ELEC 5220 Information Networks and Technology  
Lab 7 Report

Robert Skelton

# Abstract

The goal of this lab was to learn how to implement a TCP and UDP connection Client and Server using Java. The Server would be able to accept multiple Clients, and would be able to print out the details of the IP address and the port of the connection. Each step of the lab had some different requirement, all of which are detailed below.

# Introduction

The first problem I had was before I had even started. Last week, I enabled a VPN for my father to have American Internet while outside of the country. Apparently, that makes XAMPP not allowed to use port 443, so I deleted the VPN option, and continued on with the lab. I performed the entirety of this lab on my Macbook Air, running OS X 10.10. Code was editing using Sublime Text 3, and I used the zsh shell in a Terminal application to compile and run Java programs. We had four sections of this lab, and were instructed not to run the IPv6 portion of the Lab from Dr. Wu’s SCP site.

# Design

**TCP-Echo**

This step consisted of running and modifying a little bit of Client and Server code that passes a line of text from Client to Server. This would be useful in something such as an Instant Messaging Client.

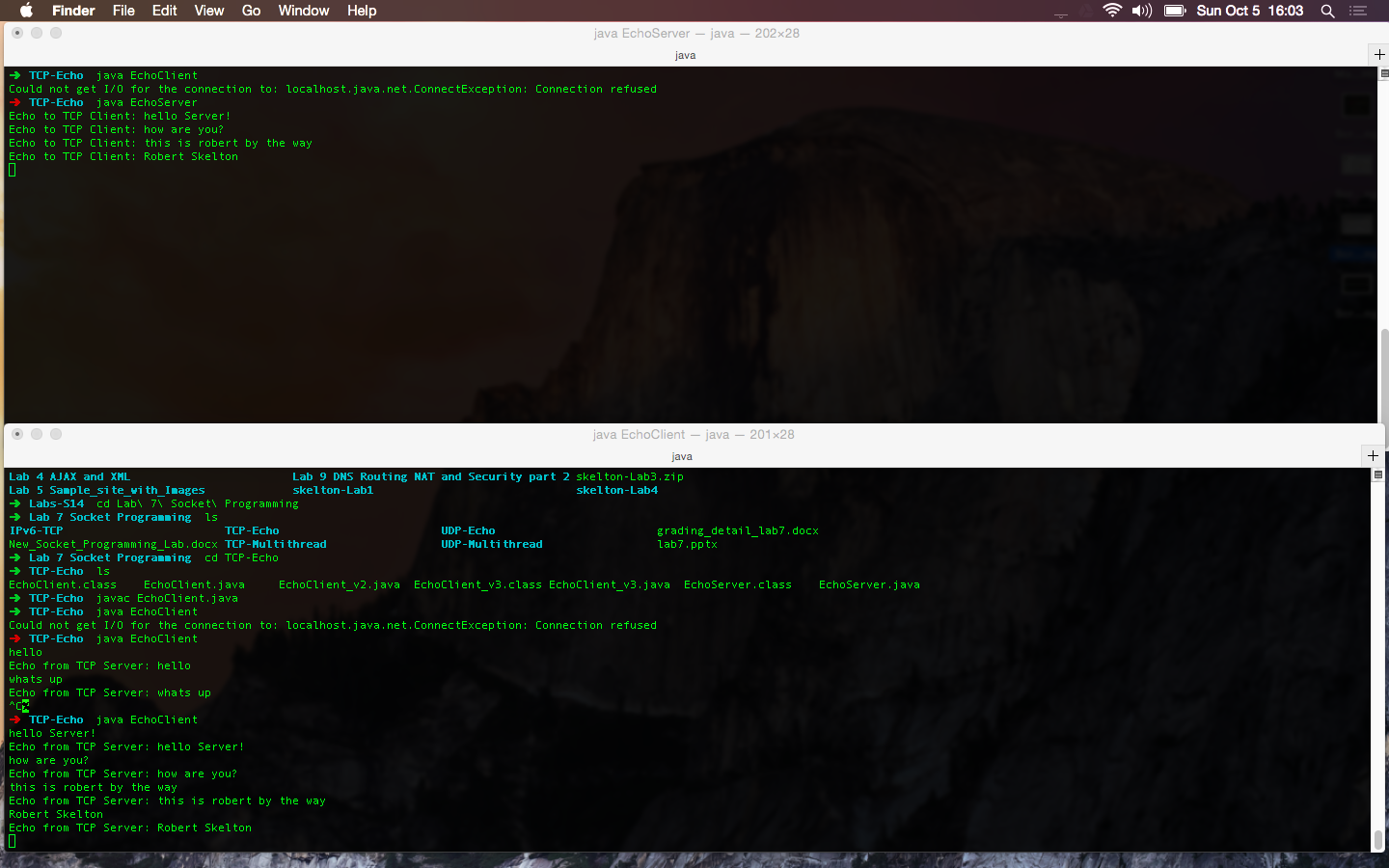


Figure 1. Running Echo Server and Client in Java.

