**TCP Syntax**

All code files must be in specific folder to be executed, write following commands:

* Cd ns3-allinone-
* Cd ns-3.x
* Cd scratch
* Touch file.py

**Run command:**

* ./waf --pyrun scratch/file.py

**Libraries:**

* import ns.applications
* import ns.core import
* ns.internet
* import ns.network
* import ns.point\_to\_point

**Functions required:**

1. **NodeContainer()**

* Functionality: This class is used to create and manage a group of nodes (hosts) in the simulation.
* Parameters: None.
* Return Type: A NodeContainer object, which holds one or more nodes.
* Usage: nodes= ns.network.nodecontainer()

1. **Create(n):**

* Creates n nodes and adds them to the NodeContainer.
* Parameters: n (int) – Number of nodes to create.
* Return Type: None
* Usage: nodes.create(n)

1. **PointToPointHelper()**

* Functionality: This class is used to configure and install point-to-point (wired) network devices between two nodes.
* **Parameters**: None.
* **Return Type**: A PointToPointHelper object.
* Usage: pointToPoint = ns.point\_to\_point.PointToPointHelper() pointToPoint.SetDeviceAttribute("DataRate", ns.core.StringValue("5Mbps")) pointToPoint.SetChannelAttribute("Delay", ns.core.StringValue("2ms"))

1. **InternetStackHelper()**

* Functionality: This class installs the Internet stack (TCP/IP) on the nodes. Without this, the nodes cannot communicate via IP/TCP.
* Parameters: None.
* Return Type: An InternetStackHelper object.
* Usage: internet = ns.internet.InternetStackHelper(), internet.Install(nodes)

1. **Ipv4AddressHelper()**

* Functionality: This class assigns IP addresses to devices connected to the network.
* Parameters: None.
* Return Type: An Ipv4AddressHelper object.
* Usage:
* address = ns.internet.Ipv4AddressHelper()
* address.SetBase(ns.network.Ipv4Address("10.1.1.0"), ns.network.Ipv4Mask("255.255.255.0"))
* interfaces = address.Assign(devices)