A

New labyrinth attraction is open in New Lostland amusement park. The labyrinth consists of n rooms connected by m passages. Each passage is colored into some color c_i. Visitors of the labyrinth are dropped from the helicopter to the room number 1 and their goal is to get to the labyrinth exit located in the room number n.

Labyrinth owners are planning to run a contest tomorrow. Several runners will be dropped to the room number 1. They will run to the room number n writing down colors of passages as they run through them. The contestant with the shortest sequence of colors is the winner of the contest. If there are several contestants with the same sequence length, the one with the *ideal path* is the winner. The path is the ideal path if its color sequence is the lexicographically smallest among shortest paths.

Andrew is preparing for the contest. He took a helicopter tour above New Lostland and made a picture of the labyrinth. Your task is to help him find the ideal path from the room number 1 to the room number n that would allow him to win the contest.

Note: A sequence $(a_1, a_2, \ldots, a_k)$ is lexicographically smaller than a sequence $(b_1, b_2, \ldots, b_k)$ if there exists i such that $a_i < b_i$, and $a_j = b_j$ for all $j < i$.

Input

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

The first line of the input file contains integers n and m — the number of rooms and passages, respectively ($2 \le n \le 100000, 1 \le m \le 200000$). The following m lines describe passages, each passage is described with three integer numbers: a_i, b_i, and c_i — the numbers of rooms it connects and its color ($1 \le a_i, b_i \le n, 1 \le c_i \le 10^9$). Each passage can be passed in either direction. Two rooms can be connected with more than one passage, there can be a passage from a room to itself. It is guaranteed that it is possible to reach the room number n from the room number 1.

Output

For each test case, the output must follow the description below.

The first line of the output file must contain k — the length of the shortest path from the room number 1 to the room number n. The second line must contain k numbers — the colors of passages in the order they must be passed in the ideal path.

Sample Input

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

Sample Output

```
2
1 3
```

B

Ciel the Commander

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Now Fox Ciel becomes a commander of Tree Land. Tree Land, like its name said, has n cities connected by $n - 1$ undirected roads, and for any two cities there always exists a path between them.

Fox Ciel needs to assign an officer to each city. Each officer has a rank — a letter from 'A' to 'Z'. So there will be 26 different ranks, and 'A' is the topmost, so 'Z' is the bottommost.

There are enough officers of each rank. But there is a special rule must obey: if x and y are two distinct cities and their officers have the same rank, then on the simple path between x and y there must be a city z that has an officer with higher rank. The rule guarantee that a communications between same rank officers will be monitored by higher rank officer.

Help Ciel to make a valid plan, and if it's impossible, output "Impossible!".

Input

The first line contains an integer n ($2 \le n \le 10^5$) — the number of cities in Tree Land.

Each of the following $n - 1$ lines contains two integers a and b ($1 \le a, b \le n, a \ne b$) — they mean that there will be an undirected road between a and b. Consider all the cities are numbered from 1 to n.

It guaranteed that the given graph will be a tree.

Output

If there is a valid plane, output n space-separated characters in a line — i-th character is the rank of officer in the city with number i.

Otherwise output "Impossible!".

Sample test(s)

input

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

output

```
A B B B
```

input

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

output

```
D C B A D C B D C D
```

Note

In the first example, for any two officers of rank 'B', an officer with rank 'A' will be on the path between them. So it is a valid solution.

C

Jollo is a simple card game which the children from Logonia love to play. It is played between two players with a normal deck of 52 cards. In the game, cards are ordered according to their rank and suit, forming a sequence of 52 distinct values.

The game is composed of three rounds, played in a best-of-three series (a player must win two rounds to win the game). At the beginning of the game the deck is shuffled and each player is given a hand of three cards. In each round the players show one card to each other and the player with the highest card wins the round. The cards shown in a round are discarded (i.e., they cannot be shown again).

The King's son loves to play the game. But he is not very smart, losing frequently to his little sister. And when he loses, he cries so loud no one can stand it. The servant who deals the cards to the Prince and his sister is afraid he will be sent to prison if the Prince continues to lose. The servant is allowed to see every card he deals, and after dealing five cards (three to the Princess and two to the Prince) he wants to know which is the lowest card he should deal to the Prince so that there is no chance he will lose the game, no matter how badly he plays.

Input

Each test case is given in a single line that contains five distinct integers A, B, C, X and Y, describing the cards dealt to the players. The first three cards are given to the Princess ($1 \leq A, B, C \leq 52$) and the last two cards are given to the Prince ($1 \leq X, Y \leq 52$). The last test case is followed by a line containing five zeros.