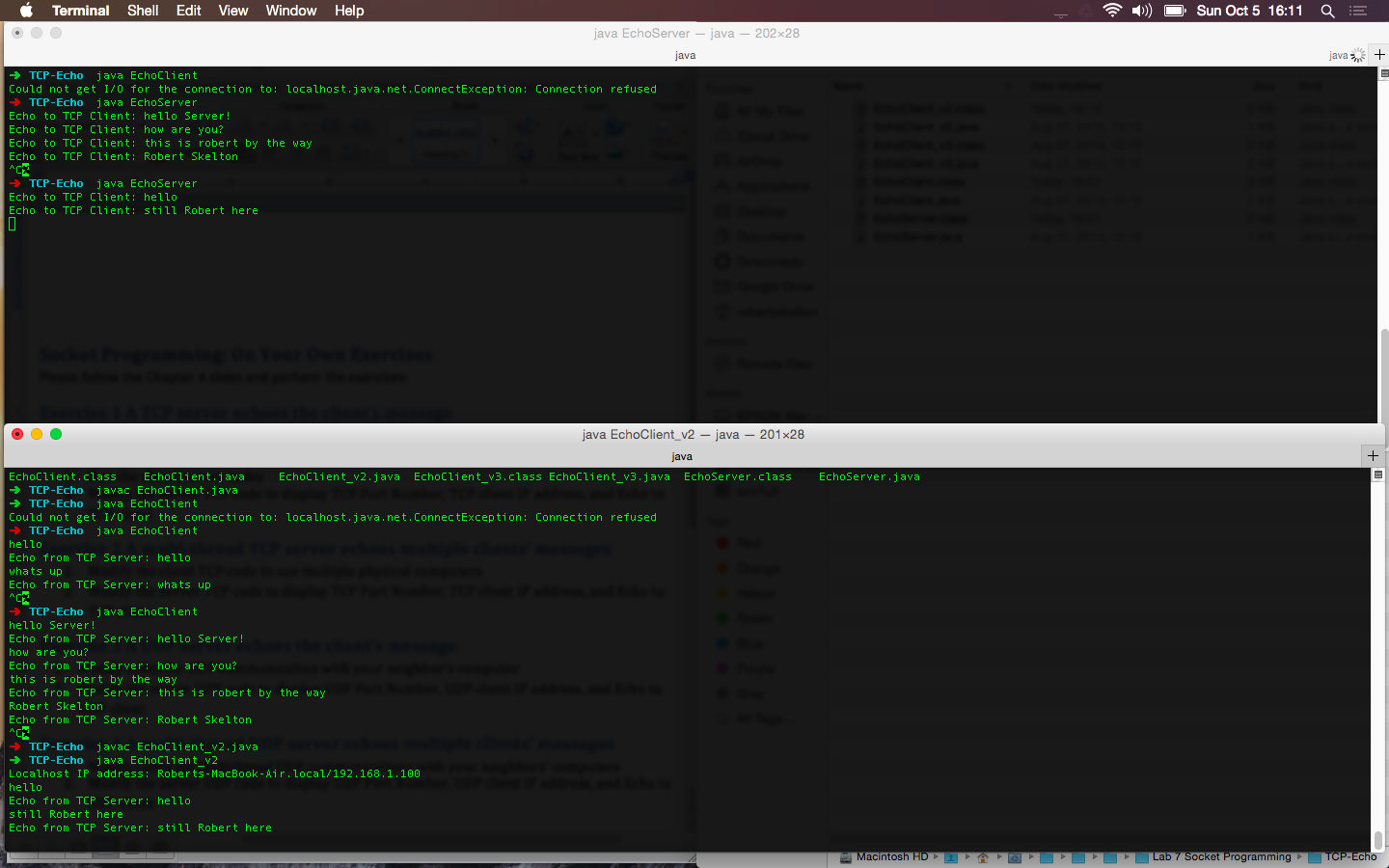


Figure 2. Running Echo Client V2 and Echo Sever in Java.

