Networks Project Report

Team Members

Hana Eldemery 55-3310 T-19 hana.eldemery@student.guc.edu.eg

Omar Ayman 55-1600 T-19 [omar.elmeligy@student.guc.edu.eg](mailto:omar.elmeligy@student.guc.edu.eg)

Both of us implemented the code together with Hana focusing on the sender class and Omar focusing on the receiver class. We then created this report equally.

Tests

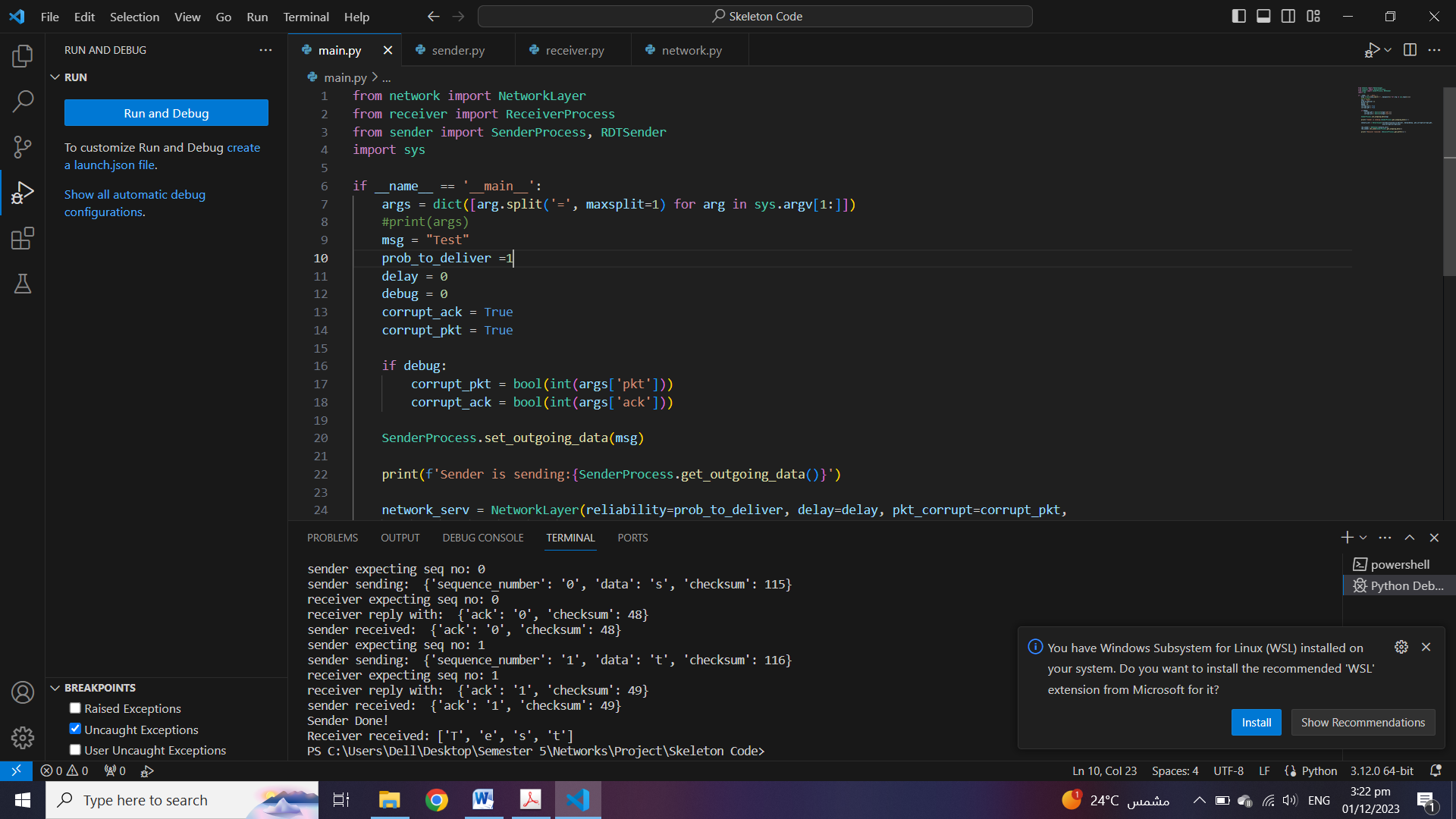
Test 1:

Msg=”Test “

Prob\_to\_deliver=1(reliability=1)

