Team members:
Amr Khaled 55-9601 T-27 amr.mowafi@student.guc.edu.eg
Khaled Amr 55-19066 T-27 khaled.amr@student.guc.edu.eg
Yousef abdelhamid 55-26578 T-27 yousef.abdellatif@student.guc.edu.eg

Contributions:It was all a team effort. We did the project all together.

Pseudocode of the reciever side:
Class: ReceiverProcess
Attribute: __buffer (a list to store received data)

Method: deliver_data(data)
Append the given data to the buffer.

Method: get_buffer()
Retrieve the buffer of received data.

Class: RDTReceiver
Attribute: sequence (a character representing the current sequence number, initialized to ’0’)

Method: __init__()
Initialize the sequence number to ’0’.

Method: is_corrupted(packet)
Check if the received packet is corrupted by comparing the ASCII code of the data with the checksum.

Method: is_expected_seq(rcv_pkt, exp_seq)
Check if the received packet has the expected sequence number.

Method: make_reply_pkt(seq, checksum)
Create a reply packet with the given sequence number and checksum.

Method: rdt_rcv(rcv_pkt)
If the received packet is corrupted or if the received packet has an unexpected sequence number:
- Print a network layer error message.
- Print the expected sequence number.
- Create and print a reply packet with the opposite sequence number and its ASCII code as check sum.

Else:
- Print the expected sequence number.
- Create and print a reply packet with the current sequence number and its ASCII code as checksum.
- Toggle the sequence number for the next iteration.
- Deliver the data to ReceiverProcess.

Return the created reply packet.

Pseudocode of the sender side:

Class: SenderProcess
Attribute: __buffer (a list to store outgoing data)

Method: set_outgoing_data(buffer)
Set the outgoing message buffer to the given list.

Method: get_outgoing_data()
Retrieve the outgoing message buffer.

Class: RDTSender
Attributes:
sequence (a character representing the current sequence number, initialized to ’0’)
net_srv (a service providing the method udt_send)

Method: __init__(net_srv)
Initialize the sequence number to ’0’ and set the network service.

Method: get_checksum(data)
Calculate the checksum for the given data (ASCII code of the character).

Method: clone_packet(packet)
Create a copy of the given packet.

Method: is_corrupted(reply)
Check if the received reply from the receiver is corrupted.

Method: is_expected_seq(reply, exp_seq)
Check if the received reply from the receiver has the expected sequence number.

Method: make_pkt(seq, data, checksum)
Create an outgoing packet with the given sequence number, data, and checksum.

Method: rdt_send(process_buffer)
For each character in the process buffer:
- Print the current sequence number the sender is expecting.
- Calculate the checksum for the character.
- Create an outgoing packet.
- Print the packet being sent.
- Create a copy of the packet to send.
- Send the packet and receive a reply.

While the received reply does not have the expected sequence number or is corrupted:
- Create a new copy of the packet to send.
- Resend the packet and receive a new reply.

Toggle the sequence number for the next iteration.

Print ’Sender Done!’

Changes of the sender side:

We implemented the get_checksum method to get the checksum of the data paremeter which is the ASCI I code of the character using the function (ord). we will use it when we make a packet using our data(char acter).

We checked if the reply is corrupted or not in the is_corrupted function by checking if the checksum of the reply is equal to the ack ASCII code

We Checked if the received reply from receiver has the expected sequence number by getting the ack of the reply and checking if it is the equal the expected sequence number or not,this is beneficial for when a network layer occurs and we want to send the same data again.

In the rdt_send method we implemented the RDT v2.2 for the sender by looping on the process buffer which is a list containing characters of the message you want to send. We got the checksum and we made the packet using the character and the checksum we just got after that we cloned the packet and sent the packet to the receiver side to receive a reply.after getting the reply we check on a loop if the reply doesnt have the expected sequence or if it is corrupted, if it satisdies either of the conditions we clone the packet again and send to the receiver again until it is sent, At last we change the sequence number,if it is 1 we make it 0 and if it is 0 we make it 1.