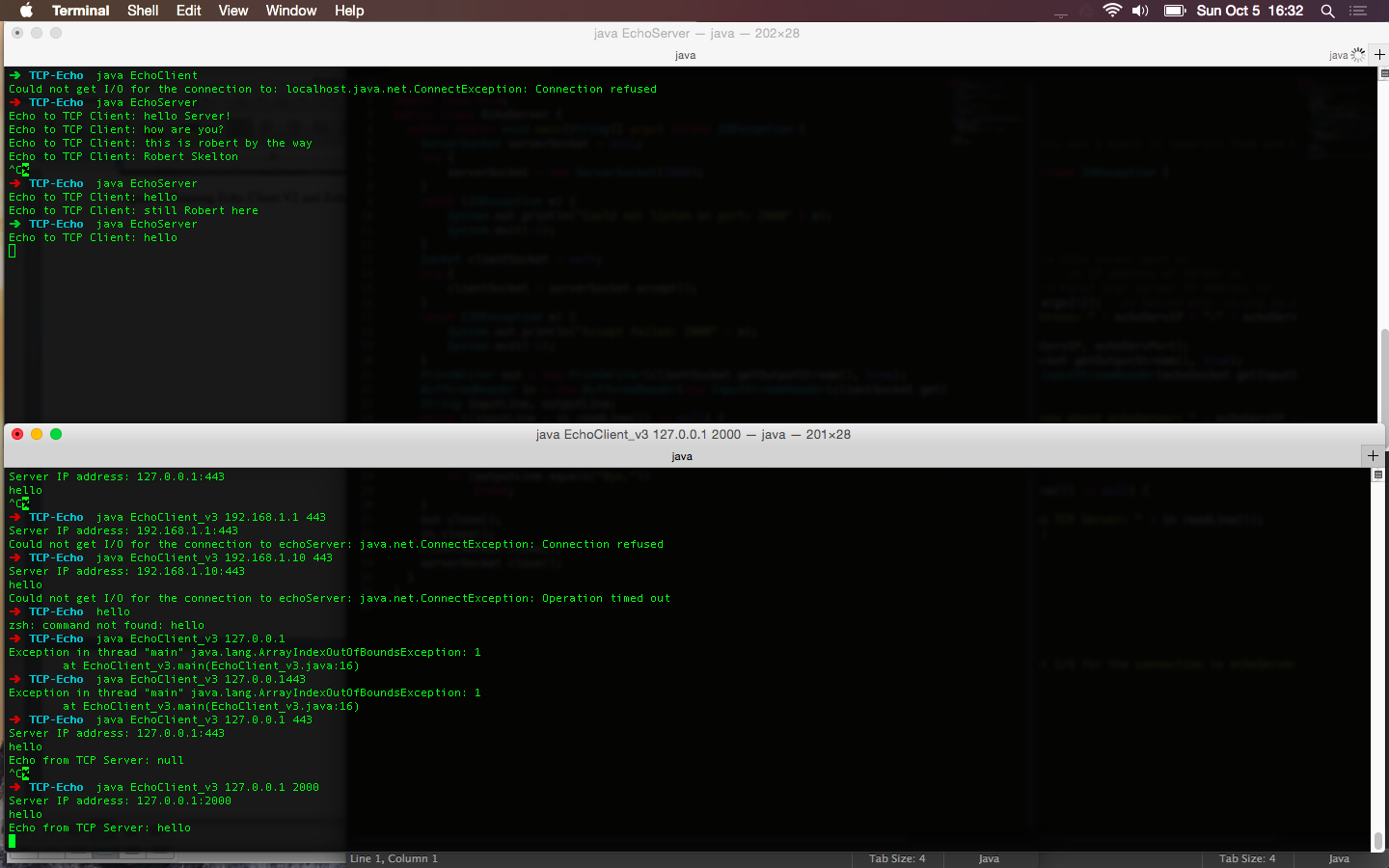


Figure 3. Running Echo Client v3 and Server in Java.

