Assignment No. 10: Depth-first search

Allocated time: 2 hours

Implementation

You are required to implement correctly and efficiently the căutare în depth-first search algorithm (DFS) (Chapter 22.3 from the book). For graph representation, you should use adjacency lists. You also have to:

- Implement the Tarjan algorithm for detecting strongly connected components (https://en.wikipedia.org/wiki/Tarjan's_strongly_connected_components_algorithm)
- Implement topological sorting (described in chapter 22.4)

Evaluation

! ! Before you start to work on the algorithm evaluation code, make sure you have a correct algorithm! Prove the correctness of your algorithm/implementation by running it on a smaller graph:

- Priting the initial graph (the adjacency lists)
- Printing all strongly connected components of the graph
- A list of nodes sorted topologically (should this list be non-empty/if it is - why so?)

Since, for a graph, both |V| and |E| may vary, and the running time of DFS depends on both (how?), we will make each analysis in turn:

1. Set |V| = 100 and vary |E| between 1000 and 5000, using a 100 increment. Generate the input graphs randomly – make sure you don't generate the same edge twice for the same graph. Run the DFS algorithm for each <|V|, |E|> pair value and count the number of operations performed; generate the corresponding chart (i.e. the variation of the number of operations with |E|).

2. Set |E| = 9000 and vary |V| between 100 and 200, using an increment equal to 10. Repeat the procedure above to generate the chart which gives the variation of the number of operations with |V|.

3. Interpret your charts.