Problem E
Foreign Exchange

Input: standard input
Output: standard output
Time Limit: 1 second

Your non-profit organization (**iCORE** - **i**nternational **C**onfederation of **R**evolver **E**nthusiasts) coordinates a very successful foreign student exchange program. Over the last few years, demand has sky-rocketed and now you need assistance with your task.

The program your organization runs works as follows: All candidates are asked for their original location and the location they would like to go to. The program works out only if every student has a suitable exchange partner. In other words, if a student wants to go from A to B, there must be another student who wants to go from B to A. This was an easy task when there were only about 50 candidates, however now there are up to **500000** candidates!

Input

The input file contains multiple cases. Each test case will consist of a line containing **n** - the number of candidates **(1≤n≤500000)**, followed by **n** lines representing the exchange information for each candidate. Each of these lines will contain **2** integers, separated by a single space, representing the candidate's original location and the candidate's target location respectively. Locations will be represented by nonnegative integer numbers. You may assume that no candidate will have his or her original location being the same as his or her target location as this would fall into the domestic exchange program. The input is terminated by a case where **n = 0**; this case should not be processed.

Output

For each test case, print **"YES"** on a single line if there is a way for the exchange program to work out, otherwise print **"NO"**.

Sample Input	Output for Sample Input
10 1 2 2 1 3 4 4 3 100 200 200 100 57 2 2 57 1 2 2 1 10 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18	YES NO

19 20 0	

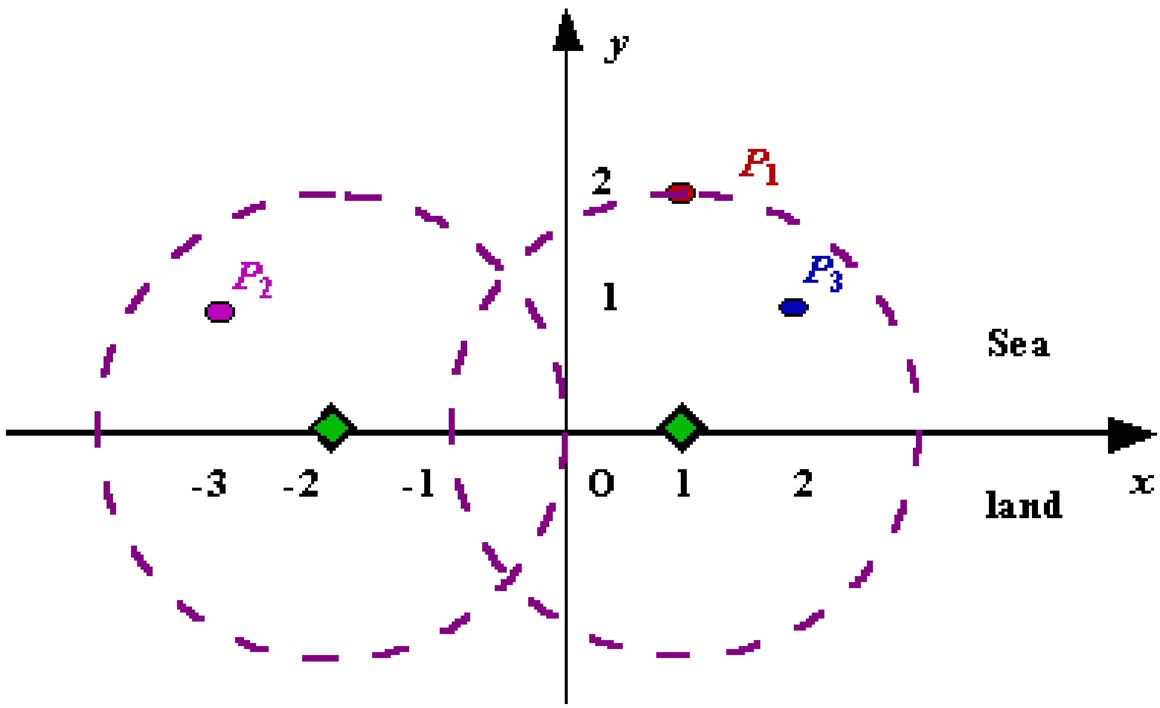
Problem setter: Gilbert Lee, University of Alberta, Canada

2519 - Radar Installation

Asia - Beijing - 2002/2003

Assume the coasting is an infinite straight line. Land is in one side of coasting, sea in the other. Each small island is a point locating in the sea side. And any radar installation, locating on the coasting, can only cover *d* distance, so an island in the sea can be covered by a radius installation, if the distance between them is at most *d* .

We use Cartesian coordinate system, defining the coasting is the *x* -axis. The sea side is above *x* -axis, and the land side below. Given the position of each island in the sea, and given the distance of the coverage of the radar installation, your task is to write a program to find the minimal number of radar installations to cover all the islands. Note that the position of an island is represented by its *x* -*y* coordinates.



