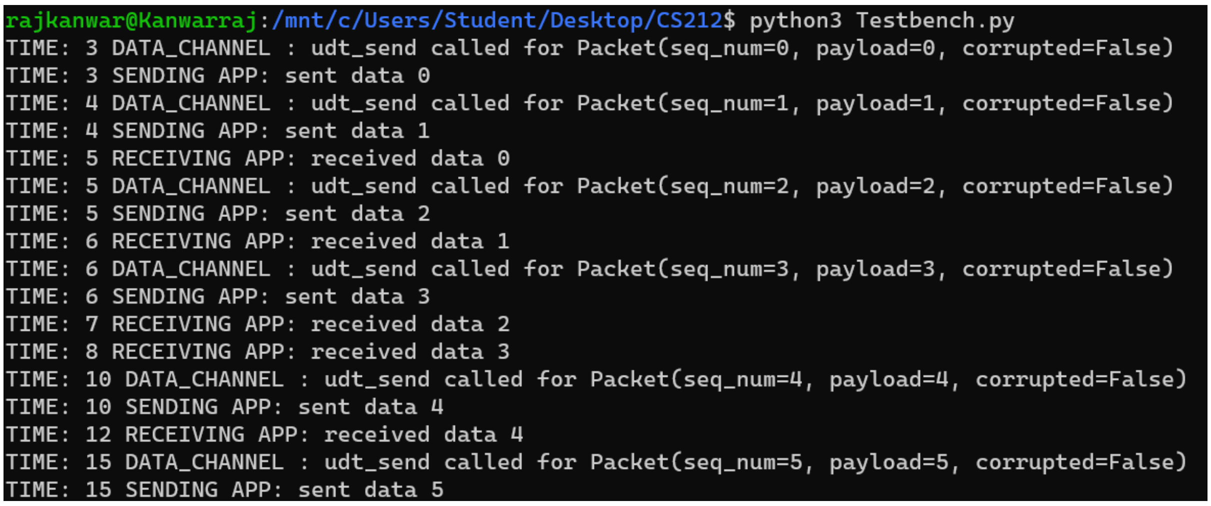
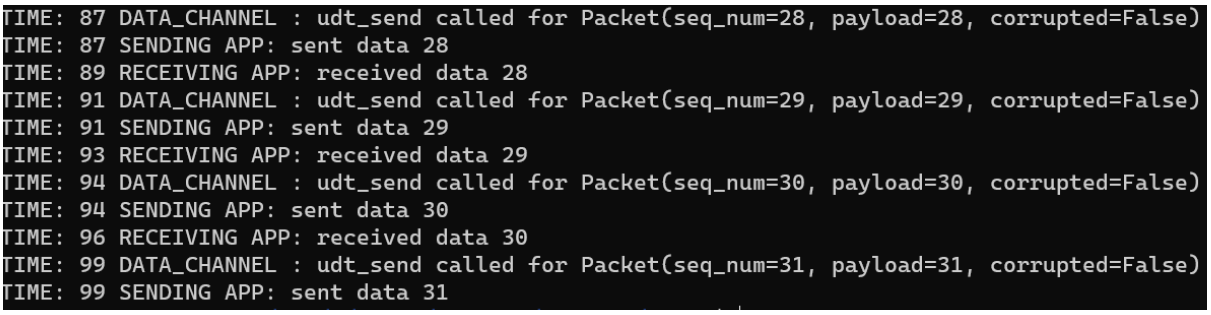
LAB3: - Principle of reliable Data Transfer

CS212(Computer networks)   Kanwar Raj Singh (1903122) & Afzal Hussain (2103103)

1. Output after setting Pc = 0, we got following output which is expected because in this case probability of loss and corruption is 0 hence, sender packet not getting corrupt and not even loss of packet, also ACK and NAK is ideal in this condition.

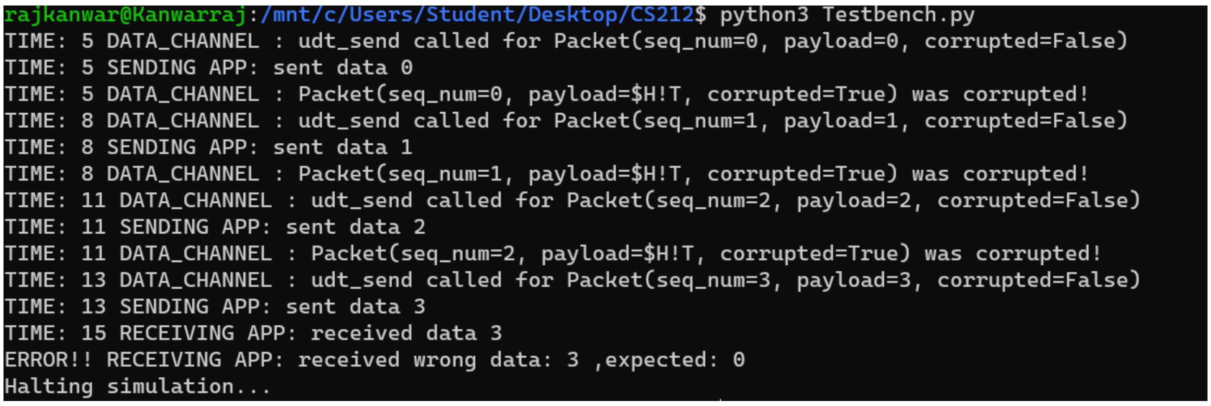


```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 SENDING APP: sent data 0
TIME: 4 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 4 SENDING APP: sent data 1
TIME: 5 RECEIVING APP: received data 0
TIME: 5 DATA_CHANNEL : udt_send called for Packet(seq_num=2, payload=2, corrupted=False)
TIME: 5 SENDING APP: sent data 2
TIME: 6 RECEIVING APP: received data 1
TIME: 6 DATA_CHANNEL : udt_send called for Packet(seq_num=3, payload=3, corrupted=False)
TIME: 6 SENDING APP: sent data 3
TIME: 7 RECEIVING APP: received data 2
TIME: 8 RECEIVING APP: received data 3
TIME: 10 DATA_CHANNEL : udt_send called for Packet(seq_num=4, payload=4, corrupted=False)
TIME: 10 SENDING APP: sent data 4
TIME: 12 RECEIVING APP: received data 4
TIME: 15 DATA_CHANNEL : udt_send called for Packet(seq_num=5, payload=5, corrupted=False)
TIME: 15 SENDING APP: sent data 5
```



```
TIME: 87 DATA_CHANNEL : udt_send called for Packet(seq_num=28, payload=28, corrupted=False)
TIME: 87 SENDING APP: sent data 28
TIME: 89 RECEIVING APP: received data 28
TIME: 91 DATA_CHANNEL : udt_send called for Packet(seq_num=29, payload=29, corrupted=False)
TIME: 91 SENDING APP: sent data 29
TIME: 93 RECEIVING APP: received data 29
TIME: 94 DATA_CHANNEL : udt_send called for Packet(seq_num=30, payload=30, corrupted=False)
TIME: 94 SENDING APP: sent data 30
TIME: 96 RECEIVING APP: received data 30
TIME: 99 DATA_CHANNEL : udt_send called for Packet(seq_num=31, payload=31, corrupted=False)
TIME: 99 SENDING APP: sent data 31
```

For Pc = 0.5 in in Protocol_rdt1, it is giving error because In rdt1 we did not implemented code for corruption of packet.

