TCP Client-Server Communication Report

1. Server Code (helloServer.py)

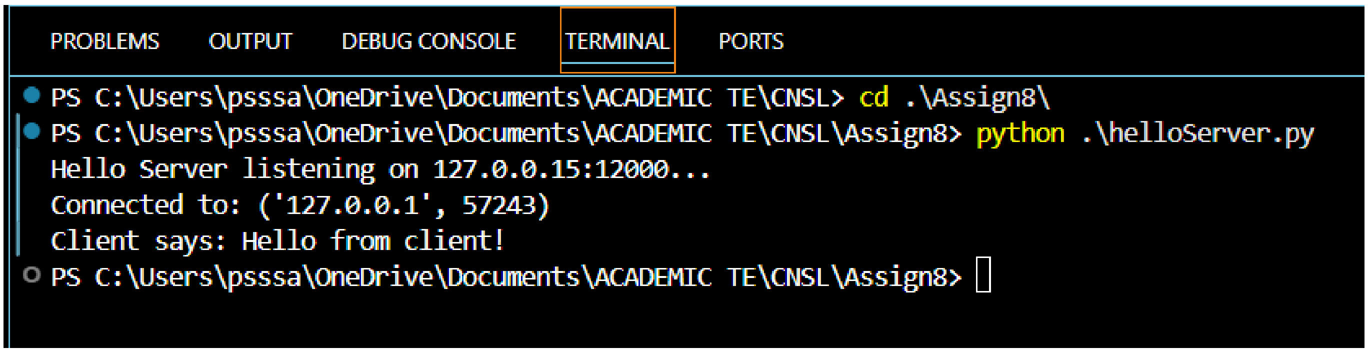
```
import socket

# Server configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 12000
BUFFER_SIZE = 1024

# Create a TCP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind socket to IP and port
server_socket.bind((SERVER_IP, SERVER_PORT))

# Listen for incoming connections
server_socket.listen(1)
print(f"Hello Server listening on {SERVER_IP}:{SERVER_PORT}...")

# Accept a client connection
conn, addr = server_socket.accept()
print(f"Connected to: {addr}")

# Exchange hello messages
conn.send(b"Hello from server!")
client_msg = conn.recv(BUFFER_SIZE).decode()
print(f"Client says: {client_msg}")

# Close connection
conn.close()
server_socket.close()
```

2. Client Code (helloClient.py)

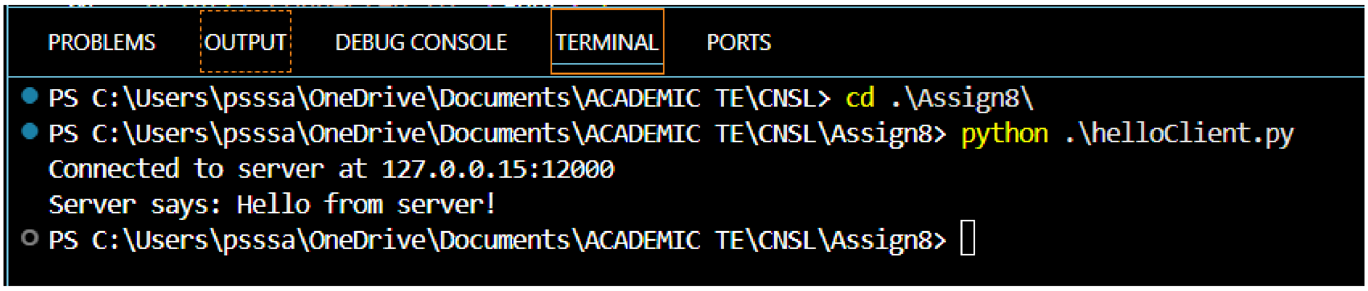
```
import socket

# Client configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 12000
BUFFER_SIZE = 1024

# Create a TCP socket
```

```
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to the server
client_socket.connect((SERVER_IP, SERVER_PORT))
print(f"Connected to server at {SERVER_IP}:{SERVER_PORT}")

# Receive message from server
server_msg = client_socket.recv(BUFFER_SIZE).decode()
print(f"Server says: {server_msg}")

# Send message to server
client_socket.send(b"Hello from client!")

# Close connection
client_socket.close()
```

