Assignment 1 Report

CZ3006: Net Centric Computing

Academic Year 2016/2017

Semester 2

Student Name: Huang Jian Wei

Matriculation Number: U1521567A

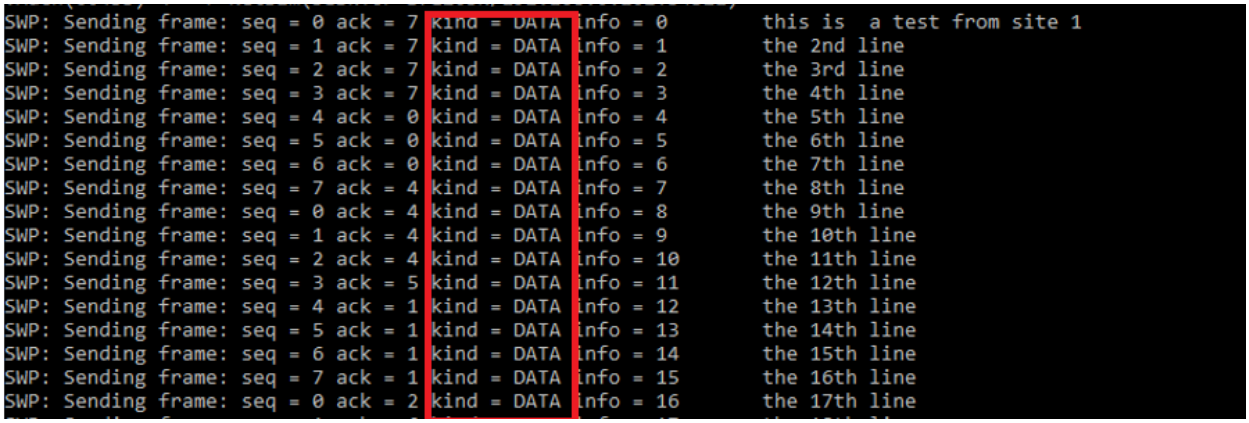
Lab Group: TS1

1. My Task

1.1 Full duplex Data Communication

Full duplex communication means that data transmission can be done in both direction, sending and receiving. One way to achieve this is to have two separate communication channel, “forward” and “reverse” channel, and use each as simplex data traffic. One disadvantage of this setup is that bandwidth of the “reverse” channel will be wasted.

In our setup, we achieve full duplex commination by setting a field in the frame header to distinguish between frame types, such as data or acknowledgement. This means that traffic can be sent and receive in both ways in a single circuit.



```
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test from site 1
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 5 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 1 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 1 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 16       the 17th line
```

Figure 1. Distinguish between the different kind of frames e.g Kind = DATA

The possible value for “Kind” field can be as such; DATA, the data that was meant for the receiver. ACK, for frames that notify the sender that data has been successful transmitted. Lastly, NAK means that sender is required to send some missing frames.

Another possible way to reduce wastage of bandwidth is through “piggybag” strategy. By delaying outgoing acknowledgement until they can get a free ride on the next outgoing frame.

Implementation:

```
s.seq = next_frame;
    s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
    if (frame_kind == PFrame.NAK) {
        no_nak = false;
    }
```

1.2 In-order delivery of packets to the network layer

To make sure that the delivery of packets is in-order to the network layer, the sliding protocol uses frame sequence number to make sure that only packets whose sequence number falls in the frame window will be received by the receiver. If packets whose sequence number does not fall into the window range, it will be discard and send a negative acknowledgement to the sender to send the correct frame.

When using Protocol 6(Selective repeat strategy), receiver may accept range that are not in the receiving window, hence resulting in packets not received in order. To solve this, protocol 6 has an array of buffer to stores frames that are within the window and only send them once all frames in the buffer are in the correct order.

Implementation:

```
if (between(expected_frame, temp_frame.seq,upper_limit) && (recieved[temp_frame.seq %
NR_BUFS] == false)) {
   recieved[temp_frame.seq % NR_BUFS] = true;
   in_buffer[temp_frame.seq % NR_BUFS] = temp_frame.info;
   while (recieved[expected_frame % NR_BUFS]) {
            to_network_layer(in_buffer[expected_frame % NR_BUFS]);
            no_nak = true;
            recieved[expected_frame % NR_BUFS] = false;

expected_frame = inc(expected_frame);
                     upper_limit = inc(upper_limit);
                        start_ack_timer();
                    }
                 }
                    }
```

1.3. Selective repeat retransmission strategy

For protocol 6 of the sliding window protocol, we use selective repeat retransmission strategy to only transmit the frame that is expired. Compared to Protocol 5 of the sliding window protocol, this could cut down on the number of frames being resend.

This strategy make use of an array of timers, one for each individual timer so that instead of transmitting all outstanding frame, it only retransmits the expired frame.

Implementation:

```
        to_physical_layer(s);
        if (frame_kind == PFrame.DATA) {
            start_timer(next_frame);
        }
private void start_timer(int seq) {
      stop_timer(seq);
      frame_timer[seq % NR_BUFS] = new Timer();
      frame_timer[seq % NR_BUFS].schedule(new ReTask(swe, seq), 200);
   }
```

However, there is still one problem in this strategy. Since sliding window move in a circular fashion, there may be an overlap of sequence number which may cause the receiver to be unable to separate the old frames and the newer frames.

