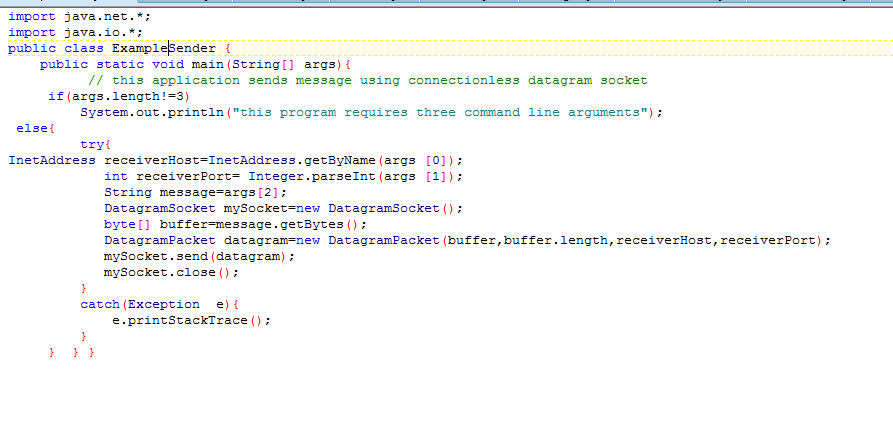
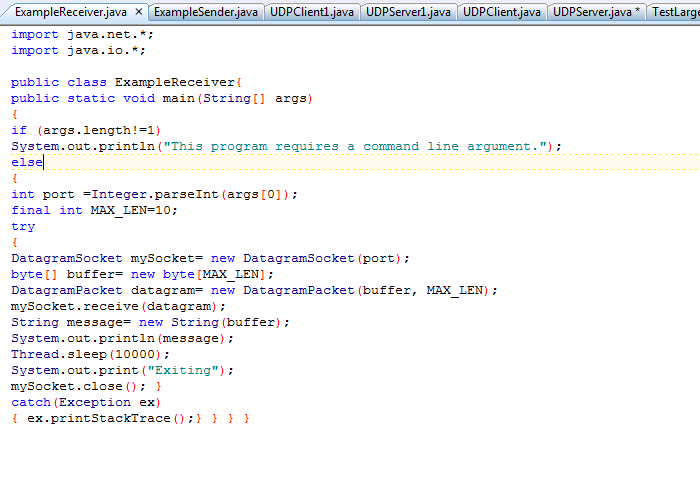
**Lab#01**

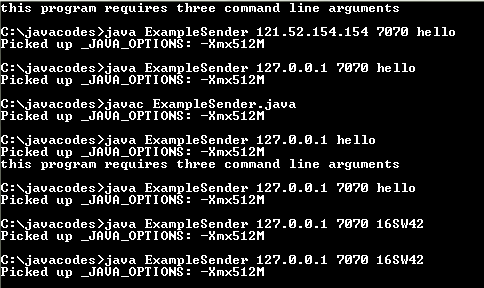
**Object: Creating applications using Datagram sockets.**

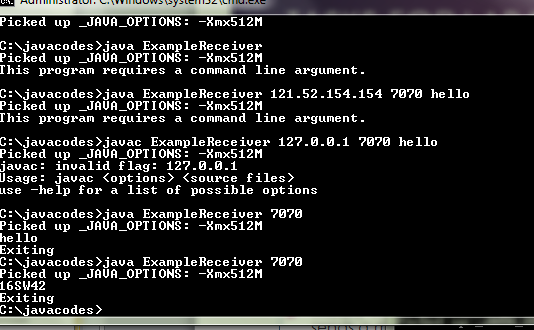
* **Task#01:** Modify the sample code so that the sender uses the same socket to send the same message to two different receivers. Start the two receivers first, then the sender. Does each receiver receive the message? Capture the code and output. Describe the outcome.
* **ExampleSender.java**

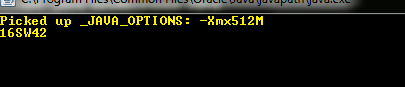
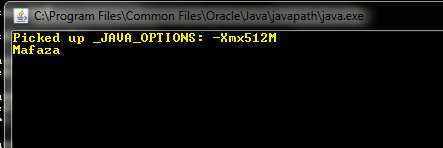


