CSCN72050 – Assignment 1

UDP/IP and Objects using Dynamic Memory, Bits

In this Assignment, you will apply your knowledge of UDP/IP Communication, Objects with Dynamic and Bitwise structures.

PART 1:

In part 1 you will complete the capabilities of the packet object, and finish the transmission of the packet over a UDP/IP socket connection.

> WARNING:
> Pay close attention to your data sizes. Don't hardcode.

Download the Visual Studio repository ***CSCN72050_A1*** folder from eConestoga and perform the following modifications to the code:

STEP#1 – Complete the Packet Class

```
Packet(char* src)
```

Complete the overloaded Packet constructor. A raw buffer of data just received over the socket interface is provided. Using the definition in packet.h, parse out the header, body and CRC information and populate the objects internal state.

```
void SetData(char* srcData, int Size)
```

Complete the SetData function in packet.h. This function receives two arguments. The first is a pointer to the address where the data to be transmitted is stored. The second is the size (in bytes) to be copied into the packet objects data buffer. Implement the allocation and copy of the data from the srcData location to the internal data location. Remember to update and set the header information accordingly.

```
char* SerializeData(int &TotalSize)
```

As we have learned in class, anytime an object has dynamic memory involved, it is important to serialize the data before transmitting it. SerializeData is to allocate enough space in the local TxBuffer of the Packet class, calculate the CRC and copy of the information/data to be transmitted into the buffer and return the TxBuffer address for the calling function to use in the transmission. This function also takes a reference to an integer, which you need to set to the total number of bytes in the TxBuffer.

```
unsigned int CalculateCRC()
```

Set the CRC data field to a value of 0xFF00FF00;

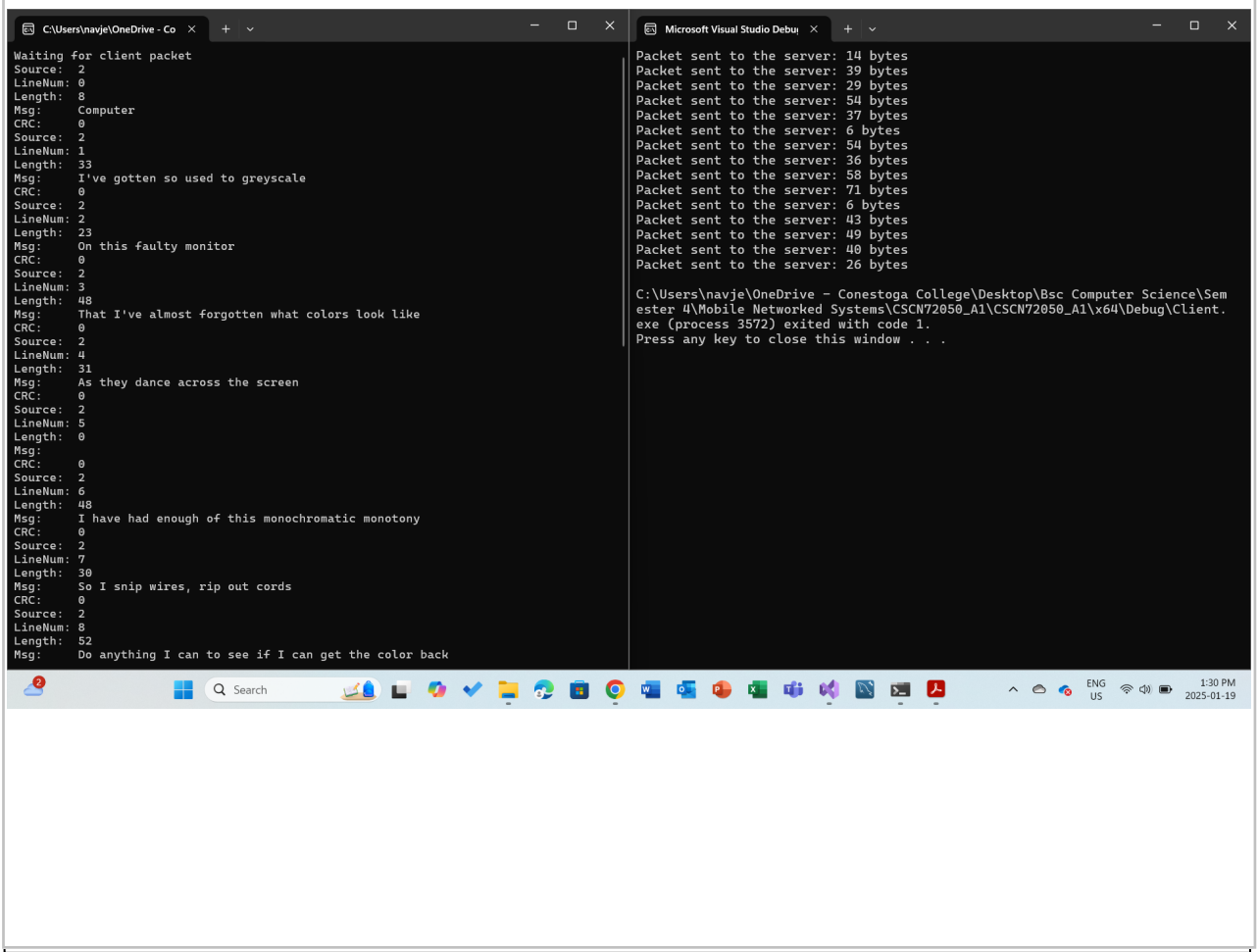
PART #2 – Complete the Server/Client CPP files

STEP #1: Complete the Data Communications

Add the UDP/IP code to send and receive the data packets of the socket connections defined. Look for the “TODO:” comment.

