**Lab Number: 01**

**Datagram Sockets**

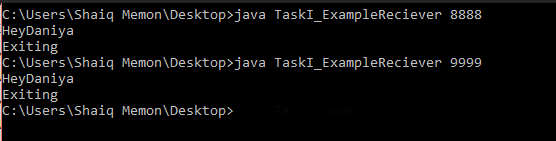
* **Modify the sample code so that the sender uses the same socket to send the same message to two different receivers. Start the two receivers first, then the sender. Does each receiver receive the message? Capture the code and output. Describe the outcome.**

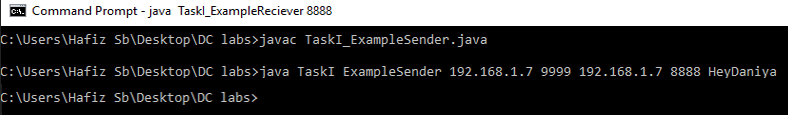
**Receiver**

****

**Sender**

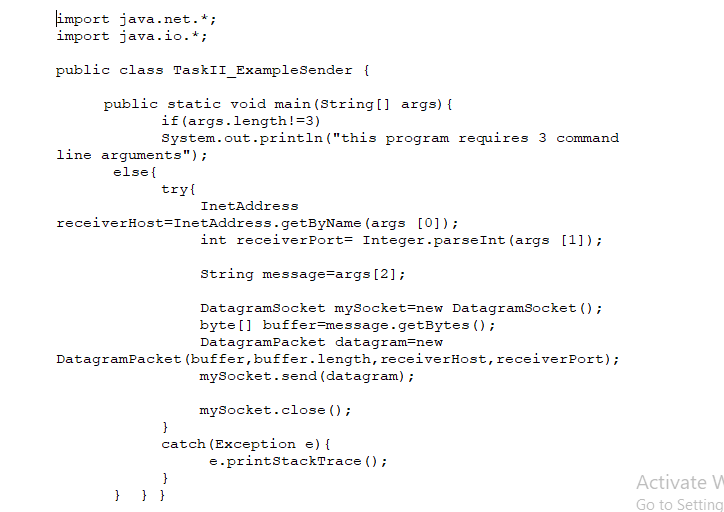
****

****

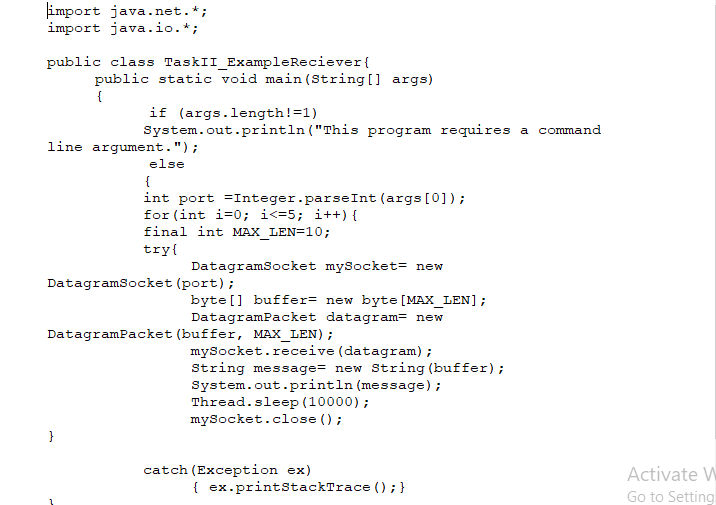
****

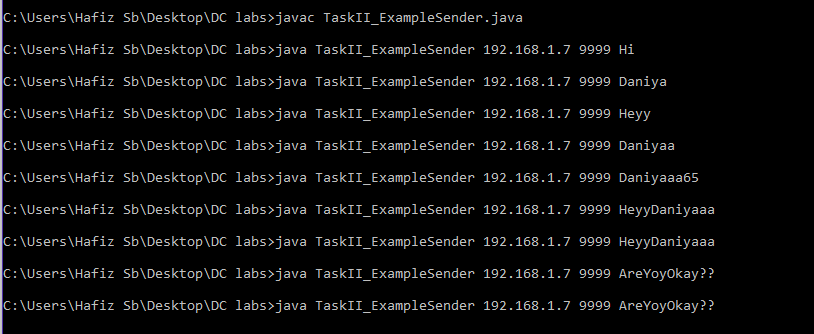
* **Modify the sample code so that the receiver loops five times to repeatedly receive then display the data received. Recompile. Then  
  i. start the receiver  
  ii. Execute the sender, sending a message “message1”, and  
  iii. In another window, start another instance of the sender, sending a message “message2”. Does the receiver receive both the messages? Capture the code and output.**