* **ExampleReceiver.java**

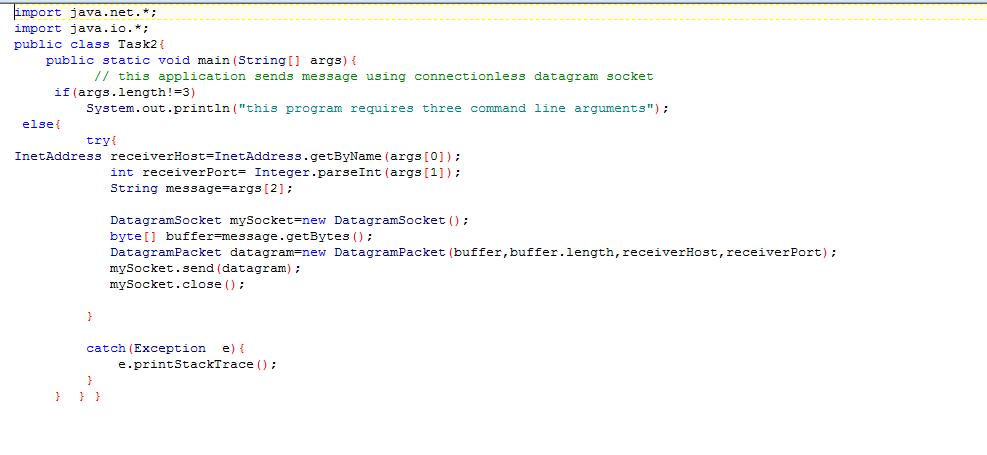


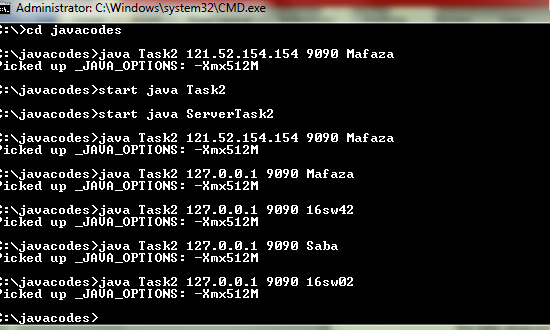


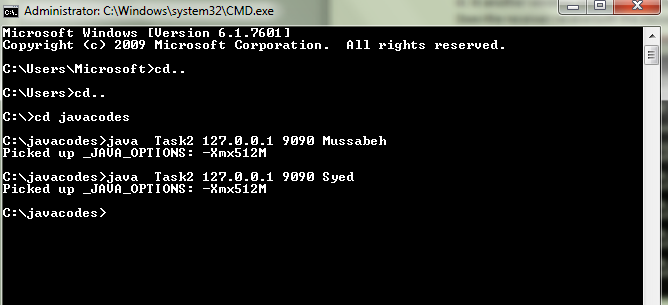




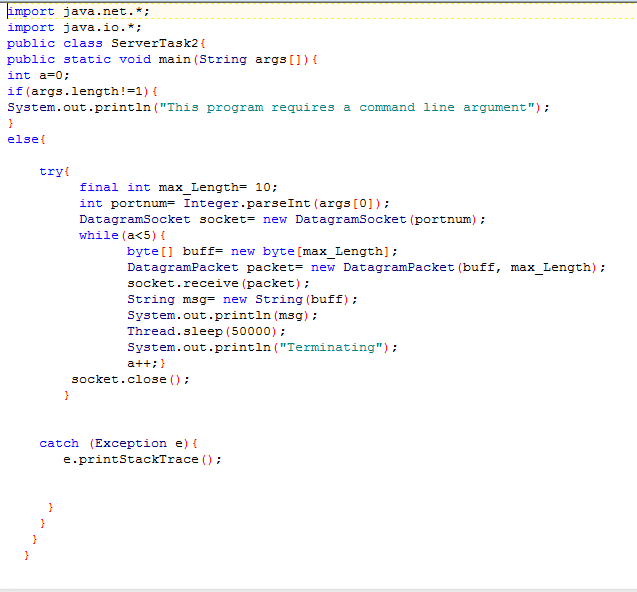
* Modify the sample code so that the receiver loops five times to repeatedly receive then display the data received. Recompile. Then:  
  i. Start the receiver  
  ii. Execute the sender, sending a message “message1”, and  
  iii. In another window, start another instance of the sender, sending a message “message2”. Does the receiver receive both the messages? Capture the code and output.
* **Task2.java**