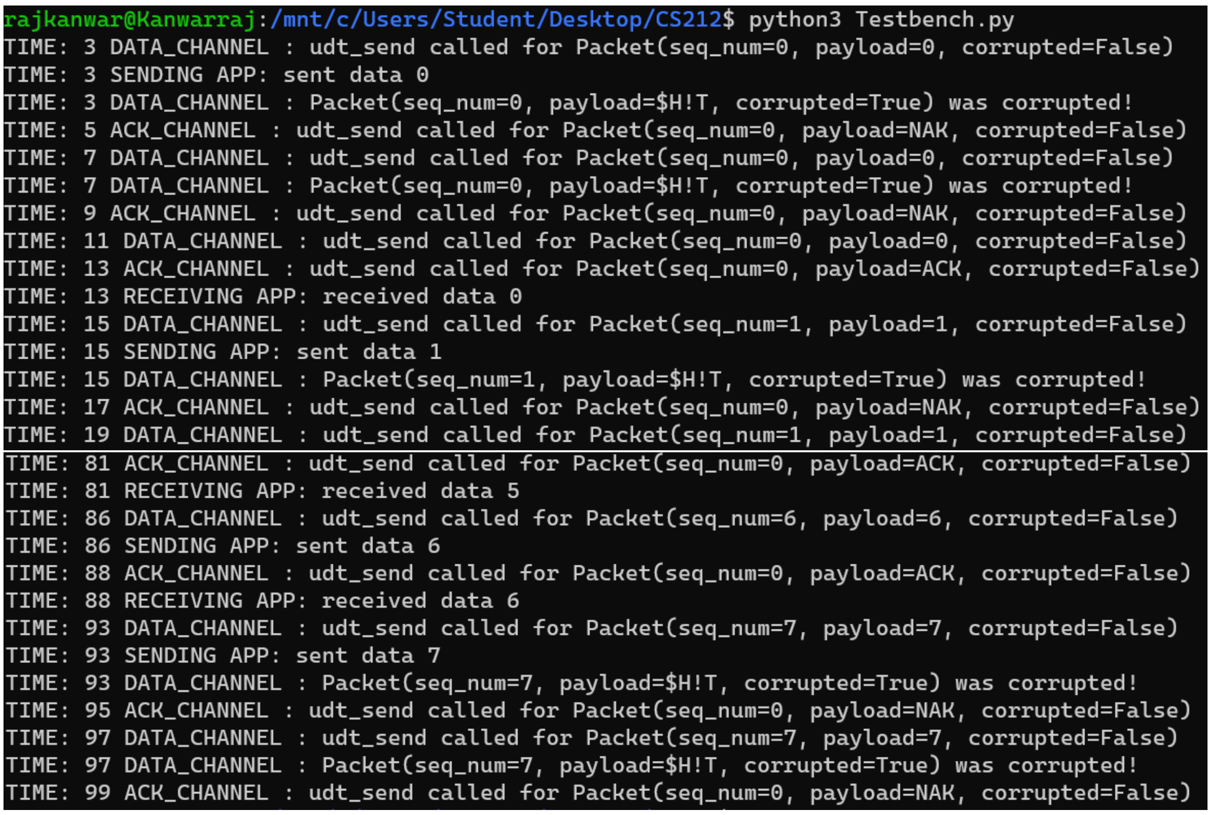


```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 5 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 5 SENDING APP: sent data 0
TIME: 5 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 8 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 8 SENDING APP: sent data 1
TIME: 8 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=2, payload=2, corrupted=False)
TIME: 11 SENDING APP: sent data 2
TIME: 11 DATA_CHANNEL : Packet(seq_num=2, payload=$H!T, corrupted=True) was corrupted!
TIME: 13 DATA_CHANNEL : udt_send called for Packet(seq_num=3, payload=3, corrupted=False)
TIME: 13 SENDING APP: sent data 3
TIME: 15 RECEIVING APP: received data 3
ERROR!! RECEIVING APP: received wrong data: 3 ,expected: 0
Halting simulation...
```

In the output we can very well see that the Protocol fails as in the end it outputs that there is some error due to receiving the wrong data packet and thus it is halting the simulation.

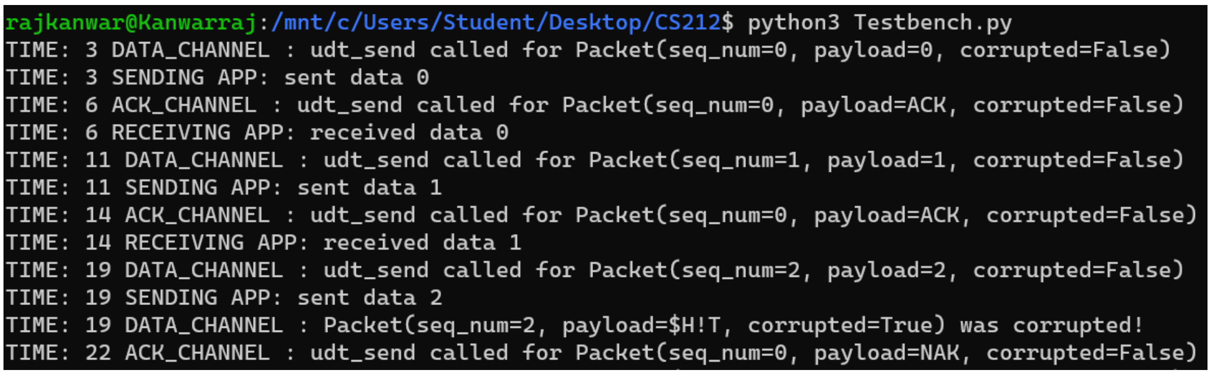
This error occurred because the first packet which was sent (packet ‘0’) got corrupted (corrupted = true, payload = $h!T) while flowing through the unreliable channel. Thus, the receiver received the corrupted packet but was waiting for the uncorrupted packet ‘0’. In the next iteration when the sender sends packet ‘1’ it does not get corrupted, now the receiver receives packet ‘1’ but it was expecting packet ‘0’ so it halts the simulation as it realizes something went wrong and it can’t rectify it.

