Report:
1– RDT 2.2 is an enhanced version of RDT 2.1 with ACK and NACK response attached to every sending packet RDT 2.2 is implemented in RDT.py with loss rate of 0.3.
As we can see there is timeout detected, sender transmits again with the same sequence number until it is received. The entire log is in abc.txt, you can refer to it.

```
Sending Data_1 - Content_1 with sequence number 0
Timeout Occurred. Retransmitting data
Sending Data_1 - Content_1 with sequence number 0
Received ACK 0. Transmission successful
Received Data_1 - Content_1 with sequence number 0
Sending Data_2 - Content_2 with sequence number 1
Received ACK 1. Transmission successful
Sending Data_3 - Content_3 with sequence number 0
Received ACK 0. Transmission successful
Received Data_3 - Content_3 with sequence number 0
Sending Data_4 - Content_4 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_4 - Content_4 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_4 - Content_4 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_4 - Content_4 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_4 - Content_4 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_4 - Content_4 with sequence number 1
Received ACK 1. Transmission successful
Received Data_4 - Content_4 with sequence number 1
Sending Data_5 - Content_5 with sequence number 0
Timeout Occurred. Retransmitting data
Sending Data_5 - Content_5 with sequence number 0
Timeout Occurred. Retransmitting data
Sending Data_5 - Content_5 with sequence number 0
Received ACK 0. Transmission successful
Received Data_5 - Content_5 with sequence number 0
Sending Data_6 - Content_6 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_6 - Content_6 with sequence number 1
Received ACK 1. Transmission successful
Received Data_6 - Content_6 with sequence number 1
Sending Data_7 - Content_7 with sequence number 0
Timeout Occurred. Retransmitting data
Sending Data_7 - Content_7 with sequence number 0
Timeout Occurred. Retransmitting data
Sending Data_7 - Content_7 with sequence number 0
Received ACK 0. Transmission successful
Received Data_7 - Content_7 with sequence number 0
Sending Data_8 - Content_8 with sequence number 1
Received ACK 1. Transmission successful
Received Data_8 - Content_8 with sequence number 1
Sending Data_9 - Content_9 with sequence number 0
Received ACK 0. Transmission successful
Received Data_9 - Content_9 with sequence number 0
Sending Data_10 - Content_10 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_10 - Content_10 with sequence number 1
Timeout Occurred. Retransmitting data
Sending Data_10 - Content_10 with sequence number 1
```

2– Congestion Control AIMD has 2 phases start phase and congestion avoidance phase, which linearly increases the congestion window size one by one and upon incurring loss the congestion window is halved. The entire log is in abc.txt, you can refer to it.

```
Initial Congestion Window Size: 1
Step 1: Congestion Window Size: 2
Step 2: Congestion Window Size: 4
Loss happened due to timeout...
Step 3: Congestion Window Size: 1
Loss happened due to timeout...
Step 4: Congestion Window Size: 1
Step 5: Congestion Window Size: 2
Step 6: Congestion Window Size: 3
Loss happened due to timeout...
Step 7: Congestion Window Size: 1
Loss happened due to timeout...
Step 8: Congestion Window Size: 1
Loss happened due to timeout...
Step 9: Congestion Window Size: 1
Step 10: Congestion Window Size: 2
Step 11: Congestion Window Size: 3
Step 12: Congestion Window Size: 4
Loss happened due to timeout...
Step 13: Congestion Window Size: 1
Loss happened due to timeout...
Step 14: Congestion Window Size: 1
Step 15: Congestion Window Size: 2
Loss happened due to timeout...
Step 16: Congestion Window Size: 1
Loss happened due to timeout...
Step 17: Congestion Window Size: 1
Step 18: Congestion Window Size: 2
Step 19: Congestion Window Size: 3
Step 20: Congestion Window Size: 4
Step 21: Congestion Window Size: 5
Loss happened due to timeout...
Step 22: Congestion Window Size: 1
Loss happened due to timeout...
Step 23: Congestion Window Size: 1
Loss happened due to timeout...
Step 24: Congestion Window Size: 1
Step 25: Congestion Window Size: 2
Loss happened due to timeout...
Step 26: Congestion Window Size: 1
Step 27: Congestion Window Size: 2
Step 28: Congestion Window Size: 3
Loss happened due to timeout...
Step 29: Congestion Window Size: 1
Loss happened due to timeout...
Step 30: Congestion Window Size: 1
Step 31: Congestion Window Size: 2
Loss happened due to timeout...
Step 32: Congestion Window Size: 1
Loss happened due to timeout...
Step 33: Congestion Window Size: 1
Step 34: Congestion Window Size: 2
Step 35: Congestion Window Size: 3
Step 36: Congestion Window Size: 4
```

3– Link State Routing is nothing but an adaptation of the Dijkstra Algorithm which finds the shortest path between source and destination. This is achieved by being greedy in nature. Picking path with least resistance

Proof of the shortest path is given in the folder