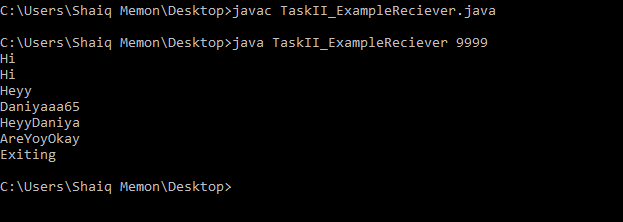
**Sender**

****

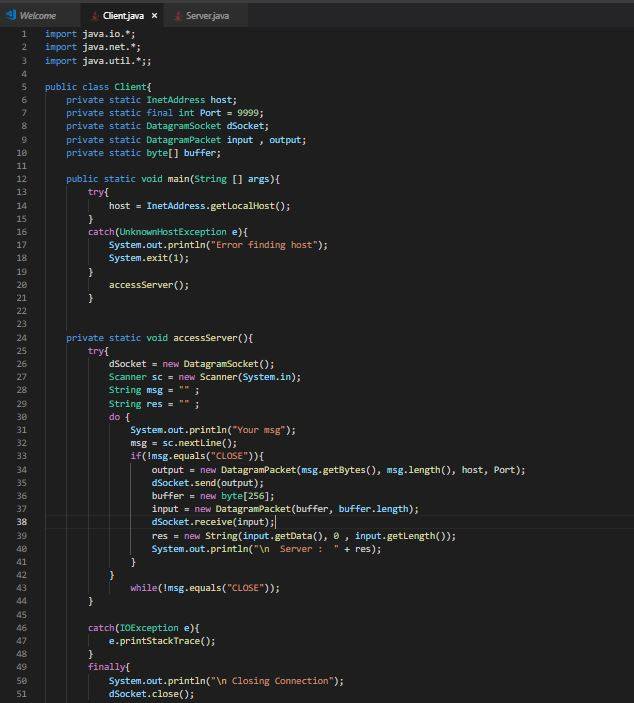
**Receiver**

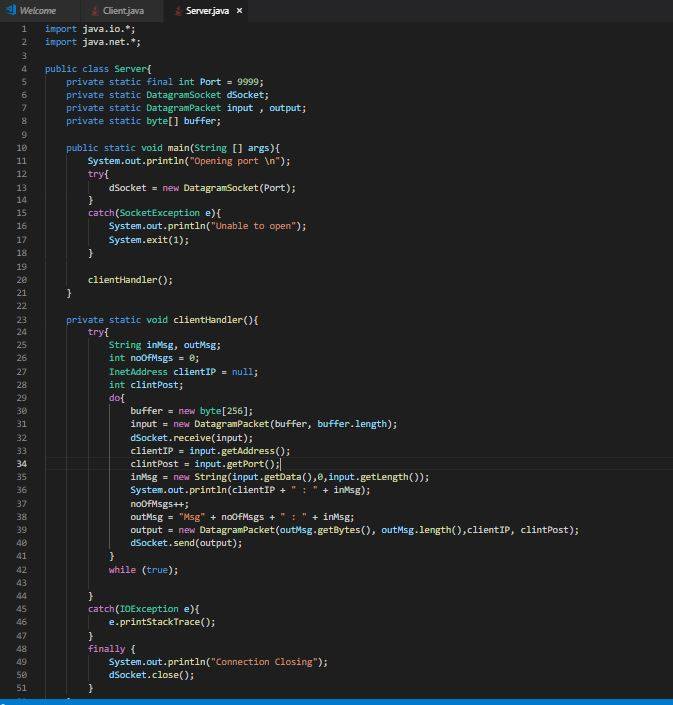
****

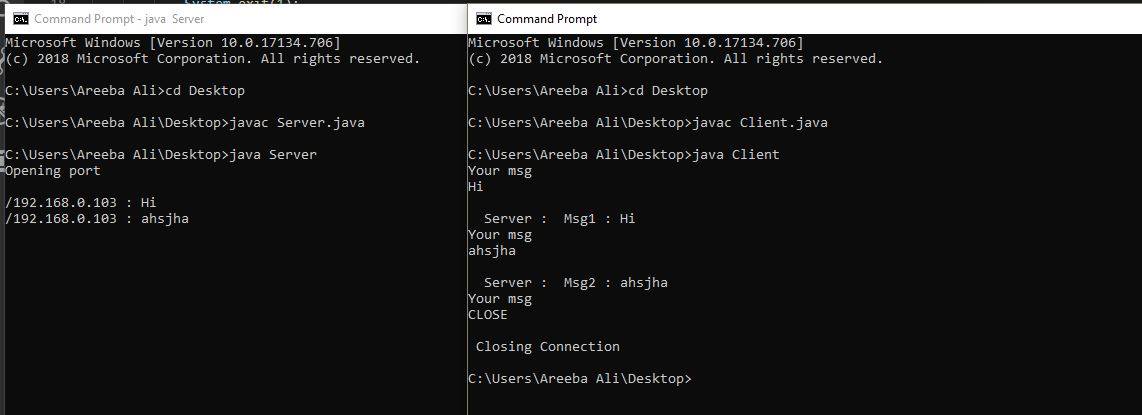
****

****

* **Modify the sample code to cater to a two way communication i.e. Sender sends a message to the Receiver, the Receiver receives the message and sends a reply to the Sender in return.**

