B. Replicating Processes

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A Large Software Company develops its own social network. Analysts have found that during the holidays, major sporting events and other significant events users begin to enter the network more frequently, resulting in great load increase on the infrastructure.

As part of this task, we assume that the social network is $4n$ processes running on the n servers. All servers are absolutely identical machines, each of which has a volume of RAM of 1 GB = 1024 MB [(1)]. Each process takes 100 MB of RAM on the server. At the same time, the needs of maintaining the viability of the server takes about 100 more megabytes of RAM. Thus, each server may have up to 9 different processes of social network.

Now each of the n servers is running exactly 4 processes. However, at the moment of peak load it is sometimes necessary to replicate the existing $4n$ processes by creating $8n$ new processes instead of the old ones. More formally, there is a set of replication rules, the i-th ($1 \le i \le 4n$) of which has the form of $a_i \to (b_i, c_i)$, where a_i, b_i and c_i ($1 \le a_i, b_i, c_i \le n$) are the numbers of servers. This means that instead of an old process running on server a_i, there should appear two new copies of the process running on servers b_i and c_i. The two new replicated processes can be on the same server (i.e., b_i may be equal to c_i) or even on the same server where the original process was (i.e. a_i may be equal to b_i or c_i). During the implementation of the rule $a_i \to (b_i, c_i)$ first the process from the server a_i is destroyed, then appears a process on the server b_i, then appears a process on the server c_i.

There is a set of $4n$ rules, destroying all the original $4n$ processes from n servers, and creating after their application $8n$ replicated processes, besides, on each of the n servers will be exactly 8 processes. However, the rules can only be applied consecutively, and therefore the amount of RAM of the servers imposes limitations on the procedure for the application of the rules.

According to this set of rules determine the order in which you want to apply all the $4n$ rules so that at any given time the memory of each of the servers contained at most 9 processes (old and new together), or tell that it is impossible.

Input

The first line of the input contains integer n ($1 \le n \le 30\,000$) — the number of servers of the social network.

Next $4n$ lines contain the rules of replicating processes, the i-th ($1 \le i \le 4n$) of these lines as form a_i, b_i, c_i ($1 \le a_i, b_i, c_i \le n$) and describes rule $a_i \to (b_i, c_i)$.

It is guaranteed that each number of a server from 1 to n occurs four times in the set of all a_i, and eight times among a set that unites all b_i and c_i.

Output

If the required order of performing rules does not exist, print "NO" (without the quotes).

Otherwise, print in the first line "YES" (without the quotes), and in the second line — a sequence of $4n$ numbers from 1 to $4n$, giving the numbers of the rules in the order they are applied. The sequence should be a permutation, that is, include each number from 1 to $4n$ exactly once.

If there are multiple possible variants, you are allowed to print any of them.

Examples

input
2 1 2 2 1 2 2 1 2 2

```
1 2 2
2 1 1
2 1 1
2 1 1
2 1 1
```

output

```
YES
1 2 5 6 3 7 4 8
```

input

```
3
1 2 3
1 1 1
1 1 1
1 1 1
2 1 3
2 2 2
2 2 2
2 2 2
3 1 2
3 3 3
3 3 3
3 3 3
```

output

```
YES
2 3 4 6 7 8 10 11 12 1 5 9
```

Note

(1) To be extremely accurate, we should note that the amount of server memory is 1 GiB = 1024 MiB and processes require 100 MiB RAM where a gibibyte (GiB) is the amount of RAM of 2^{30} bytes and a mebibyte (MiB) is the amount of RAM of 2^{20} bytes.

In the first sample test the network uses two servers, each of which initially has four launched processes. In accordance with the rules of replication, each of the processes must be destroyed and twice run on another server. One of the possible answers is given in the statement: after applying rules 1 and 2 the first server will have 2 old running processes, and the second server will have 8 (4 old and 4 new) processes. After we apply rules 5 and 6, both servers will have 6 running processes (2 old and 4 new). After we apply rules 3 and 7, both servers will have 7 running processes (1 old and 6 new), and after we apply rules 4 and 8, each server will have 8 running processes. At no time the number of processes on a single server exceeds 9.

In the second sample test the network uses three servers. On each server, three processes are replicated into two processes on the same server, and the fourth one is replicated in one process for each of the two remaining servers. As a result of applying rules $2, 3, 4, 6, 7, 8, 10, 11, 12$ each server would have 7 processes (6 old and 1 new), as a result of applying rules $1, 5, 9$ each server will have 8 processes. At no time the number of processes on a single server exceeds 9.