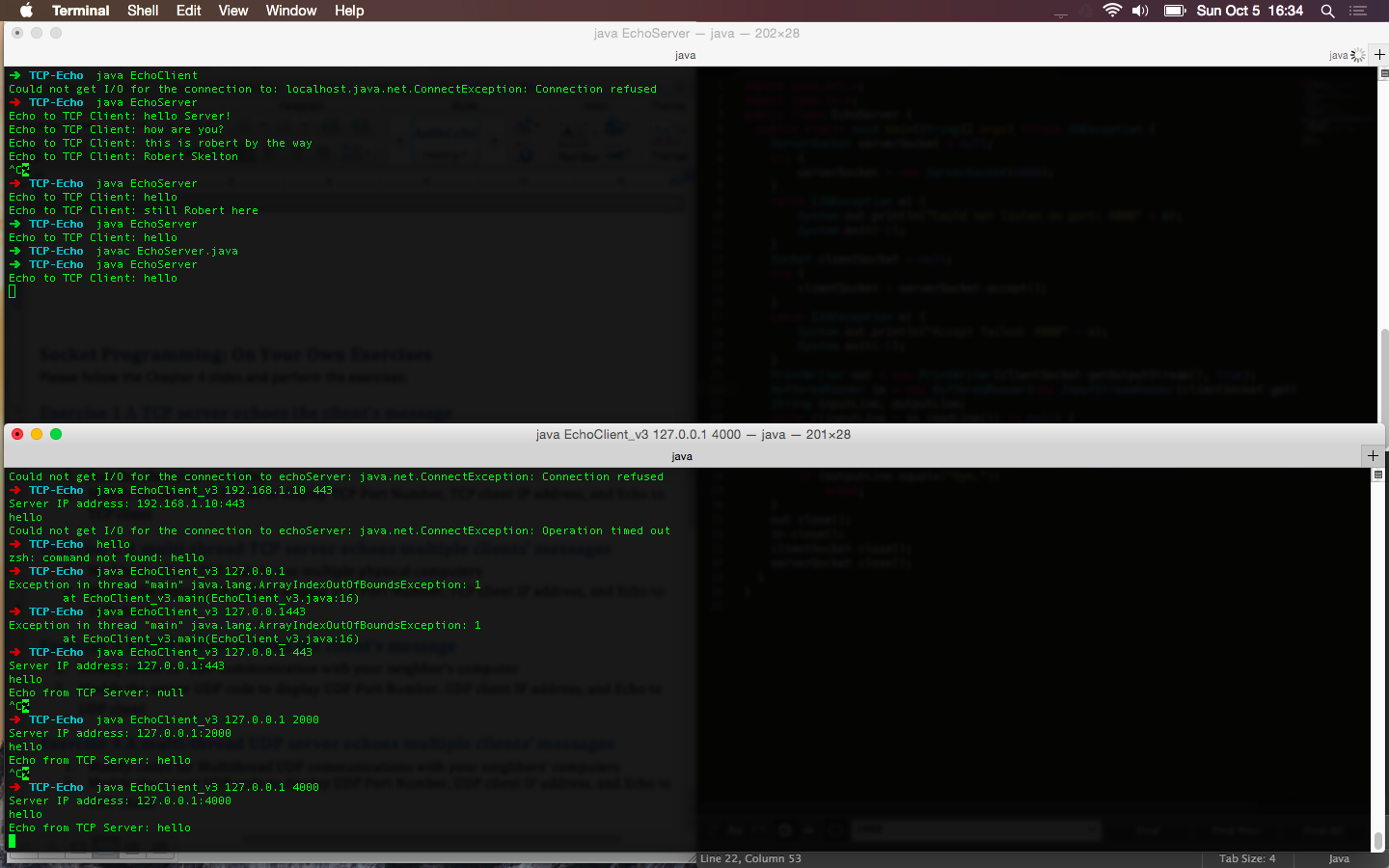


Figure 4. Changed port from 2000 to 4000.

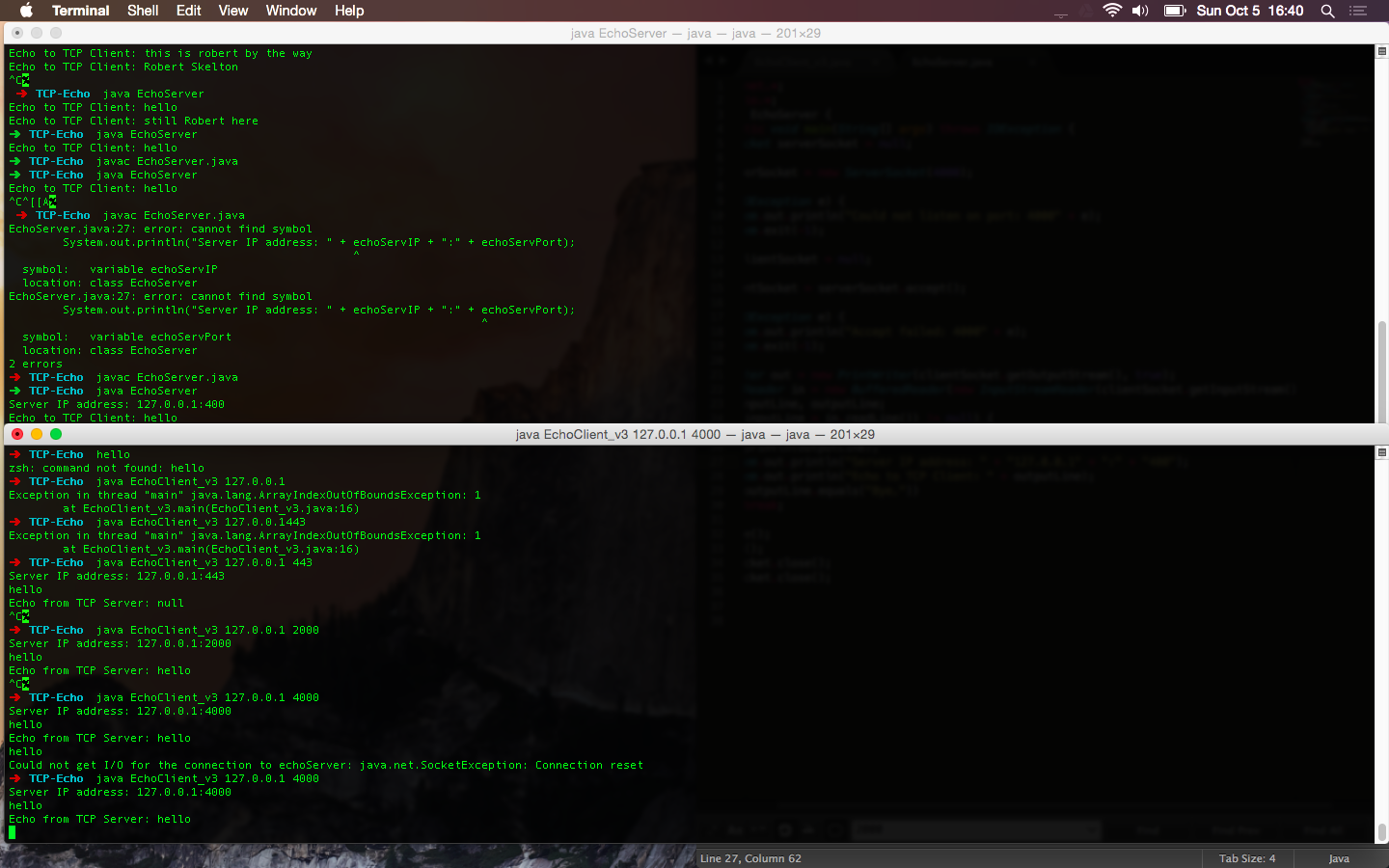


Figure 5. Code modified on Server side to display client IP and port, which is now 4000.