Output

For each test case output a single line. If there exists a card that will make the Prince win the game no matter how badly he plays, you must print the lowest such a card. Otherwise, print '`-1`'.

Sample Input

```
28 51 29 50 52
50 26 19 10 27
10 20 30 24 26
46 48 49 47 50
0 0 0 0 0
```

Sample Output

```
30
-1
21
51
```

D

A war is being lead between two countries, A and B. As a loyal citizen of C, you decide to help your countrys espionage by attending the peace-talks taking place these days (incognito, of course). There are n people at the talks (not including you), but you do not know which person belongs to which country. You can see people talking to each other, and through observing their behaviour during their occasional one-to-one conversations, you can guess if they are friends or enemies. In fact what your country would need to know is whether certain pairs of people are from the same country, or they are enemies. You may receive such questions from Cs government even during the peace-talks, and you have to give replies on the basis of your observations so far. Fortunately nobody talks to you, as nobody pays attention to your humble appearance.

Now, more formally, consider a black box with the following operations:

setFriends(x,y)	shows that x and y are from the same country
setEnemies(x,y)	shows that x and y are from different countries
areFriends(x,y)	returns true if you are sure that x and y are friends
areEnemies(x,y)	returns true if you are sure that x and y are enemies

The first two operations should signal an error if they contradict with your former knowledge. The two relations 'friends' (denoted by $\sim$) and 'enemies' (denoted by $*$) have the following properties:

$\sim$ is an equivalence relation, i.e.

1. If $x \sim y$ and $y \sim z$ then $x \sim z$ (The friends of my friends are my friends as well.
2. If $x \sim y$ then $y \sim x$ (Friendship is mutual.)
3. $x \sim x$ (Everyone is a friend of himself.)

$*$ is symmetric and irreflexive

1. If $x * y$ then $y * x$ (Hatred is mutual.)
2. Not $x * x$ (Nobody is an enemy of himself.)

Also

1. If $x * y$ and $y * z$ then $x \sim z$ (A common enemy makes two people friends.
2. If $x \sim y$ and $y * z$ then $x * z$ (An enemy of a friend is an enemy.

Operations setFriends(x,y) and setEnemies(x,y) must preserve these properties.

Input

The first line contains a single integer, n, the number of people.

Each of the following lines contains a triple of integers, cxy, where c is the code of the operation:

$c = 1$,	setFriends
$c = 2$,	setEnemies
$c = 3$,	areFriends
$c = 4$,	areEnemies

and x and y are its parameters, which are integers in the range $[0, n)$, identifying two (different) people. The last line contains '`0 0 0`'.

All integers in the input file are separated by at least one space or line break. The only constraint is $n < 10000$, the number of operations is unconstrained.

Output

For every "areFriends" and "areEnemies" operation write '`0`' (meaning no) or '`1`' (meaning yes) to the output. Also for every "setFriends" or "setEnemies" operation which contradicts with previous knowledge, output a '`-1`' to the output; note that such an operation should produce no other effect and execution should continue. A successful "setFriends" or "setEnemies" gives no output.

All integers in the output file must be separated by one line break.

Sample Input

```
10
1 0 1
1 1 2
2 0 5
3 0 2
3 8 9
4 1 5
4 1 2
4 8 9
1 8 9
1 5 2
3 5 2
0 0 0
```

Sample Output

```
1
0
1
0
0
-1
0
```

E

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 ($0 \leq N \leq 100$) and D ($1 \leq D \leq 1000000000$). 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.

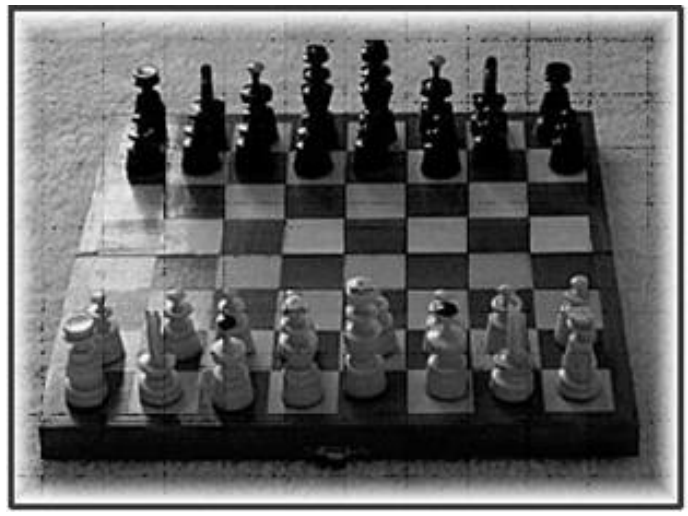
Sample Input

```
3
1 10
B-5
1 10
S-5
2 10
B-3 S-6
```