To solve this issue, we must ensure that the old and new sequence number does not overlap in the sliding window. Using *(MAX_SEQ + 1)/2*, we can make sure that the window will always be half of the maximum sequence number, hence we can differentiate between the old frames and the newer frames.

```
public static final int MAX_SEQ = 7;

public static final int NR_BUFS = (MAX_SEQ + 1)/2;
```

1.4. Synchronization with the network layer by granting credits

The sliding window protocol must be synchronized with the network layer to ensure that there is enough time to receive packets from the physical layer.

By granting credits to the network layer based on the number of buffer available, we ensure that network layer only sends the maximum number of frames within the sending window.

Initially after the protocol has been set up, the network layer will be given the credit based on the full window size, allowing the virtual machine to send up to 4 packets before processing packets that has been received from another virtual machine. Over time, the network layer will be granted credits based on the acknowledgement sent by the receiver and the amount of available buffer, allowing the network layer to be synchronized to the data linked layer, sending and receiving window.

Implementation:

```
private void enable_network_layer(int nr_of_bufs) {
//network layer is permitted to send if credit is available
        swe.grant_credit(nr_of_bufs);
}


while (between(expected_ack, temp_frame.ack,next_frame_send)) {
                stop_timer(expected_ack % NR_BUFS);
                expected_ack = inc(expected_ack);
                enable_network_layer(1);
            }
```

1.5. Negative Acknowledgement

In the selective repeat transmission protocol, receiver is allowed to be able to receive frames out-of-order, as long as the frames are within the window range. However, this means that some of frame to be “missing”, hence we will require a way to tell the sender that we are missing a certain frame.

The Boolean “no_nak” will be used to track whether if an expected packet has been received. If there is an error in receiving the packet, after a period of time a negative acknowledgement for the packet will be sent. In the source code, the no_nak will be set to false when a negative acknowledgement has to be sent.

Implementation:

```
 boolean no_nak = true; //no nak has been sent yet

  if (temp_frame.kind == PFrame.DATA)
                        {

     if ((temp_frame.seq != expected_frame) && no_nak)
                          {
        send_frame(PFrame.NAK, 0, expected_frame, out_buf);
                 }
            else
{
      start_ack_timer();
     }
```

1.6. Separate acknowledgement when the reverse traffic is light or more

Piggyback is useful to reduce bandwidth within the channel, however, when the reverse traffic is light, we should also implement a separate acknowledgement to make sure that the channel is optimize to the fullest.

We can achieve this by implementing a timer that will send a separate and individual acknowledgement frame once the timer expired.

Implementation

```
private void send_frame(int frame_kind, int next_frame, int frame_expected, Packet buffer[])
        {
    PFrame s = new PFrame();

   s.kind = frame_kind;
      if (frame_kind == PFrame.DATA) {
         s.info = buffer[next_frame % NR_BUFS];
      }
      s.seq = next_frame;
      s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
      if (frame_kind == PFrame.NAK) {
         no_nak = false;
      }
      to_physical_layer(s);
      if (frame_kind == PFrame.DATA) {
         start_timer(next_frame);
      }
      stop_ack_timer();
   }

  private void start_ack_timer( ) {
         stop_ack_timer();
      //starts another timer for sending separate ack
      ack_timer = new Timer();
      ack_timer.schedule(new AckTask(swe), 100);
  }
```

Conclusion

Hence by implementing certain functions in the protocol mentioned above, we are able to ensure that the protocol is able to withstand quality level 3 of the Network Simulator.

This report is done by Huang Jian Wei(U1521567A)

2. NetSim Tests

2.1 NetSim Quality Level 0

VMach 1 Output

C:\Windows\system32\cmd.exe - java VMach 1

```
VMach is making a connection with NetSim...
VMach(60435) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test from site 1
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 5 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 1 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 1 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 6 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 6 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 3 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 7 ack = 3 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 7 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 0 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = ACK info =
```

VMach 2 Output

C:\Windows\system32\cmd.exe - Java VMach 2

```
C:\Users\Jianwei\Downloads\ass1>Java VMach 2
VMach is making a connection with NetSim...
VMach(60423) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)

C:\Users\Jianwei\Downloads\ass1>Java VMach 2
VMach is making a connection with NetSim...
VMach(60437) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 0        this is  a test from site 2
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 2 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 7 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 7 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 1 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 5 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 5 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 5 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 5 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 7 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 0 ack = 2 kind = ACK info =
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
```

Java NetSim 0

```
C:\Users\Jianwei\Downloads\ass1>java NetSim 0
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 60435
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 60437
```