Delay=0

Debug=0



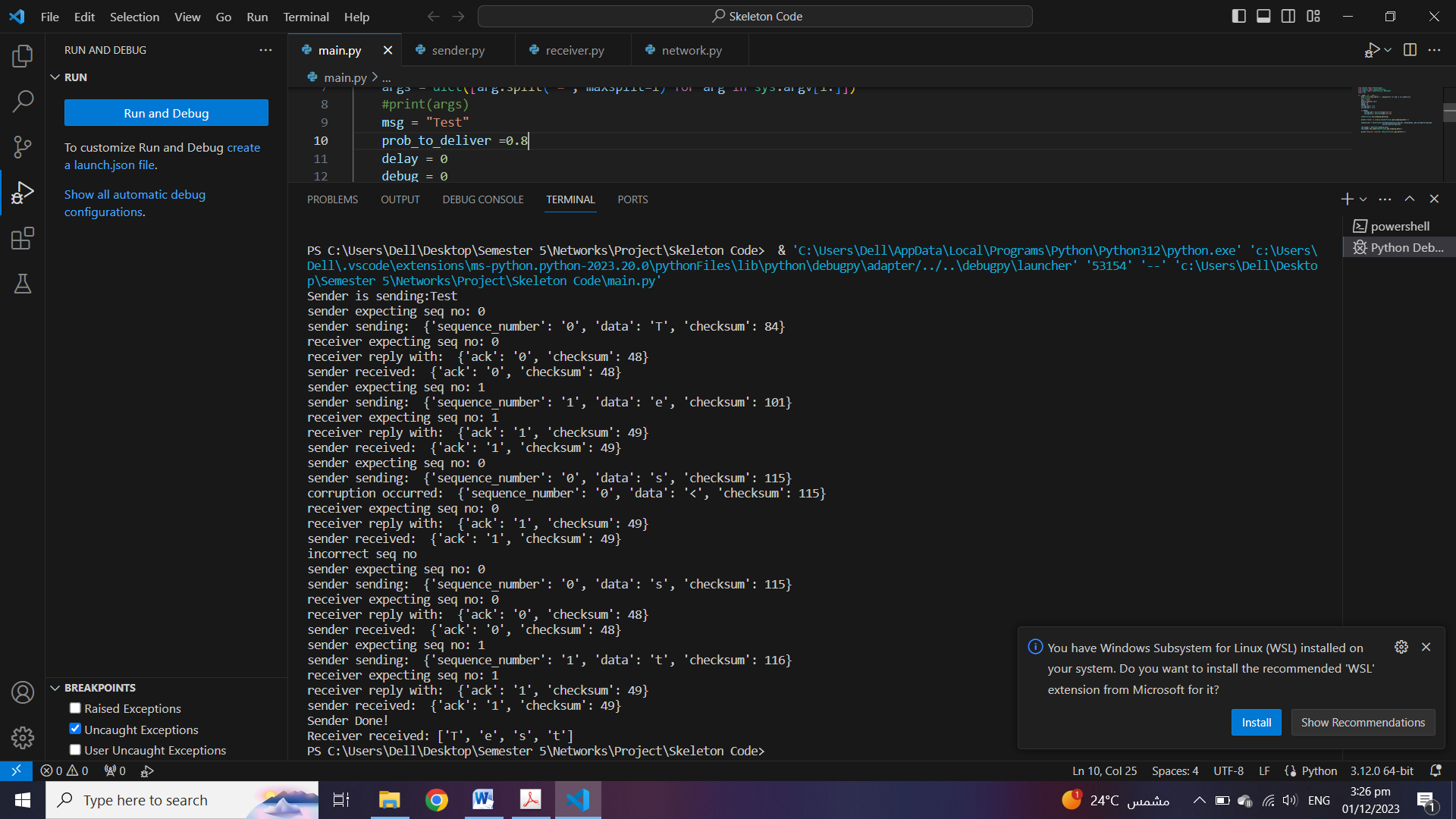
Test 2:

Msg=”Test “