**TCP-Multithread**

The goal in this section was to modify the supplied server code to allow for multiple connections (multiple computers), and to print out the IP address and port number of the computer that sent the message. The output from the modified code is displayed below in Figure 6.

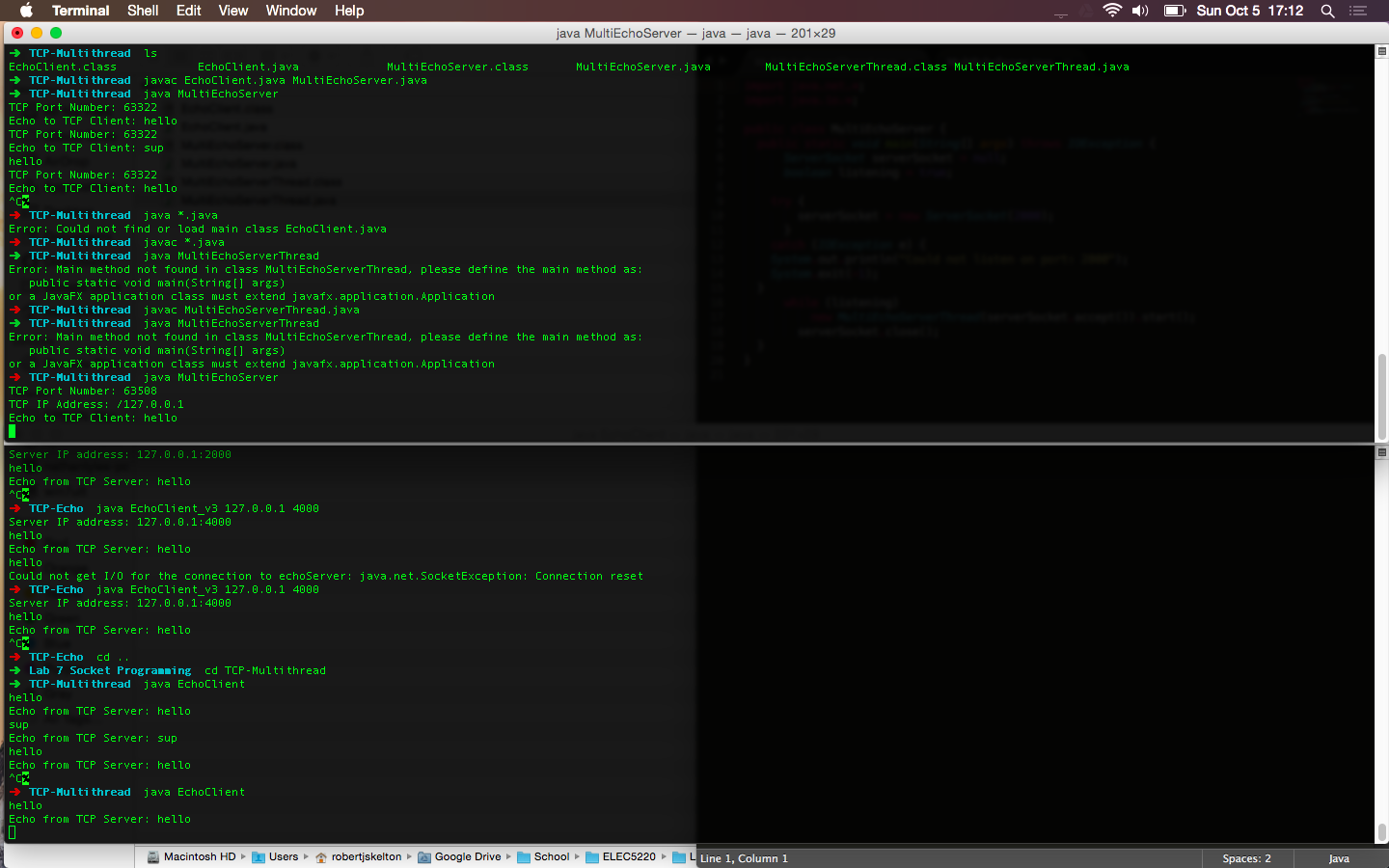


Figure 6. Code modified for multiple computers, and to show the connected IP address.

**UDP-Echo**

The required changes for this step were the same as the TCP Echo step. I had to change the code to accept multiple computer connections, and to display the connecting computer’s IP address and port number on the server. However, I did not have to change the port number for the connection this time.