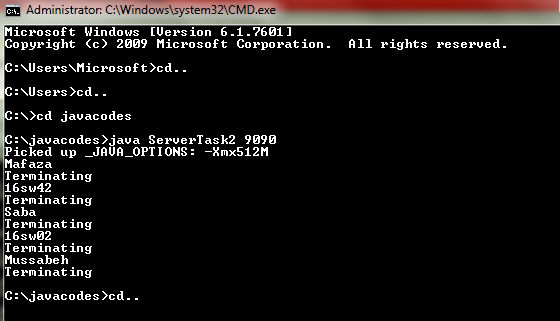




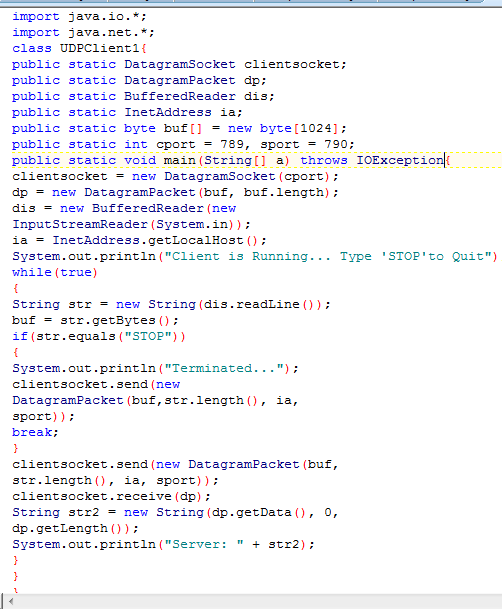


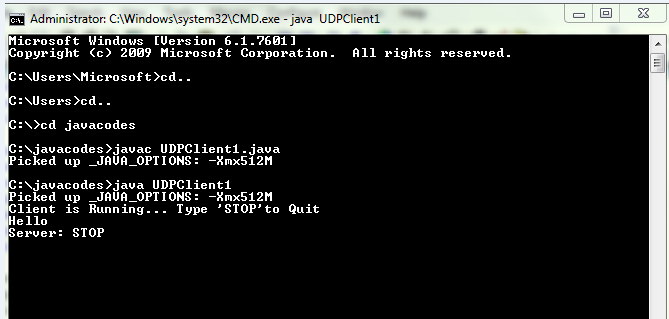
* **ServerTask2.java**



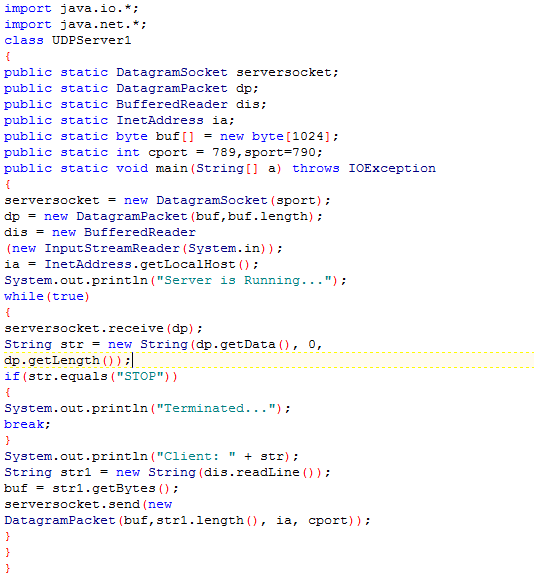


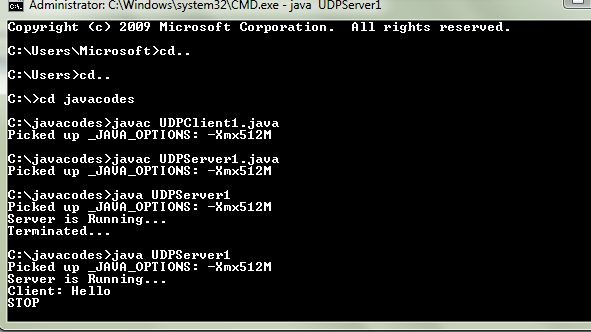
* Modify the sample code to cater to a two way communication i.e. Sender sends a message to the Receiver, the Receiver receives the message and sends a reply to the Sender in return.
* **UDPClientt1.java**



