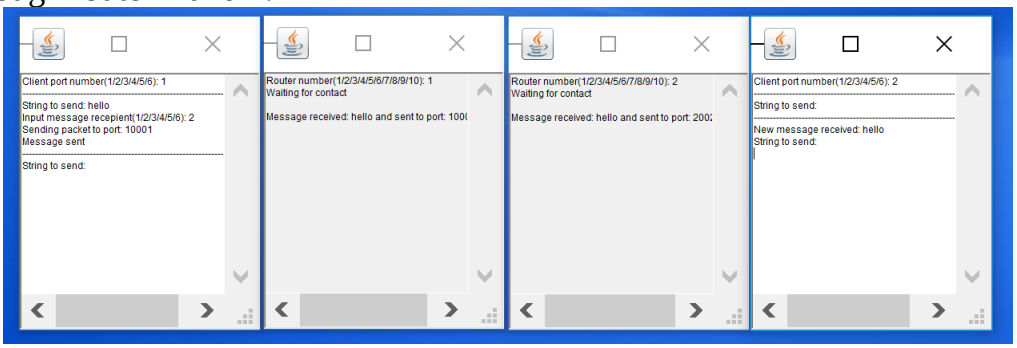
CS 2031: Telecommunications
Assignment #2

Jakub Slowinski: 16319781

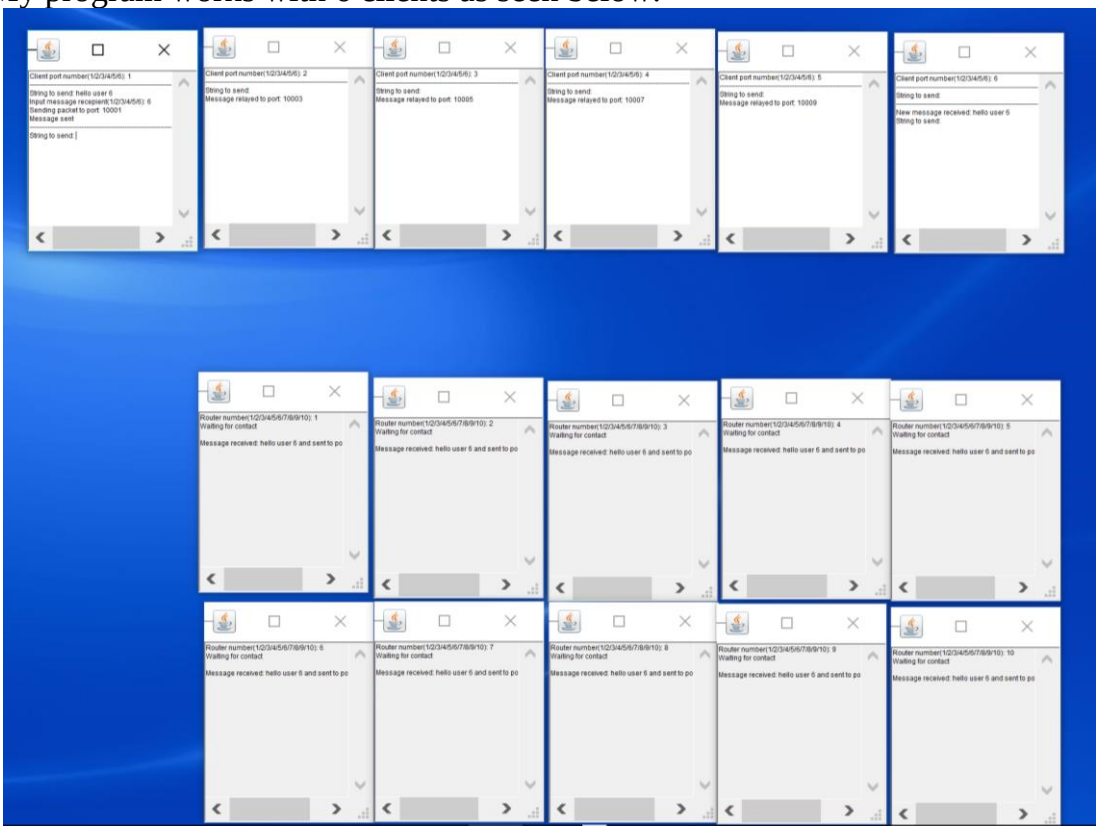
My report for the second telecoms assignment is as follows:
In my solution to the problem I tried to be as efficient as possible and to avoid a large amount of unreadable code. I opted for a very self-documented approach.. The longer I spent on this project, the more it flourished as a program and became more complete.
In my final solution, he client types in a message and a recipient number. The packet then is sent and is routed through the routers. The recipient obtains the message and then prints it.
The first Client opens a port on port number 2001, while all subsequent clients have +1 port number. The first router opens on port number 10001, while all subsequent clients have +1 port number.
My solution handles up to 6 clients with the help of 10 routers.
The router gets information from the controller on to which port to forward the data packet onto.
In between each client are 2 routers.
If using the maximum of 6 clients it looks like this(with red being clients and black being routers):
1 1 2 2 3 4 3 5 6 4 7 8 5 9 10 6