Input

The input consists of several test cases. The first line of each case contains two integers *n* ($1 \leq n \leq 1000$) and *d* , where *n* is the number of islands in the sea and *d* is the distance of coverage of the radar installation. This is followed by *n* lines each containing two integers representing the coordinate of the position of each island. Then a blank line follows to separate the cases.

The input is terminated by a line containing pair of zeros.

Output

For each test case output one line consisting of the test case number followed by the minimal number of radar installations needed. \``-1`' installation means no solution for that case.

Sample Input

```
3 2
1 2
-3 1
2 1

1 2
0 2

0 0
```

Sample Output

```
Case 1: 2
Case 2: 1
```

Beijing 2002-2003

	3171 - Oreon **Asia - Manila - 2006/2007**

In the 25th century, civilization is struck by a series of calamities that eventually led mankind to build walled cities interconnected by tunnel bridges to facilitate transportation. Each walled city possesses a unique ore required to build and repair all infrastructure including the tunnels. This material which when combined with other ores from all others cities form an almost indestructible material called ``oreon".

Outside the walled cities are uncivilized barbarians armed with antiquated but destructive weaponry which can effectively shoot down any air transport, but only damage and not penetrate tunnel bridges. Thus each city is interconnected to more than one city in order to have access redundancy in case one of its interconnecting tunnels is damaged.

If a tunnel is damaged, it becomes impassable and would require a substantial amount of ``oreon" to repair the damage. When a single city is made isolated, meaning all of its interconnections are damaged, ``oreon" cannot be manufactured which may lead to the eventual destruction of the wall fortifying the city. You, being the head of the homeland defense unit, are tasked to ensure that all cities remain accessible even by at least a single interconnecting tunnel at all times. Faced with only a limited manpower in the defense unit, you have to determine which tunnel to protect using the least number of people and ensure that no city will be isolated.

Figure 2 shows a map of the walled cities, their interconnecting tunnels and the number of security personnel.