2.2 NetSim Quality Level 1

VMach 1 Output

```
C:\Users\Jianwei\Downloads\ass1>java VMach 1
VMach is making a connection with NetSim...
VMach(55565) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test from site 1
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 0        this is  a test from site 1
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 0 ack = 6 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 4 ack = 3 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 7 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 7 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 7 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 6 ack = 7 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 7 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 4 ack = 7 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 3 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 3 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 3 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 7 ack = 3 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 0 kind = ACK info =
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 3 ack = 1 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 2 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
```

```
SWP: Sending frame: seq = 4 ack = 2 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 2 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 6 ack = 6 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 6 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 7 kind = ACK info =
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 7 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
```

VMach 2 Output

```
C:\Users\Jianwei\Downloads\ass1>java VMach 2
VMach is making a connection with NetSim...
VMach(55566) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test from site 2
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 5 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 5 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 5 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 5 kind = DATA info = 19       the 20th line
```

```
SWP: Sending frame: seq = 0 ack = 1 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 7 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 4 ack = 3 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 3 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 3 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 7 ack = 3 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 4 ack = 3 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 7 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 1 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 4 ack = 1 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 6 ack = 5 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 5 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 5 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 5 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 5 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
```

Java NetSim 1

```
C:\Users\Jianwei\Downloads\ass1>java NetSim 1
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 55565
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 55566
VMach 1 loose frame seq = 0 error counter = 1
VMach 1 loose frame seq = 1 error counter = 2
VMach 1 loose frame seq = 3 error counter = 3
VMach 2 loose frame seq = 5 error counter = 1
VMach 2 loose frame seq = 7 error counter = 2
VMach 1 loose frame seq = 1 error counter = 4
VMach 2 loose frame seq = 0 error counter = 3
VMach 2 loose frame seq = 4 error counter = 4
VMach 2 loose frame seq = 0 error counter = 5
VMach 2 loose frame seq = 1 error counter = 6
VMach 1 loose frame seq = 0 error counter = 5
VMach 1 loose frame seq = 2 error counter = 6
VMach 1 loose frame seq = 4 error counter = 7
VMach 2 loose frame seq = 0 error counter = 7
VMach 1 loose frame seq = 4 error counter = 8
VMach 2 loose frame seq = 0 error counter = 8
VMach 1 loose frame seq = 7 error counter = 9
VMach 2 loose frame seq = 1 error counter = 9
VMach 2 loose frame seq = 2 error counter = 10
VMach 2 loose frame seq = 3 error counter = 11
VMach 1 loose frame seq = 0 error counter = 10
VMach 1 loose frame seq = 5 error counter = 11
VMach 1 loose frame seq = 0 error counter = 12
VMach 1 loose frame seq = 2 error counter = 13
VMach 1 loose frame seq = 0 error counter = 14
VMach 2 loose frame seq = 3 error counter = 12
VMach 2 loose frame seq = 0 error counter = 13
VMach 1 loose frame seq = 0 error counter = 15
VMach 1 loose frame seq = 5 error counter = 16
VMach 2 loose frame seq = 4 error counter = 14
VMach 1 loose frame seq = 6 error counter = 17
VMach 1 loose frame seq = 1 error counter = 18
VMach 2 loose frame seq = 7 error counter = 15
VMach 2 loose frame seq = 0 error counter = 16
VMach 2 loose frame seq = 2 error counter = 17
VMach 2 loose frame seq = 0 error counter = 18
VMach 1 loose frame seq = 0 error counter = 19
VMach 1 loose frame seq = 1 error counter = 20
VMach 2 loose frame seq = 0 error counter = 19
VMach 1 loose frame seq = 2 error counter = 21
VMach 1 loose frame seq = 0 error counter = 22
VMach 2 loose frame seq = 0 error counter = 20
VMach 2 loose frame seq = 2 error counter = 21
```

2.3 NetSim Quality Level 2

VMach 1 Output

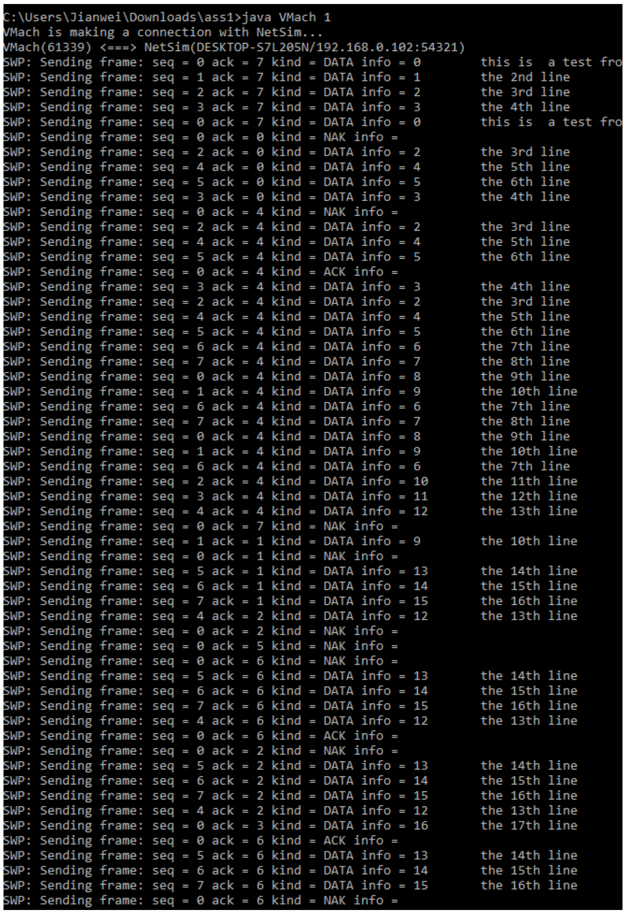
```
C:\Windows\system32\cmd.exe - java VMach 1
E:\Users\Jianwei\Downloads\ass1>java VMach 1
VMach is making a connection with NetSim...
VMach(59314) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 0        this is  a test from site 1
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 5 kind = ACK info =
SWP: Sending frame: seq = 2 ack = 5 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 1 ack = 5 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 3 ack = 5 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 4 ack = 5 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 0 ack = 5 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 5 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 2 ack = 5 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 0 ack = 1 kind = ACK info =
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 1 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 1 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 1 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 1 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 0 ack = 1 kind = ACK info =
SWP: Sending frame: seq = 2 ack = 1 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 1 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 5 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 6 ack = 5 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 5 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 6 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 2 ack = 6 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 6 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 3 ack = 6 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 6 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 6 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 6 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 25       the 26th line
```

```
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 3 ack = 2 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 2 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 0 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 1 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = ACK info =
```