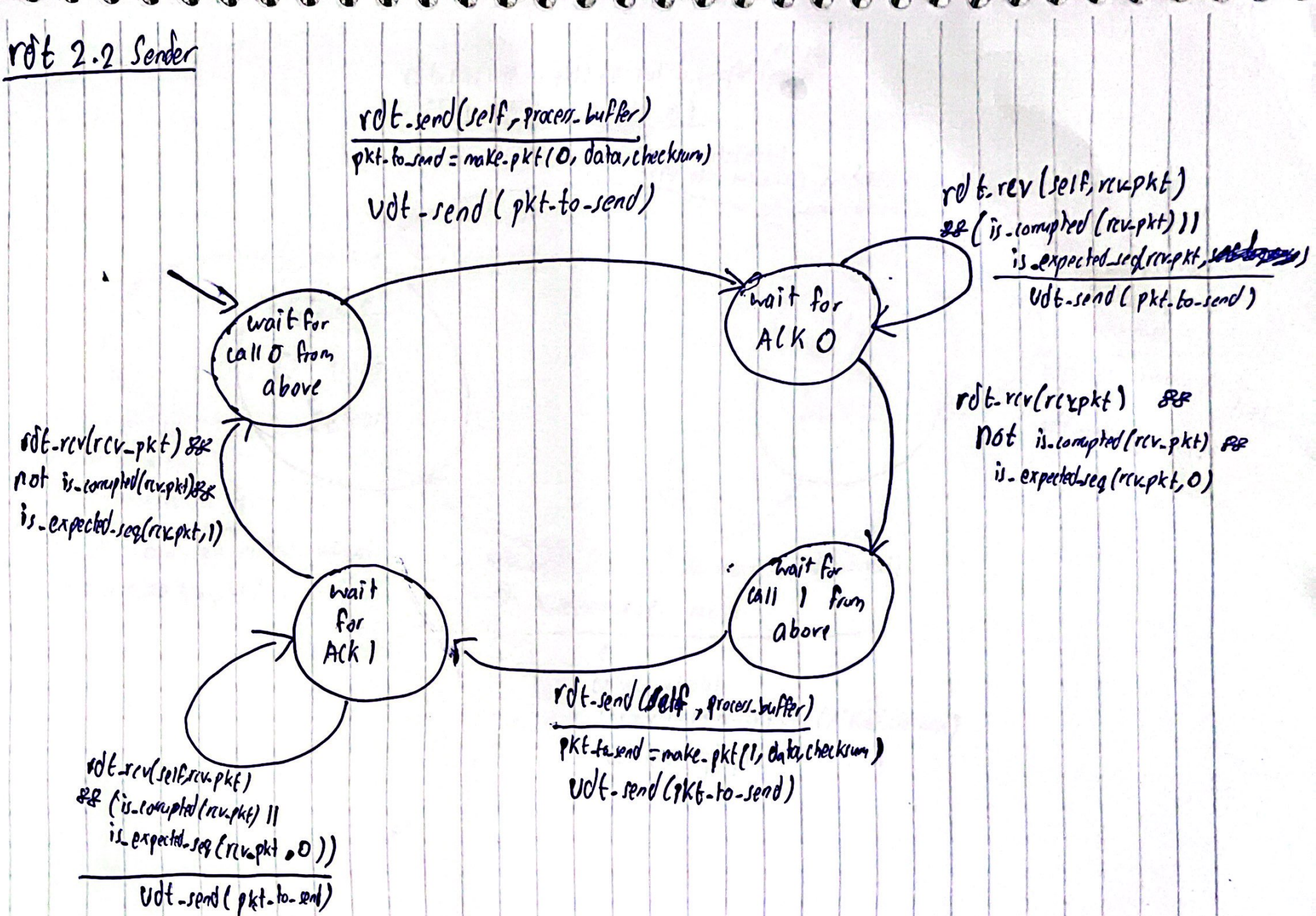
Changes of the receiver side:

We checked if the packet is corrupted or not in the is_corrupted function by checking if the checksum of the packet is equal to the data ASCII code.

