C. Lucky Tree

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

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits **4** and **7**. For example, numbers **47**, **744**, **4** are lucky and **5**, **17**, **467** are not.

One day Petya encountered a tree with n vertexes. Besides, the tree was weighted, i. e. each edge of the tree has weight (a positive integer). An edge is lucky if its weight is a lucky number. Note that a tree with n vertexes is an undirected connected graph that has exactly n - 1 edges.

Petya wondered how many vertex triples (i, j, k) exists that on the way from i to j, as well as on the way from i to k there must be at least one lucky edge (all three vertexes are pairwise distinct). The order of numbers in the triple matters, that is, the triple $(1, 2, 3)$ is not equal to the triple $(2, 1, 3)$ and is not equal to the triple $(1, 3, 2)$.

Find how many such triples of vertexes exist.

Input

The first line contains the single integer n $(1 \le n \le 10^5)$ — the number of tree vertexes. Next n - 1 lines contain three integers each: u_i v_i w_i $(1 \le u_i, v_i \le n, 1 \le w_i \le 10^9)$ — the pair of vertexes connected by the edge and the edge's weight.

Output

On the single line print the single number — the answer.

Please do not use the %lld specificator to read or write 64-bit numbers in C++. It is recommended to use the cin, cout streams or the %I64d specificator.

Examples

input

```
4
1 2 4
3 1 2
1 4 7
```

output

```
16
```

input

```
4
1 2 4
1 3 47
1 4 7447
```

output

```
24
```

Note

The 16 triples of vertexes from the first sample are: $(1, 2, 4), (1, 4, 2), (2, 1, 3), (2, 1, 4), (2, 3, 1), (2, 3, 4), (2, 4, 1), (2, 4, 3), (3, 2, 4), (3, 4, 2), (4, 1, 2), (4, 1, 3), (4, 2$

In the second sample all the triples should be counted: $4 \cdot 3 \cdot 2 = 24$.