1. `[DPV] 3.2`
2. `[DPV] 3.3`
3. What is the smallest possible depth of a leaf in a decision tree for comparison based sorting?
4. Rewrite the `Explore` algorithm so that it is non-recursive (that is, explicitly use a stack). The calls to `previsit` and `postvisit` should be positioned so that they have the same effect as in the recursive procedure.
5. You are given a binary tree $T = (V, E)$ (in adjacency list format), along with a designated root node $r \in V$. A vertex u us said to be an *ancestor* of v in the rooted tree, if the path from r to v in T passes through u.

   You wish to preprocess the tree so that queries of the form "is u an ancestor of v?" can be answered in constant time. The preprocessing itself should take linear time. How can this be done?
6. Give an efficient algorithm that takes as input a directed graph $G = (V, E)$, and determines whether or not there is a vertex $x \in V$ from which all other vertices are reachable.
7. Give a linear-time algorithm that takes as input a DAG $G = (V, E)$ and determines whether or not G contains a directed path that touches every vertex exactly once.