes one bus. Each line contains integers a and b ($0 \le a, b < n$, $a \ne b$), denoting the start and destination stations for the bus. Next are integers s and t ($0 \le s < t \le k$), giving the departure time from station a and the arrival time at station b. The last value on the line is p ($0 \le p \le 1$, with at most 10 digits after the decimal point), which denotes the probability that the bus will run as planned.

Output

Display the probability that you will catch your plane, assuming you follow an optimal course of action. Your answer must be correct to within an absolute error of 10^{-6}.

Sample Input 1

```
8 4
1000
0 1 0 900 0.2
0 2 100 500 1.0
2 1 500 700 1.0
2 1 501 701 0.1
0 3 200 400 0.5
3 1 500 800 0.1
3 0 550 650 0.9
0 1 700 900 0.1
```

Sample Output 1

```
0.3124
```

Sample Input 2

```
4 2
2
0 1 0 1 0.5
0 1 0 1 0.5
0 1 1 2 0.4
0 1 1 2 0.2
```

Sample Output 2

```
0.7
```

5133 Machine Works

You are the director of Arbitrarily Complex Machines (ACM for short), a company producing advanced machinery using even more advanced machinery. The old production machinery has broken down, so you need to buy new production machines for the company. Your goal is to make as much money as possible during the restructuring period. During this period you will be able to buy and sell machines and operate them for profit while ACM owns them. Due to space restrictions, ACM can own at most one machine at a time.

During the restructuring period, there will be several machines for sale. Being an expert in the advanced machines market, you already know the price P_i and the availability day D_i for each machines M_i . Note that if you do not buy machine M_i on day D_i, then somebody else will buy it and it will not be available later. Needless to say, you cannot buy a machine if ACM has less money than the price of the machine.

If you buy a machine M_i on day D_i, then ACM can operate it starting on day $D_i + 1$. Each day that the machine operates, it produces a profit of G_i dollars for the company.

You may decide to sell a machine to reclaim a part of its purchase price any day after you've bought it. Each machine has a resale price R_i for which it may be resold to the market. You cannot operate a machine on the day that you sell it, but you may sell a machine and use the proceeds to buy a new machine on the same day.

Once the restructuring period ends, ACM will sell any machine that it still owns. Your task is to maximize the amount of money that ACM makes during the restructuring.

Input

The input consists of several test cases. Each test case starts with a line containing three positive integers N, C, and D. N is the number of machines for sale ($N \leq 10^5$), C is the number of dollars with which the company begins the restructuring ($C \leq 10^9$), and D is the number of days that the restructuring lasts ($D \leq 10^9$).

Each of the next N lines describes a single machine for sale. Each line contains four integers D_i, P_i, R_i and G_i, denoting (respectively) the day on which the machine is for sale, the dollar price for which it may be bought, the dollar price for which it may be resold and the daily profit generated by operating the machine. These numbers satisfy $1 \leq D_i \leq D, 1 \leq R_i < P_i \leq 10^9$ and $1 \leq G_i \leq 10^9$.

The last test case is followed by a line containing three zeros.

Output

For each test case, display its case number followed by the largest number of dollars that ACM can have at the end of day $D + 1$.

Follow the format of the sample output.

Sample Input

```
6 10 20
6 12 1 3
1 9 1 2
3 2 1 2
8 20 5 4
4 11 7 4
```

```
2 10 9 1
0 0 0
```

Sample Output

```
Case 1: 44
```