Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

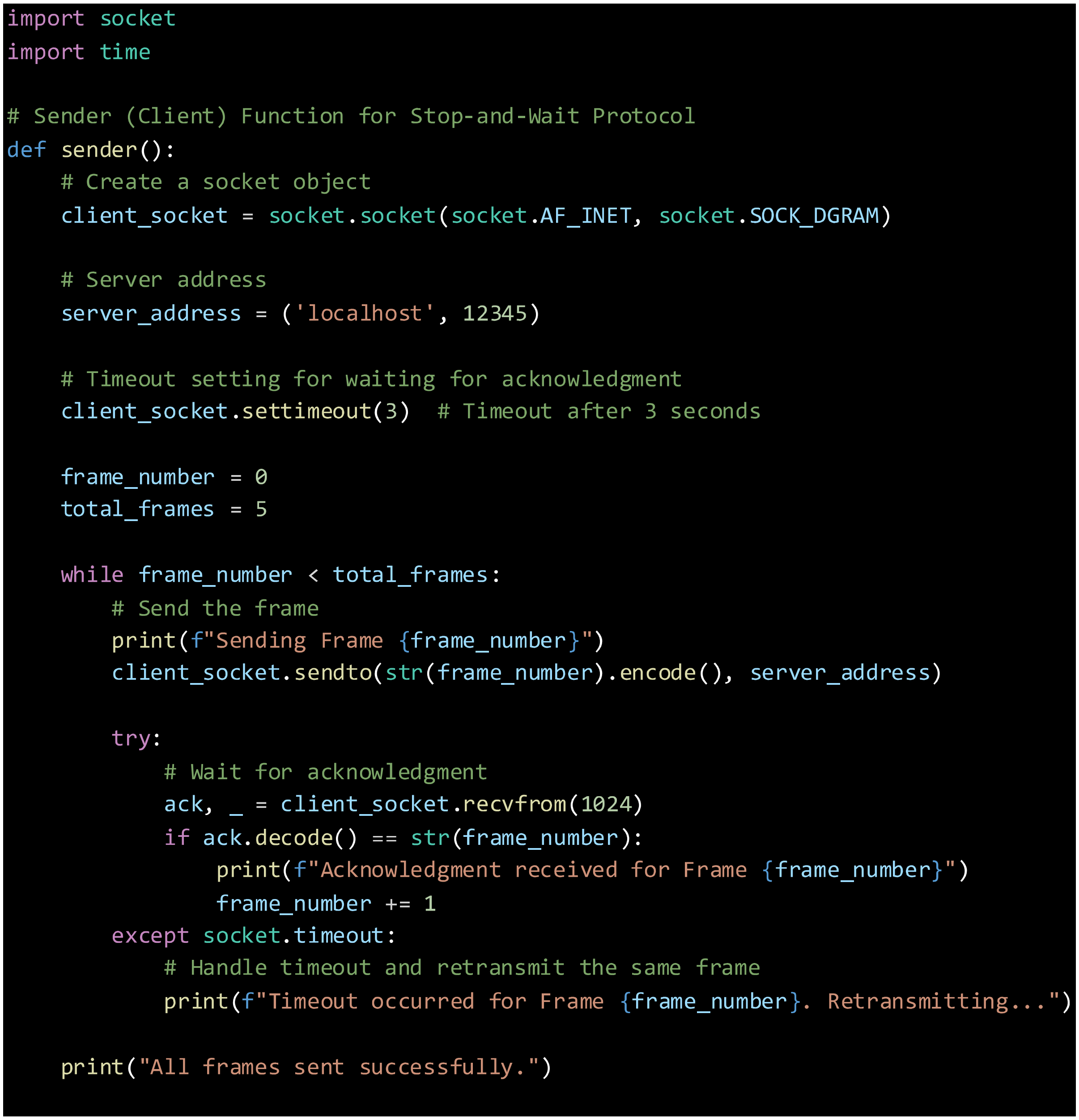
Department of Computer Engineering
Class: S.Y. B.Tech. Semester: IV
Course Code: DJ19CEL405 Course Name: Computer Networks Lab

Name: Vinit Shah	**SAP ID:60004220097**
Date of Performance:10-10-24	**Date of Submission:16-10-24**

Experiment No: 9

Aim: Write a program to implement Stop-And-Wait Protocol.