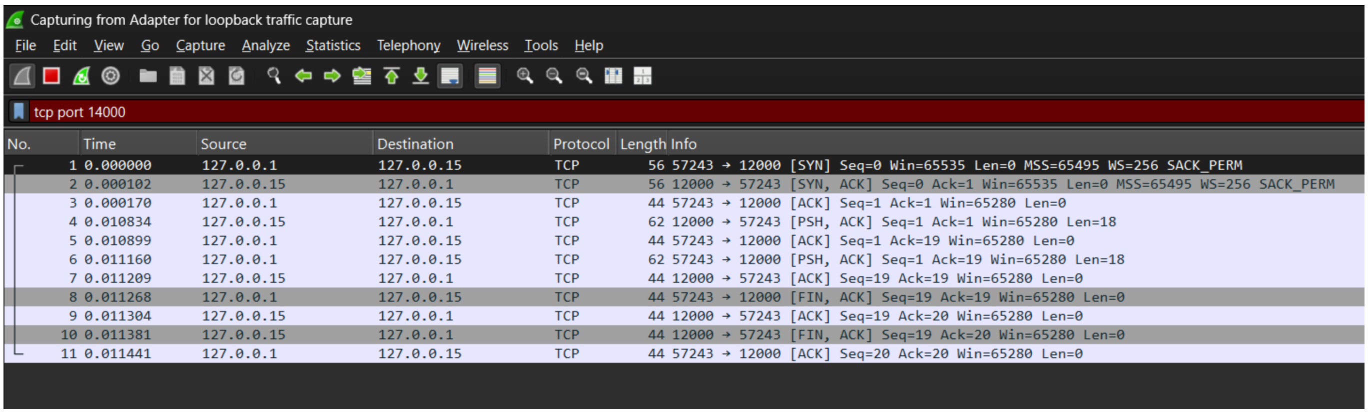
3. Execution Outputs and Wireshark Capture



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL> cd .\Assign8\
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\helloServer.py
Hello Server listening on 127.0.0.15:12000...
Connected to: ('127.0.0.1', 57243)
Client says: Hello from client!
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL> cd .\Assign8\
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\helloClient.py
Connected to server at 127.0.0.15:12000
Server says: Hello from server!
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```


Capturing from Adapter for loopback traffic capture
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp port 14000

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.15	TCP	56	57243 → 12000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
2	0.000102	127.0.0.15	127.0.0.1	TCP	56	12000 → 57243 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000170	127.0.0.1	127.0.0.15	TCP	44	57243 → 12000 [ACK] Seq=1 Ack=1 Win=65280 Len=0
4	0.010834	127.0.0.15	127.0.0.1	TCP	62	12000 → 57243 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=18
5	0.010899	127.0.0.1	127.0.0.15	TCP	44	57243 → 12000 [ACK] Seq=1 Ack=19 Win=65280 Len=0
6	0.011160	127.0.0.1	127.0.0.15	TCP	62	57243 → 12000 [PSH, ACK] Seq=1 Ack=19 Win=65280 Len=18
7	0.011209	127.0.0.15	127.0.0.1	TCP	44	12000 → 57243 [ACK] Seq=19 Ack=19 Win=65280 Len=0
8	0.011268	127.0.0.1	127.0.0.15	TCP	44	57243 → 12000 [FIN, ACK] Seq=19 Ack=19 Win=65280 Len=0
9	0.011304	127.0.0.15	127.0.0.1	TCP	44	12000 → 57243 [ACK] Seq=19 Ack=20 Win=65280 Len=0
10	0.011381	127.0.0.15	127.0.0.1	TCP	44	12000 → 57243 [FIN, ACK] Seq=19 Ack=20 Win=65280 Len=0
11	0.011441	127.0.0.1	127.0.0.15	TCP	44	57243 → 12000 [ACK] Seq=20 Ack=20 Win=65280 Len=0

