Lab#4 – Election

Objective

In this lab, you will be implementing a simulation of the ring-based election algorithm.

Task

You will need to create a *Node* class which will have methods such as returning the ID of the node, returning its neighbours' ID, and receiving and sending messages. There should also be a *Message* class for the different types of messages that are transmitted during the election process. A *Message* should have methods to return its type and content, as well as who sent it. Each node should be run as a separate process that communicate with other processes using messages. One process/node is designated as the initiator of the election. When the election is being executed, each node should print to screen messages indicating any message receive/send events and any actions taken upon such events, such as updating its local value for the largest ID seen so far or deciding the elected leader to be another process or itself, etc. This can be shown to the TA to demonstrate the correct running of your programs. Once a node has finally decided on the elected leader, it should announce to screen. Each node's message printed to screen should be labeled by its own ID to distinguish between different nodes. The final result should be that all the nodes have announced that they decided on a leader with the same ID.

Presentation to the TA:

Show the TA the running programs.