Prob\_to\_deliver=0.8(reliability=0.8)

Delay=0

Debug=0



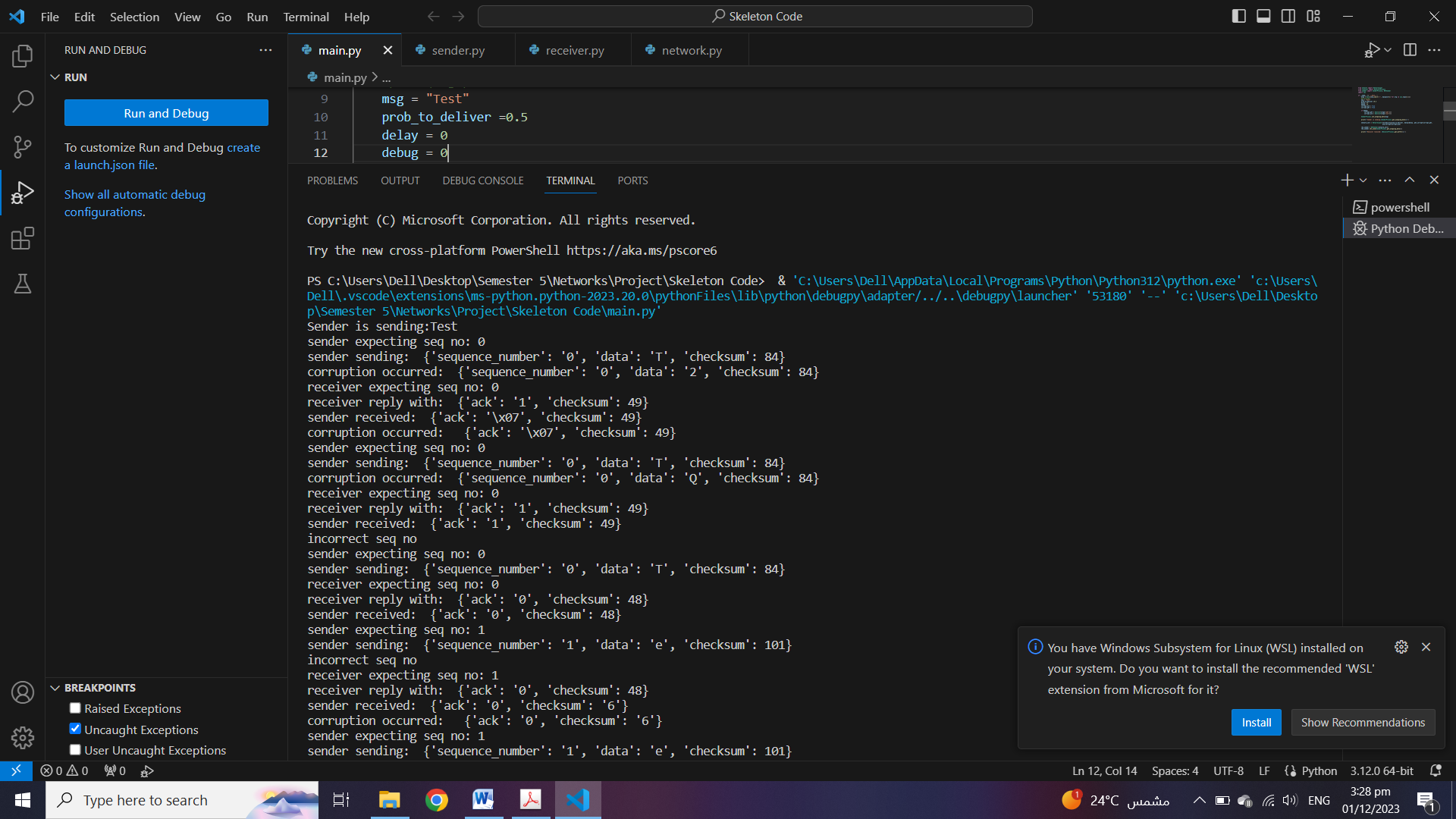
Test 3:

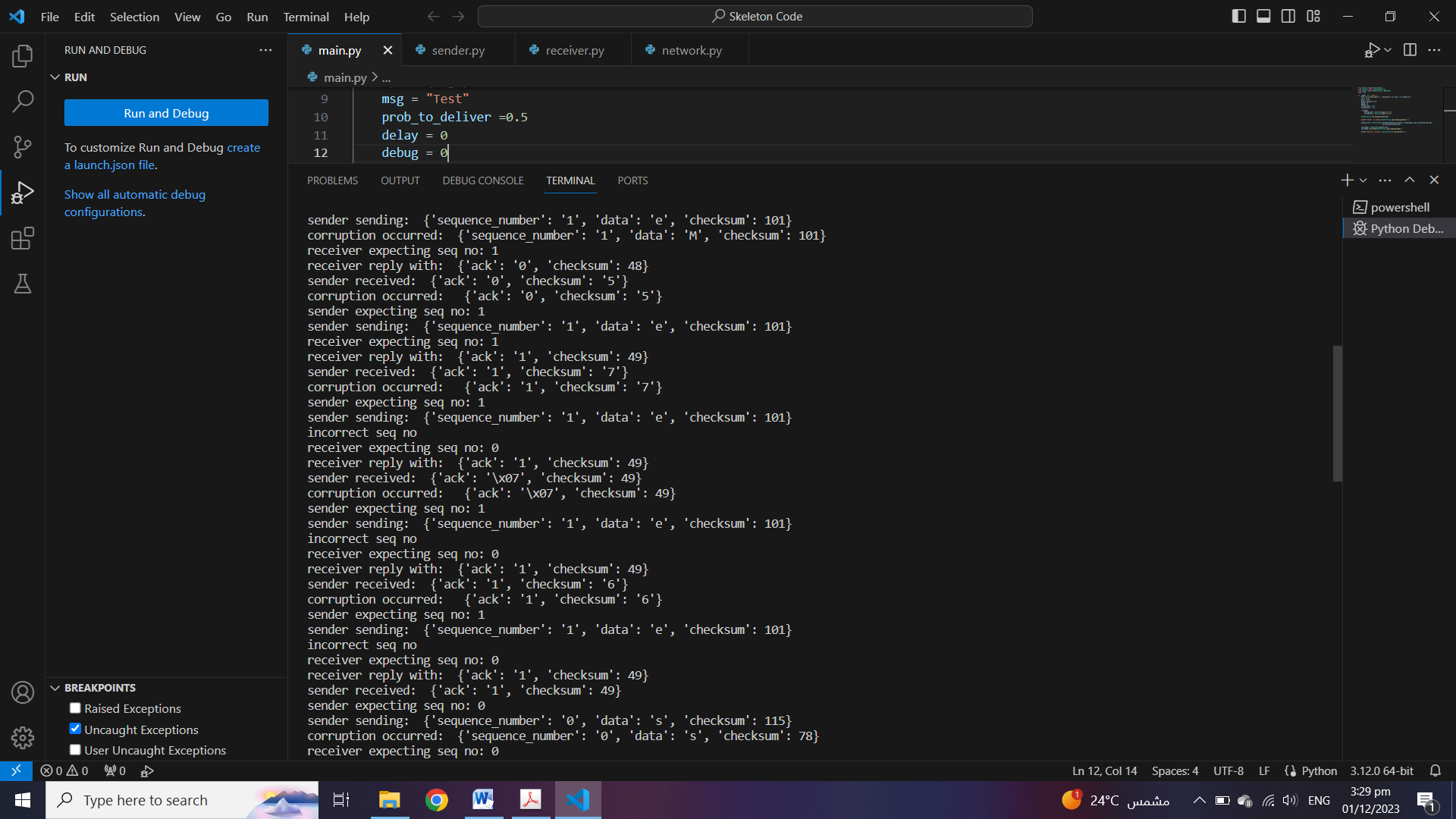
Msg=”Test “

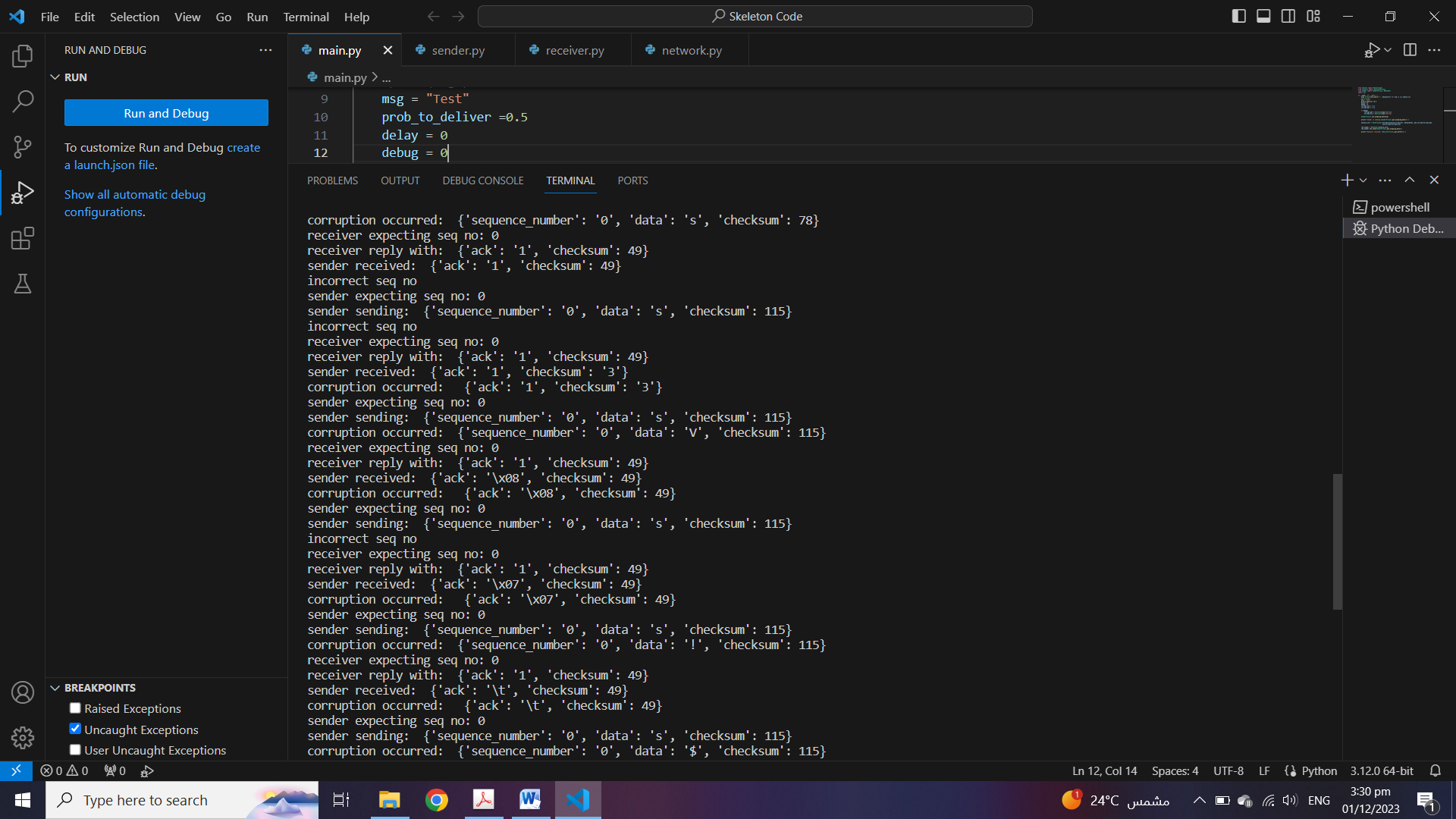
Prob\_to\_deliver=0.5(reliability=0.5)