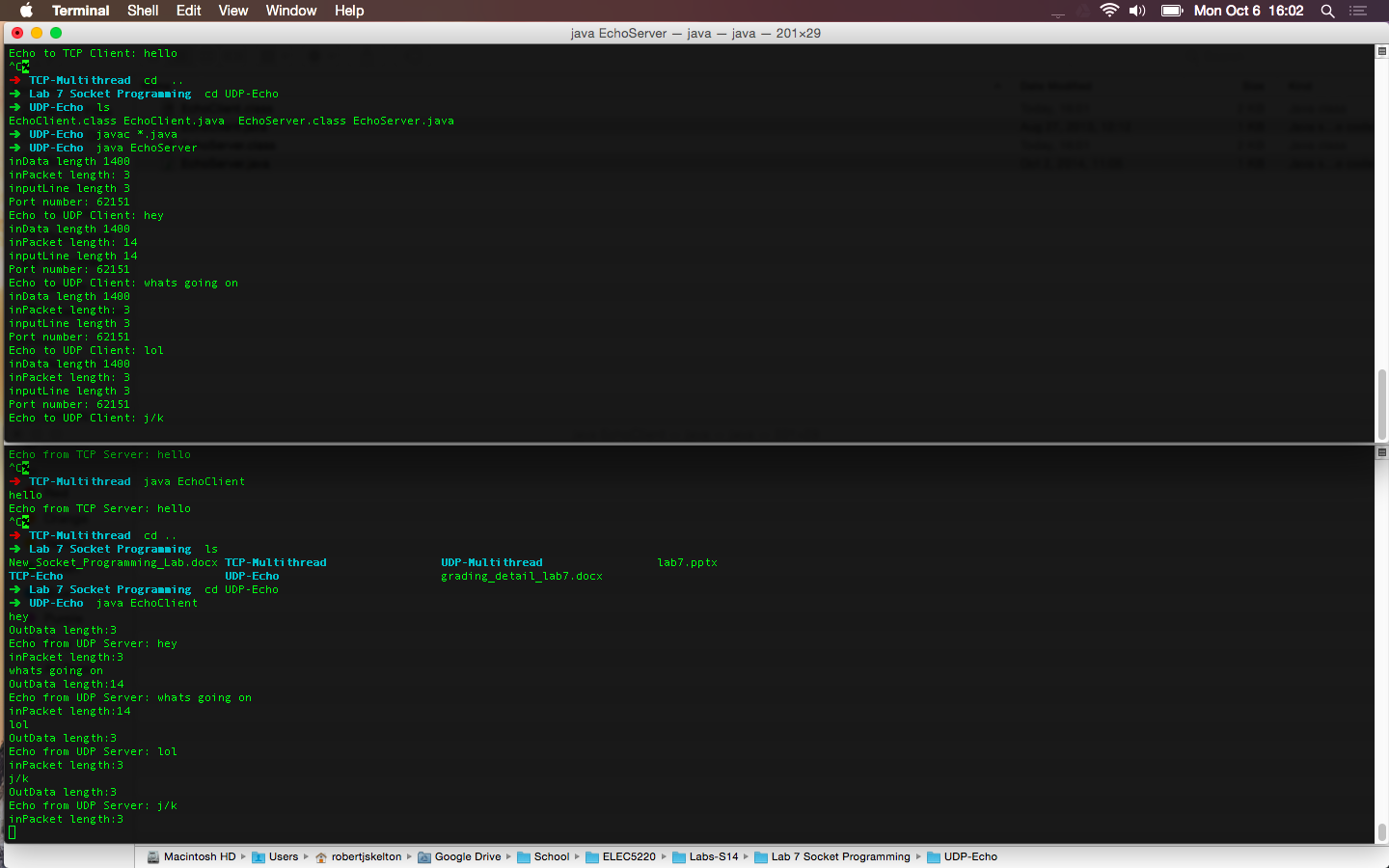


Figure 7. UDP Server and Client before any modifications are made.