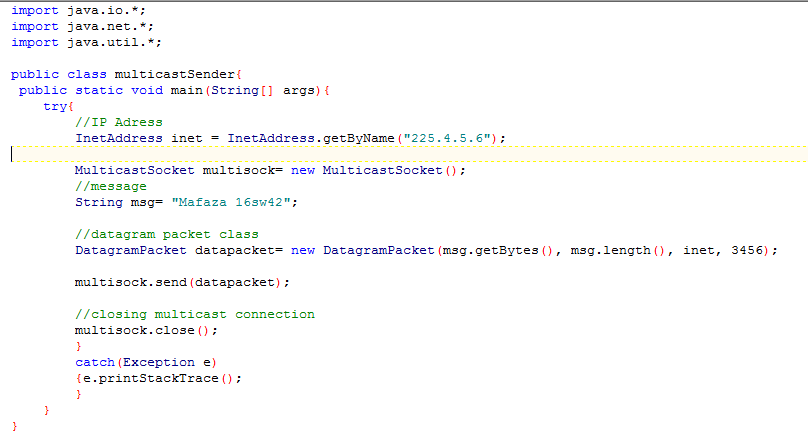


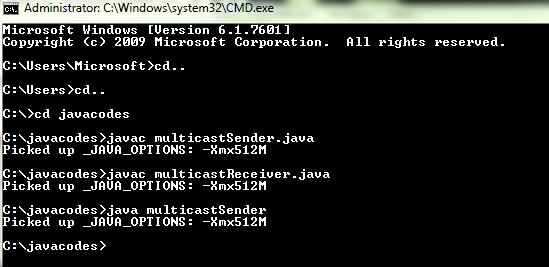
* **UDPServer1.java**



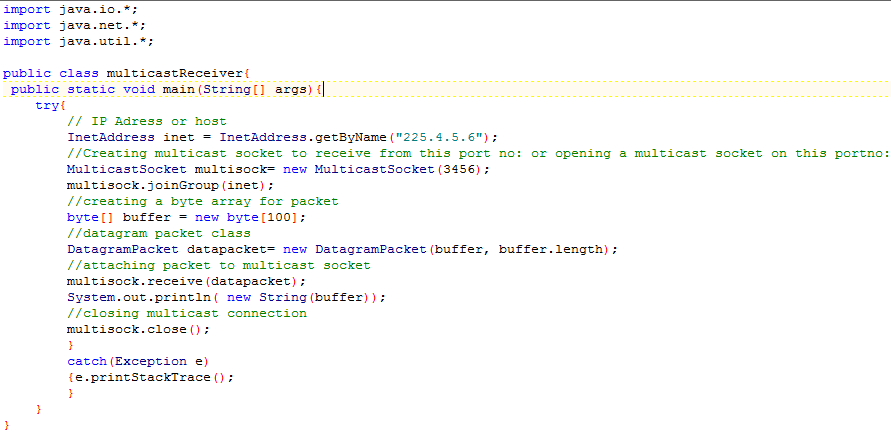


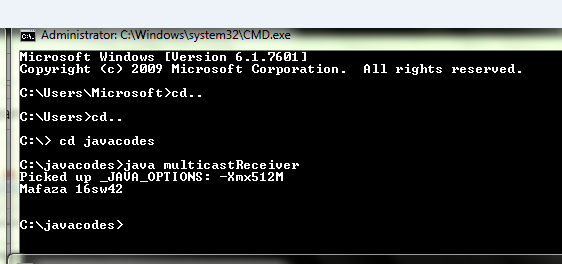
* Implement two simple programs using Java datagram sockets, which broadcasts and multicast your roll number to all or selected network nodes respectively.
* **MulticastSender.java**

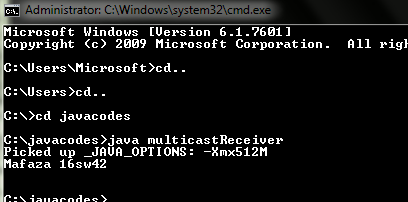




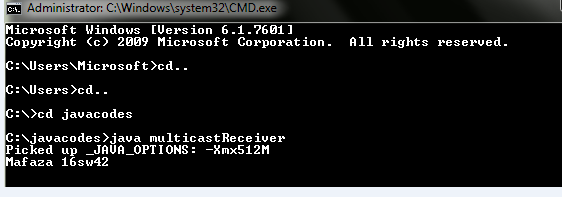
* **MulticastReceiver.java**



* **MulticastReceiver1**
* **MulticastReceiver2**

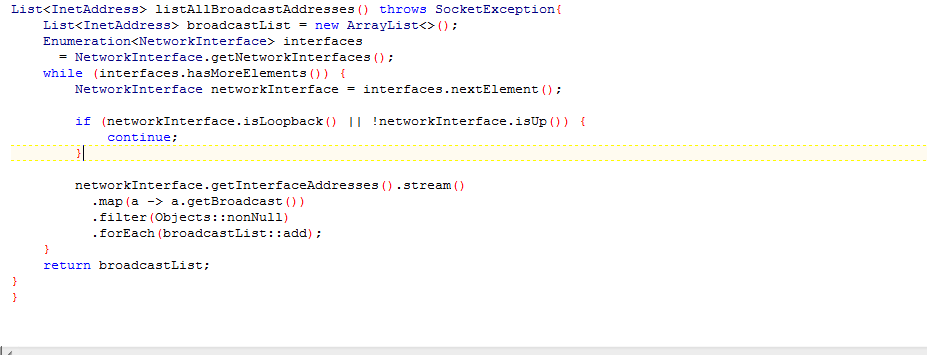


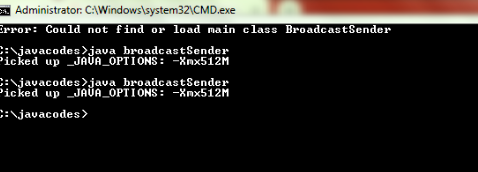
* **MulticastReceiver3**



* **Broadcasting**
* **broadcastSender.java**







* **BroadcastReceiver.java**

