INSTITUTO SUPERIOR TÉCNICO

Analysis and Synthesis of Algorithms 2023/2024

2° Project

Date enunciated: December 04, 2023
Delivery Deadline: December ~~18~~, 2023

Problem description

Professor João Caracol is carrying out a study for the government taskforce responsible for studying communicable diseases in Portugal. The taskforce is particularly interested in the topic of disease transmission among the Portuguese population, in order to study the best intervention mechanisms to contain the spread of diseases.

To do this, Professor João Caracol had access to data from the TugaNet social network, which he believes is representative of real social interactions between individuals in the Portuguese population. So, in order to study the worst case scenario for the spread of a given infection in Portugal, Professor João Caracol wants to understand the greatest number of jumps a given disease can make. However, given the density of Portuguese cities, Professor João Cara- col decided to make a simplifying assumption: individuals who know each other directly or indirectly become infected instantly.

Input

The input file contains information about the TugaNet network, which is defined as a directed graph of relationships between two individuals, as follows:

- A line containing two integers: the number n of individuals ($n \geq 2$), and the number of relationships m to indicate ($m \geq 0$);
- A list in which each line i contains two integers x and y, representing that individual x knows individual y.

Any integers on a line are separated by exactly one blank space and contain no characters other than the end of the line.

Assume that the input graphs are directed (potentially) cyclic.

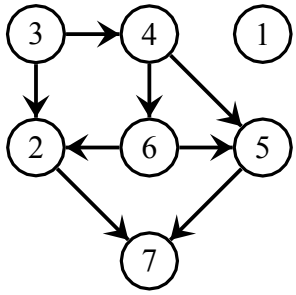
Output

The program should write to the output an integer *s* corresponding to the maximum number of jumps a disease can make on the TugaNet network.

Example 1

Input

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



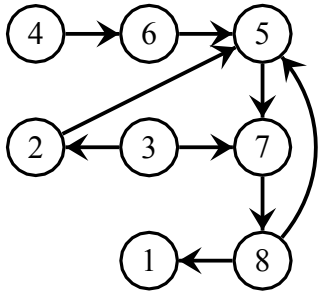
Output

```
4
```

Example 2

Input

```
8 9
2 5
3 2
3 7
4 6
5 7
6 5
7 8
8 1
8 5
```



Output

```
3
```

Implementation

The project should preferably be implemented using the C, C++ or Rust programming languages. Submissions in Java/Python will also be accepted, although strongly discouraged. Students who choose to do so should be aware that submissions in Java/Python may not pass all tests even if they implement the correct algo- pace. It should also be noted that recursive solutions can exhaust the stack limit when executed on the largest tests, so **iterative** algo-rhythms are recommended.
The time needed to implement this project is less than 15 hours.

Compilation parameters:

```
C++: g++ -std=c++11 -O3 -Wall file.cpp -
lm C: gcc -O3 -ansi -Wall file.c -lm
Javac: javac File.java
Java: java -Xss32m -Xmx256m -classpath . File
Python: python3 file.py
```

Project submission

The project submission should include a summary report and a file with the solution's source code. Information on possible programming languages is available on the Mooshak system website. The programming language is identified by the file extension. For example, a project written in c should have the extension .c. After compilation, **the resulting program should read from the standard input and write to the standard output**. Information on compilation options and restrictions can be obtained from the Mooshak system's help button. The compilation command should not produce output, otherwise it will be considered a compilation error.

Report: must be submitted via the Fénix system in PDF format with no more than **2** pages, 12pt font and 3cm margins. The report should include a description of the solution, the theoretical analysis and the experimental evaluation of the results. The report should include any reference materials that have been used in carrying out the project. Reports that are not submitted in PDF format will score 0. A report template will be released in due course.

Source code: must be submitted via the Mooshak system and the report (in PDF format) must be submitted via Fénix. The source code will be evaluated automatically by the Mooshak system (`http://acp.tecnico.ulisboa.pt/~mooshak/`). Students are encouraged to submit preliminary solutions to the Mooshak system and Fénix as soon as possible. Please note that only the last submission will be considered for evaluation. All previous submissions will be ignored: this includes the source code and the report.

Evaluation

The project must be carried out in groups of one or two students and will be assessed in two phases. In the first phase, during submission, each implementation will be run through a set of tests, which account for 85% of the final grade. In the second phase, the report will be assessed. The grade for the report contributes 15% of the final grade.

Automatic evaluation

The first phase of the project is evaluated automatically with a set of tests, which are run on a computer with the **GNU/Linux** operating system. It is essential that the source code compiles without errors and respects the input and output standards indicated above. Projects that do not respect the specified formats will be penalized and may score 0 if they fail all the tests. The tests will **not be made public before submission**. However, all tests will be made available after the deadline for submitting the project. In addition to checking the correctness of the output produced, the assessment environment **restricts the** available **memory and execution time.** Most tests run the `diff` command as follows:

```
diff output result
```

The `result` file contains the output generated by the executable from the `input` file. The `output file` contains the expected output. A program passes a test and receives the corresponding value when the `diff` command reports no differences (i.e. produces no output). The system reports a value between 0 and 170.
The grade obtained in the automatic classification may be cut if the analysis of the code shows the use of solutions adjusted to specific inputs or random/constant outputs.

Copy Detection

The evaluation of projects includes a procedure for detecting copies. Submitting a project implies a commitment that the work has been carried out exclusively by the students. Violating this commitment or attempting to submit code that was not developed by the group will result in all the students involved (including the students who provided the code) failing the course. Any attempt at cheating, direct or indirect, will be reported to the IST Pedagogical Council, to the course coordinator, and will be penalized in accordance with the rules approved by the University and published in the "Diário da República".