TCP File Transfer (Client-Server) Report

1. File Server Code (fileServer.py)

```
import socket

# Server configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 13000
BUFFER_SIZE = 1024

# Create TCP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind((SERVER_IP, SERVER_PORT))
server_socket.listen(1)
print(f"File Server listening on {SERVER_IP}:{SERVER_PORT}...")

# Accept client connection
conn, addr = server_socket.accept()
print(f"Connected to: {addr}")

try:
    # Step 1: Ask for filename
    conn.send(b"Send filename: ")
    filename = conn.recv(BUFFER_SIZE).decode().strip()

    # Step 2: Notify client server is ready
    conn.send(b"Ready to receive file...")

    # Step 3: Receive and save file
    with open(f"server_{filename}", 'wb') as f:
        while True:
            data = conn.recv(BUFFER_SIZE)
            if not data:
                break
            f.write(data)

    print(f"Received file: server_{filename}")

    # Step 4: Send acknowledgment
    conn.send(b"File received successfully.")

except ConnectionResetError:
    print("Connection closed by client unexpectedly.")
```

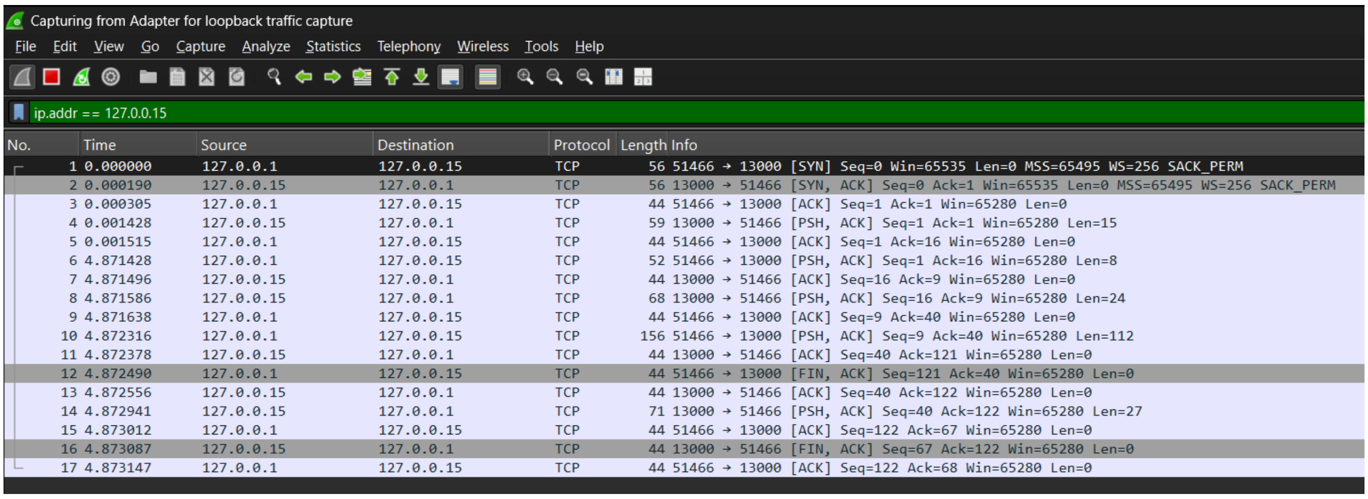
```
except Exception as e:
    print(f"Error: {e}")

# Step 5: Close connection
conn.close()
server_socket.close()
```

2. File Client Code (fileClient.py)

```
import socket
import os

# Client configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 13000
BUFFER_SIZE = 1024

# Create TCP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect((SERVER_IP, SERVER_PORT))

# Step 1: Receive prompt for filename
prompt = client_socket.recv(BUFFER_SIZE).decode()
filename = input(prompt).strip()
client_socket.send(filename.encode())

# Step 2: Wait for server ready message
ready_msg = client_socket.recv(BUFFER_SIZE).decode()
print(ready_msg)

# Step 3: Verify file exists
if not os.path.exists(filename):
    print("File not found. Please check the filename.")
    client_socket.close()
    exit()

# Step 4: Send file contents
with open(filename, 'rb') as f:
    while True:
        bytes_read = f.read(BUFFER_SIZE)
        if not bytes_read:
            break
        client_socket.sendall(bytes_read)

# Gracefully close the sending side
```