Pictures:

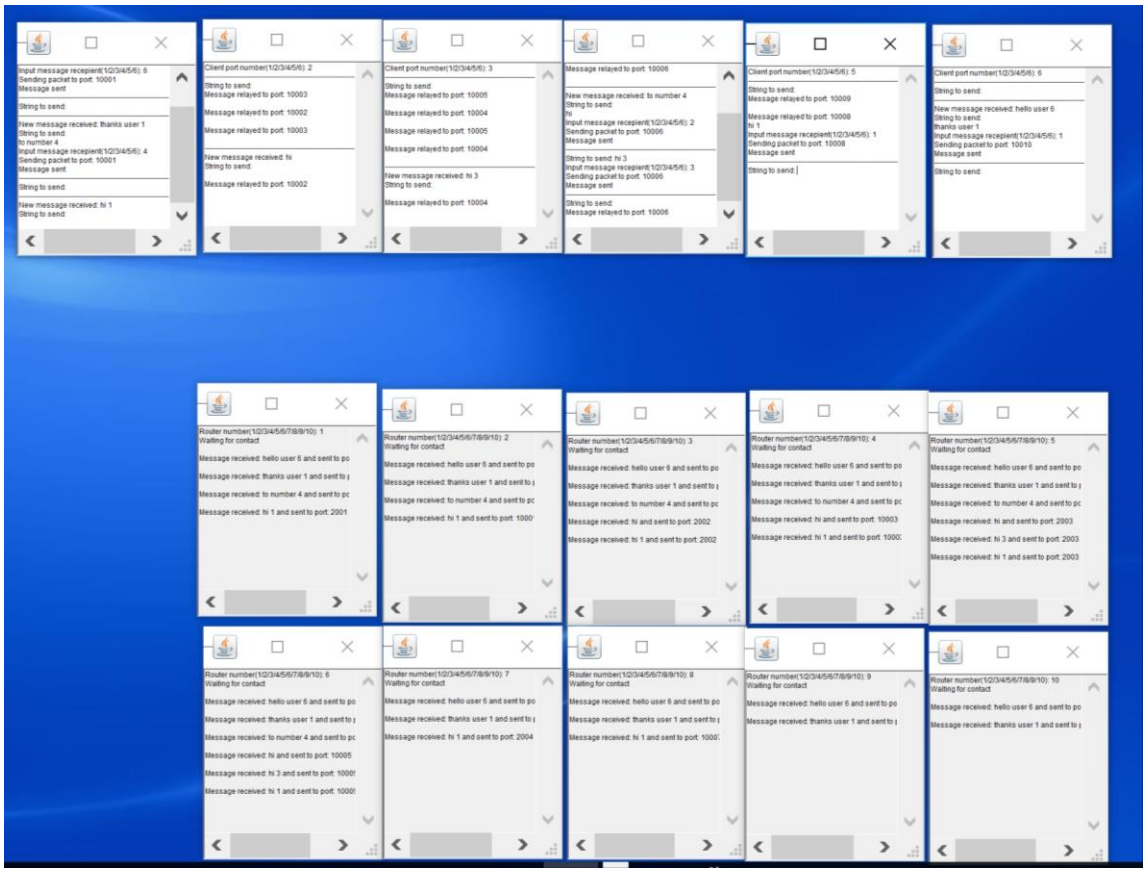
1. My program works with any amount of clients up to 6 as long as there are enough routers connecting them. This shows client 1 and 2 connected together through router 1 and 2.