=6in

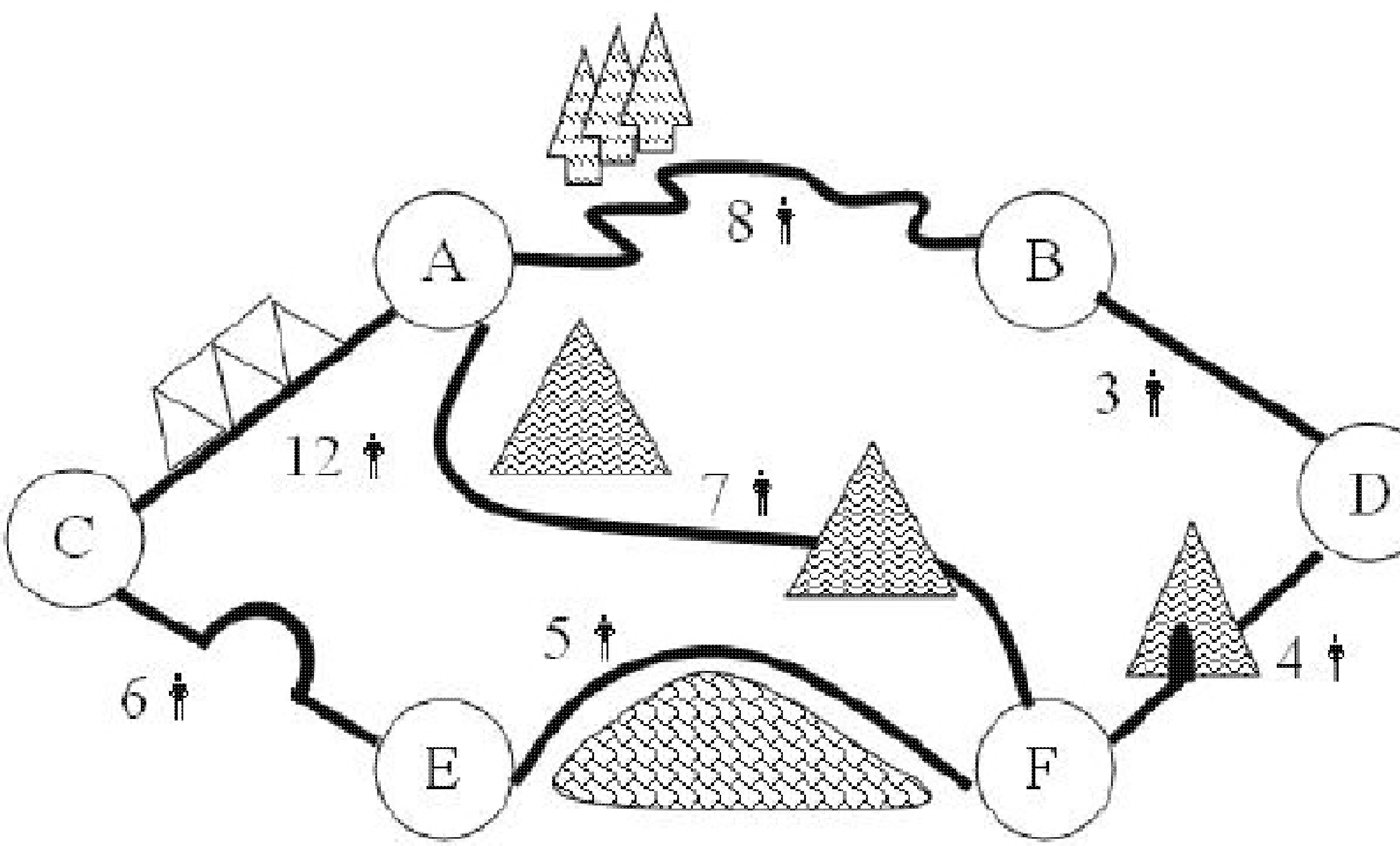


Figure 2: Map of six cities and its interconnecting tunnels

Input

The input will contain several test cases. The first line will indicate the number of test cases. Each test case begins with a number representing the number of walled cities. Cities are labeled alphabetically using the letters in the English alphabet. The subsequent lines contain the number of security personnel needed to protect the tunnel connecting each city to all other cities. A value of zero implies no security personnel needed since no tunnel exists. You are to output which tunnel should be protected and how many personnel are needed for each tunnel.

Output

The output shows the tunnel connection which is named after the cities that it connects (in alphabetical order) and the number of personel needed to protect the tunnel. Order the records in increasing order of presonal. In case two tunnels have the same number of personel, write them in lexicographycal order.

Sample Input

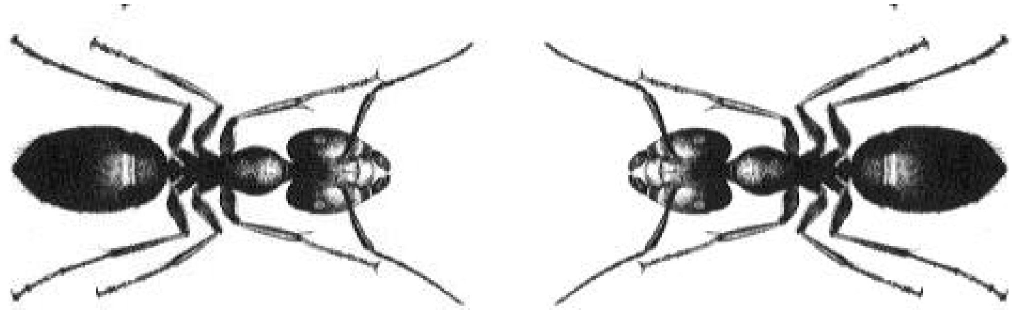
```
1
6
0, 8, 12, 0, 0, 7
8, 0, 0, 3, 0, 0
12, 0, 0, 0, 6, 0
0, 3, 0, 0, 0, 4
0, 0, 6, 0, 0, 5
7, 0, 0, 4, 5, 0
```

Sample Output

```
Case 1:
B-D 3
D-F 4
E-F 5
C-E 6
A-F 7
```

Manila 2006-2007

Problem B: Ants



An army of ants walk on a horizontal pole of length *l* cm, each with a constant speed of 1 cm/s. When a walking ant reaches an end of the pole, it immediatelly falls off it. When two ants meet they turn back and start walking in opposite directions. We know the original positions of ants on the pole, unfortunately, we do not know the directions in which the ants are walking. Your task is to compute the earliest and the latest possible times needed for all ants to fall off the pole.

The first line of input contains one integer giving the number of cases that follow. The data for each case start with two integer numbers: the length of the pole (in cm) and *n*, the number of ants residing on the pole. These two numbers are followed by *n* integers giving the position of each ant on the pole as the distance measured from the left end of the pole, in no particular order. All input integers are not bigger than 1000000 and they are separated by whitespace.

For each case of input, output two numbers separated by a single space. The first number is the earliest possible time when all ants fall off the pole (if the directions of their walks are chosen appropriately) and the second number is the latest possible such time.

Sample input

```
2
10 3
2 6 7
214 7
11 12 7 13 176 23 191
```

Output for sample input

```
4 8
38 207
```

Piotr Rudnicki

Problem ?

Dijkstra, Dijkstra.