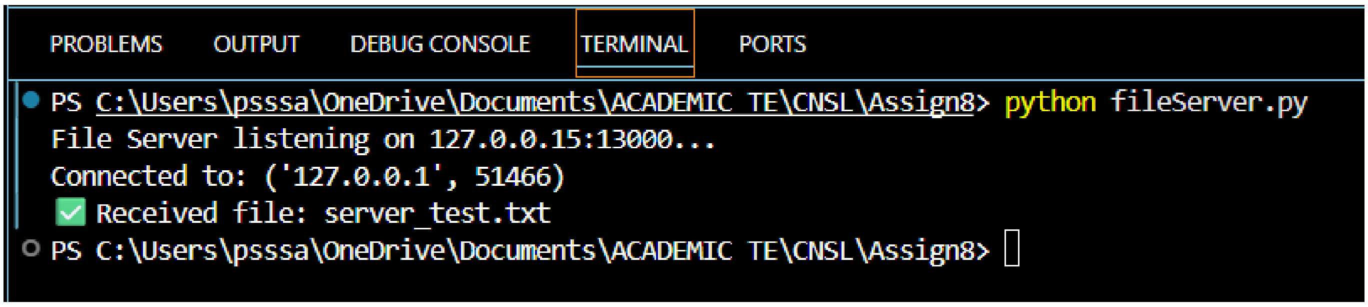
```
client_socket.shutdown(socket.SHUT_WR)

# Step 5: Receive acknowledgment
ack = client_socket.recv(BUFFER_SIZE).decode()
print(ack)

# Step 6: Close connection
client_socket.close()
```

