1.
Naman Bajaj – nb726
Roshan Aiyar – ra854

2.
We only worked with each other, and the only resources we consulted were Piazza. It was mostly concerning step 5, as that is the one we had the most trouble with.

3. We are mostly sure that our code works to the specifications above. The only part we have doubt about is step 5, as the first thing that we receive is the size of the file, however we were instructed not to change stopandwait.py to send a password as the first packet. Our first instead reads the second packet, as that actually contains the first chunk of data.

4. Our main difficulties came from testing, as some of the larger files took a long time to send all of the data (in step 1 only, as that was stop and wait sending). Also, step 5 gave us some trouble, as mentioned above. This is due to the fact that we were unsure how the data would be structured.

5. One thing we learned was how much more efficient pipelined sending would be compared to stop and wait sending. Obviously, it makes sense that sending larger chunks means be quicker than packets one by one, but we were surprised by just how much faster it would be. The long input took over 5 minutes with stop and wait (with random packet drops), compared to pipelining with a medium window size, which took under 30 seconds with the same packets being dropped. It also showed how there is a right window size for a certain amount of data. When sending large data, there were certain values that were faster, and those values all clustered near some value.

Another thing we observed was how quickly ACKs are able to be received by the sender. We thought that in class, the near zero time of ACKs being sent was purely theoretical, i.e., in real scenarios, there still is a noticeable time for ACKs to be sent. However, we noticed that ACKs were being sent very quickly, even across different computers. It showed how much theory and practice intertwine even when it seems that it can’t happen.