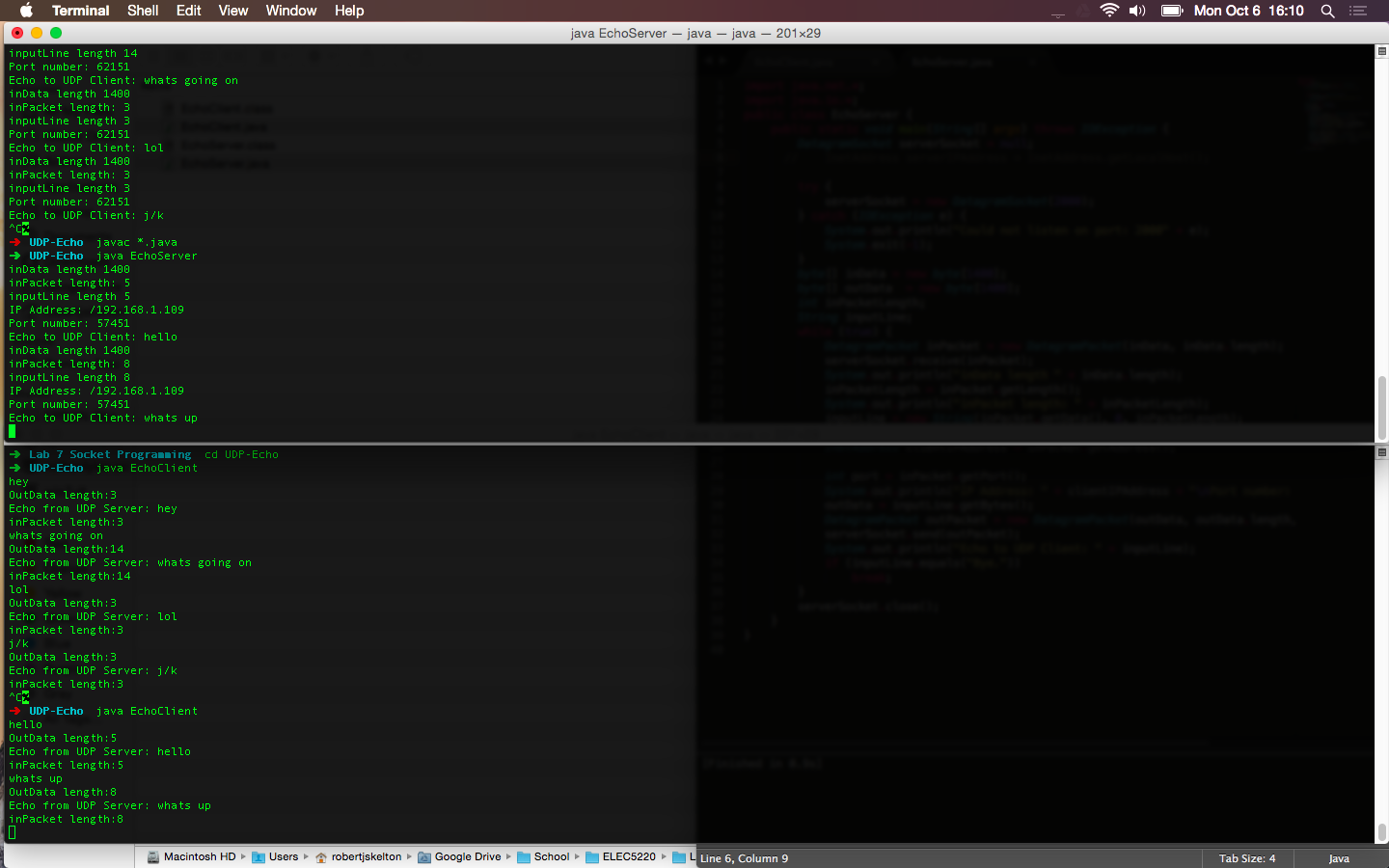


Figure 8. UDP Client and Server after IP address and multiple computer connections are enabled.

**UDP-Multithread**

For this step, we had to modify the code to display the UDP connections IP address and port number on the server’s terminal output. The before step is shown in Figure 9, and the after code is ran in Figure 10.

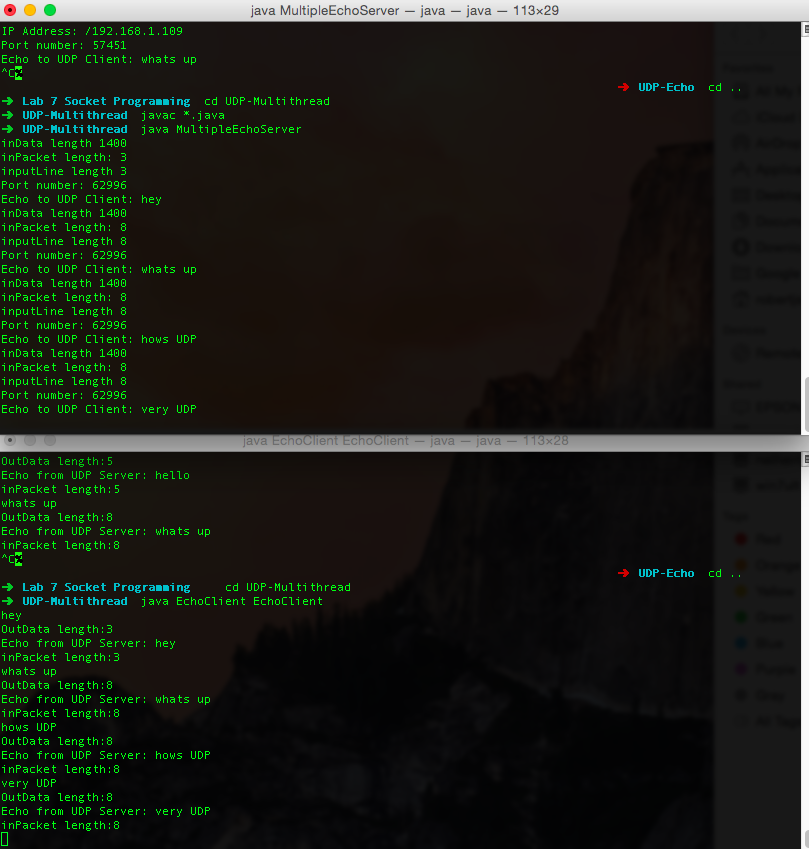


Figure 9. UDP Server and Client before modifications.