3. Execution Outputs and Wireshark Capture

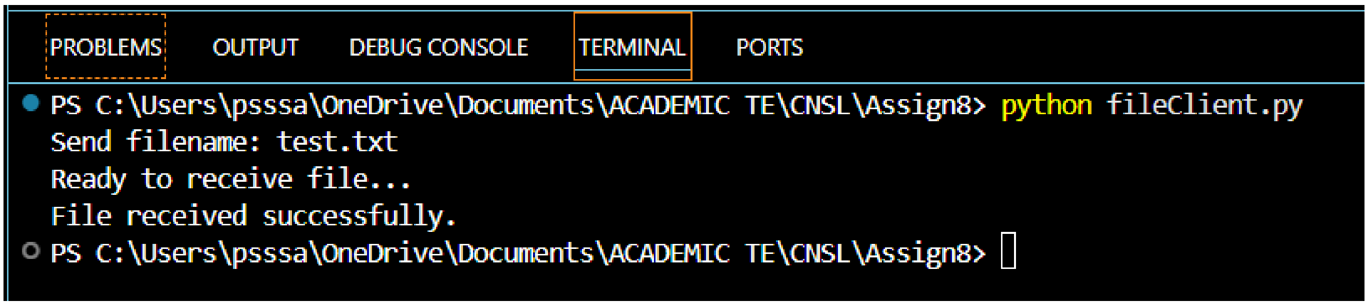


Capturing from Adapter for loopback traffic capture
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.addr == 127.0.0.15

```
No.  Time       Source      Destination  Protocol Length Info
  1 0.000000   127.0.0.1   127.0.0.15   TCP       56 51466 → 13000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
  2 0.000190   127.0.0.15  127.0.0.1    TCP       56 13000 → 51466 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
  3 0.000305   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [ACK] Seq=1 Ack=1 Win=65280 Len=0
  4 0.001428   127.0.0.15  127.0.0.1    TCP       59 13000 → 51466 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=15
  5 0.001515   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [ACK] Seq=1 Ack=16 Win=65280 Len=0
  6 4.871428   127.0.0.1   127.0.0.15   TCP       52 51466 → 13000 [PSH, ACK] Seq=1 Ack=16 Win=65280 Len=8
  7 4.871496   127.0.0.15  127.0.0.1    TCP       44 13000 → 51466 [ACK] Seq=16 Ack=9 Win=65280 Len=0
  8 4.871586   127.0.0.15  127.0.0.1    TCP       68 13000 → 51466 [PSH, ACK] Seq=16 Ack=9 Win=65280 Len=24
  9 4.871638   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [ACK] Seq=9 Ack=40 Win=65280 Len=0
 10 4.872316   127.0.0.1   127.0.0.15   TCP      156 51466 → 13000 [PSH, ACK] Seq=9 Ack=40 Win=65280 Len=112
 11 4.872378   127.0.0.15  127.0.0.1    TCP       44 13000 → 51466 [ACK] Seq=40 Ack=121 Win=65280 Len=0
 12 4.872490   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [FIN, ACK] Seq=121 Ack=40 Win=65280 Len=0
 13 4.872556   127.0.0.15  127.0.0.1    TCP       44 13000 → 51466 [ACK] Seq=40 Ack=122 Win=65280 Len=0
 14 4.872941   127.0.0.15  127.0.0.1    TCP       71 13000 → 51466 [PSH, ACK] Seq=40 Ack=122 Win=65280 Len=27
 15 4.873012   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [ACK] Seq=122 Ack=67 Win=65280 Len=0
 16 4.873087   127.0.0.15  127.0.0.1    TCP       44 13000 → 51466 [FIN, ACK] Seq=67 Ack=122 Win=65280 Len=0
 17 4.873147   127.0.0.1   127.0.0.15   TCP       44 51466 → 13000 [ACK] Seq=122 Ack=68 Win=65280 Len=0
```



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python fileServer.py
File Server listening on 127.0.0.15:13000...
Connected to: ('127.0.0.1', 51466)
✅ Received file: server_test.txt
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python fileClient.py
Send filename: test.txt
Ready to receive file...
File received successfully.
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```

TCP Calculator (Client-Server) Report

1. Calculator Server Code (calcServer.py)

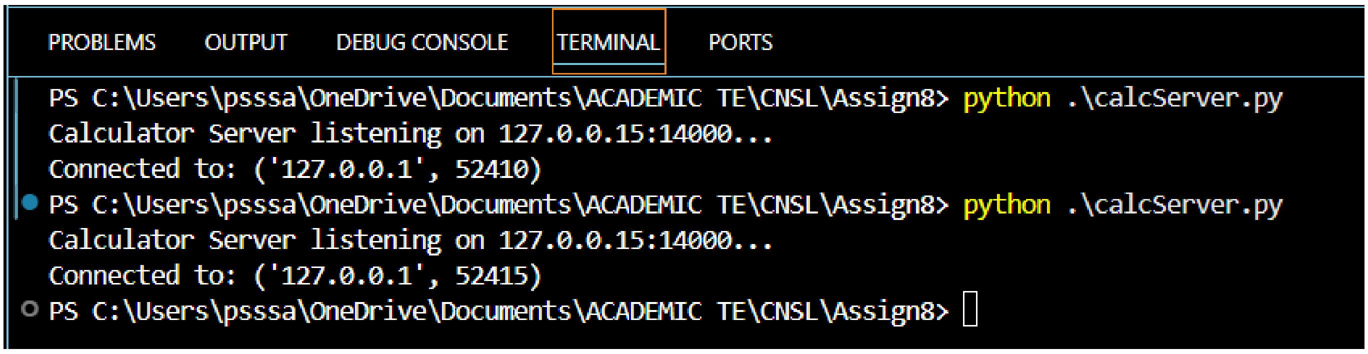
```
import socket
import math

# Server configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 14000
BUFFER_SIZE = 1024

# Create TCP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind((SERVER_IP, SERVER_PORT))
server_socket.listen(1)
print(f"Calculator Server listening on {SERVER_IP}:{SERVER_PORT}...")

# Accept client connection
conn, addr = server_socket.accept()
print(f"Connected to: {addr}")

# Ask for operation type
conn.send(b"Choose operation type (arithmetic/trigonometric): ")
op_type = conn.recv(BUFFER_SIZE).decode().strip().lower()

# --- Arithmetic Operations ---
if op_type == 'arithmetic':
    conn.send(b"Enter expression (e.g., 5 + 3): ")
    expr = conn.recv(BUFFER_SIZE).decode()

    try:
        result = eval(expr)
        conn.send(f"Result: {result}".encode())
    except Exception as e:
        conn.send(f"Error: {e}".encode())

# --- Trigonometric Operations ---
elif op_type == 'trigonometric':
    conn.send(b"Enter function and value (e.g., sin 30): ")
    func_val = conn.recv(BUFFER_SIZE).decode().split()

    if len(func_val) != 2:
        conn.send(b"Invalid input")
    else:
        func, val = func_val[0].lower(), float(func_val[1])
        val_rad = math.radians(val)
        try:
            if func == 'sin':
                result = math.sin(val_rad)
            elif func == 'cos':
                result = math.cos(val_rad)
            elif func == 'tan':
                result = math.tan(val_rad)
            else:
                conn.send(b"Unsupported function")
                conn.close()
                server_socket.close()
                exit()
            conn.send(f"Result: {result}".encode())
        except Exception as e:
            conn.send(f"Error: {e}".encode())

# --- Invalid Operation Type ---
else:
    conn.send(b"Invalid operation type.")

# Close connection
conn.close()
server_socket.close()
```

