```java
import java.net.*;
import java.io.*;
import java.util.Scanner;

// modified from https://github.com/CMU-Heinz-95702/Project-2-Client-Server
public class EchoClientUDP{
    public static void main(String args[]){
        // args give message contents and server hostname
        System.out.println("The client is running.");
        DatagramSocket aSocket = null;
        try {
            // set to localhost to host on local machine and set port
            InetAddress aHost = InetAddress.getByName("localhost");
            int serverPort = 6789;

            // input server port to use
            System.out.print("Input a server side port number: ");
            Scanner readline = new Scanner(System.in);
            serverPort = readline.nextInt();

            // set up network socket to allow communication between server and client
            aSocket = new DatagramSocket();
            aSocket.setReuseAddress(true);
            String nextLine;// read console input
            BufferedReader typed = new BufferedReader(new InputStreamReader(System.in));
            while ((nextLine = typed.readLine()) != null) {
                byte [] m = nextLine.getBytes(); // convert console in into byte packets
                DatagramPacket request = new DatagramPacket(m,  m.length, aHost, serverPort);
                aSocket.send(request); // send
```

```
29 packet
30                 byte[] buffer = new byte[1000]; //
   create a buffer to receiver server packet
31                 DatagramPacket reply = new
   DatagramPacket(buffer, buffer.length); // format
   for receiving packet
32                 aSocket.receive(reply); // receive
   from socket
33                 String replyString = new String(
   reply.getData()).substring(0,reply.getLength());
   // get proper length of string
34
35                 // halt logic
36                 if(replyString.equalsIgnoreCase("
   halt!")){
37                     System.out.println("Client side
    quitting");
38                     break;
39                 }
40                 else{
41                     System.out.println("Reply: " +
   replyString); // send reply if not halt
42                 }
43             }
44
45             // catch potential exceptions
46         }catch (SocketException e) {System.out.
   println("Socket: " + e.getMessage());
47         }catch (IOException e){System.out.println("
   IO: " + e.getMessage());
48         }finally {if(aSocket != null) aSocket.close
   ();}
49     }
50 }
```

```java
import java.net.*;
import java.io.*;
import java.util.Scanner;

// modified from https://github.com/CMU-Heinz-95702/Project-2-Client-Server
public class EchoServerUDP{
    public static void main(String args[]){
        System.out.println("The server is running."); // lab instructions
        DatagramSocket aSocket = null;
        byte[] buffer = new byte[1000];  // set up packet buffer for client message
        try{
            // set up ports
            System.out.print("Input a server port number to listen on: ");
            Scanner readline = new Scanner(System.in);
            int serverPort = readline.nextInt(); // convert to int

            // set up sockets to receive client packets
            aSocket = new DatagramSocket(serverPort);
            aSocket.setReuseAddress(true);

            DatagramPacket request = new DatagramPacket(buffer, buffer.length); // syntax for buffer
            while(true){ // loop to continue until 'halt!' is sent
                aSocket.receive(request); // receive request, and format a reply
                DatagramPacket reply = new DatagramPacket(request.getData(),
                        request.getLength(), request.getAddress(), request.getPort()); // syntax for reply from request
                String requestString = new String(
```

```
26 request.getData()).substring(0,request.getLength
   ()); // proper length
27                 System.out.println("Echoing: "+
   requestString);
28                 aSocket.send(reply); // send back
   the reply
29
30                 // halt logic
31                 if(requestString.equalsIgnoreCase("
   halt!")){
32                     System.out.print("Server side
   quitting");
33                     break;
34                 }
35             }
36             // catch potential exceptions
37         }catch (SocketException e){System.out.
   println("Socket: " + e.getMessage());
38         }catch (IOException e) {System.out.println(
   "IO: " + e.getMessage());
39         }finally {if(aSocket != null) aSocket.close
   ();}
40     }
41 }
```