2. For Protocol_rdt2.py I set Pc = 0.6, In this we are considering Pc > 0 and P(Loss) = 0,
   In this our packet can get corrupted but We didn’t implemented code for ACK & NAK corruption condition. (Packet limit = 100)
   We got following output:

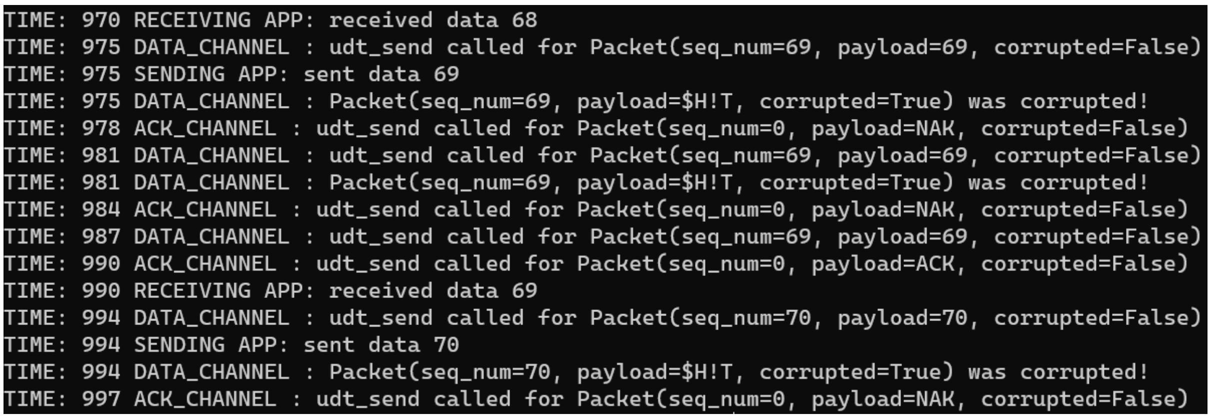


```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 SENDING APP: sent data 0
TIME: 3 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 5 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 7 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 7 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 9 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 13 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 13 RECEIVING APP: received data 0
TIME: 15 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 15 SENDING APP: sent data 1
TIME: 15 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 17 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 19 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 81 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 81 RECEIVING APP: received data 5
TIME: 86 DATA_CHANNEL : udt_send called for Packet(seq_num=6, payload=6, corrupted=False)
TIME: 86 SENDING APP: sent data 6
TIME: 88 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 88 RECEIVING APP: received data 6
TIME: 93 DATA_CHANNEL : udt_send called for Packet(seq_num=7, payload=7, corrupted=False)
TIME: 93 SENDING APP: sent data 7
TIME: 93 DATA_CHANNEL : Packet(seq_num=7, payload=$H!T, corrupted=True) was corrupted!
TIME: 95 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 97 DATA_CHANNEL : udt_send called for Packet(seq_num=7, payload=7, corrupted=False)
TIME: 97 DATA_CHANNEL : Packet(seq_num=7, payload=$H!T, corrupted=True) was corrupted!
TIME: 99 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
```

3. A) In Testbech.py we set Pc = 0.5 for packets and Pc = 0 for ACK & NAK.
   We set Delay = 3 and Number of Packets = 1000.
   Below I attached screenshot of output.



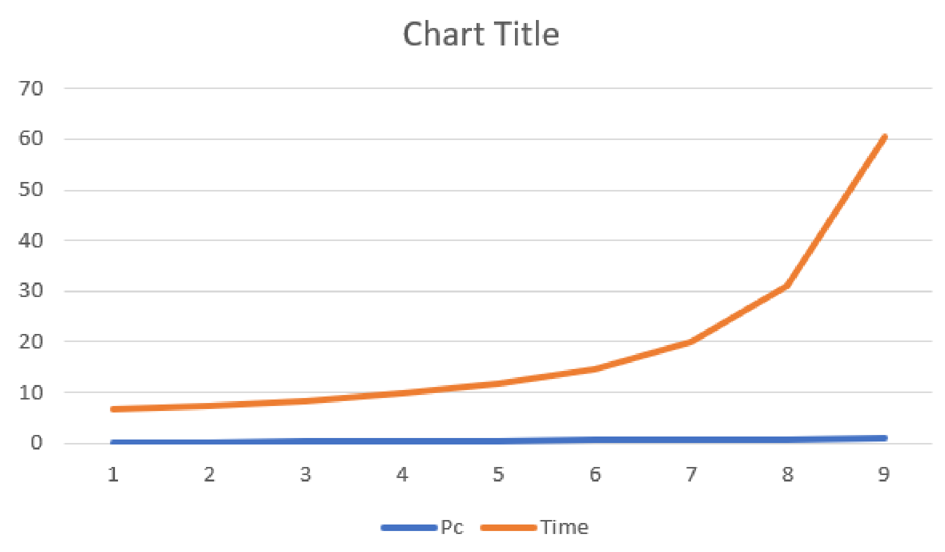
```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 SENDING APP: sent data 0
TIME: 6 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 6 RECEIVING APP: received data 0
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 11 SENDING APP: sent data 1
TIME: 14 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 14 RECEIVING APP: received data 1
TIME: 19 DATA_CHANNEL : udt_send called for Packet(seq_num=2, payload=2, corrupted=False)
TIME: 19 SENDING APP: sent data 2
TIME: 19 DATA_CHANNEL : Packet(seq_num=2, payload=$H!T, corrupted=True) was corrupted!
TIME: 22 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
```