Sample Output

```
Case 1: 5
Case 2: 10
Case 3: 7
```

F



Chess is a two-player board game believed to have been played in India as early as the sixth century. However, in this problem we will not discuss about chess, rather we will talk about a modified form of the classic n- queens problem. I know you are familiar with plotting n-queens on a chess board with the help of a classic backtracking algorithm. If you write that algorithm now you will find that there are 92 ways of plotting 8 queens in an 8×8 board provided no queens attack each other.

In this problem we will talk about injured queens who can move only like a king in horizontal and diagonal direction from current position but can reach any row from current position like a normal chess queen. You will have to find the number of possible arrangements with such injured queens in a particular $(n \times n)$ board (with some additional constraints), such that no two queens attack each other.

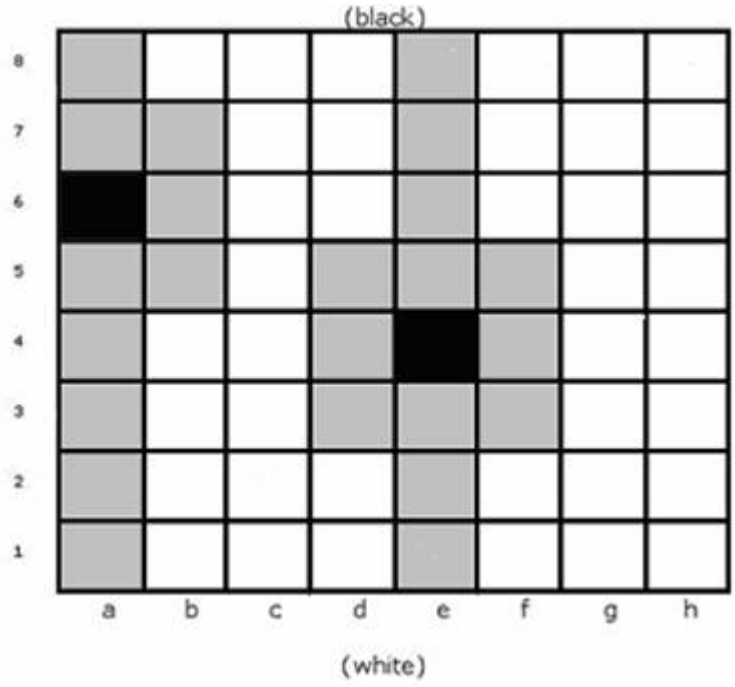


Fig: Injured Queen at a6 can reach the adjacent grey squares. Queen at e4 can reach adjacent grey squares. The injured queen positions are black and the reachable places are grey.

Input

Input file contains several lines of input. Each line expresses a certain board status. The length of these status string is the board dimension n $(0 < n \leq 15)$. The first character of the string denotes the status of first column, the second character of the string denotes the status of the second column and so on. So if the first character of the status string is 2, it means that we are looking for arrangements (no two injured queen attack each other) which has injured queen in column a, row 2. The possible numbers for rows are $1, 2, 3, \ldots, D, E, F$ which indicates row 1, 2, 3... 13, 14, 15. If any column contains '`?`' it means that in that column the injured queen can be in any row. So a status string '`1?4??3` means that you are asked to find out total number of possible arrangements in a (6×6) chessboard which has three of its six injured queens at $a1$, $c4$ and $f3$. Also note that there will be no invalid inputs. For example '`1?51`' is an invalid input because a (4×4) chessboard does not have a fifth row.

Output

For each line of input produce one line of output. This line should contain an integer which indicates the total number of possible arrangements of the corresponding input status string.

