Networks Project 1 Report

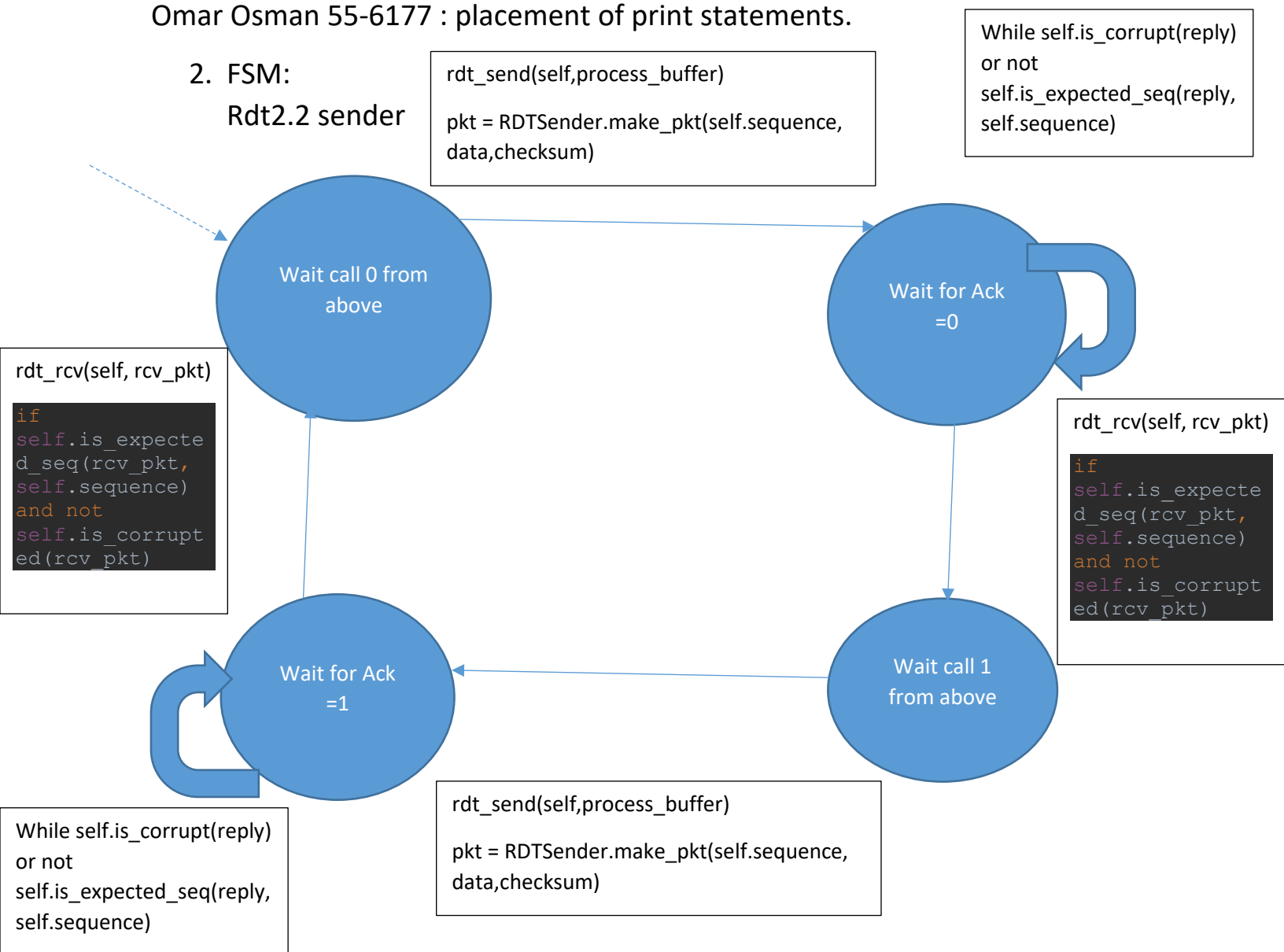
1. Names and contribution :

Kerollos Ashraf 55-25131: Sender.py rdt_send(self, process_buffer), is_corrupted(reply), is_expected_seq(reply, exp_seq), get_checksum(data), color of print statements, report.

Youssef Abdelsatar 55-11114: receiver.py rdt_rcv(self, rcv_packet), is_expected_seq(rcv_packet, exp_seq), is_corrupted(packet).