Time Limit: 10 seconds

Dexter: *"You don't understand. I can't walk... they've tied my shoelaces together."*
Topper Harley: *"A knot. Bastards!"*

`Jim Abrahams and Pat Proft, "Hot Shots! Part Deux."`

You are a political prisoner in jail. Things are looking grim, but fortunately, your jailmate has come up with an escape plan. He has found a way for both of you to get out of the cell and run through the city to the train station, where you will leave the country. Your friend will escape first and run along the streets of the city to the train station. He will then call you from there on your cellphone (which somebody smuggled in to you inside a cake), and you will start to run to the same train station. When you meet your friend there, you will both board a train and be on your way to freedom.

Your friend will be running along the streets during the day, wearing his jail clothes, so people will notice. This is why you can not follow any of the same streets that your friend follows - the authorities may be waiting for you there. You have to pick a completely different path (although you may run across the same intersections as your friend).

What is the earliest time at which you and your friend can board a train?

Problem, in short

Given a weighed, undirected graph, find the shortest path from **S** to **T** and back without using the same edge twice.

Input

The input will contain several test cases. Each test case will begin with an integer **n** (2<=**n**<=100) - the number of nodes (intersections). The jail is at node number 1, and the train station is at node number **n**. The next line will contain an integer **m** - the number of streets. The next **m** lines will describe the **m** streets. Each line will contain 3 integers - the two nodes connected by the street and the time it takes to run the length of the street (in seconds). No street will be longer than 1000 or shorter than 1. Each street will connect two different nodes. No pair of nodes will be directly connected by more than one street. The last test case will be followed by a line containing zero.

Output

For each test case, output a single integer on a line by itself - the number of seconds you and your friend need between the time he leaves the jail cell and the time both of you board the train. (Assume that you do not need to wait for the train - they leave every second.) If there is no solution, print "Back to jail".

Sample Input	Sample Output

2 1 1 2 999 3 3 1 3 10 2 1 20 3 2 50 9 12 1 2 10 1 3 10 1 4 10 2 5 10 3 5 10 4 5 10 5 7 10 6 7 10 7 8 10 6 9 10 7 9 10 8 9 10 0	Back to jail 80 Back to jail

---

Problemsetter: Igor Naverniouk

D

Next Generation Contest 3

Dynamic Frog

Time Limit: 2 seconds

With the increased use of pesticides, the local streams and rivers have become so contaminated that it has become almost impossible for the aquatic animals to survive.

Frog Fred is on the left bank of such a river. **N** rocks are arranged in a straight line from the left bank to the right bank. The distance between the left and the right bank is **D** meters. There are rocks of two sizes. The bigger ones can withstand any weight but the smaller ones start to drown as soon as any mass is placed on it. Fred has to go to the right bank where he has to collect a gift and return to the left bank where his home is situated.

He can land on every small rock at most one time, but can use the bigger ones as many times as he likes. He can never touch the polluted water as it is extremely contaminated.

Can you plan the itinerary so that the maximum distance of a single leap is minimized?

Input

The first line of input is an integer **`T(T<100)`** that indicates the number of test cases. Each case starts with a line containing two integers **`N(0≤N≤100)`** and **`D(1≤D≤1000000000)`**. The next line gives the description of the **`N`** stones. Each stone is defined by **`S-M`**. **`S`** indicates the type Big(**`B`**) or Small(**`S`**) and **`M`**(**`0<M<D)`** determines the distance of that stone from the left bank. The stones will be given in increasing order of **`M`**.

Output

For every case, output the case number followed by the minimized maximum leap.

Sample Input	Output for Sample Input
3 1 10 B-5 1 10 S-5 2 10 B-3 S-6	Case 1: 5 Case 2: 10 Case 3: 7

Problem Setter: Sohel Hafiz
Special Thanks: Jane Alam Jan

311 Packets

A factory produces products packed in square packets of the same height h and of the sizes 1×1, 2×2, 3×3, 4×4, 5×5, 6×6. These products are always delivered to customers in the square parcels of the same height h as the products have and of the size 6×6. Because of the expenses it is the interest of the factory as well as of the customer to minimize the number of parcels necessary to deliver the ordered products from the factory to the customer. A good program solving the problem of finding the minimal number of parcels necessary to deliver the given products according to an order would save a lot of money. You are asked to make such a program.

Input

The input file consists of several lines specifying orders. Each line specifies one order. Orders are described by six integers separated by one space representing successively the number of packets of individual size from the smallest size 1×1 to the biggest size 6×6. The end of the input file is indicated by the line containing six zeros.

Output

The output file contains one line for each line in the input file. This line contains the minimal number of parcels into which the order from the corresponding line of the input file can be packed. There is no line in the output file corresponding to the last "null" line of the input file.

Sample Input

```
0 0 4 0 0 1
7 5 1 0 0 0
0 0 0 0 0 0
```

Sample Output

```
2
1
```