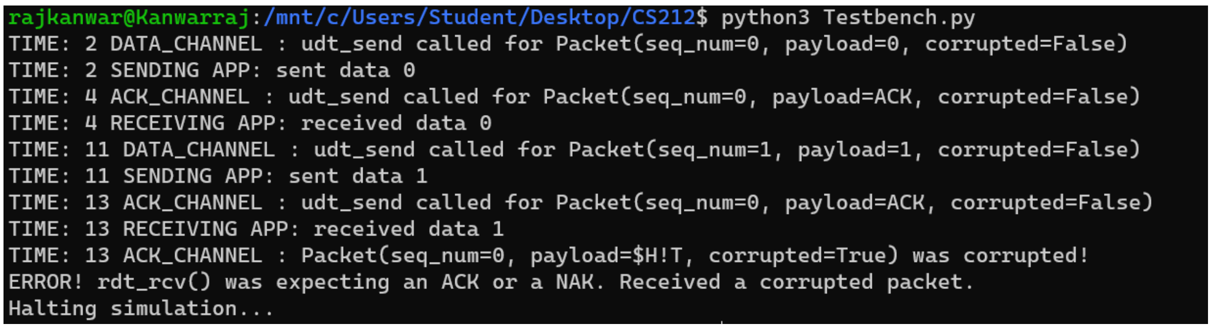
```
TIME: 970 RECEIVING APP: received data 68
TIME: 975 DATA_CHANNEL : udt_send called for Packet(seq_num=69, payload=69, corrupted=False)
TIME: 975 SENDING APP: sent data 69
TIME: 975 DATA_CHANNEL : Packet(seq_num=69, payload=$H!T, corrupted=True) was corrupted!
TIME: 978 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 981 DATA_CHANNEL : udt_send called for Packet(seq_num=69, payload=69, corrupted=False)
TIME: 981 DATA_CHANNEL : Packet(seq_num=69, payload=$H!T, corrupted=True) was corrupted!
TIME: 984 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
TIME: 987 DATA_CHANNEL : udt_send called for Packet(seq_num=69, payload=69, corrupted=False)
TIME: 990 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 990 RECEIVING APP: received data 69
TIME: 994 DATA_CHANNEL : udt_send called for Packet(seq_num=70, payload=70, corrupted=False)
TIME: 994 SENDING APP: sent data 70
TIME: 994 DATA_CHANNEL : Packet(seq_num=70, payload=$H!T, corrupted=True) was corrupted!
TIME: 997 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=NAK, corrupted=False)
```

Table of time is below attached.

Pc	Time
0.1	6.75
0.2	7.41
0.3	8.4
0.4	9.73
0.5	11.9
0.6	14.63
0.7	19.91
0.8	30.96
0.9	60.4



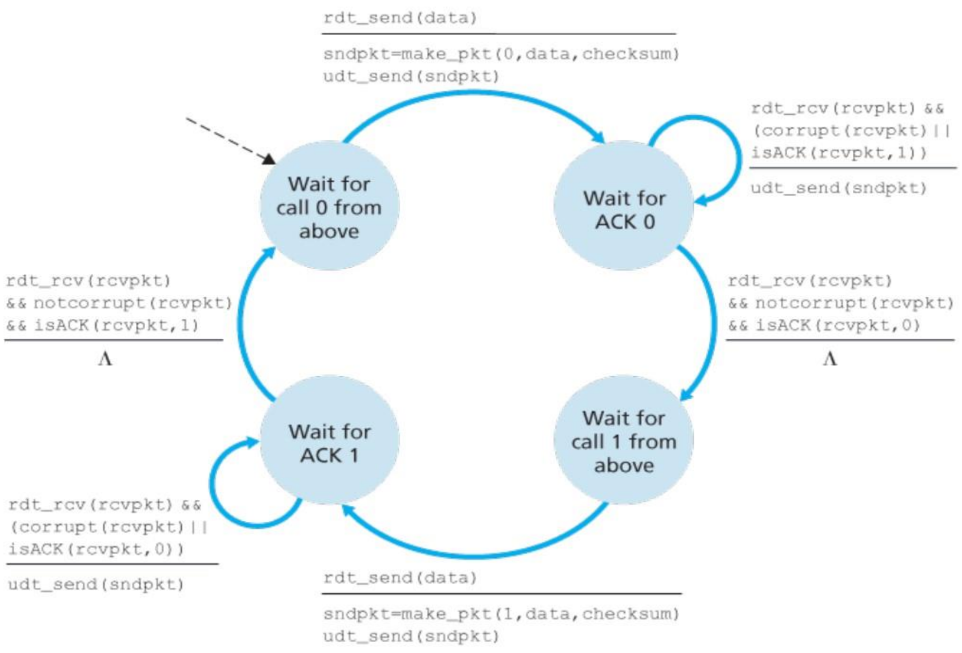
4. In protocol_rdt2.py we Implemented code for packet corruption but we consider ACK and NAK to be ideal, they cannot get corrupted but when we set Pc = 0.6 for ACK and NAK it is showing following error.



```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 2 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 2 SENDING APP: sent data 0
TIME: 4 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 4 RECEIVING APP: received data 0
TIME: 11 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 11 SENDING APP: sent data 1
TIME: 13 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 13 RECEIVING APP: received data 1
TIME: 13 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
ERROR! rdt_rcv() was expecting an ACK or a NAK. Received a corrupted packet.
Halting simulation...
```

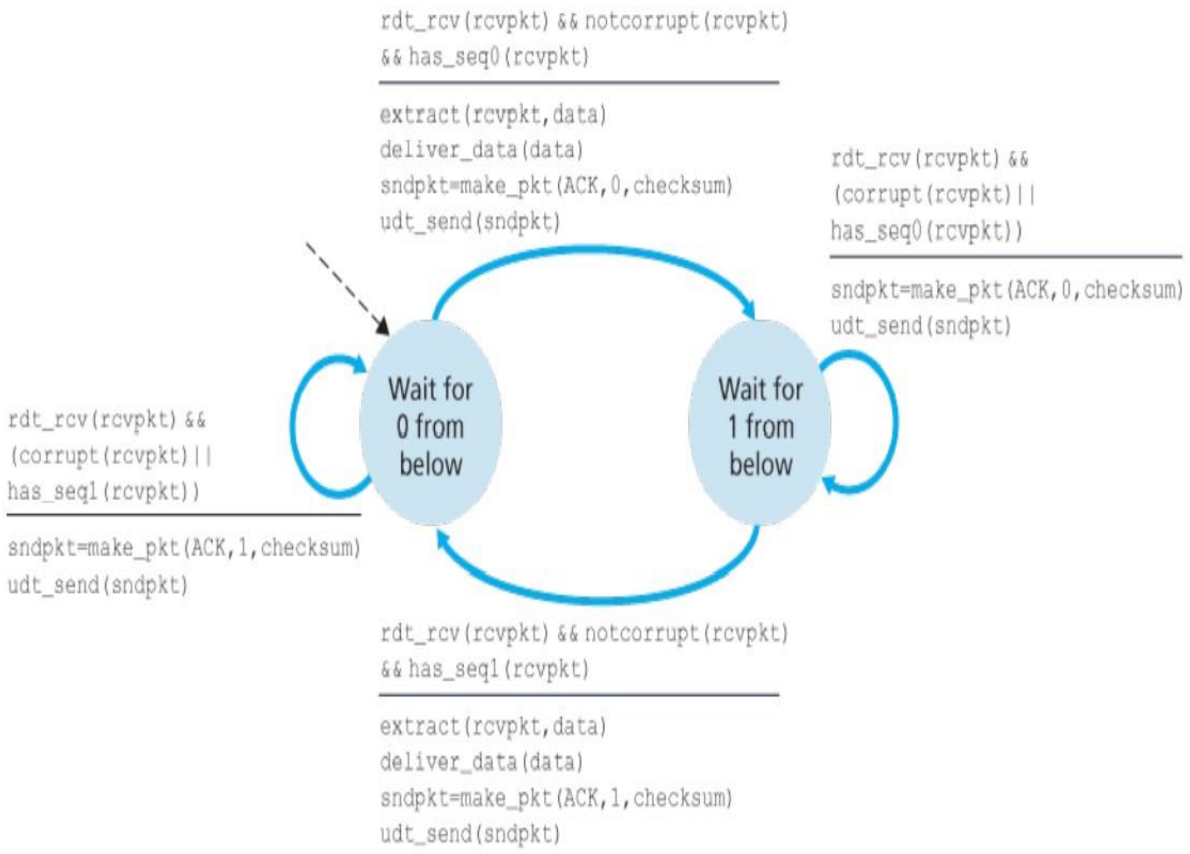
To Remove This, we must implement code for ACK & NAK corruption in protocol_rdt2.py