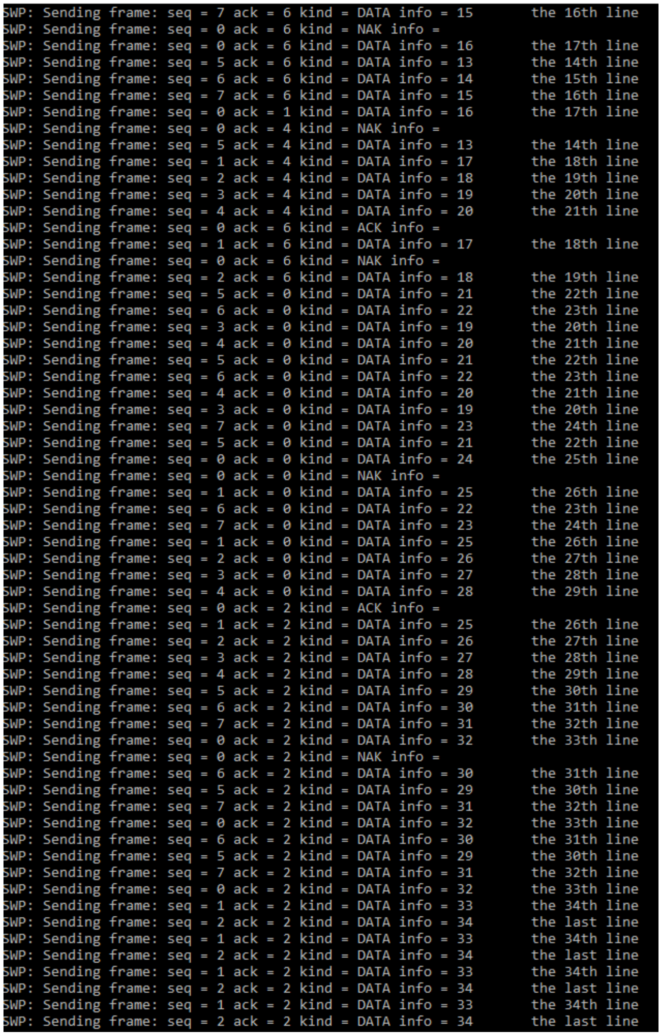
VMach 2 Output

2.4 NetSim Quality Level 3

VMach 1 Output



```
C:\Users\Jianwei\Downloads\ass1>java VMach 1
VMach is making a connection with NetSim...
VMach(61339) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test fro
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test fro
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 0 ack = 4 kind = ACK info =
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 4 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 6 ack = 4 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 6 ack = 4 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 1 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 4 ack = 2 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 5 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 6 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 6 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 4 ack = 6 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 6 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 4 ack = 2 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 0 ack = 6 kind = ACK info =
SWP: Sending frame: seq = 5 ack = 6 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 6 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
```



```
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 6 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 5 ack = 6 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 6 ack = 6 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 6 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 0 ack = 6 kind = ACK info =
SWP: Sending frame: seq = 1 ack = 6 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 0 ack = 6 kind = NAK info =
SWP: Sending frame: seq = 2 ack = 6 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 0 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 0 ack = 2 kind = ACK info =
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 2 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 2 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 6 ack = 2 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 5 ack = 2 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 1 ack = 2 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 2 kind = DATA info = 34       the last line
```