Omar Osman 55-6177 : placement of print statements.

2. FSM:
   Rdt2.2 sender

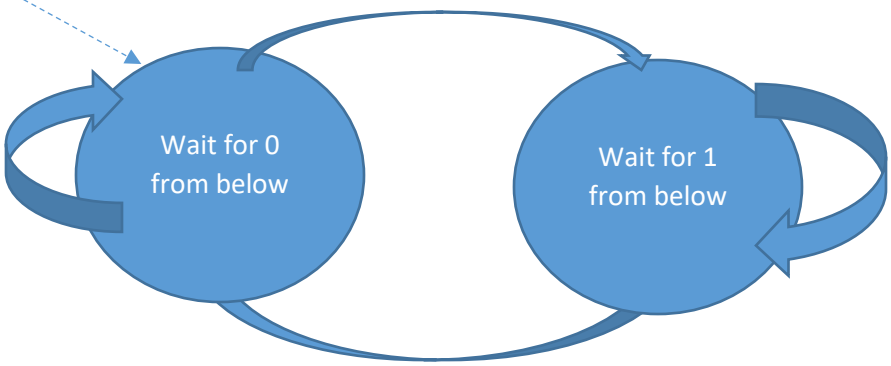


rdt2.2 receiver

```
if self.is_expected_seq(rcv_pkt, self.sequence)
and not self.is_corrupted(rcv_pkt):
ReceiverProcess.deliver_data(rcv_pkt['data'])
reply_pkt =
RDTReceiver.make_reply_pkt(self.sequence,
ord(self.sequence))
if self.sequence == '0':
    self.sequence = '1'
else:
    self.sequence = '0'
return reply_pkt
```

rdt_rcv(self, rcv_pkt) if self.is_corrupted(rcv_pkt) or not is_expected_seq(rcv_pkt, self.sequence)

```
if self.sequence ==
'0':
    reverse = '1'
else:
    reverse = '0'
```



rdt_rcv(self, rcv_pkt) if self.is_corrupted(rcv_pkt) or not is_expected_seq(rcv_pkt, self.sequence)

```
if self.sequence ==
'0':
    reverse = '1'
else:
    reverse = '0'
```

```
reply_pkt =
RDTReceiver.make_rep
ly pkt(reverse,
ord(reverse))
```

return reply_pkt

```
if self.is_expected_seq(rcv_pkt, self.sequence)
and not self.is_corrupted(rcv_pkt):
ReceiverProcess.deliver_data(rcv_pkt['data'])
reply_pkt =
RDTReceiver.make_reply_pkt(self.sequence,
ord(self.sequence))
if self.sequence == '0':
    self.sequence = '1'
else:
    self.sequence = '0'
```

```
reply_pkt =
RDTReceiver.make_rep
ly_pkt(reverse,
ord(reverse))
```

return reply_pkt

3. Pseudo code:

Sender rdt_send(self, process_buffer):

The compiler loop around the buffer to check each part of the data sent

While the compiler is in the loop it creates a checksum of the data and creates a packet to be sent. It then stores the packet in a variable so if a corruption occurs we do not lose the original packet. Then the packet is sent.

After the data it is checked to see if it has any corruption if it does the data keeps on getting sent until there is no corruption. Also the sequence number keeps getting switched for the receiver to make sure the data is not the same one that was transmitted.

Loop around buffer

Checksum of data is get_checksum(data)

Create a packet called packet with sequence number, data and checksum

Tmp = pkt

Send packet

If packet is corrupted

{Loop and keep sending till no corruption

Change sequence number}

Print sender is done.

Receiver rdt_rcv(self,rcv_pkt)

In the rdt_rcv(self,rcv_pkt) the first thing that is checked is if any corruption happened to the data if there was no corruption the receiver delivers the data to the process in the application layer and then create a reply packet then we change the sequence number to ensure that the next data sent is new.

If there is a corruption

{The compiler creates a packet with a different sequence number.

If packet is not corrupted

Deliver data of packet to application layer

Reverse sequence number}

Else

{Make reply packet with reverse sequence number}

Return the reply packet

4. Edits of skeleton:

In the sender

Firstly we edited the get_checksum(data) method so we can get the data to check it later

Secondly we edited the is_corrupted(reply) method so we can use it later on in other methods to see if the data was corrupted.

