Bi-connectivity graphs and algorithms

- An articulation point in a graph is a vertex whose removal make the graph disconnected.
- A biconnected graph is a connected, undirected graph that remains connected even if we remove any vertex.
- Equivalently a biconnected graph is a graph with no articulation points.

Tarjan's algorithm

- Tarjan's algorithm is used for finding articulation points
- The main idea is to use the DFS algorithm to check if a node is an articulation point.
- For every node U there are 2 cases for which U is an articulation point:
    - All paths (in a tree created by DFS) from A to V require U to be in the graph.
    - U is the root of the tree (starting node in DFS) and it has at least 2 children which form disconnected subgraphs.
- We need to define two numbers for each node i in the DFS tree:
    - discovery time d_i
    - low l_i
- Where discovery time denotes the order of DFS, i.e. If we start at a node A and go to nodes B then C then $d_A = 1, d_B = 2, d_C = 3$. Low of a node i is defined as the lowest (minimum) discovery time of any node that is reachable from the node i taking at most one back edge. Back edge is an edge (u, v) where v has already been visited and this edge is not a part of the DFS traversal, but the vertices u, v are connected in the graph.

- Then for any node i:
    - If i is the root of the tree (starting node of DFS) and it has more than one child then it is an articulation point.
    - Else if for any child node a it is true that $l_a \geq d_i$ then i is an articulation point.
- Time complexity of that algorithm is $O(|V| + |E|)$.

Finding biconnected components

- To find all the biconnected components of a graph just maintain the list of edges while doing DFS and run the Tarjan's algorithm as usual. Then when an articulation point is detected pop all the edges up until the edge connecting the articulation point and the child for which we found $l_a \geq d_i$ and pop that edge too.