Problem H. Hyperloop

Input file:	`standard input`
Output file:	`standard output`
Time limit:	45 seconds
Memory limit:	**64 megabytes**

The year is 2077. Artificial Intelligence has just solved the last problem of colouring hypergraphs, and the newly unemployed researchers from the Theoretical Computer Science department have been commissioned by an eccentric billionaire Melon Usk to develop software for the Hyperloop complex – a system of super-fast intercity connections.

Due to rising water levels, the only land mass left on Earth is a single island on whose shore there lie n surviving cities, which we will name for simplicity 1, 2, ..., n in the order of occurrence along the shore. Melon owns ferries that run around the island, i.e. between cities i and $i+1$ for $1 \leq i < n$ and between cities n and 1. He also owns Hyperloop connections between certain pairs of *non-neighbouring* cities[2]. The only thing left is to design the software for computing the shortest routes between cities; in the beta version it will only support travel from 1 to n.

The task might seem trivial: after all, the software is supposed to take into account both types of connections, and between the cities 1 and n there is already a *direct* ferry connection, but it doesn't have to be the fastest route! Even worse, there may be several shortest paths. In such situation the system should prefer paths which incorporate the longest possible connections between cities (the longer a single journey leg lasts, the more likely the passengers are to order Melon's coffee at inflated prices during the ride). Formally, in order to compare paths **of the same total length**, one should list the lengths of the connections in them (including repetitions), sort each sequence in non-increasing order, and select the lexicographically largest[3].

Given the description of the ferries around the island and the Hyperloop connections, find the optimal path from city 1 to city n. When comparing paths, both kinds of connections are treated equally. All the connections are bidirectional.

Observe that this task has a low memory limit – 64MB.

Input

The first line of input contains the number of test cases z ($1 \leq z \leq 600$). The descriptions of the test cases follow.

The first line of the test case contains two integers n, m ($3 \leq n \leq 100\,000$, $n \leq m \leq 300\,000$) – the number of cities and the number of connections.

Each of the following m lines contains three integers u_i, v_i, d_i ($1 \leq u_i \neq v_i \leq n$, $1 \leq d_i \leq 50\,000$), meaning in order: cities connected by a given bidirectional connection, and the time needed to travel it.

Among the connections given at the input $(1, 2)$, $(2, 3)$, ..., $(n, 1)$ will appear, corresponding to the ferry connections.

There will be at most one connection between any pair of cities.

The total number of cities in all test cases will not exceed 400 000. The total number of connections in all test cases will not exceed 800 000.

Output

For each test case, on the first line print one integer k, representing the number of cities in the optimal path. On the second line, output k different integers – the identifiers of subsequent cities on the path.

[2]Hyperloop connections use underground tunnels. The tunnels that could cross were simply laid at different depths.

[3]The string $a_1, ..., a_p$ is lexicographically smaller than the string $b_1, ..., b_q$ if it is its prefix or there exists $i \leq \min(p, q)$ such that $a_j = b_j$ for $j < i$ and $a_i < b_i$. In this task, the sequences to be compared always have the same total sum, so neither of them will be a prefix of the other.

The chosen path should start in the city 1, end in the city n, and be optimal as defined in the problem statement.

If there are multiple optimal paths, you can output any of them.

Example

standard input	standard output
2 4 6 1 2 1 1 3 2 2 3 1 2 4 2 3 4 1 1 4 4 6 11 1 2 9 2 3 12 3 4 3 4 5 5 5 6 10 6 1 22 2 4 9 3 6 1 4 6 5 2 5 2 3 5 8	3 1 2 4 5 1 2 5 3 6

Notes

In the first sample test, the minimum distance from city 1 to city 4 is 3, and it is achieved by three paths: $1 \to 2 \to 4$, $1 \to 3 \to 4$ and $1 \to 2 \to 3 \to 4$. As defined in the problem statement, the first two paths are better than the third, because lexicographically $(2, 1) > (1, 1, 1)$. Either of the first two paths is a correct answer. The direct path $1 \to 4$ is not considered as its total length is 4.

In the second example test, the paths with the minimum distance are $1 \to 2 \to 5 \to 3 \to 6$ and $1 \to 2 \to 5 \to 4 \to 3 \to 6$. The first one is optimal, as comparing their sorted lengths we obtain $(9, 8, 2, 1) > (9, 5, 3, 2, 1)$.