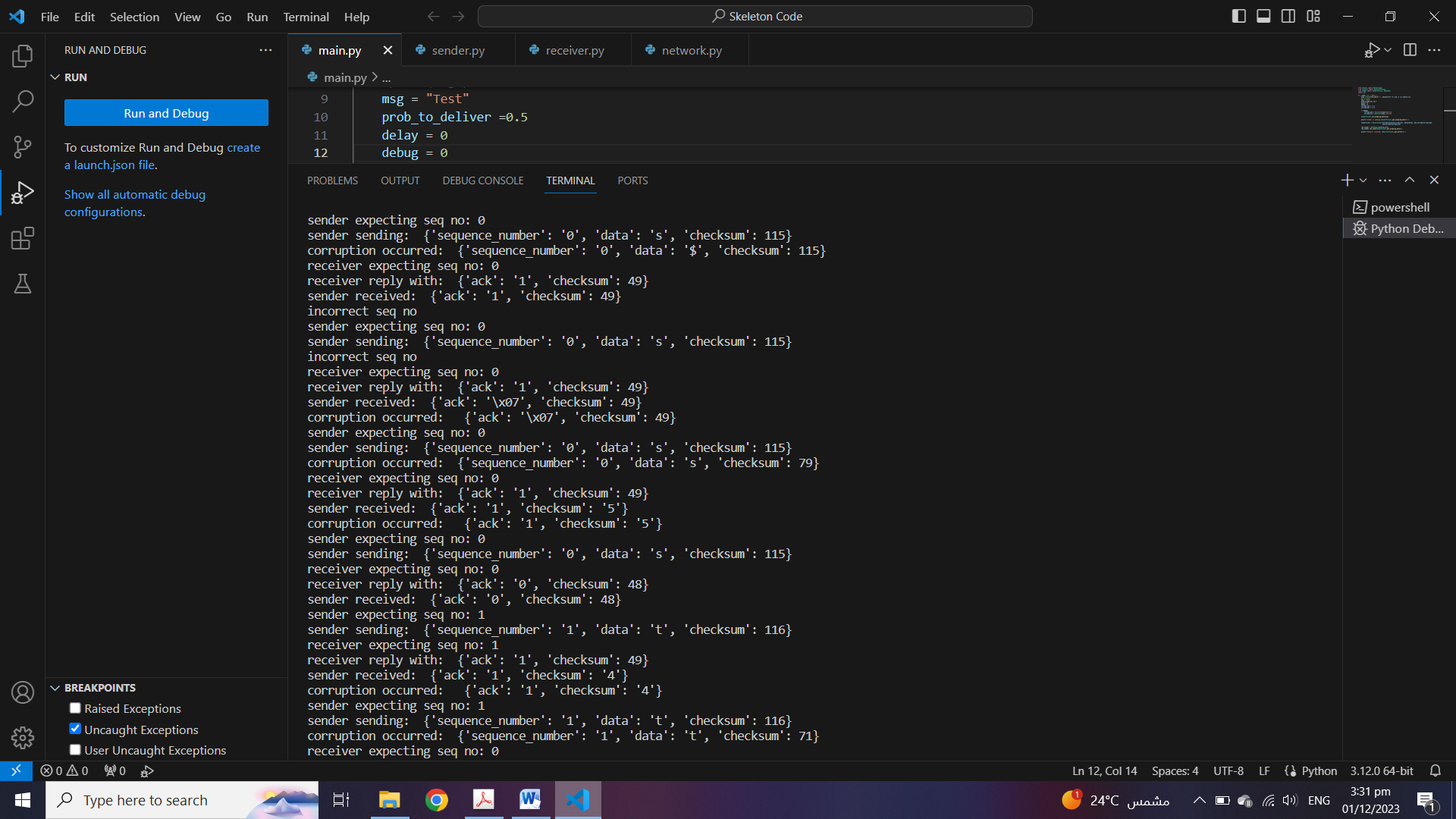
Delay=0

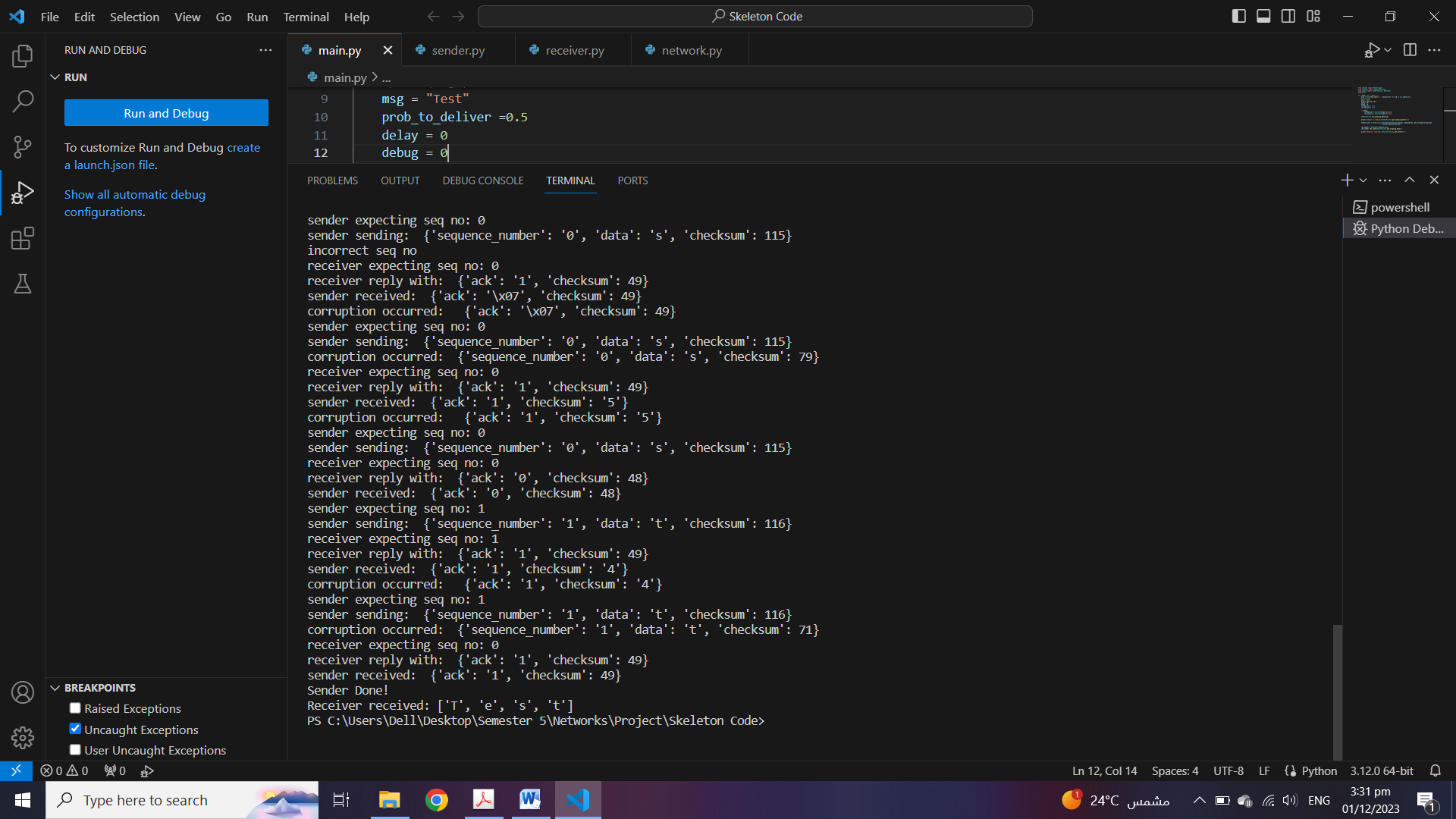