****

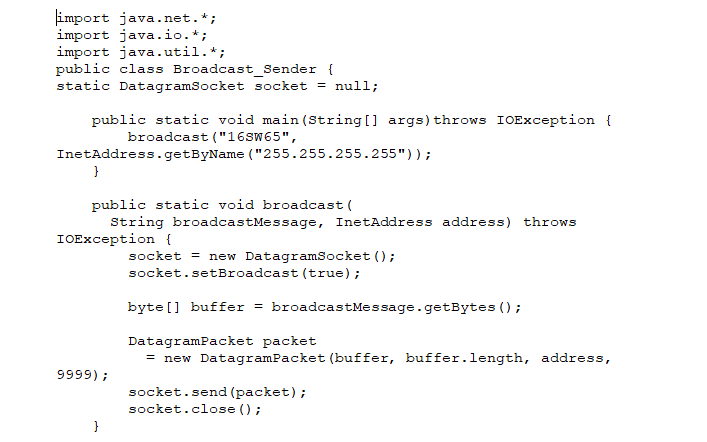
****

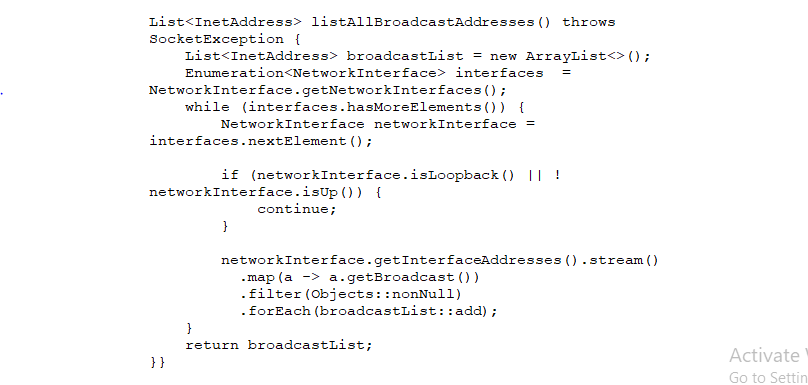
****

**Bonus Tasks**

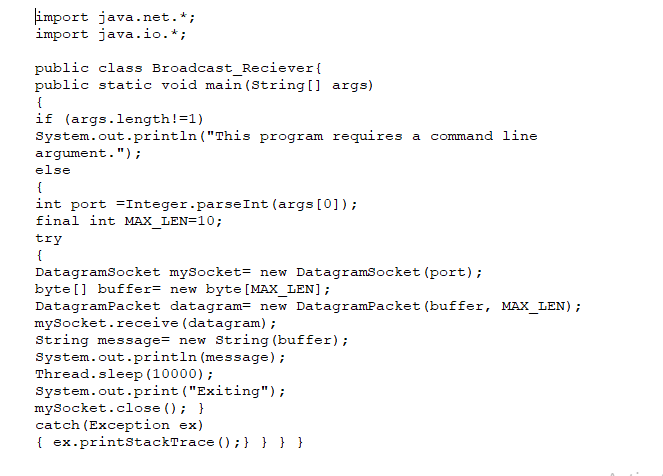
* **Broadcasting: Broadcasting is a one-to-all type of communication, i.e. the intention is to send the datagram to all the nodes in the network. Unlike in the case of point-to-point communication, we don’t have to know the target host’s IP Address. Instead, a broadcast address is used.**

