Jimmy Tran

Panther ID: 002-11-0395

Bourgeois

17 February 2017

Assignment 2

**Problem 1 Source Code:**

**Serverside Code:**

package TCPServer;

import java.io.\*;

import java.net.\*;

class TCPServer {

public static void main(String argv[]) throws Exception {

String clientSentence;

String capitalizedSentence;

String query=" Please input next string: ";

String end=" String limit reached.";

ServerSocket welcomeSocket = new ServerSocket(6789);

boolean active=true;

int i=0;

Socket connectionSocket = welcomeSocket.accept();

BufferedReader inFromClient = new BufferedReader(new InputStreamReader(connectionSocket.getInputStream()));

DataOutputStream outToClient = new DataOutputStream(connectionSocket.getOutputStream());

System.out.println("Server is running...");

while(active && i<9) {

clientSentence = inFromClient.readLine();

if(clientSentence.equalsIgnoreCase("exit")){

outToClient.writeBytes("Exit requested. Connection Closed."+'\n');

welcomeSocket.close();

i=10;

}

System.out.println("Received: " + clientSentence);

capitalizedSentence = clientSentence.toUpperCase();

outToClient.writeBytes("From Server: "+capitalizedSentence+query+'\n');

i++;

}

if(i==9){

clientSentence = inFromClient.readLine();

if(clientSentence.equalsIgnoreCase("exit")){

outToClient.writeBytes("Exit requested. Connection Closed."+'\n');

welcomeSocket.close();

}

System.out.println("Received: " + clientSentence);

capitalizedSentence = clientSentence.toUpperCase();

outToClient.writeBytes("From Server: "+capitalizedSentence+end+'\n');

i++;

}

welcomeSocket.close();

System.out.println("Connection Closed.");

}

}

**Clientside Code:**

package TCPClient;

import java.io.\*;

import java.net.\*;

import java.util.concurrent.TimeUnit;

class TCPClient {

public static void main(String argv[]) throws Exception {

String sentence;

String modifiedSentence;

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

Socket clientSocket = new Socket("localhost", 6789);

DataOutputStream outToServer = new DataOutputStream(clientSocket.getOutputStream());

BufferedReader inFromServer = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

int i=0;

System.out.println("Please input a string: ");

while(i<10){

sentence = inFromUser.readLine();

outToServer.writeBytes(sentence + '\n');

modifiedSentence = inFromServer.readLine();

System.out.println(modifiedSentence);

if(sentence.equalsIgnoreCase("exit")){

i=10;

}

i++;

}

//delay so that server closes socket before client to prevent error

TimeUnit.SECONDS.sleep(1);

clientSocket.close();

System.out.println("Connection Closed.");