VMach 2 Output

```
C:\Users\Jianwei\Downloads\ass1>java VMach 2
VMach is making a connection with NetSim...
VMach(61340) <===> NetSim(DESKTOP-S7L205N/192.168.0.102:54321)
SWP: Sending frame: seq = 0 ack = 7 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 7 kind = DATA info = 0        this is  a test from site 2
SWP: Sending frame: seq = 1 ack = 7 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 7 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 7 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 0 ack = 1 kind = NAK info =
SWP: Sending frame: seq = 4 ack = 1 kind = DATA info = 4        the 5th line
SWP: Sending frame: seq = 1 ack = 1 kind = DATA info = 1        the 2nd line
SWP: Sending frame: seq = 2 ack = 1 kind = DATA info = 2        the 3rd line
SWP: Sending frame: seq = 3 ack = 1 kind = DATA info = 3        the 4th line
SWP: Sending frame: seq = 5 ack = 1 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 1 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 1 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 1 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 0 ack = 1 kind = ACK info =
SWP: Sending frame: seq = 5 ack = 5 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 5 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 5 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 5 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 5 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 0 ack = 5 kind = ACK info =
SWP: Sending frame: seq = 5 ack = 5 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 5 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 5 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 5 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 0 ack = 0 kind = ACK info =
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 5        the 6th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 6        the 7th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 7        the 8th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 8        the 9th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 9        the 10th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 3 ack = 0 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 4 ack = 3 kind = DATA info = 12       the 13th line
SWP: Sending frame: seq = 5 ack = 3 kind = DATA info = 13       the 14th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 10       the 11th line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 3 kind = ACK info =
```

```
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 3 kind = ACK info =
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 3 ack = 3 kind = DATA info = 11       the 12th line
SWP: Sending frame: seq = 6 ack = 3 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 6 ack = 3 kind = DATA info = 14       the 15th line
SWP: Sending frame: seq = 7 ack = 3 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 7 ack = 3 kind = DATA info = 15       the 16th line
SWP: Sending frame: seq = 0 ack = 3 kind = DATA info = 16       the 17th line
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 17       the 18th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 18       the 19th line
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 19       the 20th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 20       the 21th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 21       the 22th line
SWP: Sending frame: seq = 6 ack = 4 kind = DATA info = 22       the 23th line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 0 ack = 4 kind = ACK info =
SWP: Sending frame: seq = 7 ack = 4 kind = DATA info = 23       the 24th line
SWP: Sending frame: seq = 0 ack = 4 kind = DATA info = 24       the 25th line
SWP: Sending frame: seq = 1 ack = 4 kind = DATA info = 25       the 26th line
SWP: Sending frame: seq = 2 ack = 4 kind = DATA info = 26       the 27th line
SWP: Sending frame: seq = 3 ack = 4 kind = DATA info = 27       the 28th line
SWP: Sending frame: seq = 4 ack = 4 kind = DATA info = 28       the 29th line
SWP: Sending frame: seq = 5 ack = 4 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 6 ack = 0 kind = DATA info = 30       the 31th line
SWP: Sending frame: seq = 7 ack = 0 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 0 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 5 ack = 0 kind = DATA info = 29       the 30th line
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 7 ack = 2 kind = DATA info = 31       the 32th line
SWP: Sending frame: seq = 0 ack = 2 kind = DATA info = 32       the 33th line
SWP: Sending frame: seq = 0 ack = 2 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 3 kind = ACK info =
SWP: Sending frame: seq = 1 ack = 3 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 3 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 3 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 5 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 5 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 5 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 0 kind = NAK info =
SWP: Sending frame: seq = 1 ack = 0 kind = DATA info = 33       the 34th line
SWP: Sending frame: seq = 2 ack = 0 kind = DATA info = 34       the last line
SWP: Sending frame: seq = 0 ack = 4 kind = NAK info =
SWP: Sending frame: seq = 0 ack = 5 kind = NAK info =
```