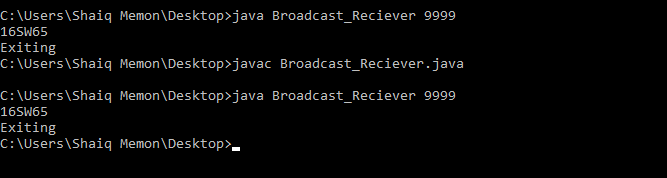
**Sender**

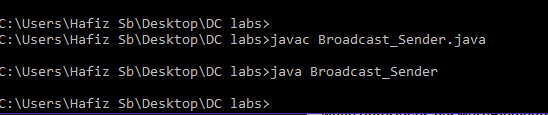
****

****

**Receiver**

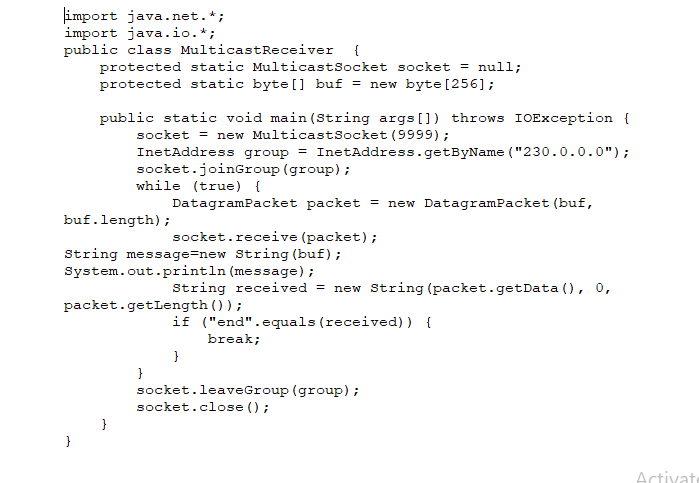
****

****

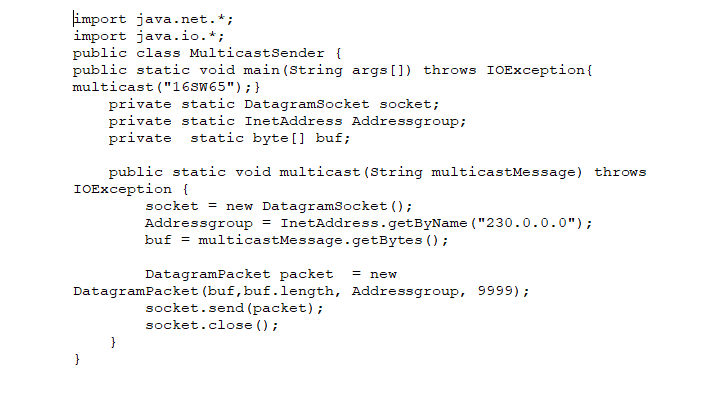
****

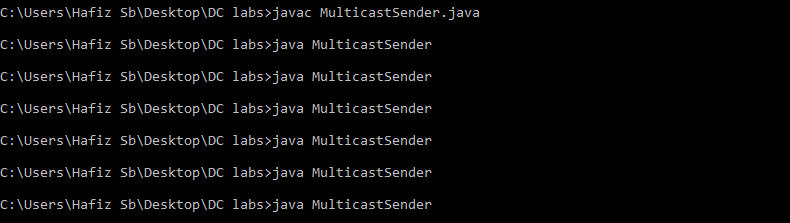
* **Multicasting: Broadcasting is inefficient as packets are sent to all nodes in the network, irrespective of whether they are interested in receiving the communication or not. This may be a waste of resources. Multicasting sends packets to only those nodes which are interested. Multicasting is based on a group membership concept, where a multicast address represents each group.**

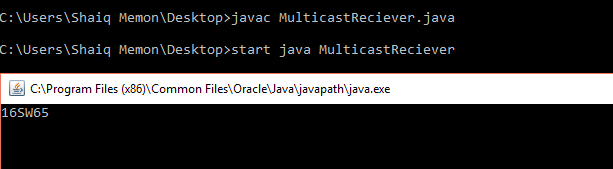
**Receiver**

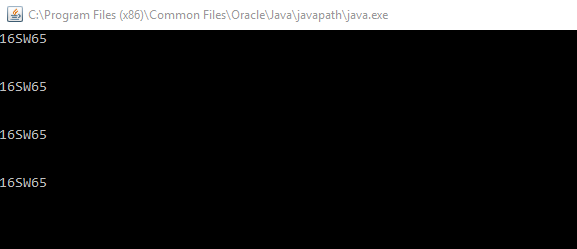
****

**Sender**

****

****

****

****