Sample Input

```
??????
????????????????
???8?????
43?????
```

Sample Output

```
2642
22696209911206174
2098208
0
```

G

You are given two non-empty strings S and T of equal lengths. S contains the characters '0', '1' and '?', whereas T contains '0' and '1' only. Your task is to convert S into T in minimum number of moves. In each move, you can

1. change a '0' in S to '1'
2. change a '?' in S to '0' or '1'
3. swap any two characters in S

As an example, suppose S = "01??00" and T = "001010". We can transform S into T in 3 moves:

- Initially S = "01??00"
- – Move 1: change $S[2]$ to '1'. S becomes "011?00"
- – Move 2: change $S[3]$ to '0'. S becomes "011000"
- – Move 3: swap $S[1]$ with $S[4]$. S becomes "001010"
- S is now equal to T

Input

The first line of input is an integer C ($C \leq 200$) that indicates the number of test cases. Each case consists of two lines. The first line is the string S consisting of '0', '1' and '?'. The second line is the string T consisting of '0' and '1'. The lengths of the strings won't be larger than 100.

Output

For each case, output the case number first followed by the minimum number of moves required to convert S into T. If the transition is impossible,output '-1' instead.

Sample Input

```
3
01??00
001010
01
10
110001
000000
```

Sample Output

```
Case 1: 3
Case 2: 1
Case 3: -1
```

H

Cow Program

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Farmer John has just given the cows a program to play with! The program contains two integer variables, x and y, and performs the following operations on a sequence $a_1, a_2, ..., a_n$ of positive integers:

1. Initially, $x = 1$ and $y = 0$. If, after any step, $x \leq 0$ or $x > n$, the program immediately terminates.
2. The program increases both x and y by a value equal to a_x simultaneously.
3. The program now increases y by a_x while decreasing x by a_x.
4. The program executes steps 2 and 3 (first step 2, then step 3) repeatedly until it terminates (it may never terminate). So, the sequence of executed steps may start with: step 2, step 3, step 2, step 3, step 2 and so on.

The cows are not very good at arithmetic though, and they want to see how the program works. Please help them!

You are given the sequence $a_2, a_3, ..., a_n$. Suppose for each i $(1 \leq i \leq n - 1)$ we run the program on the sequence $i, a_2, a_3, ..., a_n$. For each such run output the final value of y if the program terminates or `-1` if it does not terminate.

Input

The first line contains a single integer, n $(2 \leq n \leq 2 \cdot 10^5)$. The next line contains $n - 1$ space separated integers, $a_2, a_3, ..., a_n$ $(1 \leq a_i \leq 10^9)$.

Output

Output $n - 1$ lines. On the i-th line, print the requested value when the program is run on the sequence $i, a_2, a_3, ...a_n$.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
4 2 4 1

output
3 6 8

input
3 1 2

output
-1 -1

Note

In the first sample

1. For $i = 1$, x becomes $1 \rightarrow 2 \rightarrow 0$ and y becomes $1 + 2 = 3$.
2. For $i = 2$, x becomes $1 \rightarrow 3 \rightarrow -1$ and y becomes $2 + 4 = 6$.
3. For $i = 3$, x becomes $1 \rightarrow 4 \rightarrow 3 \rightarrow 7$ and y becomes $3 + 1 + 4 = 8$.

The kingdom of ByteLand is in trouble. The enemies are going to attack ByteLand. The enemies know that ByteLand has exactly N cities and exactly M bidirectional roads and it is possible to go from any city to every other city directly or via other cities. They also know that any pair of cities can be directly connected by at most one road. But they do not have any information about which road connects which two cities. They are planning to destroy all critical roads of ByteLand. A road is critical if after destroying that road only at least one pair of cities become disconnected. They are very optimistic so they expect to destroy maximum number of critical roads. What is the maximum number of critical roads that can be present in ByteLand according only to the information the enemies have about ByteLand?

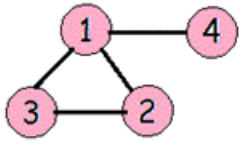
Input

The first line of input contains T $(1 \leq T \leq 50)$ which is the number of tests cases. Each case contains two integers N which is the number of cities and M which is the number of roads $\left(2 \leq N \leq 10^5 and 1 \leq M \leq \frac{N(N}{}\right.$

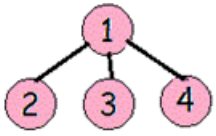
Output

For each test case output one integer the maximum number of critical roads that could be present in ByteLand.

Explanation of Sample Cases



For sample 1, one road network with maximum possible 1 critical road(1-4) is shown. There can be more valid networks, but none of them has more than 1 critical road.



For sample 2, one road network where every road is critical is shown.

Sample Input

```
2
4 4
4 3
```

Sample Output

```
1
3
```