Java NetSim 3

```
C:\Users\Jianwei\Downloads\ass1>java NetSim 3
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 61339
NetSim(Port= 54321) is waiting for connection ...
NetSim accepted connection from: DESKTOP-S7L205N : 61340
VMach 1 Check sum error for seq = 0 error counter = 1
VMach 1 loose frame seq = 2 error counter = 2
VMach 1 Check sum error for seq = 3 error counter = 3
VMach 2 loose frame seq = 1 error counter = 1
VMach 1 Check sum error for seq = 0 error counter = 4
VMach 1 Check sum error for seq = 2 error counter = 5
VMach 1 loose frame seq = 4 error counter = 6
VMach 1 Check sum error for seq = 5 error counter = 7
VMach 2 Check sum error for seq = 2 error counter = 2
VMach 1 Check sum error for seq = 2 error counter = 8
VMach 2 Check sum error for seq = 5 error counter = 3
VMach 2 Check sum error for seq = 7 error counter = 4
VMach 2 loose frame seq = 0 error counter = 5
VMach 1 Check sum error for seq = 3 error counter = 9
VMach 2 loose frame seq = 5 error counter = 6
VMach 2 Check sum error for seq = 6 error counter = 7
VMach 1 Check sum error for seq = 6 error counter = 10
VMach 1 Check sum error for seq = 0 error counter = 11
VMach 2 Check sum error for seq = 0 error counter = 8
VMach 1 Check sum error for seq = 1 error counter = 12
VMach 1 loose frame seq = 6 error counter = 13
VMach 2 loose frame seq = 0 error counter = 9
VMach 2 loose frame seq = 5 error counter = 10
VMach 2 Check sum error for seq = 6 error counter = 11
VMach 2 Check sum error for seq = 7 error counter = 12
VMach 2 Check sum error for seq = 0 error counter = 13
VMach 1 Check sum error for seq = 1 error counter = 14
VMach 2 loose frame seq = 0 error counter = 14
VMach 1 Check sum error for seq = 4 error counter = 15
VMach 2 loose frame seq = 6 error counter = 15
VMach 2 loose frame seq = 2 error counter = 16
VMach 2 loose frame seq = 3 error counter = 17
VMach 1 Check sum error for seq = 5 error counter = 16
VMach 1 Check sum error for seq = 6 error counter = 17
VMach 1 Check sum error for seq = 4 error counter = 18
VMach 2 Check sum error for seq = 3 error counter = 18
VMach 2 Check sum error for seq = 6 error counter = 19
VMach 2 loose frame seq = 7 error counter = 20
VMach 1 Check sum error for seq = 0 error counter = 19
VMach 1 Check sum error for seq = 5 error counter = 20
VMach 1 Check sum error for seq = 6 error counter = 21
VMach 1 loose frame seq = 4 error counter = 22
VMach 1 Check sum error for seq = 0 error counter = 23
VMach 2 loose frame seq = 1 error counter = 21
VMach 1 Check sum error for seq = 5 error counter = 24
VMach 1 Check sum error for seq = 6 error counter = 25
VMach 2 loose frame seq = 2 error counter = 22
VMach 1 loose frame seq = 7 error counter = 26
VMach 1 loose frame seq = 0 error counter = 27
VMach 1 Check sum error for seq = 0 error counter = 28
VMach 1 Check sum error for seq = 5 error counter = 29
VMach 1 loose frame seq = 6 error counter = 30
VMach 2 loose frame seq = 7 error counter = 23
VMach 2 Check sum error for seq = 1 error counter = 24
VMach 1 loose frame seq = 0 error counter = 31
```

```
VMach 2 Check sum error for seq = 2 error counter = 25
VMach 1 loose frame seq = 5 error counter = 32
VMach 1 Check sum error for seq = 7 error counter = 33
VMach 2 loose frame seq = 0 error counter = 26
VMach 2 Check sum error for seq = 5 error counter = 27
VMach 1 loose frame seq = 0 error counter = 34
VMach 1 loose frame seq = 1 error counter = 35
VMach 2 Check sum error for seq = 7 error counter = 28
VMach 1 Check sum error for seq = 3 error counter = 36
VMach 1 loose frame seq = 4 error counter = 37
VMach 2 loose frame seq = 0 error counter = 29
VMach 2 Check sum error for seq = 1 error counter = 30
VMach 1 loose frame seq = 0 error counter = 38
VMach 2 loose frame seq = 2 error counter = 31
VMach 1 loose frame seq = 2 error counter = 39
VMach 1 loose frame seq = 5 error counter = 40
VMach 2 loose frame seq = 0 error counter = 32
VMach 1 Check sum error for seq = 6 error counter = 41
VMach 2 loose frame seq = 0 error counter = 33
VMach 1 loose frame seq = 4 error counter = 42
VMach 2 loose frame seq = 0 error counter = 34
VMach 2 loose frame seq = 1 error counter = 35
VMach 2 loose frame seq = 2 error counter = 36
VMach 1 loose frame seq = 5 error counter = 43
VMach 1 Check sum error for seq = 6 error counter = 44
VMach 1 loose frame seq = 3 error counter = 45
VMach 2 Check sum error for seq = 0 error counter = 37
VMach 2 loose frame seq = 1 error counter = 38
VMach 1 loose frame seq = 1 error counter = 46
VMach 1 Check sum error for seq = 1 error counter = 47
VMach 1 Check sum error for seq = 2 error counter = 48
VMach 1 Check sum error for seq = 0 error counter = 49
VMach 2 loose frame seq = 2 error counter = 39
VMach 1 loose frame seq = 6 error counter = 50
VMach 2 loose frame seq = 0 error counter = 40
VMach 2 Check sum error for seq = 0 error counter = 41
VMach 1 Check sum error for seq = 0 error counter = 51
VMach 1 loose frame seq = 6 error counter = 52
VMach 1 loose frame seq = 5 error counter = 53
VMach 1 loose frame seq = 7 error counter = 54
VMach 1 loose frame seq = 0 error counter = 55
VMach 1 loose frame seq = 5 error counter = 56
VMach 1 Check sum error for seq = 7 error counter = 57
VMach 1 Check sum error for seq = 0 error counter = 58
VMach 1 Check sum error for seq = 1 error counter = 59
VMach 1 Check sum error for seq = 2 error counter = 60
VMach 1 loose frame seq = 1 error counter = 61
VMach 2 Check sum error for seq = 0 error counter = 42
VMach 1 Check sum error for seq = 2 error counter = 62
```