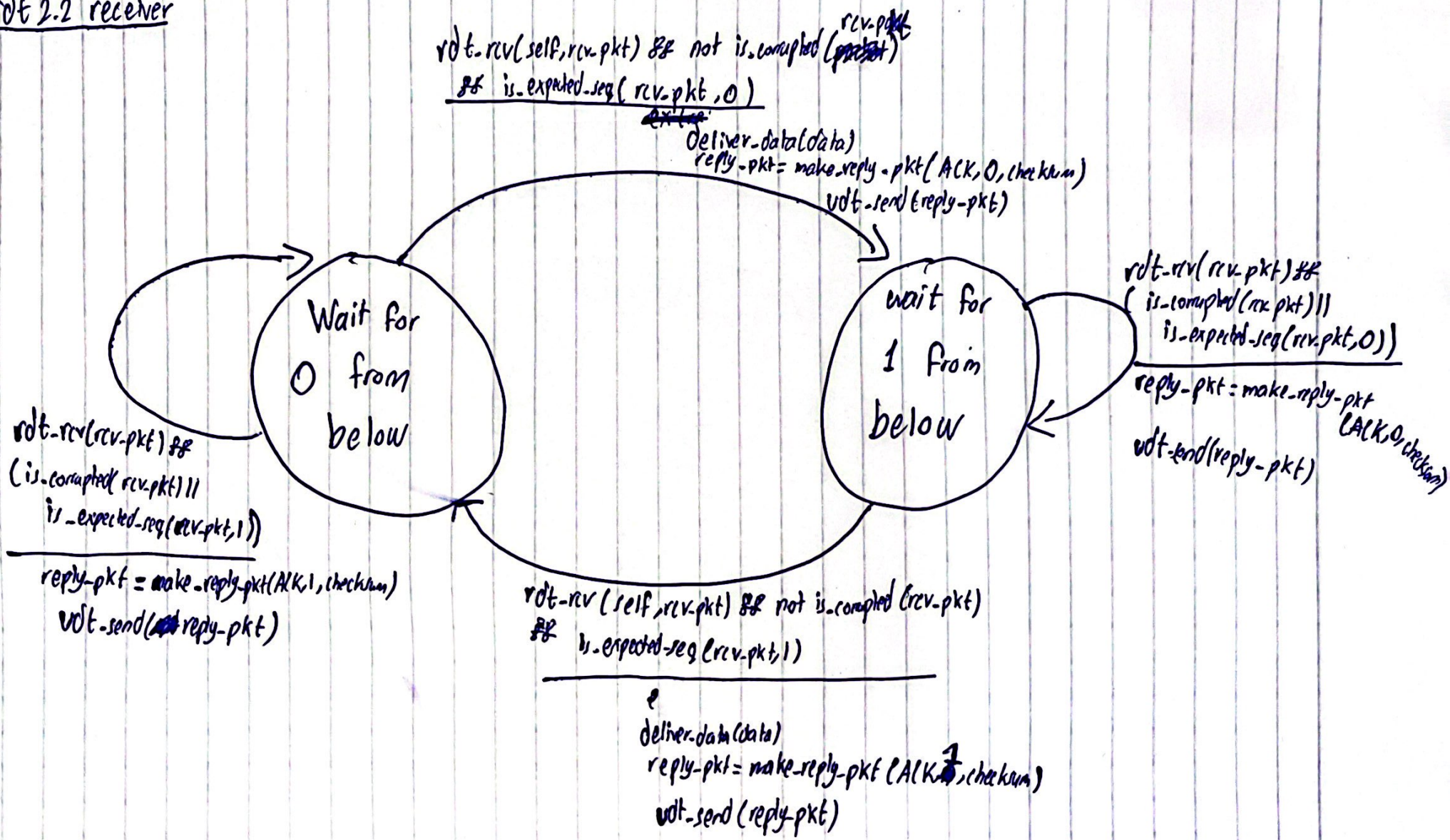
We Checked if the received packet from the sender has the expected sequence number by getting the sequence number of the packet and checking if it is the equal the expected sequence number or not.

In the rdt_rcv function we receive the packet in the paremeter so we check if it is corrupted or has the wrong sequnce number and if yes we print that a network error occured and make a reply packet with changed sequnced numbers and if no we make a reply packet without changing the sequence numbers and we change the self.sequence then we deliver the data of the received packet and return the reply packet.

