Robert likes his job, but, as inevitably happens to everyone at some point, he is late for work today. Thus, he wants to get from his home to the office in the fastest way. For this purpose, he has at his disposal his car to which he has attached his bike. For the purpose of the problem we will assume that the town in which Robert resides is a connected graph with ***М*** undirected edges (two-way streets), whose vertices (crossroads) are labeled with the numbers from 1 to ***N***. Robert‘s home is in vertex 1 and the office is in vertex *N*. In this manner Robert is going to set off from vertex 1 with his car and if needs, he will stop the car and continue with his bike to vertex *N,* so once he’s off the car he can‘t get back on it in the future and has to continue his route only by bike. It is allowed to not use his car or bike at all, and the time to change from car to bike is 0. For every edge between vertices *ui* and *vi* (1 ≤ i ≤ *M*), there are two numbers *ai* and *bi* , respectively the time to pass with a car and the time to pass with a bike. For example, on a straight path with little traffic the car would be way faster than the bike, while on a busy road with a lot of turns the bike could be the faster option. We assume that Robert can stop his car on each one of the vertices. Help Robert by writing a program, which takes as input values N, M, and the data for the edges, and finds the shortest path between vertices 1 and N.

**Input**

On the first row of the input file hurry.in is a number 𝑇. Next 𝑇 test cases follow.

On the first row of every test case there are two natural numbers *N* и *М.* On every next row i (1 ≤ i ≤ *M*) there are 4 natural numbers *ui, vi, ai, bi*– respectively the vertices, which the edge connects, the time to pass the edge with the car and the time to pass it with the bike.

**Output**

For every test case, print in the output file hurry.out one natural number – the minimum time needed for Robert to get from vertex 1 to vertex N.

**Restrictions**

The sum of N over all test cases does not exceed 105

**Time limit: 1 sec.**

**Memory limit: 256 MB.**

**Example test**

|  |  |
| --- | --- |
| **Input (hurry.in)** | **Output (hurry.out)** |
| 3  5 6  1 2 2 1  2 3 3 5  3 4 9 3  1 3 6 6  2 5 7 6  3 5 8 1  5 5  1 2 1 1  2 3 2 2  3 5 1 4  4 5 2 5  1 4 3 4  5 7  1 2 6 2  1 3 3 4  1 4 4 5  2 3 5 2  2 4 3 1  2 5 7 4  4 5 3 2 | 6  4  5 |

**Explanation**

In the first test case the optimal path for Robert is 1-2-3-5 with a switch from car to bike on edge 3. In the second test case the optimal path for Robert is 1-2-3-5, when he uses only the car. In the third test case the optimal path is 1-2-4-5, when Robert uses only the bike.