# CHN12

**Praveen falls from a tall tree**

Praveen has climbed a tall tree and now he can't get down! While he is waiting for Arjun to bring a ladder, he has decided to amuse himself by numbering the **N** nodes of the tree from from **1** to **N** and associating a value **S[i]** with every vertex.

For that, he applies the following procedure.

val = 1
Let T be our tree.
while T is not empty:
- Identify the branching nodes of tree T. A node of tree T is said to be a branching node if its degree > 2.
- Choose all the nodes of T which have a path to any leaf node not passing through any of the branching nodes.
- Remove all of these chosen nodes from the tree T.
  - Set S value of all these removed nodes to be val.
- Increase val by 1, i.e. val += 1

Note that first step of identification of branching nodes is re-done in each execution of the while loop.

Please check the example problem statement to understand how this process works. Note that Praveen cannot actually remove the nodes from the tree. He just simulates the procedure in his head's supercomputer.

After Arjun comes back with the ladder, Praveen decides to ask him **Q** queries about the tree. Each query will contain two nodes **u, v**. Let the path in the tree from node **u** to **v**, be **u, $u_1$, $u_2$, .., $u_r$, v**. Consider the array **S[u], S[$u_1$],..,S[$u_r$], S[v]**. For each query, Arjun has to find the number of inversion pairs in this array. **i** and **j** form an inversion pair, if **i > j** and **S[i] < S[j]**. All these hours sitting atop the tree have made Praveen light in the head, and he refuses to come down until Arjun answers all his

queries.

Unfortunately, Arjun does not have a supercomputer in his head like Praveen does and he has also left his laptop at home. So please help Arjun answer these queries and get Praveen down.

**Input**

The first line of input contains a single integer **T** denoting the number of test cases.

For each test case:

- The first line contains two space separated integers **N, Q** as defined in the statement
- Each of next **N - 1** lines contains two space separated integers **u, v** denoting that there is an edge between vertex **u** and **v** in the tree.
- Each of next **Q** lines will contain two space separated integers **u, v**, the nodes corresponding to the query.

**Output**

- For each test case, print **Q** lines corresponding to the answers of the queries in separate lines.

**Constraints**

- **$1 \leq T \leq 10^5$**
- **$1 \leq N, Q \leq 10^5$**
- **$1 \leq u, v \leq N$**
- Sum of all **N**, as well as sum of all **Q**, won't exceed $10^5$.

**Example**

**Input 1:**

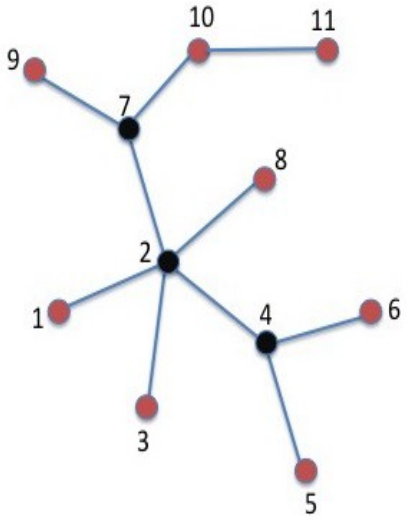
```
1
11 6
1 2
2 3
2 4
4 5
4 6
```

2 8
2 7
7 9
7 10
11 10
5 10
1 9
6 10
3 7
2 9
9 2

**Output 1:**

3
2
3
0
2
0

**Explanation**




The figure is the tree corresponding to the given sample input.

The nodes marked red have S value 1 and the black nodes have S Value 2. In the first step, the branching nodes are 2, 4 and 7. Nodes 1, 3, 5, 6, 8, 9, 11 are leaf nodes and can be removed, i.e they have S value 1. Even 10 can be removed and it also has S value 1. The branching nodes cannot be touched. In the next iteration, with val = 2, there are no branching nodes and we can remove nodes 2, 4 and 7 giving them S Value 2.

The first query is (5,10). The path between nodes numbered 5 and 10 passes through nodes {5,4,2,7,10} with S values {1,2,2,2,1}. The number of inversions in this array is 3 formed by 3 pairs of {2,1}.

The second query is (1,9). The path between nodes numbered 1 and 9 passes through nodes {1,2,7,9} with S values {1,2,2,1}. The number of inversions in this array is 2 formed by 2 pairs of {2,1}.

The third query is (6,10). The path between nodes numbered 6 and 10 passes through nodes {6,4,2,7,10} with S values {1,2,2,2,1}. The number of inversions in this array is 3 formed by 3 pairs of {2,1}.

The fourth query is (3,7). The path between nodes numbered 3 and 7 passes through nodes {3,2,7} with S values {1,2,2}. There are no inversions in this array.

The fifth query is (2,9). The path between nodes numbered 2 and 9 passes through nodes {2,7,9} with S values {2,2,1}. The number of inversions in this array is 2 formed by 2 pairs of {2,1}.

The last query is (9,2). The path between nodes numbered 9 and 2 passes through nodes {9,7,2} with S values {1,2,2}. The number of inversions in this array is zero.

**Input 2**:
1
20 4
1 2
2 3
2 4
4 6
5 6
6 20

8 20
7 20
4 9
9 10
10 19
17 19
18 19
10 11
11 12
11 13
13 14
13 15
15 16
4 10
1 16
17 18
11 11

**Output 2:**

0
18
1
0