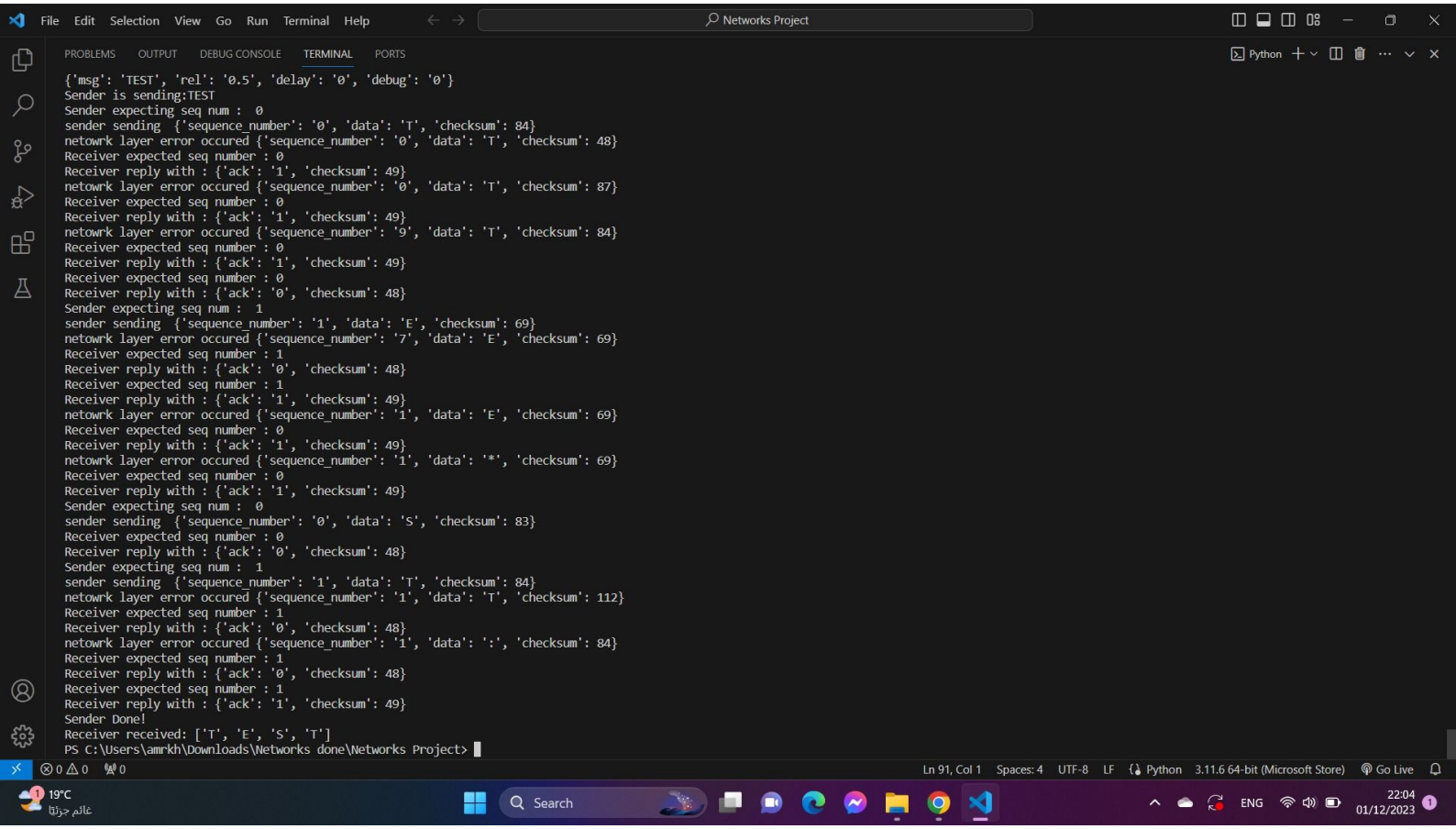
rdt 2.2 Sender

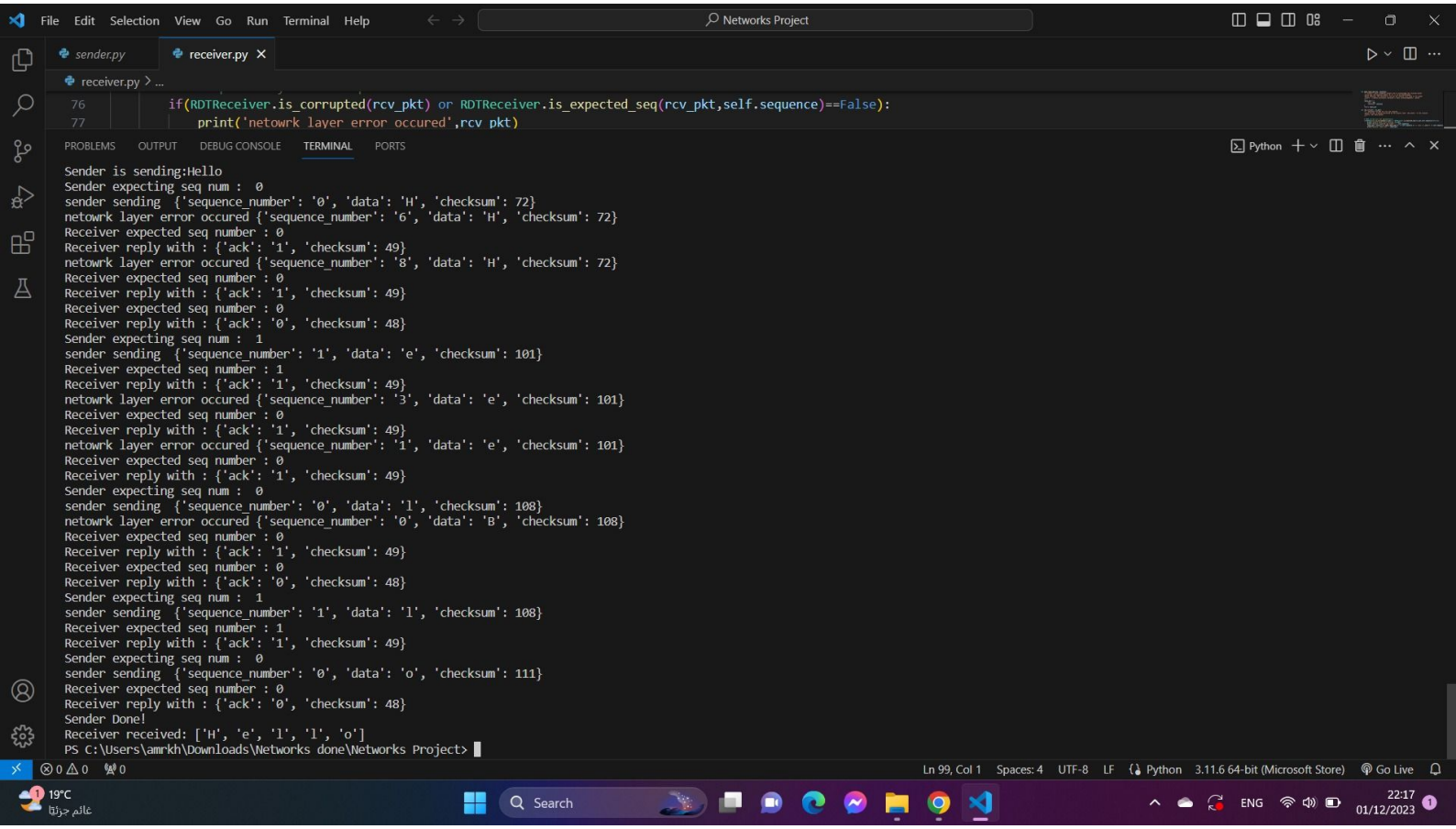


rdt 2.2 receiver



```
C:\Users\khale\Downloads\Networks Project>python main.py msg=TEST rel=1 delay=0 debug=0
{'msg': 'TEST', 'rel': '1', 'delay': '0', 'debug': '0'}
Sender is sending:TEST
Sender expecting seq num :  0
sender sending  {'sequence_number': '0', 'data': 'T', 'checksum': 84}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num :  1
sender sending  {'sequence_number': '1', 'data': 'E', 'checksum': 69}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender expecting seq num :  0
sender sending  {'sequence_number': '0', 'data': 'S', 'checksum': 83}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num :  1
sender sending  {'sequence_number': '1', 'data': 'T', 'checksum': 84}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T']
```


