J Tour de France

Your company has gotten a grand and prestigious contract: you are to write a program for the organisation of the Tour de France! With the large amount of money that the organisation of the Tour has, your marketing department became really fanatic, trying everything to get some work for your company that they could sell for way too much money. You, the company's best programmer, has been assigned to this job.

When you read the assignment however, both respect for the marketing department and horror at the assignment strikes you. Your marketing department has somehow managed to convince the organisation of the Tour de France that they absolutely need a program that will compute the route of the Tour through the selected French cities for them. In particular, the organisation of the Tour now believes that the best such tour would be the *shortest* one. Additionally, they have decided that the Tour should really be a tour as understood in computer science, in that the tour ends at the starting point, and that each stage (= a day of cycling from one city to the next) starts at the finish of the previous one.

You quickly look up how many stages they plan on having, see that it's a huge number (they lengthened the Tour) and you fall to your knees, burying your face in your hands in despair. Given your Computer Science background, you know that this is precisely the famous Travelling Salesman problem, famous for requiring exponential time to solve (given $\mathbb{P} \neq \mathbb{NP}$), so there is almost surely no algorithm fast enough for this many stages.

You hurry to your boss, a huge man with an equally huge temper. You start explaining how this new assignment is exactly a famous hard problem, how there are simply too many tours through these cities and that there's no way you can find the shortest one before the end of the century (skipping the dynamic programming algorithm that does better than trying all tours, because it's too complicated to explain quickly and would still not solve the problem in time). Your boss is starting to get annoyed by you, clearly convinced you must have some other motive for saying all this, such as laziness. When you say that basically the entire Computer Science community agrees with this assessment and that finding a better algorithm is a famous open problem for decades, he seems to feel that you might not be lying.

He says: "Don't worry about it. They promised us something in later negotiations: as it's hard to set out so many possible routes between cities, they decided that every city can only choose two cities as potential destinations. Furthermore, as assistance is needed from the destination city to set up a route for a stage, every destination city need only cooperate with at most two cities that want to set out a route to them. That should take care of your 'there's too many possible tours' problem. Now scram and don't bother me again."

He slams the door in your face. You feel a bit better now: apparently, the problem is now one on a directional graph, where every vertex has at most two incoming and outgoing edges… A quick calculation reveals that you still can't try all possible tours or the dynamic programming algorithm, but maybe something is known on this restriction of the Travelling Salesman problem? A quick search of the internet yields no results. The longer, more frantic search that follows also yields no results, and after cursing your luck and life in general for a minute, you resign yourself to your fate: finding some way of solving the problem fast enough by yourself in an attempt to avoid incurring the wrath of your boss.

Input

The input starts with a line containing an integer T, the number of test cases. Then for each test case:

- One line with two space-separated integers: N, the number of cities ($3 \leq N \leq 36$) and M, the number of edges ($N \leq M \leq 2 \times N$).
- M lines, each with three space-separated integers i, j, d, indicating that there is a directed edge from city i to city j with length d. ($0 \leq i, j < N$, $i \neq j$ and $1 \leq d \leq 10{,}000$). For any pair of cities a, b, there will be at most one line a, b, x (i.e. no duplicate edges), but there can be a line b, a, y (the backward edge).

Output

For each test case, output one line with one integer indicating the length of the shortest tour through all N cities. Note: for every test case, there will always be at least one tour possible.

Sample input and output

Input	Output
2	9
3 5	5
0 1 2	
0 2 1	
1 0 1	
1 2 3	
2 0 4	
5 10	
0 2 1	
0 4 5	
1 0 1	
1 2 2	
2 4 1	
2 3 3	
3 1 1	
3 0 4	
4 3 1	
4 1 6	