Problem Statement for **Help The Tourist**

Problem:
During summer time, old cities in Europe are swarming with tourists who roam the streets and visit points of interest.

Many old cities were built organically and not according to some architecture plan, but, strangely, their growth exhibits a similar pattern: the cities started from three points of interest, with each pair being connected by a bidirectional street; then, gradually, new points of interest were added. Any new point of interest was connected by two new bidirectional streets to two different previous points of interest which were already directly connected by a street.

A tourist visiting such a city would like to do a tour visiting as many points of interest as possible. The tour can start at any point of interest and must end at the same point of interest. The tour may visit each street at most once and each point of interest at most once (with the exception of the first point of interest which is visited exactly twice).

You are given the description of how the city grew. Find the largest number of different points of interest a single tour can visit in this city.

Input:
The first line of the input file contains the number of cases, T. T test cases follow.

Each case begins with the integer N - the total number of points of interest in the city. Points are denoted with numbers from 1 to N; numbers 1, 2, and 3 denote the three original points when the city started, while numbers 4, ..., N denote the other points in the order they were added to the city.

The next N-3 lines each contain a pair of space-separated integers A, B, indicating that the corresponding point of interest was connected by streets to points A and B. First of these lines corresponds to point number 4, second to point number 5, etc.

Output:
For each test case, output one line containing the largest number of points of interest a tour can visit in this city.

Constraints:
1 <= T <= 50
4 <= N <= 15
Compilation time: 10 seconds,
Execution time: 5 seconds.
Memory usage: 256 MB.

Examples:

Input:
2
5
1 2
2 1
6
1 2
1 4

```
4 5
```

Output:

```
4
6
```