Assignment No. 10: Depth-First Search

Allocated time: 2 hours

Implementation

You are required to implement **correctly** and **efficiently** the *Depth-First Search (DFS)* graph algorithm (Section 22.3 from the book[1]). For graph representation, you should use adjacency lists.
You are also required to:

- Perform edge labeling on the graph (look for *Classification of edges* sub-section in section 22.3)
- Implement the *topological sort* algorithm (section 22.4 from the book[1])

Evaluation

! Before you start to work on the algorithm evaluation code, make sure you have a correct algorithm! You will have to prove your algorithm works on a small-sized graph (which you may hardcode in your main function); you should display:

- The original graph – the adjacency lists
- The edge labels for all edges, following a DFS run
- Does your graph have a topological sort or not? If it does, which is it? If not, why?

We will analyze the running time for DFS in the average case. 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 110 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.

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. Introduction to Algorithms