Graphs, Trees and Recursion

Assignment 1 (recursion):

Implement the following algorithm:

Find the depth of a binary tree with **N** elements.

For the choice of the algorithm, please look at theory lesson. It's one of the two mentioned algorithms. We expect that you use recursion for this algorithm. You can implement the searching algorithm on your own or reuse an implementation from the web. In the latter case, don't forget to mention the references that you used. Please think and design first.

Input Format

Input will be read from the standard input (cin) and will have the following format:

First line of the input contains the number of tree elements **N**
Next **N** lines contain 3 space separated integers **X**, **Y**, **Z** where **X** is root, **Y** is left node of **X** and **Z** is right node of **X**

Output Format

Depth of the tree with the root node number 1.

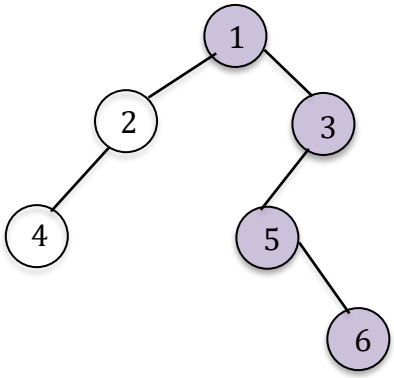
Constraints

- 0 < N < 20

Example

Input:
6
1 2 3
2 4 0
3 5 0
4 0 0
5 0 6
6 0 0
Output:
4

Explanation:



The tree defined in the example can be seen in the picture above. The depth of the tree is 4, the deepest path consists of nodes 1,3,5,6.

Delivery

Code in the file called ass5_1.c. Your design choices and reflection described in a document.

Tip:
You can test your code by using the following input files:
in5_1 should give result 4
in5_2 should give result 5
in5_3 should give result 6
in5_4 should give result 9

or just by running

make test

If necessary, please make test script executable (chmod +x test).

Don't forget to test with your own test data. Passing examples files doesn't guarantee correctness of the algorithm.

Assignment 2 (graphs - shortest path):

Implement the following algorithm:

We have a network of **N** nodes numbered from 1 to **N**. Some of the network nodes are connected with a **bidirectional** connection. Your task is to find an shortest route (path) from node **1** to node **N**. Output of your program will show the number of the connections in the shortest route from 1 to N.

For the choice of the algorithm, please look at one of the two algorithms mentioned in the theory lesson. You can implement the searching algorithm on your own or reuse an implementation from the web. In the latter case, don't forget to mention the references that you used.

Tip: probably the simplest algorithm to implement this is explained here (https://medium.com/basecs/going-broad-in-a-graph-bfs-traversal-959bd1a09255).

Tip 1:

Search on Internet can lead you to Dijkstra's shortest path algorithm. It's not recommended to use it for this assignment, it's definitely overkill for this problem.

To implement this algorithm, think well how to structure your data. C++ containers mentioned in the theory lesson can be useful to make well readable and compact code.

Please think and design first. Good design is worth points even without perfect implementation.

Tip 2:

You can test your code by running

```
make test
```

correct.txt file shows the correct results per file.

Input Format

Input will be read from the standard input (cin) and will have the following format:

First line contains **T**. **T** test cases follow.
First line of each test case contains two space-separated integers **N**, **M**.
Each of the next **M** lines contains two space-separated integers **X** and **Y**, denoting that there is a connection between node **X** and node **Y**.

Output Format

Answer to each test case in a new line.

Constraints

- $1 \leq \mathbf{T} \leq 10$
- $1 \leq \mathbf{N} \leq 10^4$
- $1 \leq \mathbf{M} \leq 10^5$
- $1 \leq \mathbf{X}, \mathbf{Y} \leq \mathbf{N}$

Example

Input:

Contents of File	Meaning
2	2 --> 2 testcases
3 2	3 2 --> 3 Nodes, the next 2 lines contain the connection
1 2	1 2 --> connection
2 3	2 3 --> connection
4 4	4 4 --> 4 Nodes, the next 4 lines contain the connection
1 2	1 2 --> connection
2 3	2 3 --> connection
3 4	3 4 --> connection
4 2	4 2 --> connection

Output:
2
2

Explanation:
There are 2 test cases, first case has N=3, M=2 followed by 2 [X,Y] entries, second test case has N=4, M=4 followed by 4 [X,Y] entries.

In the first case, the shortest route to 3 is 1 - 2 - 3, so 2 connections are used.

In the second case, the shortest route to 4 is 1 – 2 – 4, so again 2 connections are used.

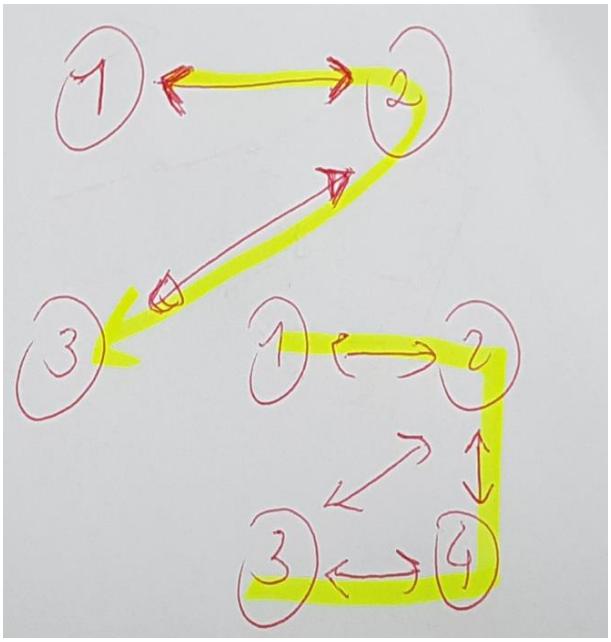


Figure 1: sketch of the testcases

Delivery

Code in the file called ass5_2.c. Your design choices and reflection described in a document.

For the delivery of both assignments, you can best zip the whole ass3 directory that has been cleaned from executables (make clean).