Network Systems | Challenge 2

Lars Ran: s1403192
Anne van den Boom: s2674475
Challenge group number: 40

Stop-and-wait tryout
We started out simple by having the sender send a packet until the complete fileContents were sent. The receiver would send an ACK packet whenever it received a packet from the sender, indicating to the sender that the next packet could be sent.

More packets at a time and AckReceiver
Now that we managed to send files it is time to put the bandwidth to good use. Therefore we want to send multiple files at a time. In order to do this we used the header bytes to number the packets. This made a lot of things easier. First of all, we could easily resend a packet once we knew what packet was lost. Secondly, the receiver could already store the packet in the resulting array as he knew exactly where it should go. And last but not least, we could send more packets at a time while keeping track of which were ack'ed. In order to do this we made a helper class called AckReceiver. This class kept track of the acks received in a concurrent thread. The sender could then always ask which files were not received and whether all files were received.

Window and choosing timeouts
Whereas previously, the sender just tried to send many packets consecutively in the *for* loop, we introduced a window of packets that can be "in circulation" at the same time. Furthermore, we added a small *sleep* period between sending packets. An issue that we found was that the sender would time-out too quickly and would thus retransmit packets that weren't actually lost, just slightly delayed. By having the window of only 32 packets being in circulation and having a little bit longer in between sending of packets, this solved that problem. The window size was estimated from the fact that the RTT was around 720 ms, and having a 20 ms delay in between sending packets.

The program determines that the complete file is sent when the receiver receives a packet that is smaller than the set datasize and there are no missing packets. If the last packet can fit exactly within the determined datasize, an "empty" packet is sent as the last frame.