Problem A:

Sightseeing

As a computer science student you are of course very outdoorsie, so you decided to go hiking. For your vacation this year, you located an island full of nice places to visit. You already identified a number of very promising tracks, but are still left with some problems. The number of choices is so overwhelming, that you had to select only a "small" subset of at most 100 000 sights. And if that is not enough, you are very picky about the order in which you want to visit the sights. So you have already decided on an order in which you want to visit the preselected tracks. The problem you are left with is to decide in which direction to travel along each single track, and whether you may have to reduce your choice of tracks even further. After identifying the travel time between the endpoints of different tracks, you decide to write a program to figure out if you can make all your trips within the time you have planned for your vacation. Since you also do not want to waste any precious time, you only care about an optimal solution to your problem. Furthermore, the tracks can get pretty challenging. Thats why you do not want to hike along a track more than once.

Input

The first line of the input gives the number of test cases C ($0 < C \leq 100$). The first line of each such test case holds two integers N, T the number of tracks of the current hiker ($1 \leq N \leq 100\,000$) and the maximal time spent hiking throughout the vacation ($0 \leq T \leq 1\,000\,000$) Each of the following N lines holds five integers c_p, c_{bb}, c_{be}, c_{eb} and c_{ee} that describe a track (in order of importance). c_p gives the length of the track in minutes. c_{xy} gives the travel time of the official **b**egin or **e**nd of a track to the **b**eginning or **e**nd of the next most important track, where x and y are either b or e. All values given are non-negative integers not greater than 1 000 000. Since you have to get back to your car, the list is circular. Furthermore, we will ignore the time it takes you to get to the start of your trip with your car.

Output

For each test case print one line. The output should contain a list of "`F`" and "`B`" for every track (in order) indicating whether you have to hike the track in forward direction or backward direction. If you cannot make the full trip within the planned time T, you should print "`IMPOSSIBLE`" to indicate that these trips are just too much hiking.

Sample Input

```
3
2 100
4 7 8 2 3
1 4 6 1 2
2 20
4 2 3 7 8
1 1 2 4 6
3 5
1 2 2 2 1
1 1 2 2 2
1 2 2 1 2
```

Sample Output

```
FF
BB
IMPOSSIBLE
```