Version 1.7

Distributed Systems – Elliott
Sample code for InetServer and InetClient

These running programs are provided as a guide. **TYPE THEM IN** and get them running on your own machine. Along the way you will likely make some mistakes in transcription that have to be repaired. Once you have them running, you can play with them by making modifications.

The byte-compiled .class files from these programs are also available in the course directories, so that you can use, e.g., the existing client to test your own server, or the existing server to test your own client.

I do not require you to type in, and get running, or turn in, the standalone InetAddresser program. However this is a first step in getting this program running, and, once you have typed it in and gotten it running, then much of it will be useful in the InetServer and InetClient programs via copy/paste.

I recommend that you first make sure that Java and TCP/IP are running correctly on your machine by loading and running the InetAddresser, and the InetServer and InetClient byte-compiled .class files. These should work "out of the box." You may find that you have to turn off, or play with, your firewall. If you are concerned about security you can unplug your local home network from the internet, but leave the connections between your local home machines active.

Then, if you are advanced programmer you can write your own java multi-threaded server and client. Otherwise **TYPE IN THE PROGRAMS** and get them running. This is an excellent way to learn. If you are clever you can probably figure a way to scan the text from the .pdf file, but this will not help you to learn the material and I strongly do NOT recommend it.

Use your own variable names, and alter the logic as you see fit, as long as it runs, and has comments.

Once you have the client and server running, CHANGE THE PORT number to something higher and announce the port when the server starts up. Run your client and server on different machines if you have more than one available. Run several clients at once with the single server.

Tip: start small and get something running first, then gradually add functionality, and complexity. But, always keep your program running.

These programs are explicitly given without many comments. This is intentional because I want YOU to figure out exactly what is going on, and ***write your own comments***, which is an important part of your grade.

Good luck. Have fun.