Thirdly we edited the is_expected_sequence(reply, exp_seq) to check if the sequence number of the reply has the expected sequence.

Finally in the sender in the rdt_send(self, process_buffer) method we added a loop , some print statements and some extra functionality so the code could work.

In the receiver

Firstly we edited the method is_corrupted(packet) so we can later on check if the data of the packet is corrupted or not

Secondly we edited the method is_expected_sequence(rcv_pkt, exp_seq) to check if the reply packet has the expected sequence number

Finally we edited the rdt_rcv(self,rcv_pkt) so that we change the sequence number to make sure the data is not retransmitted.

5. Screen shots of test cases in cmd:

Low reliability :

```
C:\Users\kerol\OneDrive\Desktop\networks\networks>python main.py msg=TEST_report rel=0.2 delay=0 debug=0 pkt=1 ack=1
{'msg': 'TEST_report', 'rel': '0.2', 'delay': '0', 'debug': '0', 'pkt': '1', 'ack': '1'}
Sender is sending:TEST_report
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'T', 'checksum': 84}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'E', 'checksum': 69}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'S', 'checksum': 83}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'T', 'checksum': 84}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': '_', 'checksum': 95}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'e', 'checksum': 101}
network layer:corruption occurred
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'p', 'checksum': 112}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'o', 'checksum': 111}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 't', 'checksum': 116}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T', '_', 'r', 'e', 'p', 'o', 'r', 't']
```

Medium reliability:

```
C:\Users\kerol\OneDrive\Desktop\networks\networks>python main.py msg=TEST_report rel=0.5 delay=0 debug=0 pkt=1 ack=1
{'msg': 'TEST_report', 'rel': '0.5', 'delay': '0', 'debug': '0', 'pkt': '1', 'ack': '1'}
Sender is sending:TEST_report
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'T', 'checksum': 84}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'E', 'checksum': 69}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'S', 'checksum': 83}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'T', 'checksum': 84}
network layer:corruption occurred
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': '_', 'checksum': 95}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'e', 'checksum': 101}
network layer:corruption occurred
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'p', 'checksum': 112}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'o', 'checksum': 111}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
network layer:corruption occurred
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 't', 'checksum': 116}
network layer:corruption occurred
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T', '_', 'r', 'e', 'p', 'o', 'r', 't']
```

High reliability:

```
C:\Users\kerol\OneDrive\Desktop\networks\networks>python main.py msg=TEST_report rel=0.8 delay=0 debug=0 pkt=1 ack=1
{'msg': 'TEST_report', 'rel': '0.8', 'delay': '0', 'debug': '0', 'pkt': '1', 'ack': '1'}
Sender is sending:TEST_report
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'T', 'checksum': 84}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'E', 'checksum': 69}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'S', 'checksum': 83}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'T', 'checksum': 84}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': '_', 'checksum': 95}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'e', 'checksum': 101}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'p', 'checksum': 112}
network layer:corruption occurred
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'o', 'checksum': 111}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 't', 'checksum': 116}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T', '_', 'r', 'e', 'p', 'o', 'r', 't']
```

Certain reliability:

```
C:\Users\kerol\OneDrive\Desktop\networks\networks>python main.py msg=TEST_report rel=1 delay=0 debug=0 pkt=1 ack=1
{'msg': 'TEST_report', 'rel': '1', 'delay': '0', 'debug': '0', 'pkt': '1', 'ack': '1'}
Sender is sending:TEST_report
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'T', 'checksum': 84}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'E', 'checksum': 69}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'S', 'checksum': 83}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'T', 'checksum': 84}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': '_', 'checksum': 95}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'e', 'checksum': 101}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'p', 'checksum': 112}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 'o', 'checksum': 111}
Receiver:expecting seq num:0
Receiver: reply with:{'ack': '1', 'checksum': 49}
sender expecting seq num:1
sender sending:{'sequence_number': '1', 'data': 'r', 'checksum': 114}
Receiver:expecting seq num:1
Receiver: reply with:{'ack': '0', 'checksum': 48}
sender expecting seq num:0
sender sending:{'sequence_number': '0', 'data': 't', 'checksum': 116}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T', '_', 'r', 'e', 'p', 'o', 'r', 't']
```