5. Following is the FSM model, we need to develop in our code:


rdt_send(data)
sndpkt=make_pkt(0,data,checksum)
udt_send(sndpkt)
Wait for call 0 from above
Wait for ACK 0
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,1))
udt_send(sndpkt)
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,1)
Λ
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,0)
Λ
Wait for ACK 1
Wait for call 1 from above
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,0))
udt_send(sndpkt)
rdt_send(data)
sndpkt=make_pkt(1,data,checksum)
udt_send(sndpkt)

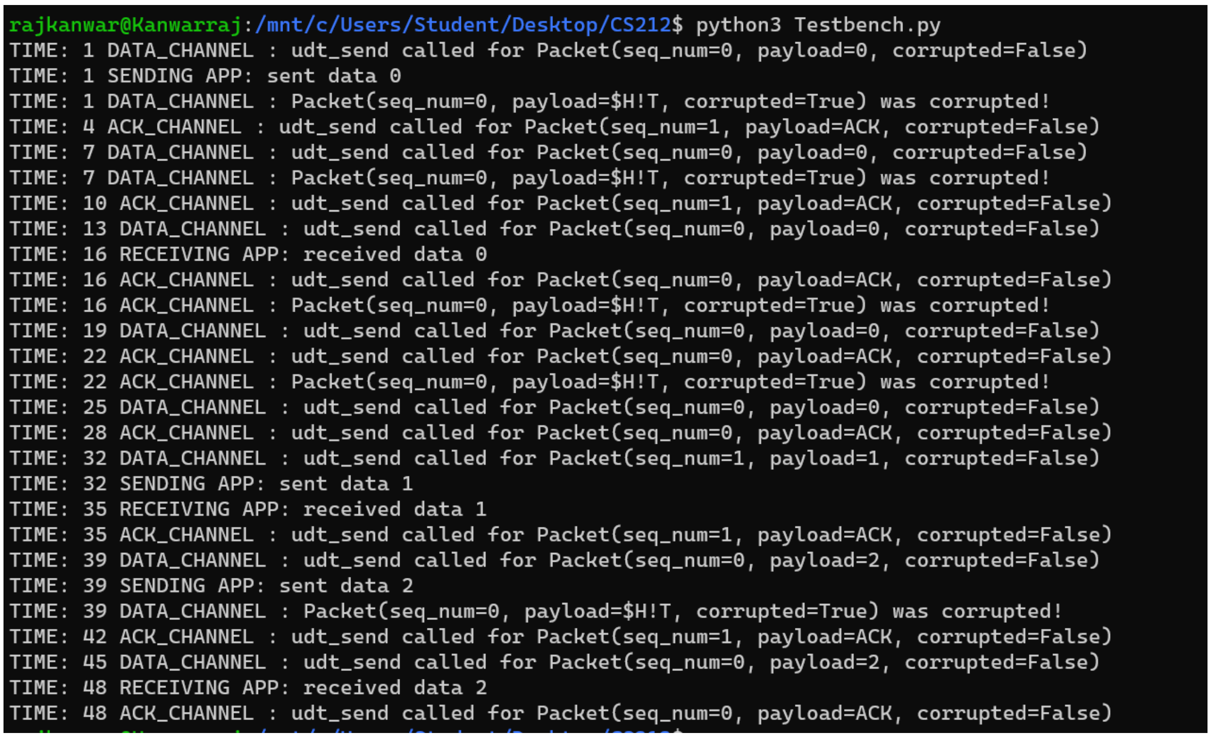
Now there is no need for NAK, instead, we can give ACK of that packet which was not expected to serve as NAK.

Similarly, we develop at receiver:


rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq0(rcvpkt)
extract(rcvpkt,data)
deliver_data(data)
sndpkt=make_pkt(ACK,0,checksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || has_seq0(rcvpkt))
sndpkt=make_pkt(ACK,0,checksum)
udt_send(sndpkt)
Wait for 0 from below
Wait for 1 from below
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || has_seq1(rcvpkt))
sndpkt=make_pkt(ACK,1,checksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq1(rcvpkt)
extract(rcvpkt,data)
deliver_data(data)
sndpkt=make_pkt(ACK,1,checksum)
udt_send(sndpkt)

The protocol Works well for both channels having probability of corruption

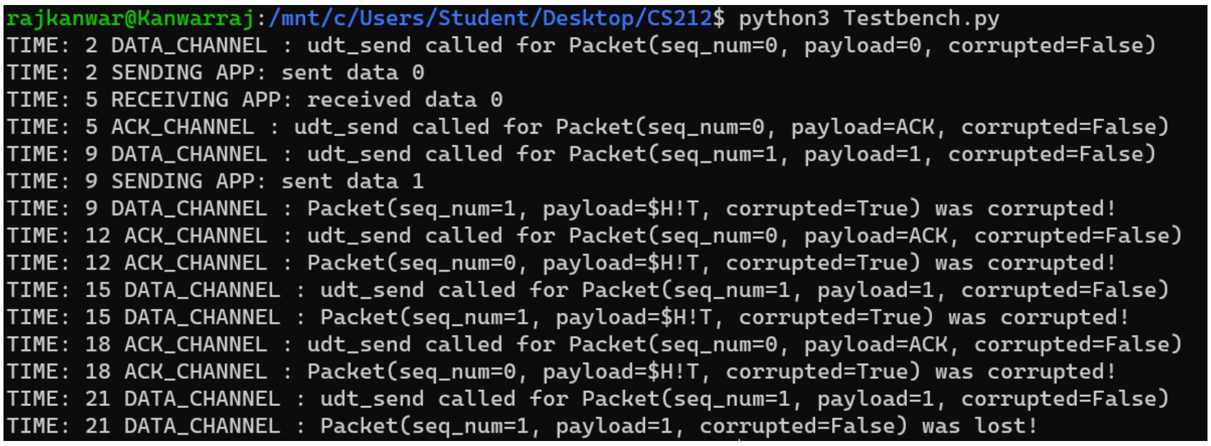
Screenshot of output (I set packets == 50)



```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 1 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 1 SENDING APP: sent data 0
TIME: 1 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 4 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK, corrupted=False)
TIME: 7 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 7 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 10 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK, corrupted=False)
TIME: 13 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 16 RECEIVING APP: received data 0
TIME: 16 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 16 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 19 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 22 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 22 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 25 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 28 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 32 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 32 SENDING APP: sent data 1
TIME: 35 RECEIVING APP: received data 1
TIME: 35 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK, corrupted=False)
TIME: 39 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=2, corrupted=False)
TIME: 39 SENDING APP: sent data 2
TIME: 39 DATA_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 42 ACK_CHANNEL : udt_send called for Packet(seq_num=1, payload=ACK, corrupted=False)
TIME: 45 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=2, corrupted=False)
TIME: 48 RECEIVING APP: received data 2
TIME: 48 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
```