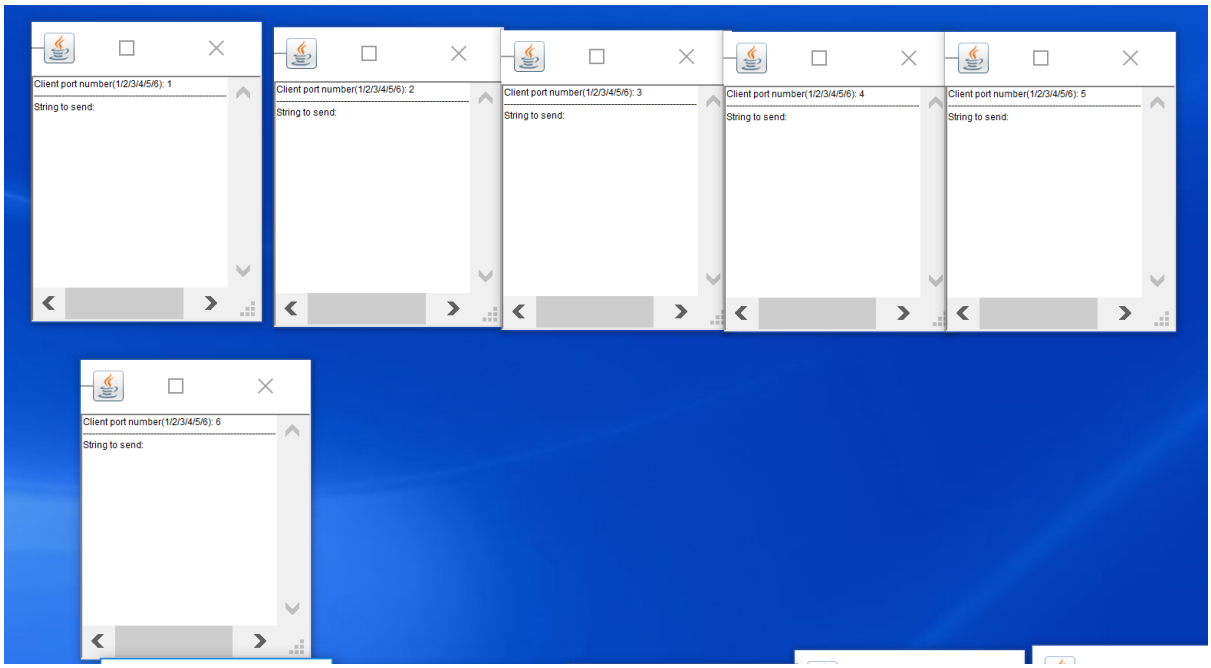
2. My program works with 6 clients as seen below.



3. More examples

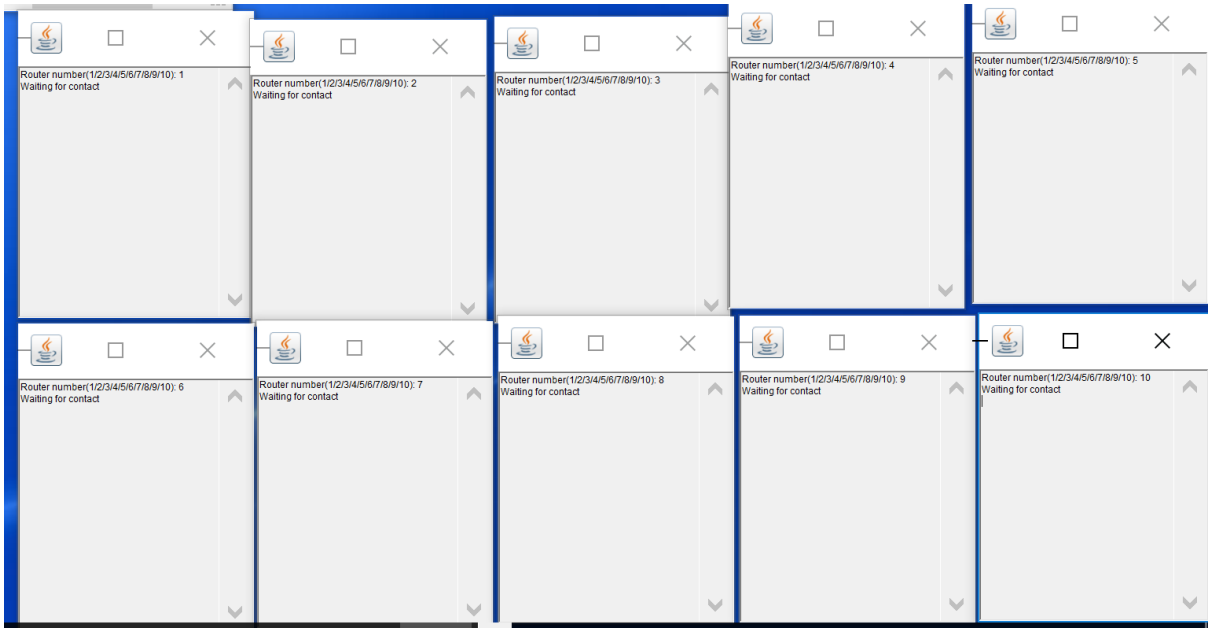


Client class:



The clients operates from port 2000-2006.
The clients all have to be initially set up by giving every client its corresponding number(1-6).
My client class takes in a string which becomes the payload of the message being send to the router. It also takes a final client destination as a byte.
The client class puts the final desintation into header[1], which means that the intended port number is the byte+2000.
The packet number gets forwarded through routers and clients. This makes my solution peer to peer as messages go through clients on their way to their destination.
When a client receives a message it checks is the final destination the same as its own client number. If that's the case it prints the message, else it forwards it securely without printing the message.

Router class:



A router is a networking device that forwards data packets between computer networks.
The router operates on ports 10001-10010.
The router waits to receive something and you cant type into it after initialising the port number.

```
int dstPort;
byte[] buffer = packet.getData();
int dNumber = buffer[1] & 0xff;
double R = this.port-10000;
R = (Math.ceil(R-R/2));
if (dNumber>R)
    dstPort= outs[0];
else
    dstPort= outs[1];
```

This code above determines which was to send the message.
My router class contacts the controller to know which addresses to forward the message. The message goes through routers and clients. The router prints what it receives and shows what port it sends it to.

Controller class:

My client contains a hardcoded 2D array which serves as the routing table

```
//router/source/destination
private static int[][] ROUTINGTABLE = {{0,1,6}, {2001,0,10001}, {10001,2001,10002}, {10002, 10001, 2002},{2002,10002,10003}, {10003, 2002, 10004},
        {10004,10003,2003},{2003,10004,10005},{10005,2003,10006},{10006,10005,2004},{2004, 10006, 10007},{10007,2004,10008},{10008,10007,2005},
        {2005,10008,10009},{10009,2005,10010},{10010,10009,2006},{2006,10010,0}};
```

The very first array is used to determine the direction.

Every array after that is sorted as follows: router/source/destination.
Example the router of port number 10001(router1)/ the message to it came from port 2001 (client 1)/ the messages destination is 10002 (router 2).
It is all stored as ints.
The controller contains 2 overloaded methods. One takes in just a port number and one takes in a port number and final destination. The one with 2 inputs has one output and is used if a router or client doesn't know a destination for a packet.
The method with one input is called upon starting a router and returns an array of port numbers which are the possible destinations of packets which can possibly go through that router. They both return elements from the routing table.

Packet content class:

My packet content class serves as an interface. It possesses the toSting() andtoDatagramPacket() methods. It also holds the length of the header as HEADERLENGTH.

String content class:

String content implements the packet content class. It returns the string and makes a datagram packet through.

Node class:

The listener function in the node class listens for incoming packets on a datagram socket and informs registered receivers about incoming packets. It listens for incoming packets and informs receivers upon arrival.

Errors:

No matter how much research I did I was not able to get a working distance vectoring approach. This made my overall project less efficient.
I encountered many problems in the project but with the help of my teaching assistants I got over most of them.

Advantages:

The advantages to my protocol is I used a linear peer to peer structure to send the packets to the intended target. My program works as shown by the above images. Each client and router is told which number it is so it is able to determine its port number without human error involved. Because you have to manually open the clients by running the class it makes sure there will be a perfect amount of the program running.

Disadvantages:

Each client and router has to be told which number it is before it determines each port number, this could be seen as tedious. My protocol doesn't use the link vector approach, this probably slows down the efficiency of my program.

Conclusion:

I spent approximately 20 hours on this project including the report.
I am very happy that I was able to deliver a working solution.
All in all this was a very challenging assignment to work on. I started with a half working first assignment and no knowledge of how multiple clients and routers are possible. As time went on I understood more and more of how what I was doing was designing a protocol, both packet layout and handling for the communication between multiple client through multiple routers. This assignment made me see how the study of telecommunications can be practically applied to real life situations. It was challenging yet fun and hence one of my favourite assignments in college to date. I think my efficient implementation worked well for me as I found an effective solution with not too many messy lines of code. I hope you enjoy reading my code as much as I enjoyed writing it.