-CE

```
/** File is: InetAddresser.java, Version 1.7

A standalone InetAddresser. Elliott, after Hughes, Shoffner, Winslow

This will not run unless TCP/IP is loaded on your machine.
---------------------------------------------------------------------*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

public class InetAddresser {
  public static void main (String args[]) {
    printLocalAddress ();

    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

    try {
      String name;
      do {
        System.out.print("Enter a hostname or an IP address, (quit) to end: ");
        System.out.flush ();
        name = in.readLine ();
        if (name.indexOf("quit") < 0)
          printRemoteAddress (name);
      } while (name.indexOf("quit") < 0);
      System.out.println ("exit");
    } catch (IOException x) {x.printStackTrace ();}
  }

  static void printLocalAddress () {
    try {
      System.out.print("Clark Elliott's INET addresser program, 1.7\n");
      InetAddress me = InetAddress.getLocalHost ();
      System.out.println("My local name is:       " + me.getHostName ());
      System.out.println("My local IP address is: " + toText(me.getAddress ()));
    } catch (UnknownHostException x) {
      System.out.println ("I appear to be unknown to myself. Firewall?:");
      x.printStackTrace ();
    }
  }


  static void printRemoteAddress (String name) {
    try {
      System.out.println ("Looking up " + name + "...");
      InetAddress machine = InetAddress.getByName (name);
      System.out.println ("Host name : " + machine.getHostName ());
      System.out.println ("Host IP : " + toText (machine.getAddress ()));
    } catch (UnknownHostException ex) {
      System.out.println ("Failed to lookup " + name);
    }
  }

  static String toText (byte ip[]) {  /* Make portable for 128 bit format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
```

```
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }
}
```

```
/** File is: InetServer.java, Version 1.5

A multithreaded server for InetClient. Elliott, after Hughes, Shoffner, Winslow

This will not run unless TCP/IP is loaded on your machine.
-------------------------------------------------------------------------*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

class Worker extends Thread {    // Class definition
  Socket sock;                   // Class member, socket, local to Worker.
  Worker (Socket s) {sock = s;}  // Constructor, assign arg s to local sock

  public void run(){
    // Get I/O streams in/out from the socket:
    PrintStream out = null;
    BufferedReader in = null;
    try {
      in = new BufferedReader
        (new InputStreamReader(sock.getInputStream()));
      out = new PrintStream(sock.getOutputStream());
      // Note that this branch might not execute when expected:
      if (InetServer.controlSwitch != true){
        System.out.println
          ("Listener is now shutting down as per client request.");
        out.println("Server is now shutting down. Goodbye!");
      }
      else try {
        String name;
        name = in.readLine ();
        if (name.indexOf("shutdown") > -1){
          InetServer.controlSwitch = false;
          System.out.println("Worker has captured a shutdown request.");
          out.println("Shutdown request has been noted by worker.");
          out.println("Please send final shutdown request to listener.");
        }
        else{
          System.out.println("Looking up " + name);
          printRemoteAddress(name, out);
        }
      } catch (IOException x) {
        System.out.println("Server read error");
        x.printStackTrace ();
      }
      sock.close(); // close this connection, but not the server;
    } catch (IOException ioe) {System.out.println(ioe);}
  }

  static void printRemoteAddress (String name, PrintStream out) {
    try {
      out.println("Looking up " + name + "...");
      InetAddress machine = InetAddress.getByName (name);
      out.println("Host name : " + machine.getHostName ());
      out.println("Host IP : " + toText (machine.getAddress ()));
    } catch(UnknownHostException ex) {
```

```
      out.println ("Failed in atempt to look up " + name);
    }
  }

  // Not interesting to us:
  static String toText (byte ip[]) {  /* Make portable for 128 bit format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }
}

public class InetServer {

  public static boolean controlSwitch = true;

  public static void main(String a[]) throws IOException {
    int q_len = 6; /* Number of requests for OpSys to queue */
    int port = 1565;
    Socket sock;

    ServerSocket servsock = new ServerSocket(port, q_len);

    System.out.println
      ("Clark Elliott's Inet server starting up, listening at port 1565.\n");
    while (controlSwitch) {
      sock = servsock.accept(); // wait for the next client connection
      if (controlSwitch) new Worker(sock).start(); // Spawn worker to handle it
      // Uncomment to see shutdown oddity:
      // try{Thread.sleep(10000);} catch(InterruptedException ex) {}
    }
  }
}
```

```
/** File is: InetClient.java, Version 1.7

A client for InetServer. Elliott, after Hughes, Shoffner, Winslow

This will not run unless TCP/IP is loaded on your machine.
-------------------------------------------------------------------------*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

public class InetClient{
  public static void main (String args[]) {
    String serverName;
    if (args.length < 1) serverName = "localhost";
    else serverName = args[0];

    System.out.println("Clark Elliott's Inet Client, 1.7.\n");
    System.out.println("Using server: " + serverName + ", Port: 1565");
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    try {
      String name;
      do {
        System.out.print
          ("Enter a hostname or an IP address, (quit) to end: ");
        System.out.flush ();
        name = in.readLine ();
        if (name.indexOf("quit") < 0)
          getRemoteAddress(name, serverName);
      } while (name.indexOf("quit") < 0); // quitSmoking.com?
      System.out.println ("Cancelled by user request.");
    } catch (IOException x) {x.printStackTrace ();}
  }

  static String toText (byte ip[]) {  /* Make portable for 128 bit format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }

  static void getRemoteAddress (String name, String serverName){
    Socket sock;
    BufferedReader fromServer;
    PrintStream toServer;
    String textFromServer;

    try{
      /* Open our connection to server port, choose your own port number.. */
      sock = new Socket(serverName, 1565);

      // Create filter I/O streams for the socket:
      fromServer =
        new  BufferedReader(new InputStreamReader(sock.getInputStream()));
      toServer   = new PrintStream(sock.getOutputStream());
```

```
      // Send machine name or IP address to server:
      toServer.println(name); toServer.flush();

      // Read two or three lines of response from the server,
      // and block while synchronously waiting:
      for (int i = 1; i <=3; i++){
        textFromServer = fromServer.readLine();
        if (textFromServer != null) System.out.println(textFromServer);
      }

      sock.close();
    } catch (IOException x) {
      System.out.println ("Socket error.");
      x.printStackTrace ();
    }
  }
}
```