2. Calculator Client Code (calcClient.py)

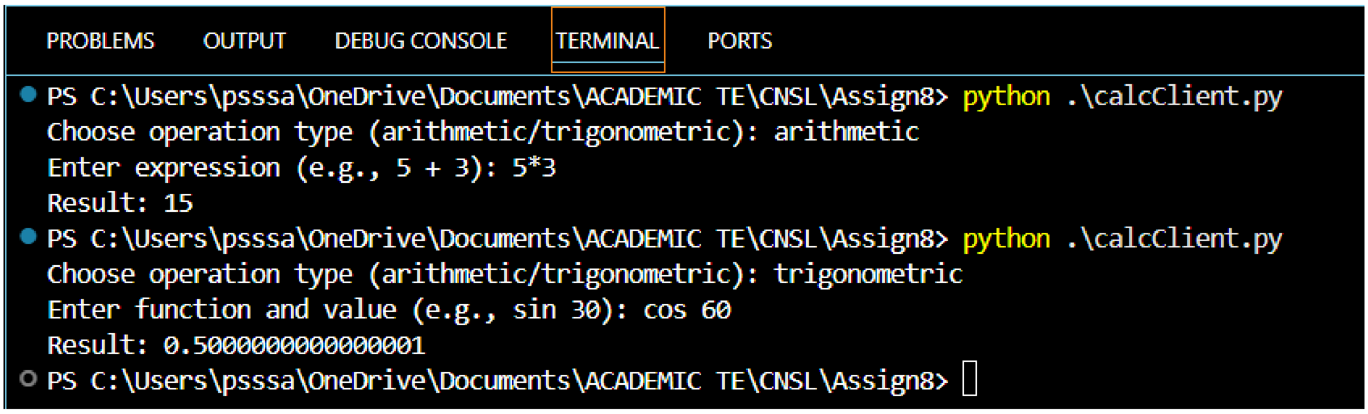
```
import socket

# Client configuration
SERVER_IP = '127.0.0.15'
SERVER_PORT = 14000
BUFFER_SIZE = 1024

# Create TCP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect((SERVER_IP, SERVER_PORT))

# Step 1: Receive prompt for operation type
prompt = client_socket.recv(BUFFER_SIZE).decode()
op_type = input(prompt)
client_socket.send(op_type.encode())
```

```
# Step 2: Receive second prompt
prompt2 = client_socket.recv(BUFFER_SIZE).decode()
expr = input(prompt2)
client_socket.send(expr.encode())

# Step 3: Receive and display result
result = client_socket.recv(BUFFER_SIZE).decode()
print(result)

# Close connection
client_socket.close()
```

