```
id = -1 # init to start at id = 0
g = adjacency list with undirected edges
n = size of the graph
oec = 0 # root outEdgeCount during DFS
# In these arrays index i represents node i
low = [0, 0, … 0, 0]             # Length n
ids = [0, 0, … 0, 0]             # Length n
visited = [false, …, false] # Length n
isArt   = [false, …, false] # Length n
findArtPoints()
print(isArt)

function findArtPoints():
  for (i = 0; i < n; i = i + 1):
    if (!visited[i]):
      id, oec = dfs(root=i, at=i, id, oec=0)
      isArt[i] = (oec>1) # Override with T/F
  return
```

```
# Perform DFS to find articulation points.
function dfs(root, at, id, oec):
  visited[at] = true
  id += 1
  low[at] = ids[at] = id
  # For each edge from node ‘at’ to node ‘to’
  for (to : g[at]):
    if (!visited[to]):
        if at == root:
           oec += 1 # Count outgoing edges for root
        g[to].remove(at) # remove backward link
        id, oec = dfs(root, to, id, oec)
        low[at] = min(low[at], low[to])
        # Articulation point found via bridge with <
        #   and cycle with ==
        if (ids[at]<=low[to]:
             isArt[at] = true
     else:
        low[at] = min(low[at], ids[to])
  Return id, oec
```