Debug=0











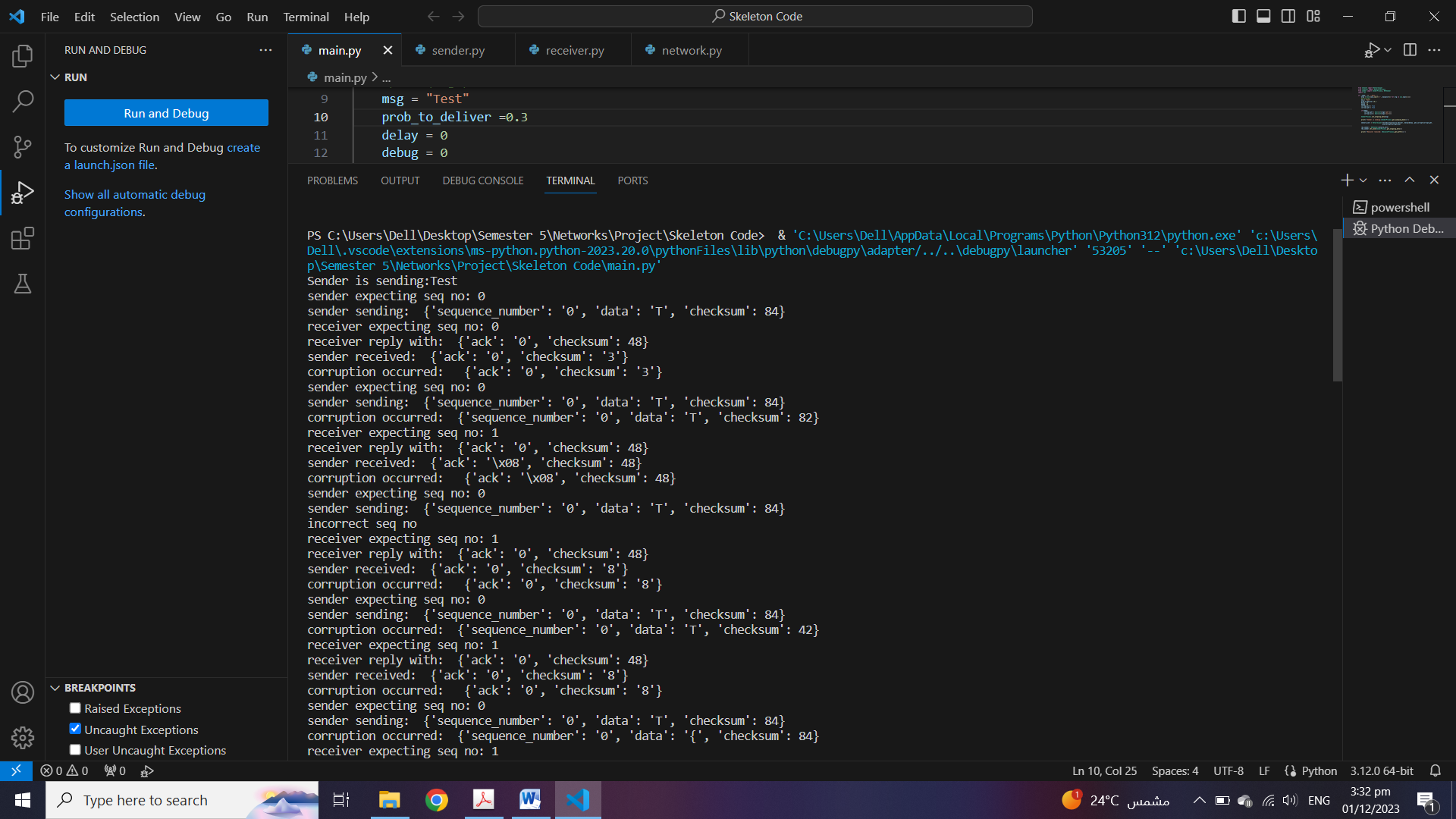
Test 4:

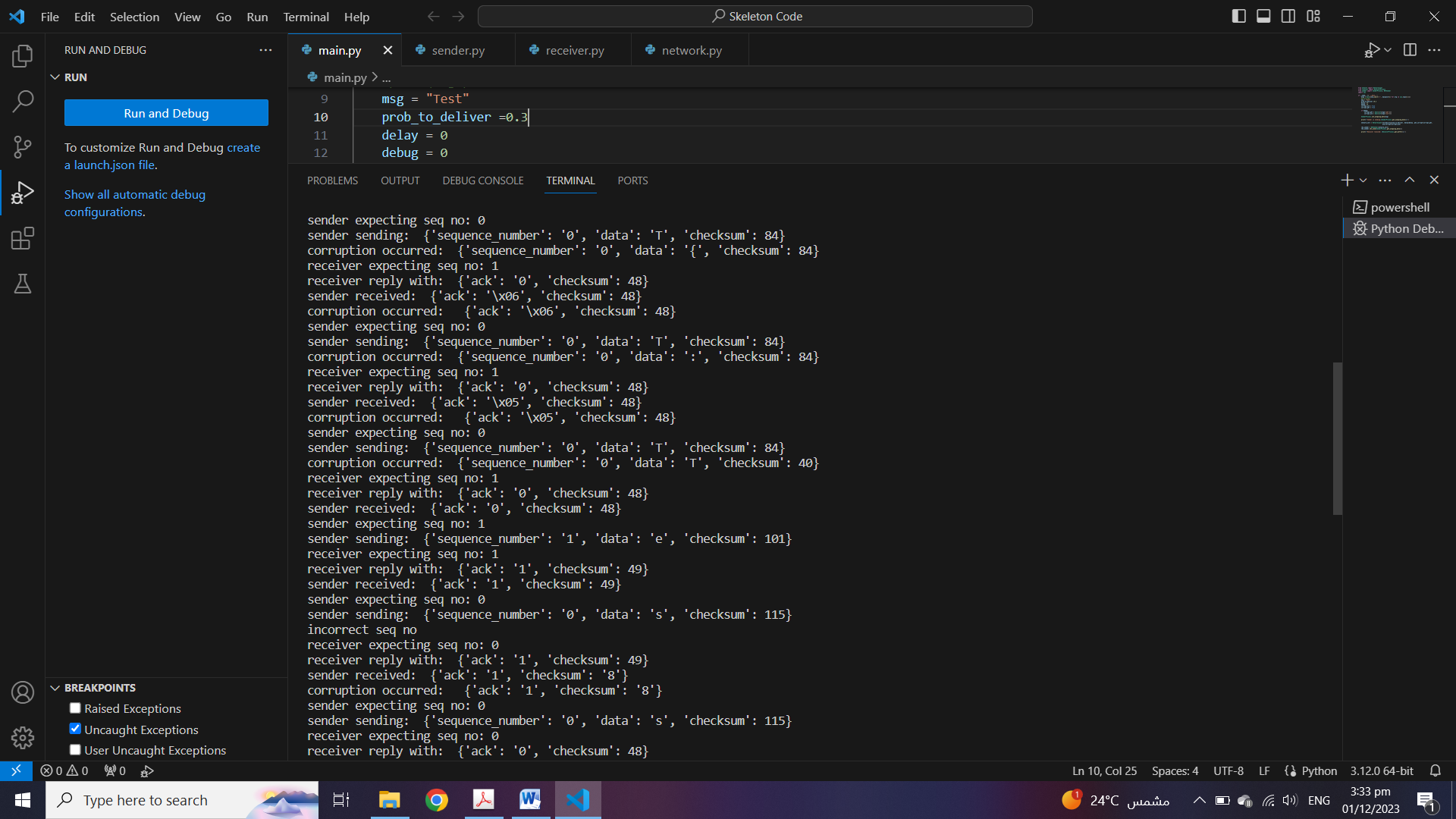
Msg=”Test “

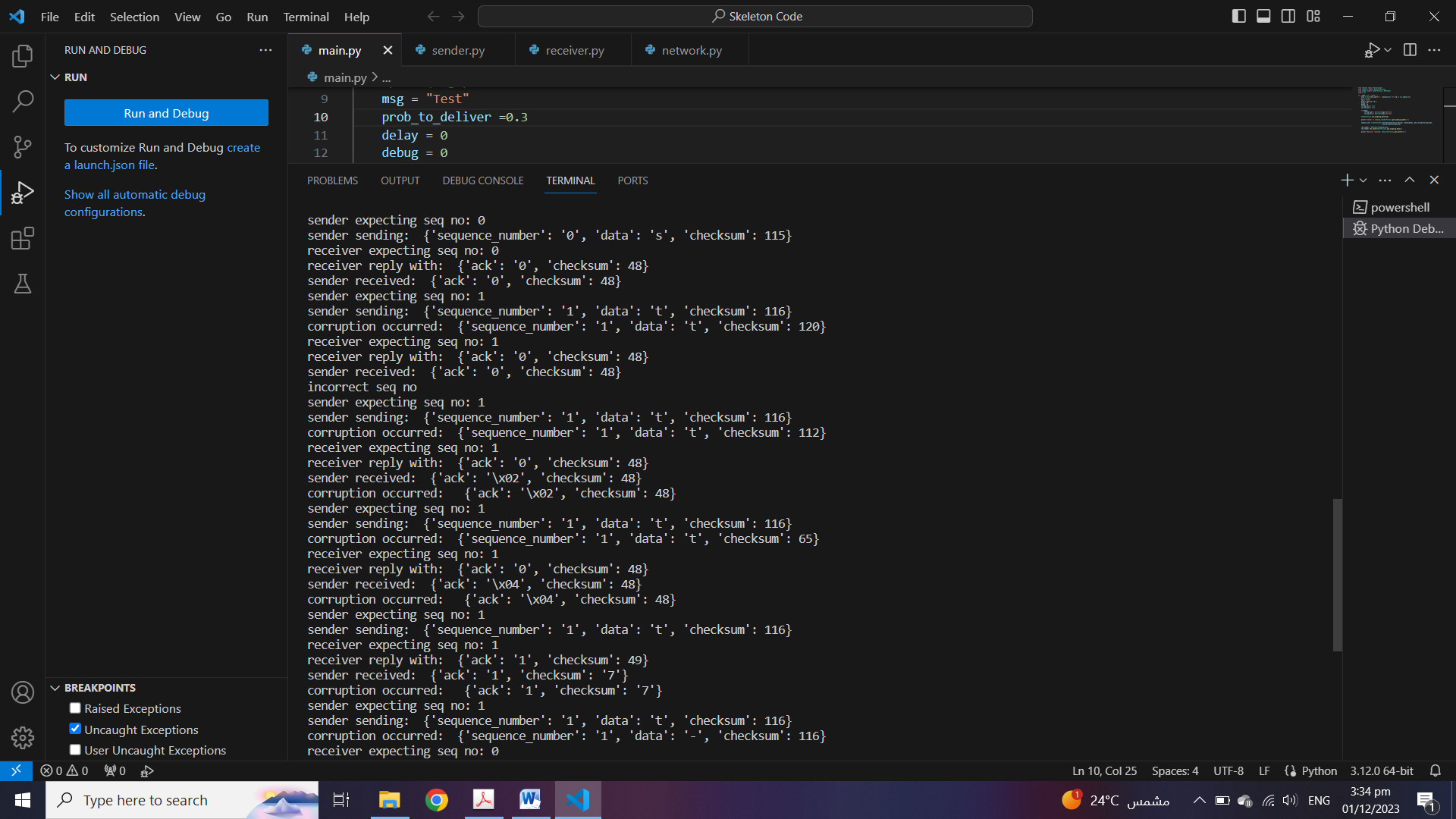
Prob\_to\_deliver=0.3(reliability=0.3)