File Edit Selection View Go Run Terminal Help
Networks Project
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
{'msg': 'TEST', 'rel': '0.5', 'delay': '0', 'debug': '0'}
Sender is sending:TEST
Sender expecting seq num : 0
sender sending {'sequence_number': '0', 'data': 'T', 'checksum': 84}
netowrk layer error occured {'sequence_number': '0', 'data': 'T', 'checksum': 48}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '0', 'data': 'T', 'checksum': 87}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '9', 'data': 'T', 'checksum': 84}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num : 1
sender sending {'sequence_number': '1', 'data': 'E', 'checksum': 69}
netowrk layer error occured {'sequence_number': '7', 'data': 'E', 'checksum': 69}
Receiver expected seq number : 1
Receiver reply with : {'ack': '0', 'checksum': 48}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '1', 'data': 'E', 'checksum': 69}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '1', 'data': '*', 'checksum': 69}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender expecting seq num : 0
sender sending {'sequence_number': '0', 'data': 'S', 'checksum': 83}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num : 1
sender sending {'sequence_number': '1', 'data': 'T', 'checksum': 84}
netowrk layer error occured {'sequence_number': '1', 'data': 'T', 'checksum': 112}
Receiver expected seq number : 1
Receiver reply with : {'ack': '0', 'checksum': 48}
netowrk layer error occured {'sequence_number': '1', 'data': ':', 'checksum': 84}
Receiver expected seq number : 1
Receiver reply with : {'ack': '0', 'checksum': 48}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender Done!
Receiver received: ['T', 'E', 'S', 'T']
PS C:\Users\amrkh\Downloads\Networks done\Networks Project>
Ln 91, Col 1 Spaces: 4 UTF-8 LF Python 3.11.6 64-bit (Microsoft Store) Go Live
19°C
Search
ENG
22:04
01/12/2023


File Edit Selection View Go Run Terminal Help
Networks Project
sender.py
receiver.py
receiver.py > ...
76 if(RDTReceiver.is_corrupted(rcv_pkt) or RDTReceiver.is_expected_seq(rcv_pkt,self.sequence)==False):
77 print('netowrk layer error occured',rcv pkt)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Sender is sending:Hello
Sender expecting seq num : 0
sender sending {'sequence_number': '0', 'data': 'H', 'checksum': 72}
netowrk layer error occured {'sequence_number': '6', 'data': 'H', 'checksum': 72}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '8', 'data': 'H', 'checksum': 72}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num : 1
sender sending {'sequence_number': '1', 'data': 'e', 'checksum': 101}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '3', 'data': 'e', 'checksum': 101}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
netowrk layer error occured {'sequence_number': '1', 'data': 'e', 'checksum': 101}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender expecting seq num : 0
sender sending {'sequence_number': '0', 'data': 'l', 'checksum': 108}
netowrk layer error occured {'sequence_number': '0', 'data': 'B', 'checksum': 108}
Receiver expected seq number : 0
Receiver reply with : {'ack': '1', 'checksum': 49}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender expecting seq num : 1
sender sending {'sequence_number': '1', 'data': 'l', 'checksum': 108}
Receiver expected seq number : 1
Receiver reply with : {'ack': '1', 'checksum': 49}
Sender expecting seq num : 0
sender sending {'sequence_number': '0', 'data': 'o', 'checksum': 111}
Receiver expected seq number : 0
Receiver reply with : {'ack': '0', 'checksum': 48}
Sender Done!
Receiver received: ['H', 'e', 'l', 'l', 'o']
PS C:\Users\amrkh\Downloads\Networks done\Networks Project>
Ln 99, Col 1 Spaces: 4 UTF-8 LF Python 3.11.6 64-bit (Microsoft Store) Go Live
19°C
Search
ENG
22:17
01/12/2023