6. A) I set number of packets == 1000, Pc of packets = 0.5 and Pc of ACK = 0.5 and

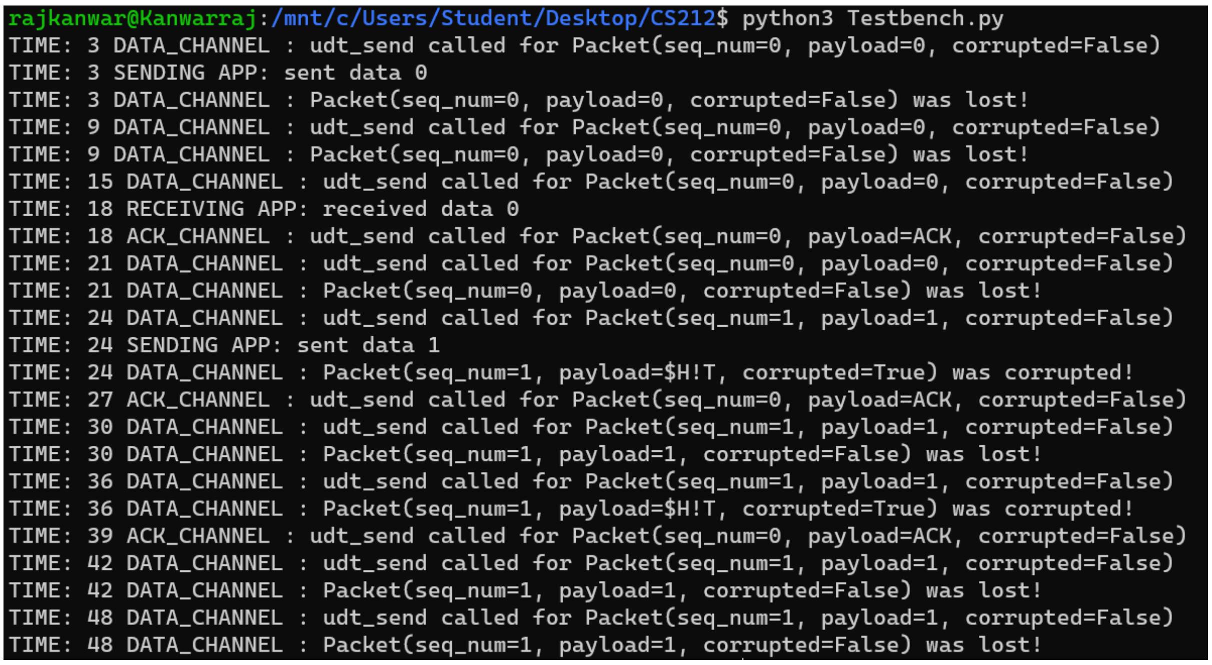
   P(Loss) = 0.4 and got following output because in Protocol_rdt22.py we didn’t consider packet loss.



```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 2 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 2 SENDING APP: sent data 0
TIME: 5 RECEIVING APP: received data 0
TIME: 5 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 9 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 9 SENDING APP: sent data 1
TIME: 9 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 12 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 12 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 15 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 15 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 18 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 18 ACK_CHANNEL : Packet(seq_num=0, payload=$H!T, corrupted=True) was corrupted!
TIME: 21 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 21 DATA_CHANNEL : Packet(seq_num=1, payload=1, corrupted=False) was lost!
```

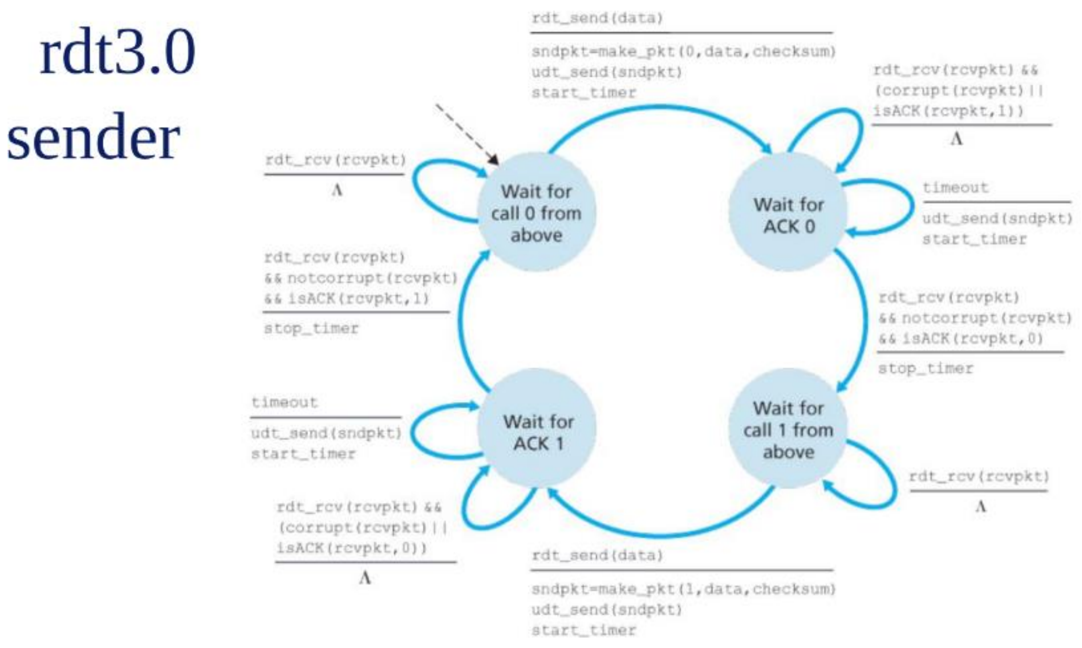
After time 21, there program terminated.