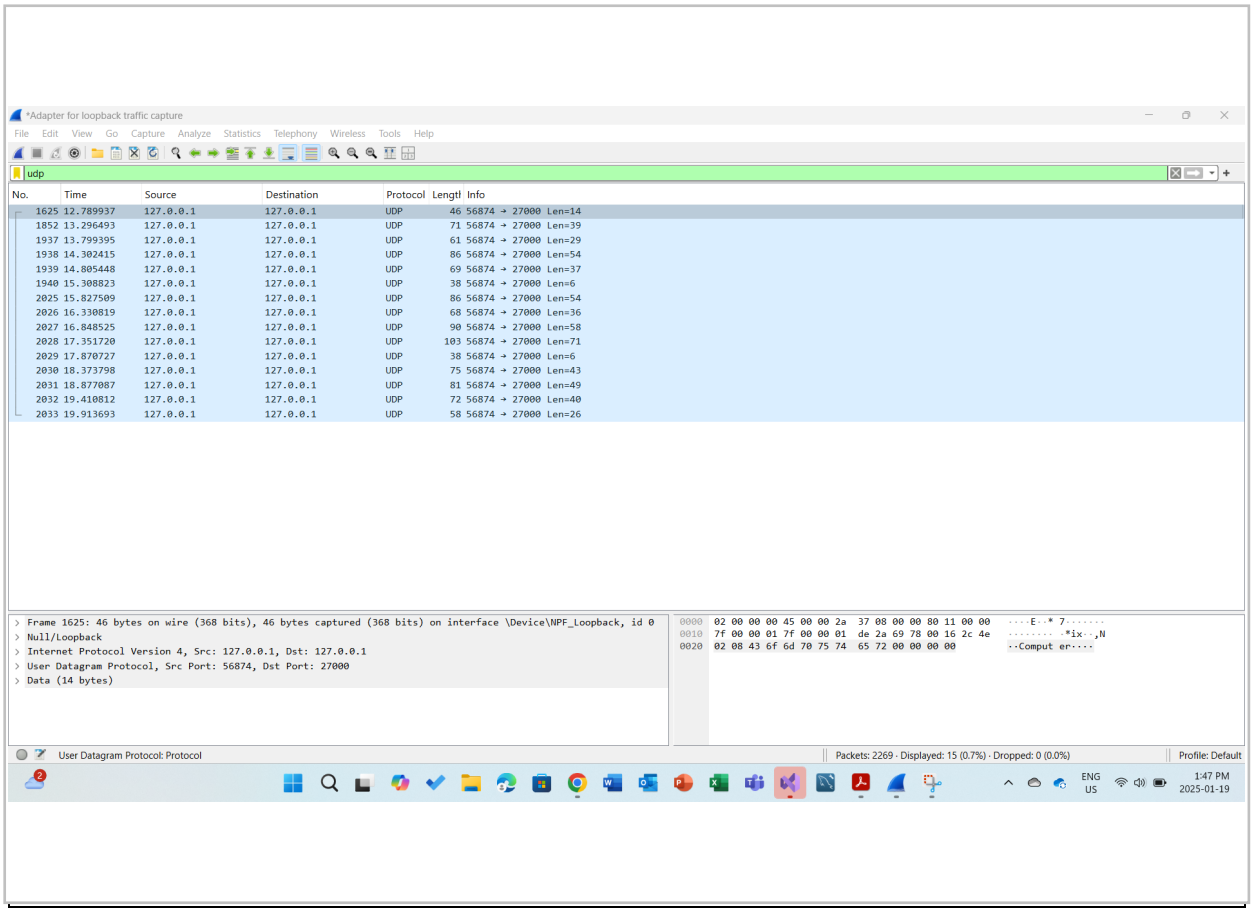
STEP #2: Show Running Code

Run your Client and Server applications side by side. Take a screenshot (at any point during the execution) of your client and server applications (as shown in the Appendix below) and paste it in the box below.



STEP #3:

Use Wireshark and capture the data during the execution of your program. Filter your Wireshark file and find the communication packets showing your data transmission. Upload a screenshot to the box below.



What to Hand In

Once you have completed your lab create and upload the following files:

- This PDF form completed (without any modifications)
- Packet.h
- Client.cpp
- Server.cpp

Do not compress these files in anyway. Uploading a Zip, Tar, 7z, etc… file will result in a 10% penalty for not following instructions.

APPENDIX

Your final screenshot should look something like this.