3. Source Code

```
/*==================================================================*

File: SWP.java
This class implements the sliding window protocol
Used by VMach class
Uses the following classes: SWE, Packet, PFrame, PEvent
Author: Professor SUN Chengzheng
School of Computer Engineering
Nanyang Technological University
Singapore 639798
 *==================================================================*/
import java.util.Timer;
import java.util.TimerTask;
public class SWP {

/*=====================================================================
=====*
 the following are provided, do not change them!!

*======================================================================
====*/
   //the following are protocol constants.
   public static final int MAX_SEQ = 7;
   public static final int NR_BUFS = (MAX_SEQ + 1)/2;

   // the following are protocol variables
   private int oldest_frame = 0;
   private PEvent event = new PEvent();
   private Packet out_buf[] = new Packet[NR_BUFS];
           //declare in_buffer which is a packet data type
   private Packet in_buffer[] = new Packet[NR_BUFS];
   //the following are used for simulation purpose only
   private SWE swe = null;
   private String sid = null;
   //Constructor
   public SWP(SWE sw, String s){
      swe = sw;
      sid = s;
   }

   //the following methods are all protocol related
   private void init(){
```

```
    for (int i = 0; i < NR_BUFS; i++){
            out_buf[i] = new Packet();
    }
  }

  private void wait_for_event(PEvent e){
    swe.wait_for_event(e); //may be blocked
    oldest_frame = e.seq;  //set timeout frame seq
  }

  private void enable_network_layer(int nr_of_bufs) {
  //network layer is permitted to send if credit is available
        swe.grant_credit(nr_of_bufs);
  }

  private void from_network_layer(Packet p) {
    swe.from_network_layer(p);
  }

  private void to_network_layer(Packet packet) {
        swe.to_network_layer(packet);
  }

  private void to_physical_layer(PFrame fm)  {
    System.out.println("SWP: Sending frame: seq = " + fm.seq +
                            " ack = " + fm.ack + " kind = " +
                            PFrame.KIND[fm.kind] + " info = " + fm.info.data );
    System.out.flush();
    swe.to_physical_layer(fm);
  }

  private void from_physical_layer(PFrame fm) {
    PFrame fm1 = swe.from_physical_layer();
        fm.kind = fm1.kind;
        fm.seq = fm1.seq;
        fm.ack = fm1.ack;
        fm.info = fm1.info;
  }

/*==================================================================
========*
implement your Protocol Variables and Methods below:

*==================================================================
======*/
```

```
  boolean no_nak = true; //no nak has been sent yet
 //between method that is similar to protocol 5. Checks the frame numbers if it is valid
public static boolean between(int a, int b, int c){
        return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a));
   }


  //A java method to send the frames
private void send_frame(int frame_kind, int next_frame, int frame_expected, Packet buffer[])
        {
 // Construct and send a data, ack, or nak frame.
     PFrame s = new PFrame();

    s.kind = frame_kind; // 3 kind of frames == data, ack, or nak
       if (frame_kind == PFrame.DATA) {
          s.info = buffer[next_frame % NR_BUFS];
       }
       s.seq = next_frame; //only meaningful for data frames
       s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
       if (frame_kind == PFrame.NAK) {
          no_nak = false; //one nak per frame
       }
       to_physical_layer(s); //transmit the frame
       if (frame_kind == PFrame.DATA) {
          start_timer(next_frame);
       }
       stop_ack_timer(); //no need for separate ack frame
    }



   public void protocol6() {
       init();

          int expected_ack;                    //expected frame acknowledgement
          int next_frame_send;                 //Frame number of the next frame to be sent
          int expected_frame;                  //Frame expected to be recieved by reciever
          int index;                           //index of the buffer
          int upper_limit;                     //upper limit of the buffer


          boolean recieved[] = new boolean[NR_BUFS];       //To Keep track of frames arrived

          PFrame temp_frame = new PFrame();                //declare the temporary frame
```

```
  enable_network_layer(NR_BUFS);                    //initialize network layer

  //initialize the counter variables
  expected_ack = 0;
  next_frame_send = 0;
  expected_frame = 0;
  upper_limit = NR_BUFS;
  index = 0;
  for(int i = 0;i<NR_BUFS;i++)
          recieved[i] = false;

  while(true) {
wait_for_event(event);
    switch(event.type) {
      case (PEvent.NETWORK_LAYER_READY):
            //When network layer is ready, fetch new packets from network layer and put the
            packets in the out buffer of the sender
            from_network_layer(out_buf[next_frame_send%NR_BUFS]);
          //transmit data fetched, which is in the senders' buffer from the network layer over
          the network

            send_frame(PFrame.DATA,next_frame_send,expected_frame,out_buf);
            //Next frame to be sent from the buffer will be incremented
            next_frame_send = inc(next_frame_send);
       break;
      case (PEvent.FRAME_ARRIVAL ):
            from_physical_layer(temp_frame);





            if (temp_frame.kind == PFrame.DATA)
            {
   //A complete and undamaged frame is recieved