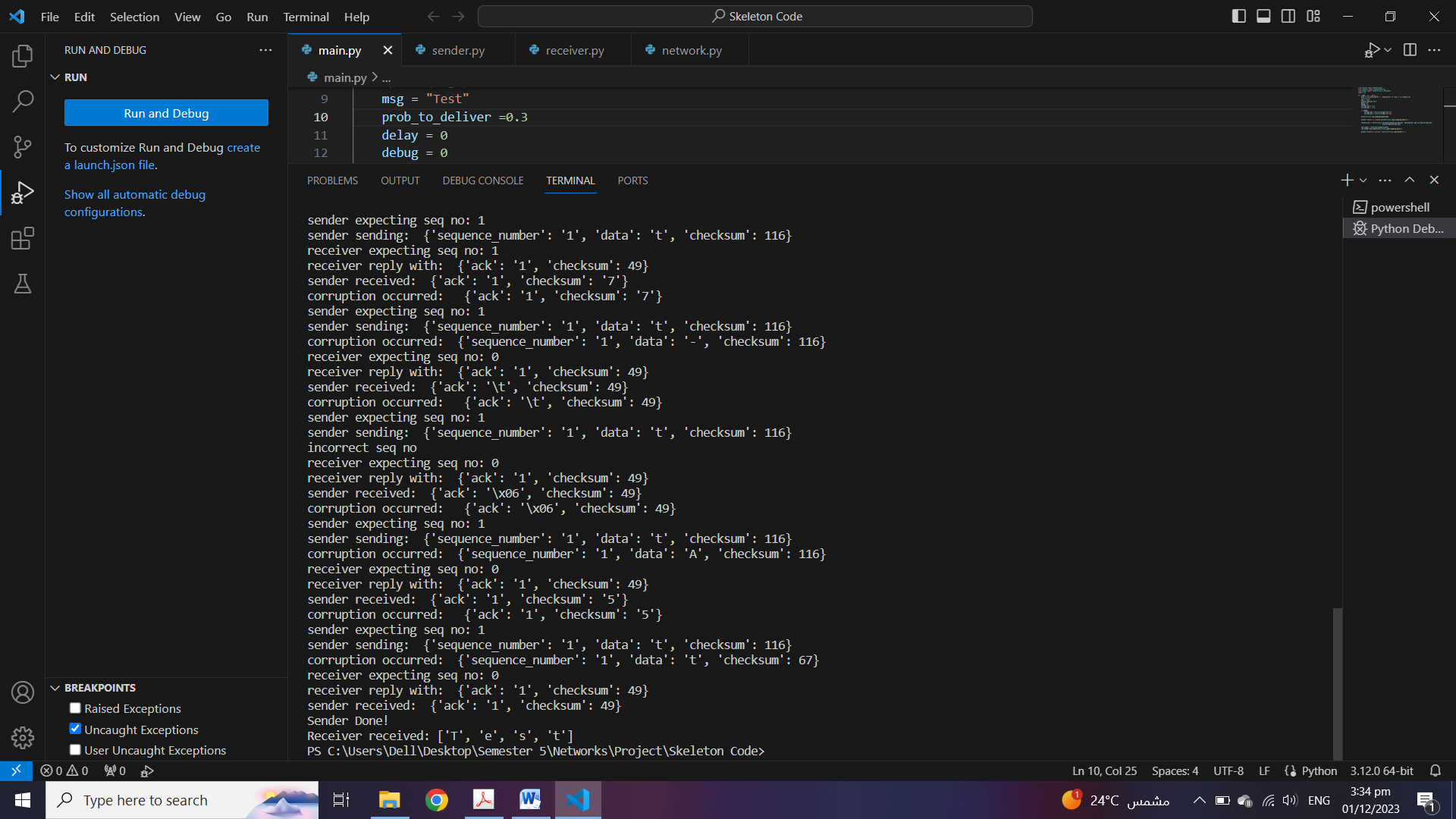
Delay=0

Debug=0









Pseudo Code

Sender

 @staticmethod

    def get\_checksum(data):

        checksum = ord(data)  #converts a character into ascii code

        return checksum

this method gets the ascii code (checksum) of a character

this static method takes in as a input a character that is going to be outgoing data and returns the ascii code (checksum) of it using the ord function that returns the ascii code of a character

@staticmethod

    def is\_corrupted(reply):

        if reply['checksum']==ord(reply['ack']):

            return False

        else:

            return True

this method checks whether the acknowledgement received from the receiver is corrupted or not

this static method takes in as an input a reply which is a python dictionary and compares its checksum with the ascii code (checksum) of its acknowledgement if they are equal then the packet is not corrupted and it returns false however if they are not equal then the packet is corrupted and it returns true

@staticmethod

    def is\_expected\_seq(reply, exp\_seq):

        if reply['ack']==exp\_seq:

            return True

        else:

            return False

this method checks whether the received reply has the expected sequence number by checking that the acknowledgement received by the receiver has the expected number

this static method takes in as an input the reply which is a python dictionary and the expected sequence number and compares the reply’s acknowledgement with the expected sequence number if they are equal then we return true because the acknowledgment in the reply matches the expected sequence number however if they are not equal then we return false because the acknowledgment in the reply doesn’t match the expected sequence number

def rdt\_send(self, process\_buffer):

this method prepares a packet that is going to be sent and initializes it with a sequence number (of 0) and sends out this packet to the transport layer

it receives an acknowledgemet back and makes some checks

if it’s not corrupted and with the correct expected sequence then the acknowledgement is correct so we toggle the sequence number for the next state and break out of the while loop

however if it’s corrupted or with the wrong expected sequence number then the acknowledgement is incorrect so we stay in the loop and retransmit the packet until the correct acknowledgment is received

Receiver

@staticmethod

    def is\_corrupted(packet):

        if packet['checksum']==ord(packet['data']):

            return False

        else:

            return True

this method checks whether a received packet is corrupted or not

this static method takes in as an input a packet which is a python dictionary and compares its checksum with the ascii code (checksum) of its data (character) are equal then the packet is not corrupted and it returns false however if they are not equal then the packet is corrupted and it returns true

@staticmethod

    def is\_expected\_seq(rcv\_pkt, exp\_seq):

        if rcv\_pkt['sequence\_number']==exp\_seq:

            return True

        else:

            return False

this method checks whether the received reply has the expected sequence number by checking that the sequence number received by the receiver has the expected number

this static method takes in as an input the received packet and the expected sequence number and compares the received packet’s sequence number with the expected sequence number if they are equal then we return true because the sequence number in the received packet matches the expected sequence number however if they are not equal then we return false because the sequence number in the received packet doesn’t match the expected sequence number

def rdt\_rcv(self, rcv\_pkt):

this method checks if the received packet is corrupted or with the wrong sequence number, if this is true we create a reply packet with the toggled sequence number with its checksum

however if the received packet is not corrupted and with the correct sequence number then we create a reply packet with the correct sequence number with its checksum then we toggle the sequence number for the next state