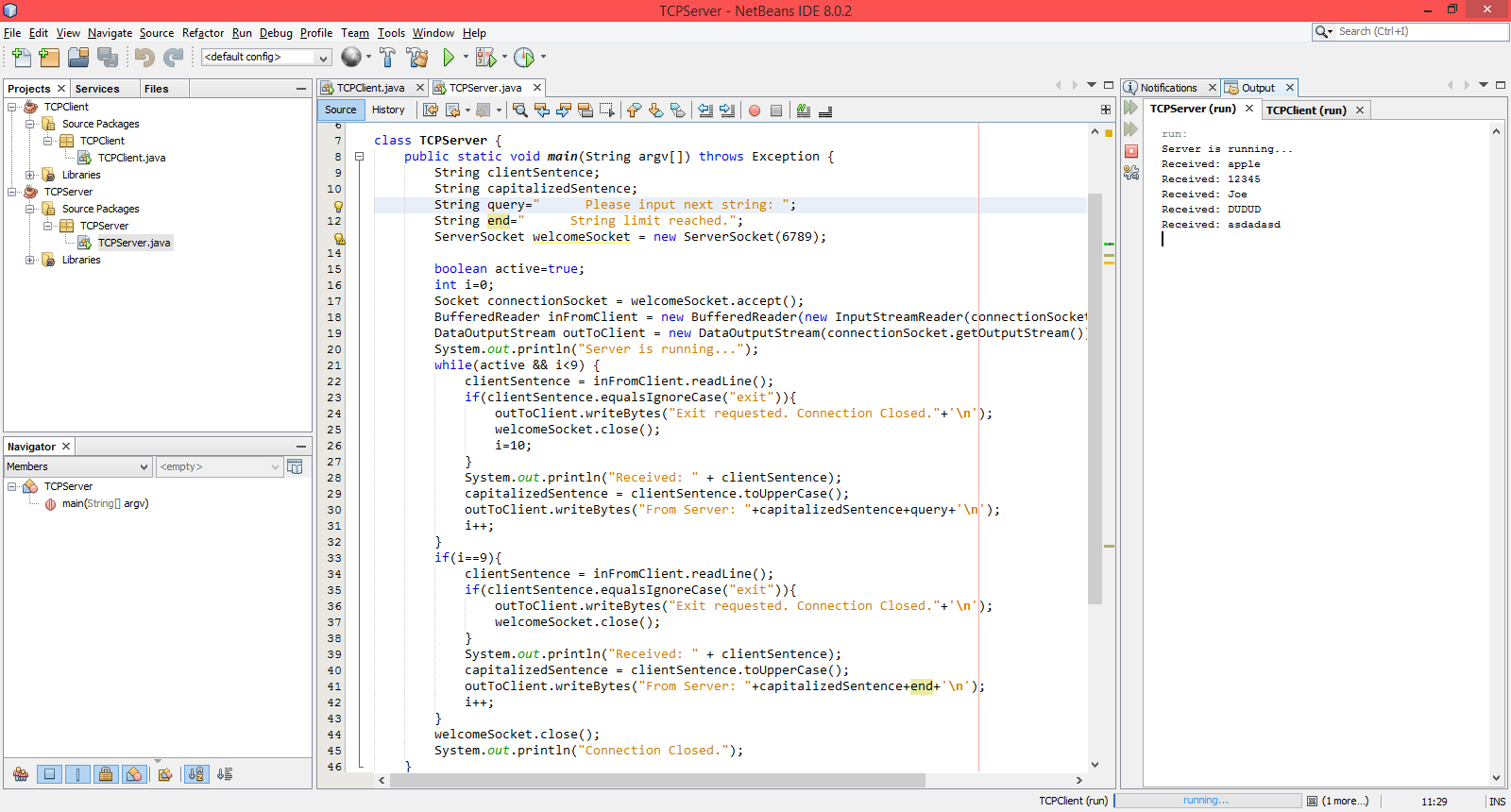
}

}

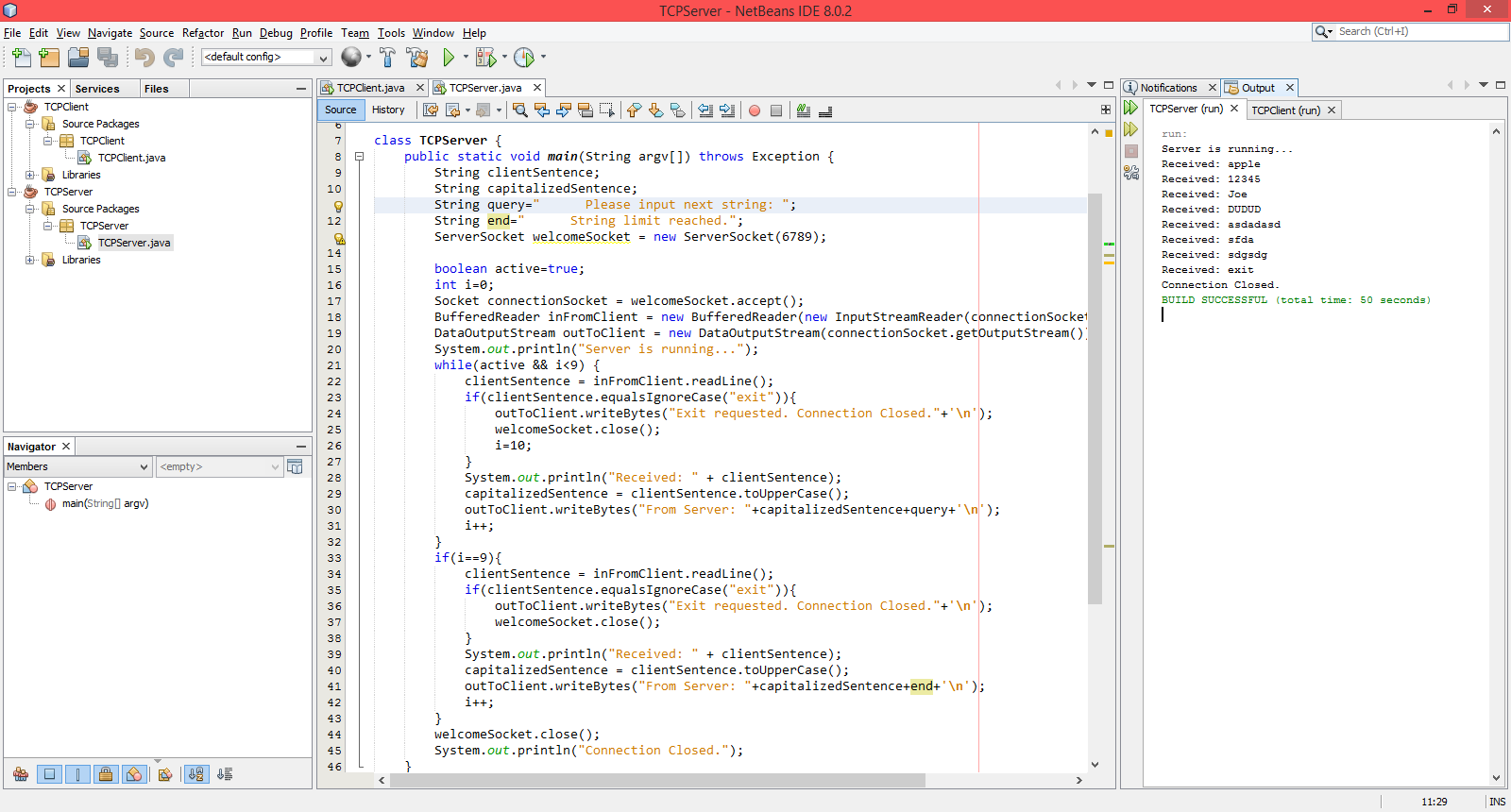
**Screenshots:**

**Serverside:**

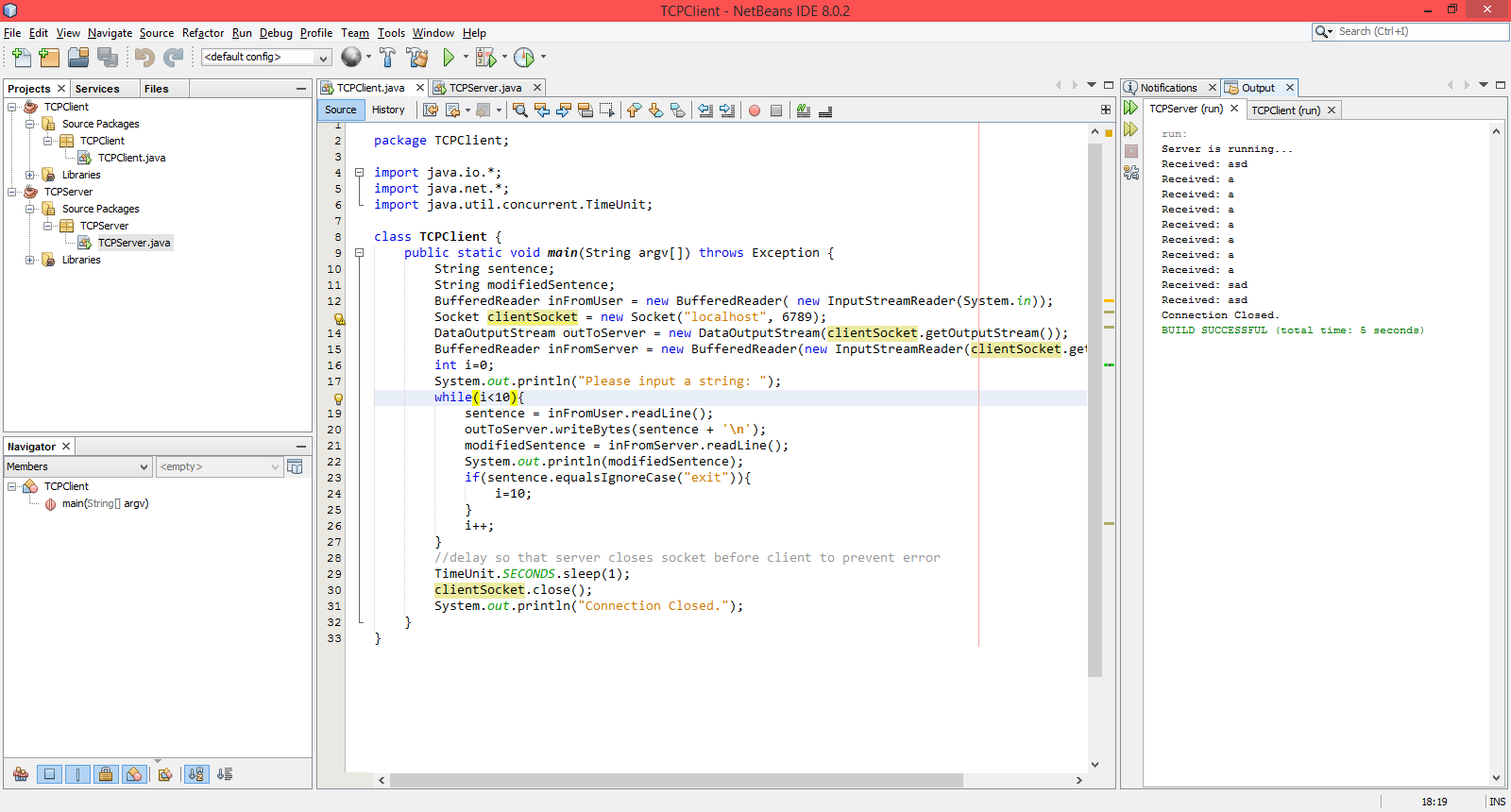
**Five strings in:**



**Exit requested:**

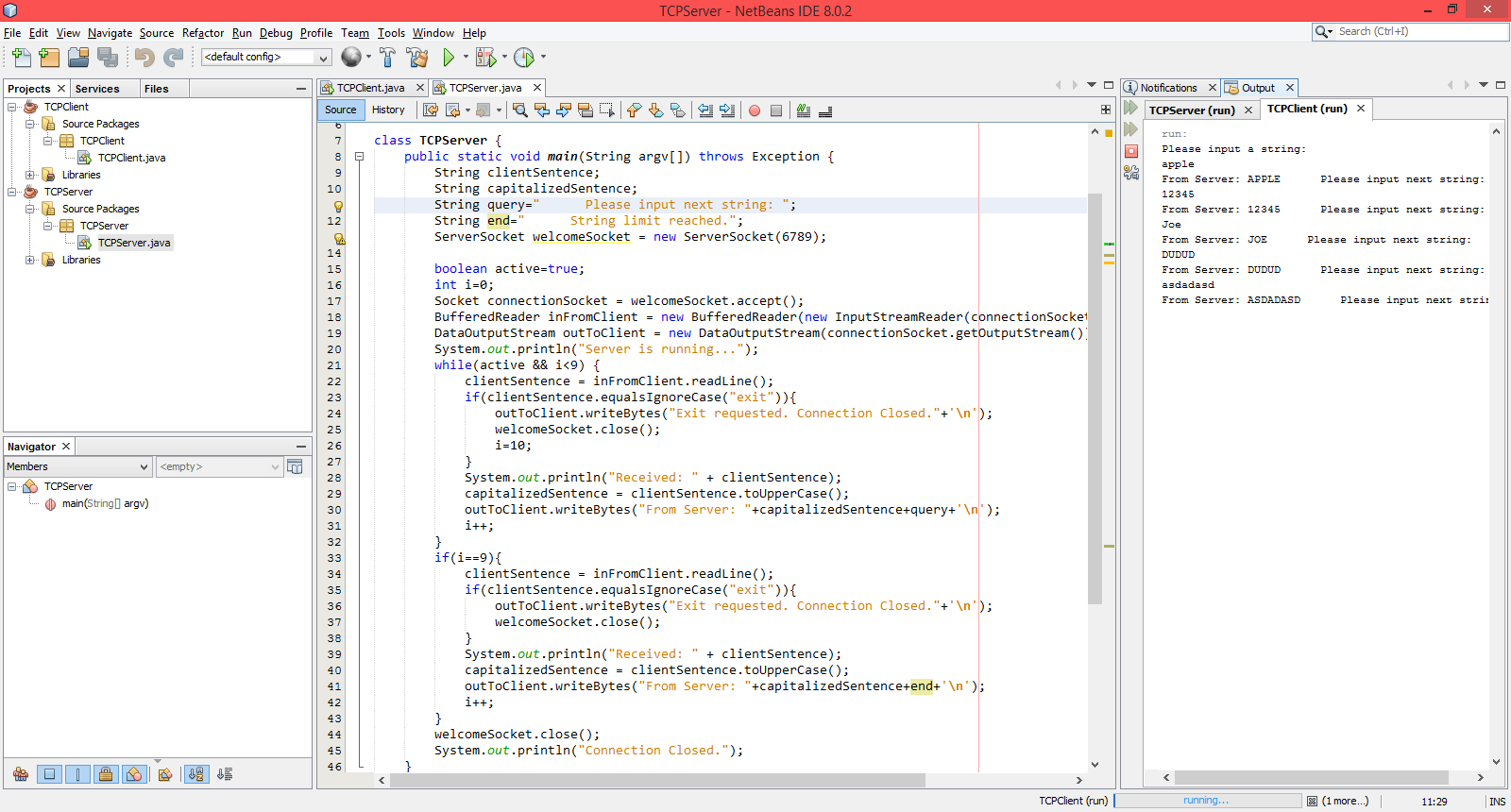


**Ten strings in:**

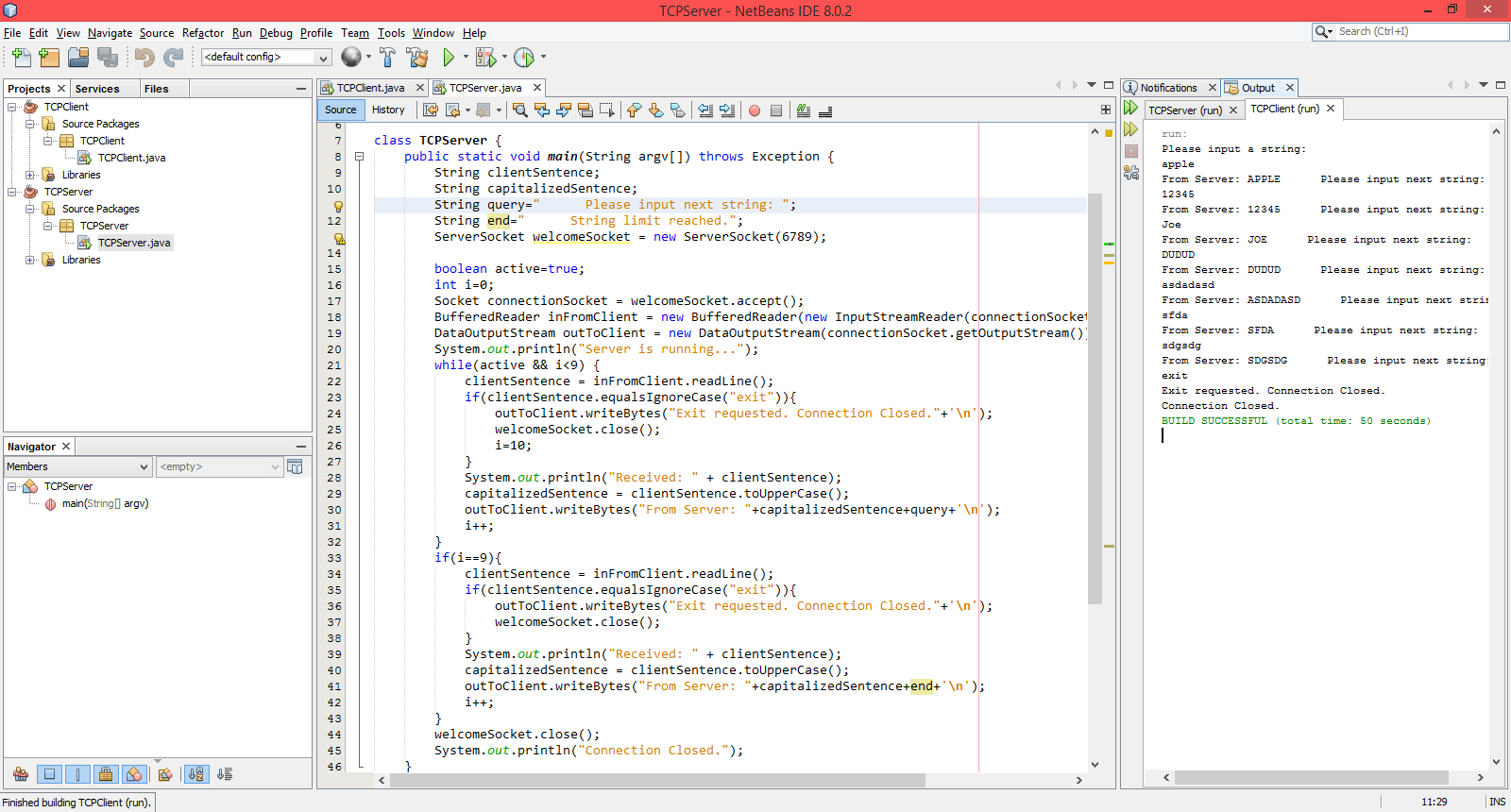