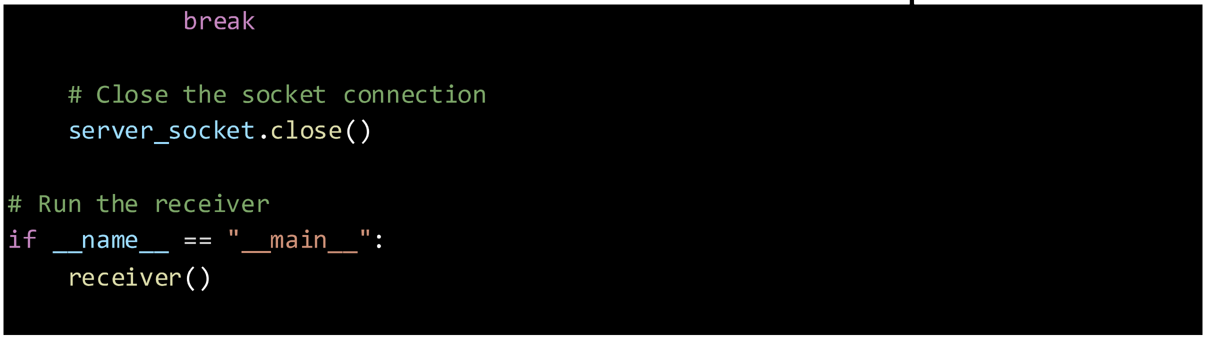
Theory:



```
import socket
import time

# Sender (Client) Function for Stop-and-Wait Protocol
def sender():
    # Create a socket object
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    # Server address
    server_address = ('localhost', 12345)

    # Timeout setting for waiting for acknowledgment
    client_socket.settimeout(3)  # Timeout after 3 seconds

    frame_number = 0
    total_frames = 5

    while frame_number < total_frames:
        # Send the frame
        print(f"Sending Frame {frame_number}")
        client_socket.sendto(str(frame_number).encode(), server_address)

        try:
            # Wait for acknowledgment
            ack, _ = client_socket.recvfrom(1024)
            if ack.decode() == str(frame_number):
                print(f"Acknowledgment received for Frame {frame_number}")
                frame_number += 1
        except socket.timeout:
            # Handle timeout and retransmit the same frame
            print(f"Timeout occurred for Frame {frame_number}. Retransmitting...")

    print("All frames sent successfully.")
    # Close the socket connection
    client_socket.close()

# Run the sender
if __name__ == "__main__":
    sender()
```

