/\*--------------------------------------------------------

1. Name / Date: Jessica Bender / Version 1: Febuary 7th 2021 11:58am

2. Java version used, if not the official version for the class:

My Java:

java version "9.0.4"

Java(TM) SE Runtime Environment (build 9.0.4+11)

Java HotSpot(TM) 64-Bit Server VM (build 9.0.4+11, mixed mode)

3. Precise command-line compilation examples / instructions:

> javac MiniWebserver.java

4. Precise examples / instructions to run this program:

5. List of files needed for running the program.

a. MiniWebserver.java

6. Notes:

----------------------------------------------------------\*/

/\*Explain how MIME-types are used to tell the browser what data is coming.

\*/

/\*Explain how you would return the contents of requested files (web pages) of type HTML (text/html)

\*/

/\*Explain how you would return the contents of requested files (web pages) of type TEXT (text/plain)

\*/

import java.io.\*; //imports all directories located in java.io

import java.net.\*; //imports all directories located in java.io

class ListenWorker extends Thread { //start of ListenWorker class

Socket sock; //new Socket called sock

ListenWorker (Socket s) {sock = s;}

public void run(){ //start of run

PrintStream out = null; // initialize a PrintStream called out and sets its value to null

BufferedReader in = null; //initialize a BufferedReader called in and sets its value to null

try {//Start try statement. If try doesnt work moved down to catch statement

out = new PrintStream(sock.getOutputStream()); //makes a new PrintStream and assigns it to out and gets the OutputStream from the Socket sock

in = new BufferedReader //makes a new BufferedReader and assigns it to in

(new InputStreamReader(sock.getInputStream())); //gets the InputStream from the Socket sock and makes a new InputStreamReader and puts in inside the BufferedReader in

System.out.println("Sending the HTML Reponse now: " +

Integer.toString(WebResponse.i) + "\n" );

String HTMLResponse = "<html> <h1> Hello Browser World! " +

Integer.toString(WebResponse.i++) + "</h1> <p><p> <hr> <p>";

out.println("HTTP/1.1 200 OK");

out.println("Connection: close");

out.println("Content-Length: 400");

out.println("Content-Type: text/html \r\n\r\n");

out.println(HTMLResponse);

for(int j=0; j<6; j++){

out.println(in.readLine() + "<br>\n");

} //closes for statement

out.println("</html>"); //prints message to PrintStream out

sock.close(); // close the connection to the Socket sock, but does not colose the connecton to the server;

} //closes try statement

catch (IOException x) { //start catch. Catches an IOException if the try fails.

System.out.println("Error: Connetion reset. Listening again...");//prints message if try fails

}//closes catch statement

}//closes run statement

}//closes ListenWorker class

public class MiniWebserver { //start of MiniWebserver class. This is a public class.

static int i = 0; //new int called i and sets it to 0

public static void main(String a[]) throws IOException { //start of main void method

int q\_len = 6; //sets q\_len to 6

int port = 2540; //sets port to 2540

Socket sock; //new Socket called sock

ServerSocket servsock = new ServerSocket(port, q\_len); //new ServerSocket called servsock. also sets it equal to the port and q\_len that we se above

System.out.println("Jess Bender's WebResponse running at " + port + ".");//Prints statement. Changed code to have my name and changed the statement so that the variable port would be printed and it was not hardcoded in the string statement.

System.out.println("Point Firefox browser to http://localhost:2540/abc.\n");//prints message

while (true) { //starts wile true statement to look for the next client connection

sock = servsock.accept();

new ListenWorker (sock).start();// starts new ListenWorker

}//clsoes while true

}//closes main void statement

}//closes miniwebserver class.