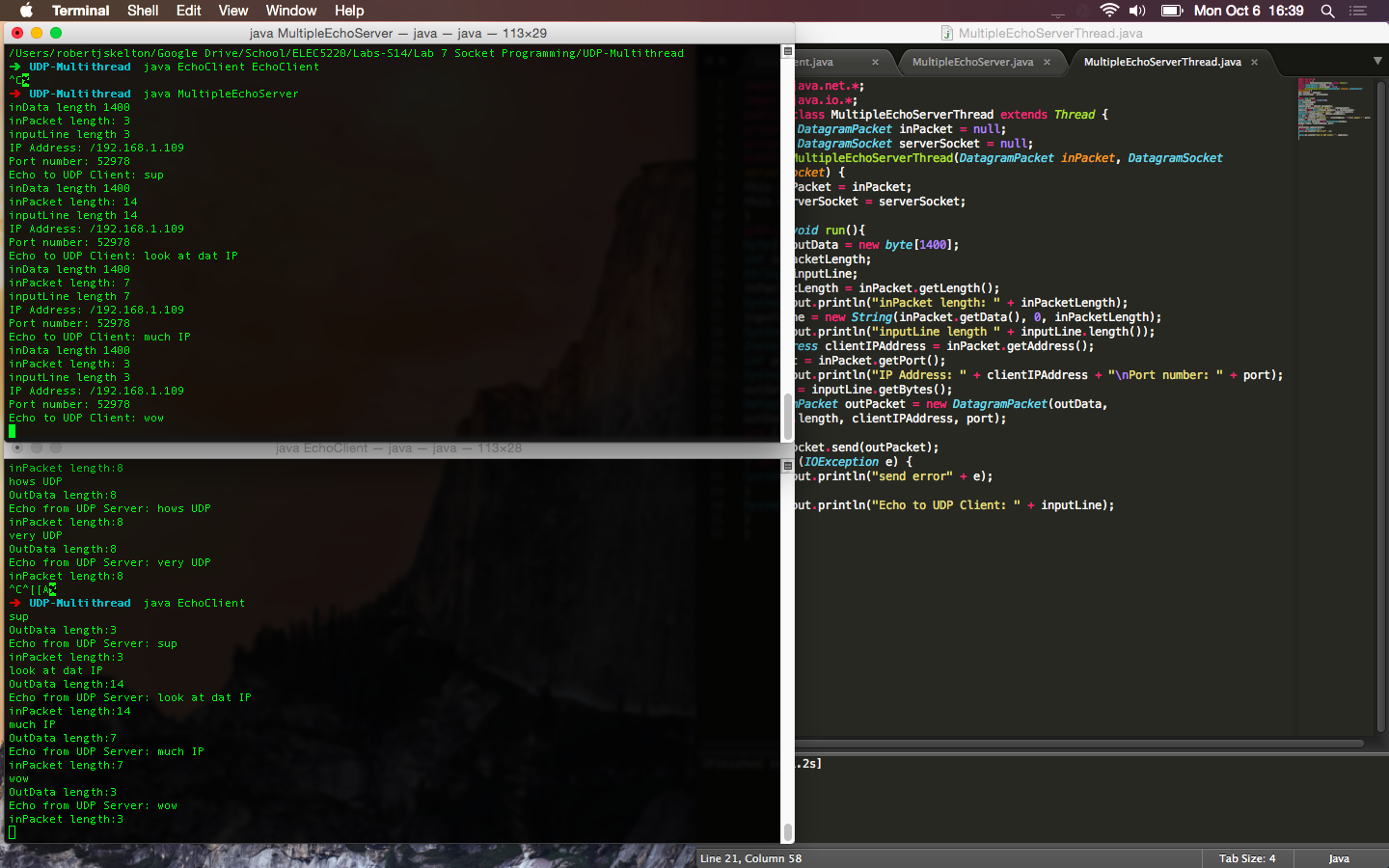


Figure 10. UDP Server printing out connected client’s IP address.

# Result

These four exercises were a good introduction to the practical uses of TCP and UDP datagram protocols. This will be a good foundation for my future career in Information Technology. The getInetAddress in Java was very useful for finding the IP address, and luckily it was defined already in most of the Server Java codes. This meant that all I had to do was use System.out.println to print out the IP address.

# Conclusion

This lab was a very straightforward process. The Server’s should all accept multiple connections at a time by default. As Mr. Hou explained, this was a quick and simple lab compared to many we have done previously throughout the semester, and I needed no external sources other than what was provided.

# References

* Dr. Wu’s included slides and lab materials