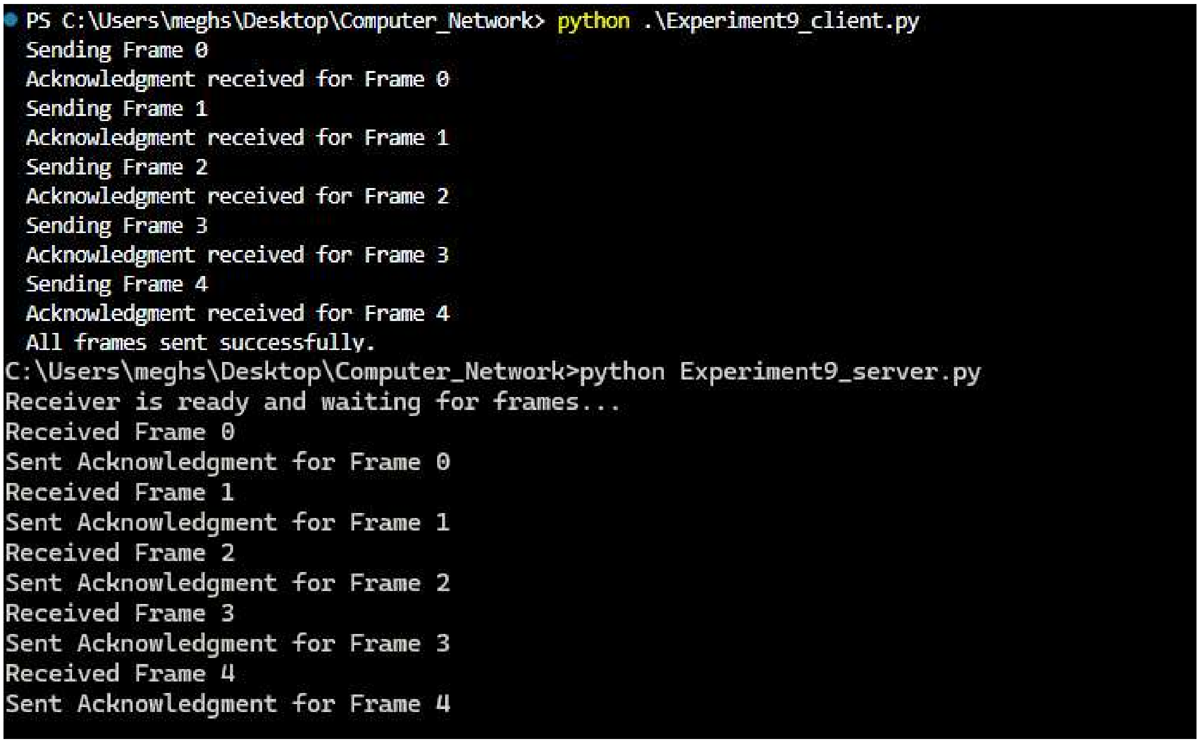
Server

```
import socket
import random
import time

# Receiver (Server) Function for Stop-and-Wait Protocol
def receiver():
    # Create a socket object
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    # Bind to an address and port
    server_socket.bind(('localhost', 12345))

    expected_frame = 0

    print("Receiver is ready and waiting for frames...")

    while True:
        # Receive a frame
        frame, client_address = server_socket.recvfrom(1024)
        frame = frame.decode()

        print(f"Received Frame {frame}")

        # Simulate acknowledgment loss or delay
        if random.random() > 0.2:  # 80% chance to send acknowledgment
            if int(frame) == expected_frame:
                time.sleep(1)  # Simulate processing delay
                server_socket.sendto(frame.encode(), client_address)
                print(f"Sent Acknowledgment for Frame {frame}")
                expected_frame += 1
        else:
            print(f"Lost Acknowledgment for Frame {frame}")

        if expected_frame >= 5:
```



```
            break

    # Close the socket connection
    server_socket.close()

# Run the receiver
if __name__ == "__main__":
    receiver()
```

Output:-



```
PS C:\Users\meghs\Desktop\Computer_Network> python .\Experiment9_client.py
 Sending Frame 0
 Acknowledgment received for Frame 0
 Sending Frame 1
 Acknowledgment received for Frame 1
 Sending Frame 2
 Acknowledgment received for Frame 2
 Sending Frame 3
 Acknowledgment received for Frame 3
 Sending Frame 4
 Acknowledgment received for Frame 4
 All frames sent successfully.
C:\Users\meghs\Desktop\Computer_Network>python Experiment9_server.py
Receiver is ready and waiting for frames...
Received Frame 0
Sent Acknowledgment for Frame 0
Received Frame 1
Sent Acknowledgment for Frame 1
Received Frame 2
Sent Acknowledgment for Frame 2
Received Frame 3
Sent Acknowledgment for Frame 3
Received Frame 4
Sent Acknowledgment for Frame 4
```

Conclusion:

Thus, we have successfully studied and implemented stop-and-wait protocol.