Why is \( u \xrightarrow{G} w \) or \( u \xrightarrow{GR} w \) because \( u \) has highest post-order.

Either \( u \xrightarrow{GR} w \) or \( w \xrightarrow{GR} u \).
k-connected graph (undirected)

k-edge connected

The network remains connected despite the removal of any k-1 edges in the graph

A tree is 1-connected

k-vertex connected

----- despite removal of any k-1 vertices in the graph
2-edge connected: unconnected

Menger's theorem: A $k$-edge-connected graph has at least $k$ edge-disjoint paths between any pair of vertices.
Given a graph \( G = (V, E) \), how do we determine if \( G \) is biconnected?

From the defn of 2-connectedly, we can run \( n \) instances of DFS

\[ \implies O(n(n+m)) \] runtime time

Question: Can we design a faster algorithm.

Consider the biconnected components of a graph.

\( e_i \) is related \( e_2 \), \( e_i, e_2 \in E \)

iff there is a common cycle containing \( e_i \) and \( e_2 \).
This relation is an equivalence relation and defines the maximal bi-connected components.

$e_1, e_2, e_3$

$x, y$ are articulation points.

The component graph is a Tree.

A DFS with some extra bookkeeping lets us identify the bi-connected components. Running time: $O(m+n)$.