**Clientside:**

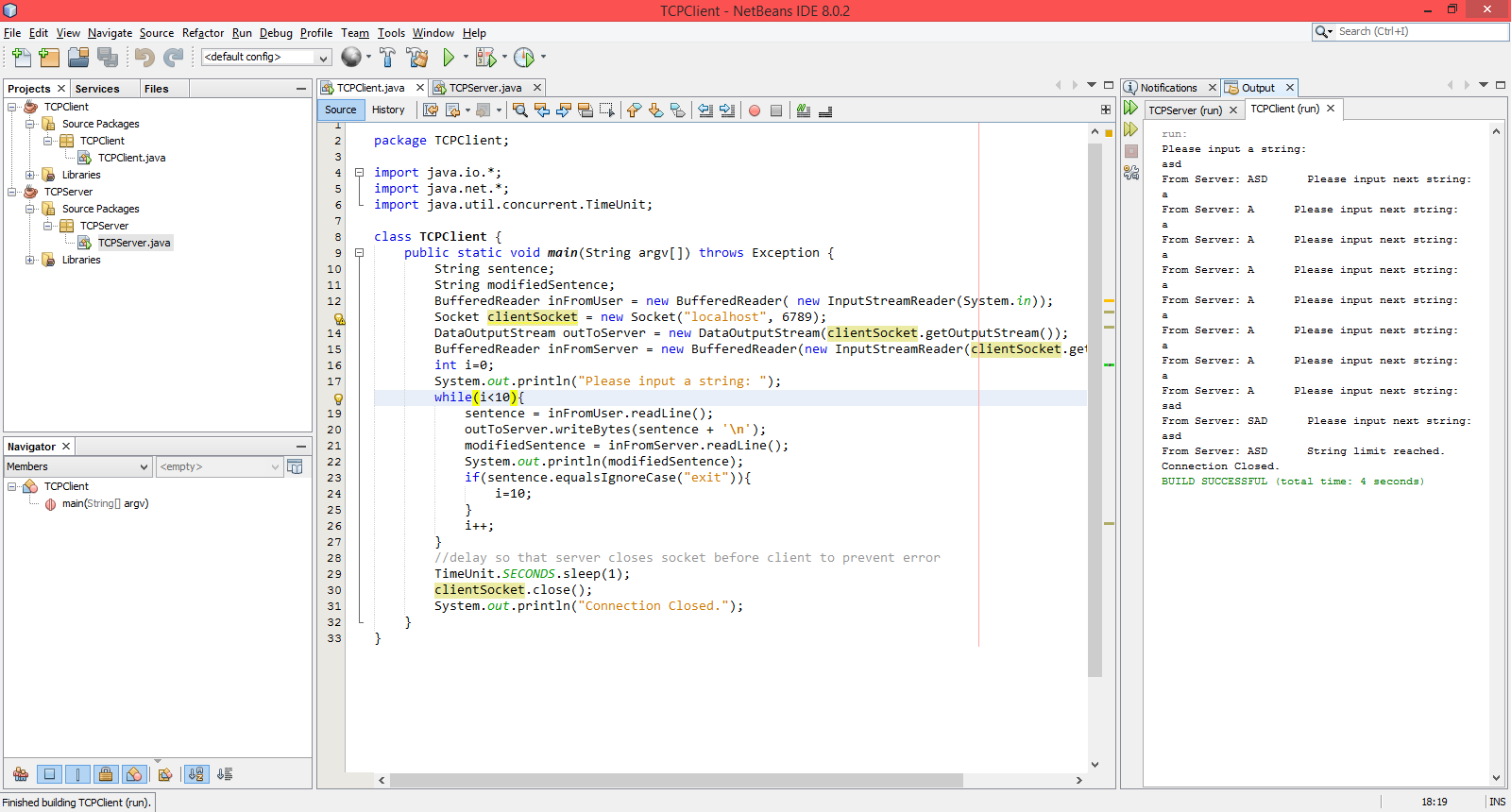
**Five strings in:**



**Exit requested:**



**Ten strings in:**



**Problem 2 Source Code:**

package HTTPServer;

import java.io.\*;

import java.net.\*;

import java.net.ServerSocket;

public class HTTPServer {

public static void main(String[] args) throws Exception {

ServerSocket server = new ServerSocket(8000);

System.out.println("Listening for connection on port 8000...");

Boolean serverStatus=true;

FileInputStream fileIn;

OutputStream httpResponse;

String fileName;

while (serverStatus){

//create socket on client request

Socket client = server.accept();

BufferedReader input = new BufferedReader(new InputStreamReader(client.getInputStream()));

DataOutputStream output = new DataOutputStream(client.getOutputStream());

try{

//receive http request from client

fileName = input.readLine();

System.out.println("HTTP Request=" + fileName);

try{

//parse request and try to get file requested

//separate 1st header elements into array

String[]data=fileName.split(" ");

fileName=data[1];

//find and print file name

fileName=fileName.substring(1);

System.out.println("File Name: "+fileName);

File toSend = new File(fileName);

