3rd experiment much shorter version with output from chatgpt

import socket

import threading

SERVER PROGRAM:

def handle\_client(conn):

recv\_buffer = ''

missing = []

expected = 1

buffer\_size = 8192

try:

while True:

data = conn.recv(buffer\_size).decode().split()

if not data:

break

for seq in map(int, data):

if seq == expected:

expected += 1

elif seq in missing:

missing.remove(seq)

else:

missing.append(expected)

expected += 1

recv\_buffer += str(seq)

conn.sendall((str(expected) + ' ').encode())

except:

pass

finally:

conn.close()

def server():

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

sock.bind((socket.gethostname(), 4001))

sock.listen(5)

print("Server Running ......")

while True:

conn, addr = sock.accept()

print('Connected to:', addr)

threading.Thread(target=handle\_client, args=(conn,)).start()

if \_\_name\_\_ == '\_\_main\_\_':

server()

CLIENT PROGRAM:

import socket

import random

import time

from collections import deque

# Constants

TOTAL\_PACKETS = 100

PACKET\_SIZE = 4

DROP\_PROBABILITY = 0.01

def connect():

host = socket.gethostname()

port = 4001

try:

conn = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

conn.connect((host, port))

return conn

except Exception as e:

print("Error:", str(e))

return None

def process\_packets(conn):

seq\_num = 0

pkt\_success\_sent = 0

sent\_count = 0

window\_size = 1

dropped\_pkt = deque()

while pkt\_success\_sent < TOTAL\_PACKETS:

to\_send = []

for \_ in range(window\_size):

seq\_num += 1

to\_send.append(seq\_num)

for pkt in to\_send:

sent\_count += 1

if random.random() < DROP\_PROBABILITY:

dropped\_pkt.append(pkt)

else:

conn.sendall(f"{pkt} ".encode())

pkt\_success\_sent += 1

try:

ack\_str = conn.recv(8192).decode()

acks = list(map(int, ack\_str.split()))

for ack in acks:

if ack and window\_size < 2048:

window\_size += 1

except socket.timeout:

pass

if dropped\_pkt:

window\_size = max(1, window\_size // 2)

if sent\_count % 10 == 0:

print(f"{sent\_count} packets sent...")

print("Execution Completed")

print(f"Packets sent: {sent\_count}")

conn.close()

def main():

start\_time = time.time()

conn = None

while conn is None:

conn = connect()

time.sleep(1)

if conn:

process\_packets(conn)

end\_time = time.time()

print("Runtime:", end\_time - start\_time)

if \_\_name\_\_ == "\_\_main\_\_":

main()