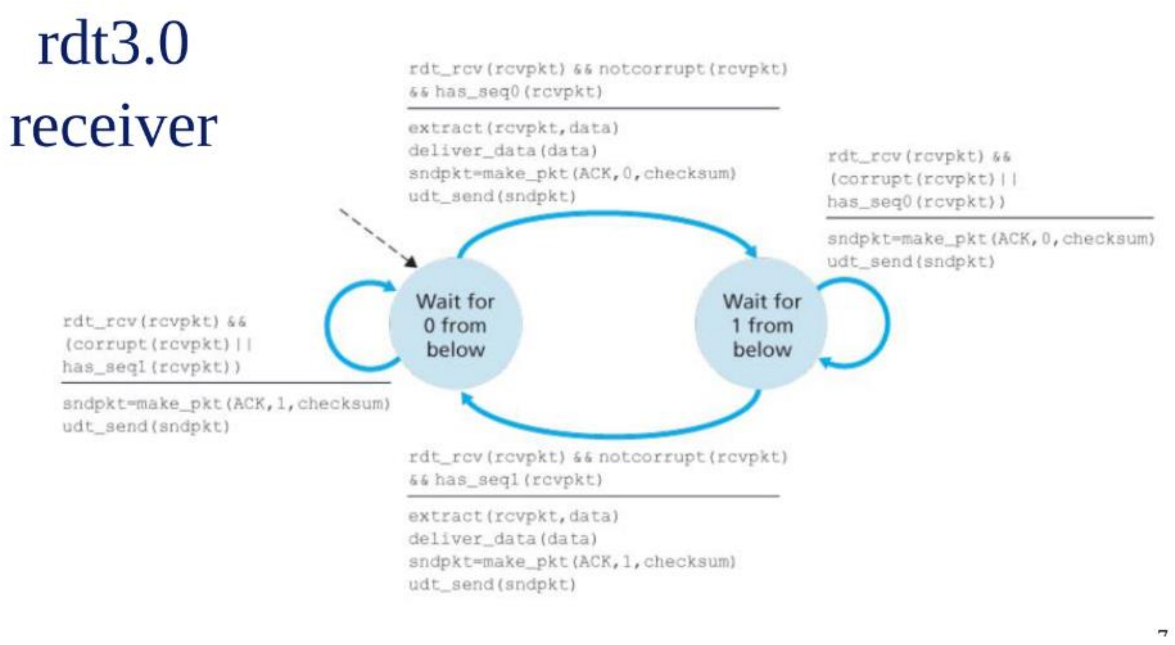
B) screenshot of output of rdt3.0



```
rajkanwar@Kanwarraj:/mnt/c/Users/Student/Desktop/CS212$ python3 Testbench.py
TIME: 3 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 3 SENDING APP: sent data 0
TIME: 3 DATA_CHANNEL : Packet(seq_num=0, payload=0, corrupted=False) was lost!
TIME: 9 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 9 DATA_CHANNEL : Packet(seq_num=0, payload=0, corrupted=False) was lost!
TIME: 15 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 18 RECEIVING APP: received data 0
TIME: 18 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 21 DATA_CHANNEL : udt_send called for Packet(seq_num=0, payload=0, corrupted=False)
TIME: 21 DATA_CHANNEL : Packet(seq_num=0, payload=0, corrupted=False) was lost!
TIME: 24 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 24 SENDING APP: sent data 1
TIME: 24 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 27 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 30 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 30 DATA_CHANNEL : Packet(seq_num=1, payload=1, corrupted=False) was lost!
TIME: 36 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 36 DATA_CHANNEL : Packet(seq_num=1, payload=$H!T, corrupted=True) was corrupted!
TIME: 39 ACK_CHANNEL : udt_send called for Packet(seq_num=0, payload=ACK, corrupted=False)
TIME: 42 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 42 DATA_CHANNEL : Packet(seq_num=1, payload=1, corrupted=False) was lost!
TIME: 48 DATA_CHANNEL : udt_send called for Packet(seq_num=1, payload=1, corrupted=False)
TIME: 48 DATA_CHANNEL : Packet(seq_num=1, payload=1, corrupted=False) was lost!
```

Following FSM model, we used to code rdt_3.py

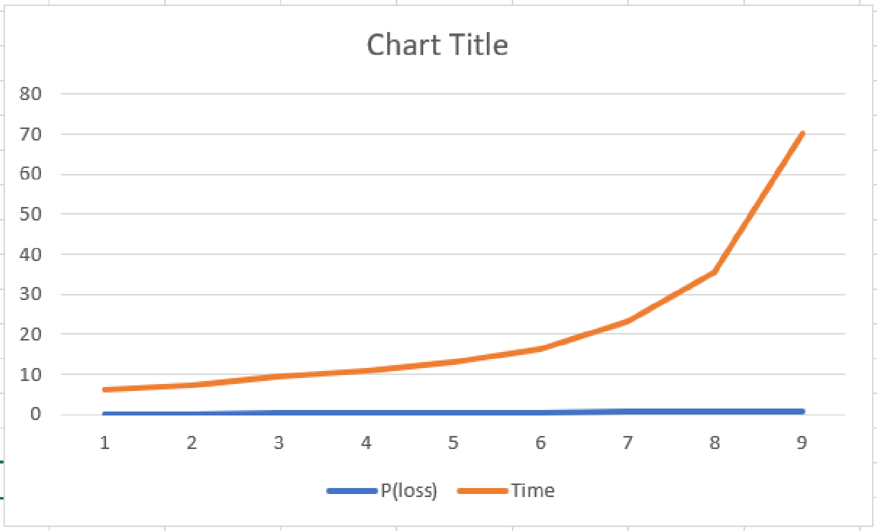

rdt3.0
sender
rdt_send(data)
sndpkt=make_pkt(0,data,checksum)
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,1))
Λ
rdt_rcv(rcvpkt)
Λ
Wait for call 0 from above
Wait for ACK 0
timeout
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,1)
stop_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,0)
stop_timer
timeout
udt_send(sndpkt)
start_timer
Wait for ACK 1
Wait for call 1 from above
rdt_rcv(rcvpkt)
Λ
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,0))
Λ
rdt_send(data)
sndpkt=make_pkt(1,data,checksum)
udt_send(sndpkt)
start_timer



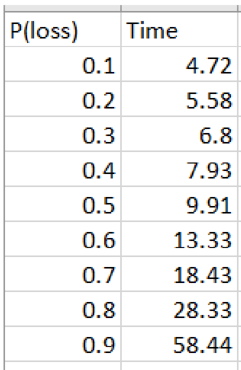
Source: - Class Slides.

