Problem B. Travelling

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

There are n cities, numbered from 1 to n. There are also $n-1$ bidirectional roads connecting these cities, so that for every two cities x and y, there exists a path from x to y. The i-th road connects cities u_i and v_i, and has a cost c_i. Every time you pass through road i, you spend c_i coins.

You are now at city 1 and want to travel to visit all cities. Every second, if you are at city x, you can choose a city y that is directly connected with city x by a road, move to city y, and pay the cost of the respective road. Each road can be passed **at most twice**. When you have visited all cities and are at city 1, you can finish your travel. The cost of the travel is the total cost of all the moves you made.

There is also an integer k. During your travel, you can use a free coupon once, making your following k moves have no cost.

For every k from 0 to $2n-2$, you should calculate the minimum cost of a travel that visits all cities and uses the coupon optimally.

Input

The first line contains a single integer t ($1 \le t \le 400$), the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 2000$), denoting the number of cities.

The next $n-1$ lines describe the roads. The i-th of these lines contains three integers, u_i, v_i, and c_i ($1 \le u_i, v_i \le n$, $0 \le c_i \le 10^9$), denoting the cities connected by the i-th road and the cost of that road.

There is a path between any pair of cities. The sum of n does not exceed 2000.

Output

For each test case, output one line containing $2n-1$ integers: the answers for k from 0 to $2n-2$.

Example

standard input	*standard output*
2 5 1 2 2 2 3 3 2 4 1 4 5 3 7 1 2 1 1 3 1 1 4 4 3 5 5 3 6 1 6 7 4	18 15 12 10 8 5 3 2 0 32 27 22 21 17 13 12 8 4 3 2 1 0

Note

In the first test case, we can always travel along the path $1, 2, 3, 2, 4, 5, 4, 2, 1$, with the cost sequence $2, 3, 3, 1, 3, 3, 1, 2$. The cost-free segments for each k are shown below.

k	cost-free segment	answer
0	[]	18
1	[3]	15
2	[3, 3]	12
3	[2, 3, 3]	10
4	[3, 3, 1, 3]	8
5	[3, 3, 1, 3, 3]	5
6	[2, 3, 3, 1, 3, 3]	3
7	[2, 3, 3, 1, 3, 3, 1]	2
8	[2, 3, 3, 1, 3, 3, 1, 2]	0