11.1. Learn how to find LCA in a link-cut tree in $O(\log n)$ time.

11.2. Learn how to check if two nodes belong to the same link-cut tree in $O(\log n)$ time.

11.3. See how the potential changes after the `link` and `cut` operations, prove that the amortized running time of these operations is $O(\log n)$.

11.4. Link-cut tree, edges are black or white. Check that the edges alternate on the way from v to the root.

11.5. Link-cut tree, each edge has weight. Find on the path from v to the root the nearest edge with weight at most d.

11.6. Add operation to the link-cut tree that changes the root of the tree (that is, all edges become oriented so that the given node becomes the root). (Hint: see which edges change direction and what happens to the corresponding paths, think about how to make this change in the splay tree).

11.7. There is a graph of n vertices, initially empty. Edges are added to it. After each addition, you need to recalculate the minimum spanning tree.

11.8. Given a tree, a number is written at each vertex. Requests: 1) the vertex v and the number x are given. Add to all descendants u of the vertex v the number $x - d(u, v)$, where $d(u, v)$ is the distance between the vertices, 2) find the value of the number at the vertex. Answer both queries in $O(\log n)$ time.

11.9. Given a matrix $d[i, j]$. Construct a weighted tree such that the distance from i to j is equal to $d[i, j]$.