C) chart and table attached below.

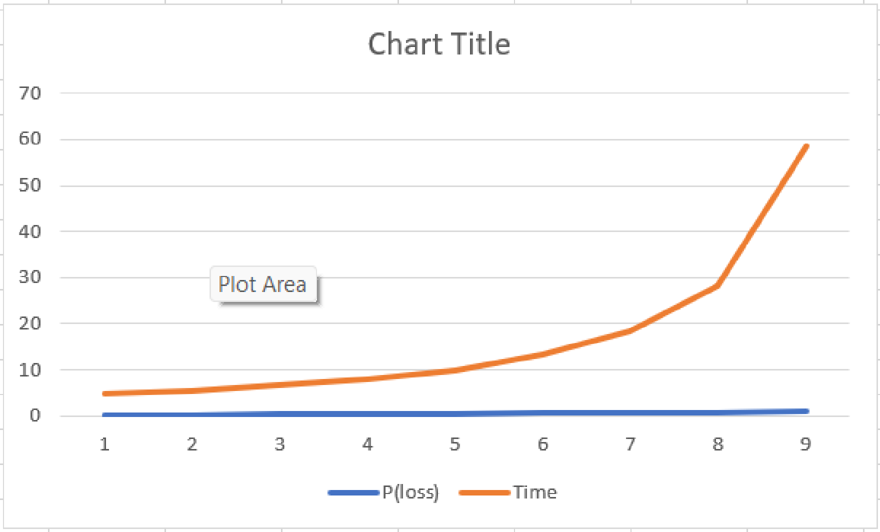
P(loss)	Time
0.1	6.17
0.2	7.32
0.3	9.33
0.4	10.79
0.5	13.18
0.6	16.4
0.7	23.06
0.8	35.66
0.9	70.18



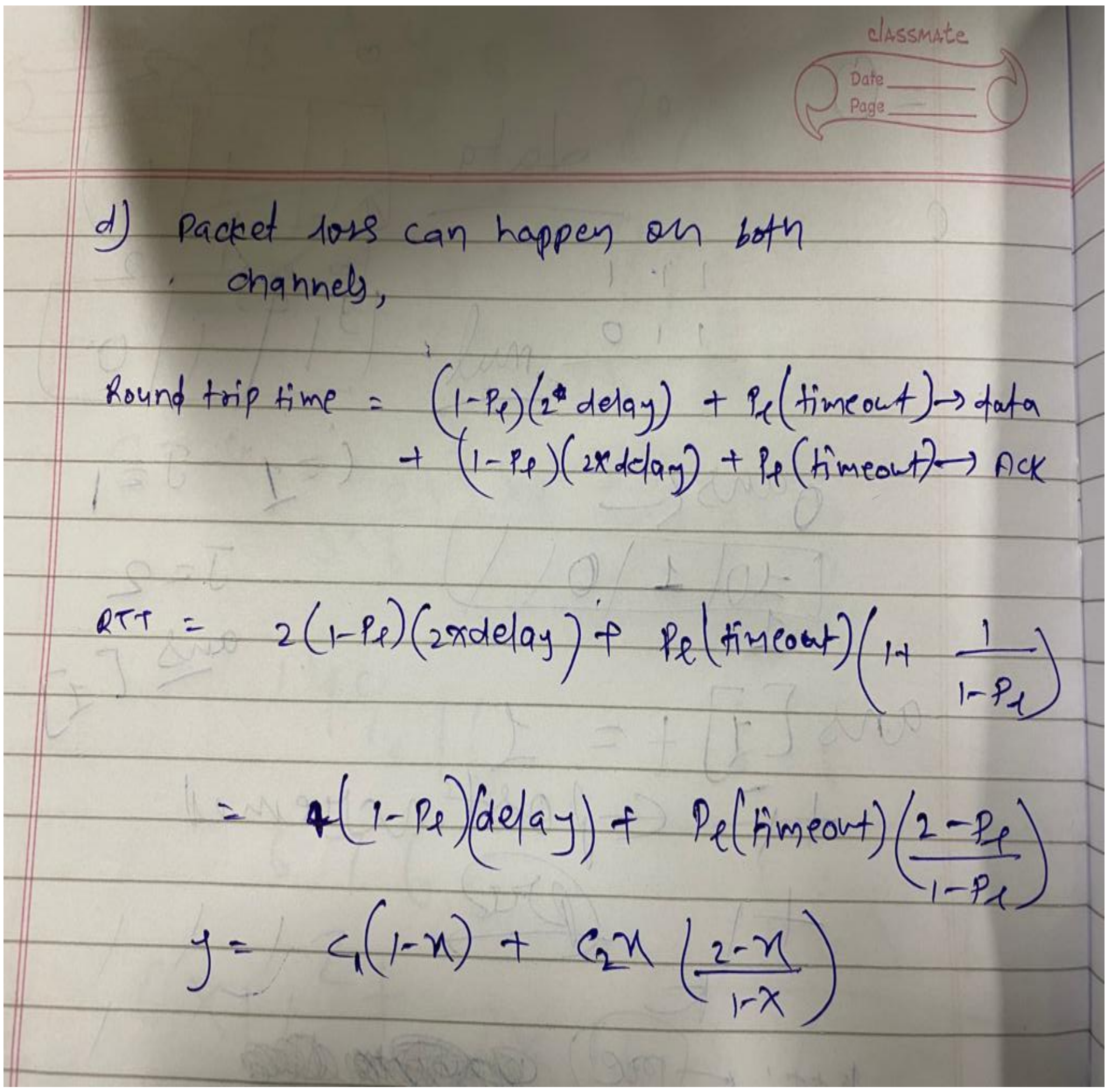
d) Graph between P(loss) and Average time is attached below;



P(loss)	Time
0.1	4.72
0.2	5.58
0.3	6.8
0.4	7.93
0.5	9.91
0.6	13.33
0.7	18.43
0.8	28.33
0.9	58.44



Expression for this is below:



d) Packet loss can happen on both channels,

Round trip time = $(1-P_\ell)(2*\text{delay}) + P_\ell(\text{timeout}) \rightarrow \text{data}$

$+ (1-P_\ell)(2\times\text{delay}) + P_\ell(\text{timeout}) \rightarrow \text{ACK}$

$$RTT = 2(1-P_\ell)(2\times\text{delay}) + P_\ell(\text{timeout})\left(1+\frac{1}{1-P_\ell}\right)$$

$$= 4(1-P_\ell)(\text{delay}) + P_\ell(\text{timeout})\left(\frac{2-P_\ell}{1-P_\ell}\right)$$

$$y = c_1(1-x) + c_2 x\left(\frac{2-x}{1-x}\right)$$