   if ((temp_frame.seq != expected_frame) && no_nak)
              {
   send_frame(PFrame.NAK, 0, expected_frame, out_buf);
        }
  else
  //If the recieved frame is not a data frame
  {
   start_ack_timer();
   }
```

```
//Check if the frame recieved is between the expected frames of the sliding window and recieved
frame is not duplicated

if (between(expected_frame, temp_frame.seq,upper_limit) && (recieved[temp_frame.seq %
NR_BUFS] == false)) {

  //This allows the frames to be accepted in any order of arrival in the reciever
   recieved[temp_frame.seq % NR_BUFS] = true;

  //If the data frame recieved is not damaged and falls between the expected frame and the limit
  of the sliding window buffer, add the frame into the input buffer

   in_buffer[temp_frame.seq % NR_BUFS] = temp_frame.info;
   while (recieved[expected_frame % NR_BUFS]) {

//Pass frames recieved from the physical layer by the sender to the network layer
to_network_layer(in_buffer[expected_frame % NR_BUFS]);
no_nak = true;
//mark the undamaged data frame recieved by the reciever as recieved
recieved[expected_frame % NR_BUFS] = false;

//Increment the lower expected in the sliding window when the complete expected frame is
recieved by the reciever machine
expected_frame = inc(expected_frame);
//When a complete and undamaged frame is recieved, the upper edge of the sliding window is
also incremented
                    upper_limit = inc(upper_limit);
                   //start the ack timer
                      start_ack_timer();
                   }
                }
                   }
//If the frame in the reciever's side is NAK, we will check that the frame is in between the current
sliding window and resend the data of the frame which the NAK has been recieved
        if ((temp_frame.kind == PFrame.NAK)
                && between(expected_ack,
                ((temp_frame.ack + 1) % (MAX_SEQ   +1)), next_frame_send))
                   {

//send the data of the frame for which NAK has been recieved by the sender of the data
     send_frame(PFrame.DATA, ((temp_frame.ack + 1)%(MAX_SEQ + 1)),
     expected_frame, out_buf);
             }

while (between(expected_ack, temp_frame.ack,next_frame_send)) {
            //If a complete and undamaged frame is recieved
```

```
            stop_timer(expected_ack % NR_BUFS);
        //In the sender's sliding window increase the expected sliding window for the ack to be
        recieved for the sent data frames
            expected_ack = inc(expected_ack);

                //always free 1 buffer slot if ack has been done
            enable_network_layer(1);
          }

        break;
          case (PEvent.CKSUM_ERR):
          if (no_nak) {
            //damaged frame
            send_frame(PFrame.NAK, 0, expected_frame,out_buf);
          }
          break;
          case (PEvent.TIMEOUT):
        //If the timer is expired for the oldest frame,then resend the data for the frame for which
        the timer has expired
          send_frame(PFrame.DATA, oldest_frame,expected_frame, out_buf);
              break;
            case (PEvent.ACK_TIMEOUT):
        //If ack timer expired in the reciever's side for the frame that has been recieved, send the
        ACK again
          send_frame(PFrame.ACK, 0, expected_frame,out_buf);
            break;
        default:
        System.out.println("SWP: undefined event type = " + event.type);
        System.out.flush();
          }
    }
  }

/* Note: when start_timer() and stop_timer() are called,
   the "seq" parameter must be the sequence number, rather
   than the index of the timer array,
   of the frame associated with this timer,
  */
   Timer frame_timer[] = new Timer[NR_BUFS];
   Timer ack_timer;

  public static int inc(int num) {
      num = ((num + 1) % (MAX_SEQ + 1));
      return num;
   }
```

```
private void start_timer(int seq) {
   stop_timer(seq);
   //create new timer
   frame_timer[seq % NR_BUFS] = new Timer();
   //schedule the task for execution after 200ms
   frame_timer[seq % NR_BUFS].schedule(new ReTask(swe, seq), 200);
}

private void stop_timer(int seq) {
        if (frame_timer[seq % NR_BUFS] != null) {
        frame_timer[seq % NR_BUFS].cancel();
      }
}

private void start_ack_timer( ) {
        stop_ack_timer();
     //starts another timer for sending separate ack
     ack_timer = new Timer();
     ack_timer.schedule(new AckTask(swe), 100);
}

private void stop_ack_timer() {
        if (ack_timer != null) {
        ack_timer.cancel();
     }
}

//for retransmission timer
 class ReTask extends TimerTask {
     private SWE swe = null;
     public int seqnr;

     public ReTask(SWE sw, int seq) {
        swe = sw;
        seqnr = seq;
     }

     public void run() {
        //stops this timer and discard any
        //scheduled tasks for the current seqnr
        stop_timer(seqnr);
        swe.generate_timeout_event(seqnr);
     }
 }

 //for ack timer
```

```
    class AckTask extends TimerTask {

        private SWE swe = null;

        public AckTask(SWE sw) {
            swe = sw;
        }

        public void run() {
            // stop the timer
            stop_ack_timer();
            swe.generate_acktimeout_event();
        }
    }

}//End of class

/* Note: In class SWE, the following two public methods are available:
   . generate_acktimeout_event() and
   . generate_timeout_event(seqnr).

   To call these two methods (for implementing timers),
   the "swe" object should be referred as follows:
     swe.generate_acktimeout_event(), or
     swe.generate_timeout_event(seqnr).
*/
```