String location=toSend.getAbsolutePath();

System.out.println("File Location: "+location);

//create http response to send file to client

fileIn = new FileInputStream(location);

httpResponse = client.getOutputStream();

//send over tcp

httpResponse.write(fileIn.read());

httpResponse.flush();

System.out.println("File Successfully Delivered.");

}

//catch for missing file

catch (FileNotFoundException e){

String missing="File Not Found";

System.err.println(missing);

String errResponse = "HTTP/1.1 404 OK\r\n\r\n"+missing;

client.getOutputStream().write(errResponse.getBytes("UTF-8"));

}

}

//catch all just in case

catch (Exception e){

System.err.println("Server error.");

}

}

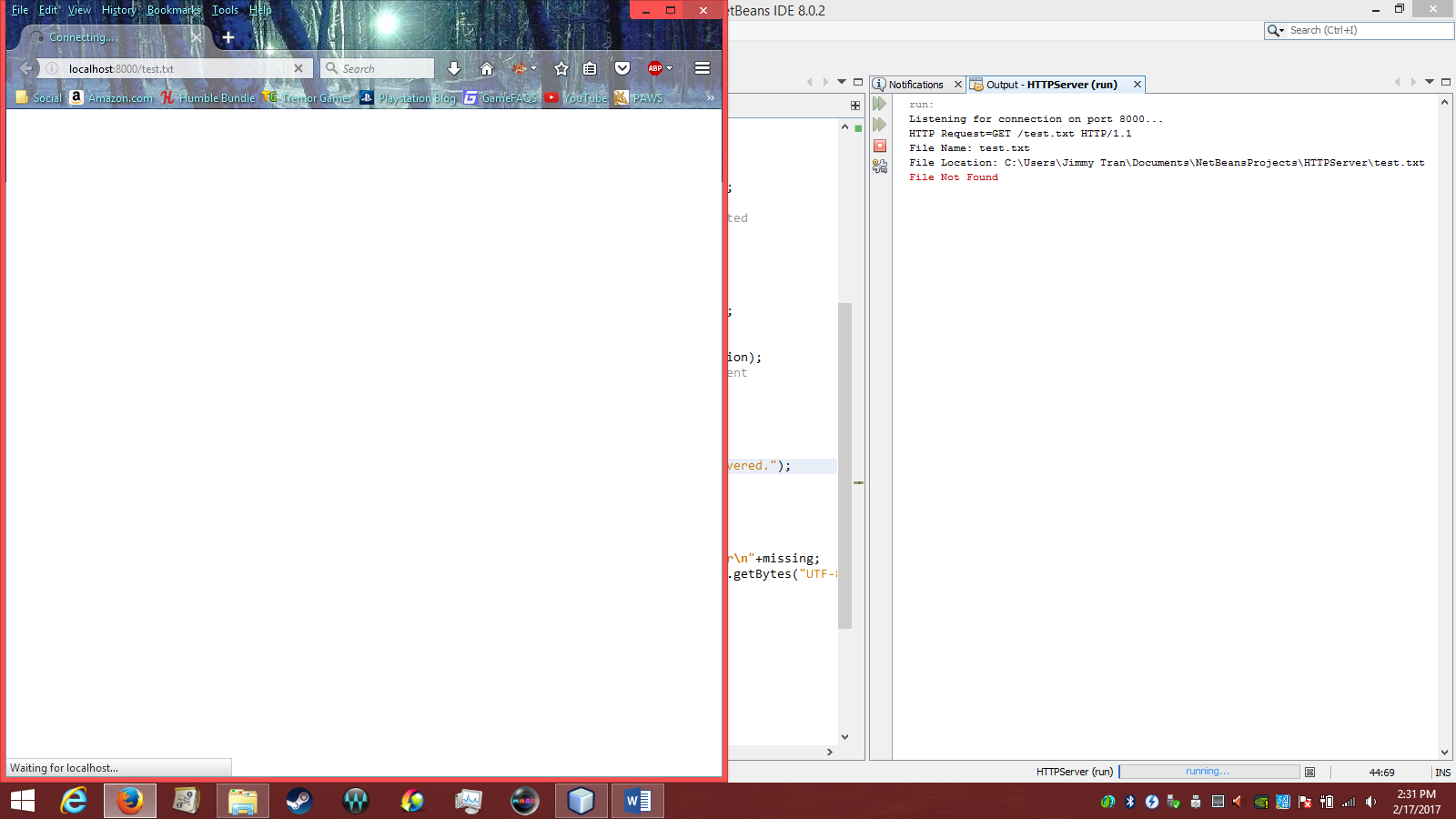
}

}

**Screenshots:**

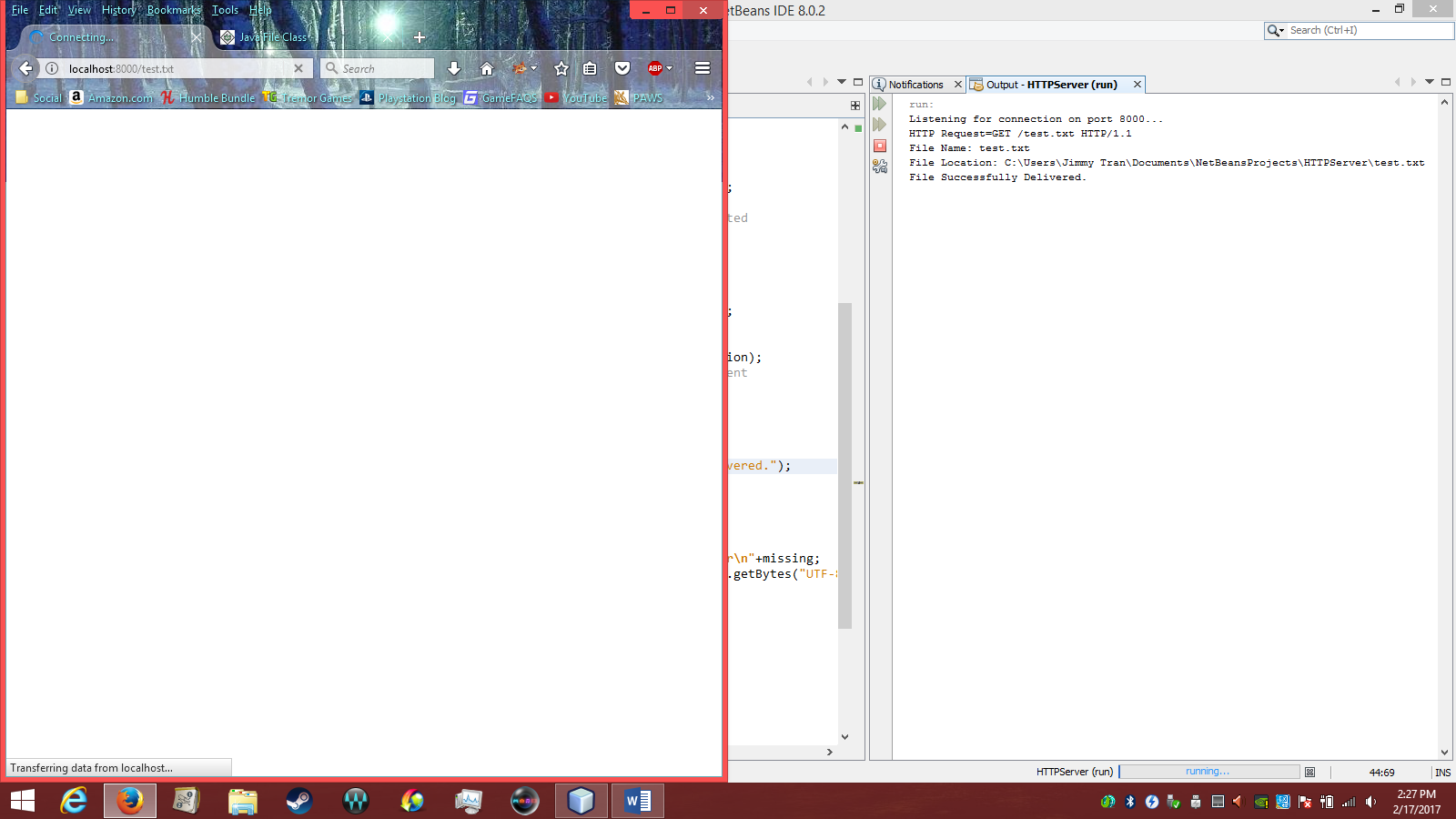
**File Not Found:**

If the file doesn’t exist in the given location when the server looks for it, the File Not Found error results as below. I couldn’t figure out how to get the browser to show its error page, but my program did send the correct 404 error message as evident in my source code.



**File Found and Sent:**

File is found and sent to browser, although it seems my browser can’t actually open or read the text file. It has been sent though, according to the output window. Otherwise, an error would have occurred. Oddly enough, the program looked two levels above where my HTTPServer.java file was for test.txt. Could probably just specify any location though since my program finds the absolute path name anyway - I think.



To further confirm: I couldn’t move the file while the browser was still accessing it.