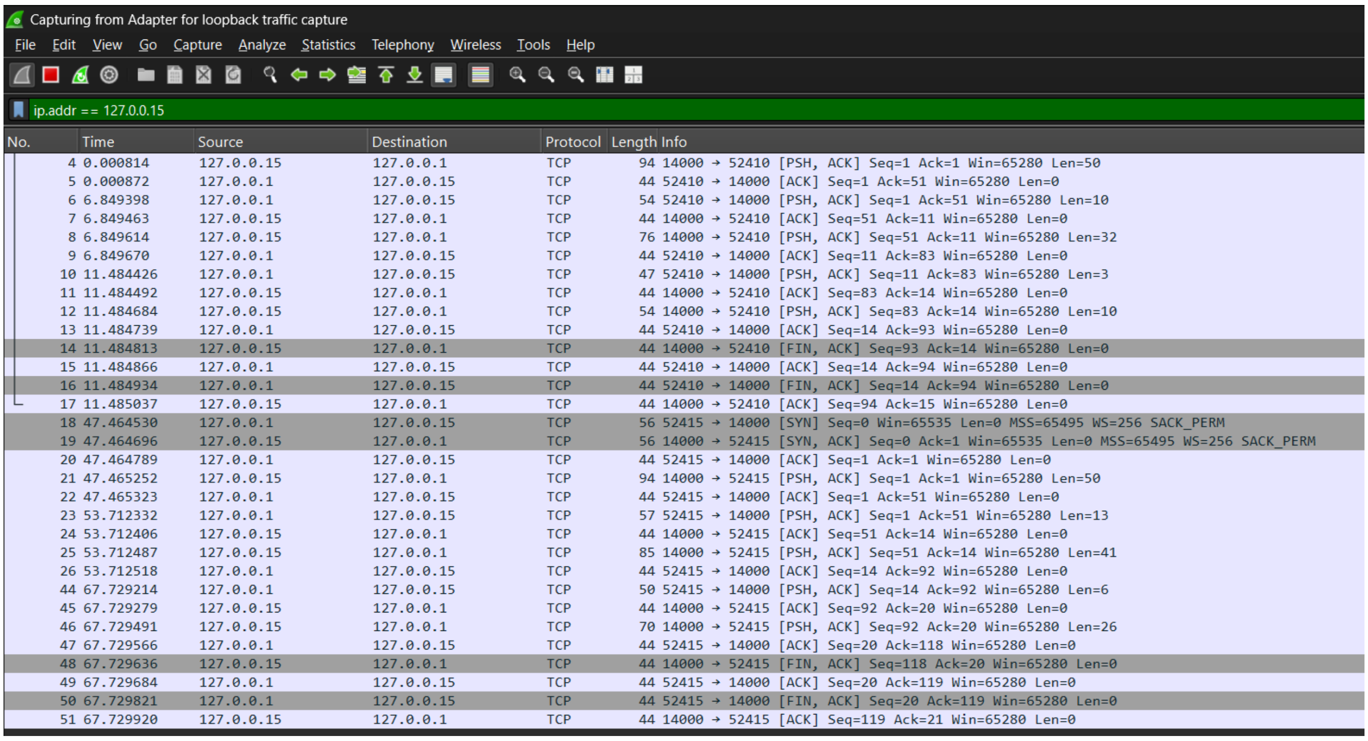
3. Execution Outputs and Wireshark Capture



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\calcServer.py
Calculator Server listening on 127.0.0.15:14000...
Connected to: ('127.0.0.1', 52410)
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\calcServer.py
Calculator Server listening on 127.0.0.15:14000...
Connected to: ('127.0.0.1', 52415)
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```



```
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\calcClient.py
Choose operation type (arithmetic/trigonometric): arithmetic
Enter expression (e.g., 5 + 3): 5*3
Result: 15
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8> python .\calcClient.py
Choose operation type (arithmetic/trigonometric): trigonometric
Enter function and value (e.g., sin 30): cos 60
Result: 0.5000000000000001
PS C:\Users\psssa\OneDrive\Documents\ACADEMIC TE\CNSL\Assign8>
```



```
ip.addr == 127.0.0.15
```

```
No.     Time        Source       Destination  Protocol Length Info
     4 0.000814    127.0.0.15   127.0.0.1    TCP     94 14000 → 52410 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=50
     5 0.000872    127.0.0.1    127.0.0.15   TCP     44 52410 → 14000 [ACK] Seq=1 Ack=51 Win=65280 Len=0
     6 6.849398    127.0.0.1    127.0.0.15   TCP     54 52410 → 14000 [PSH, ACK] Seq=1 Ack=51 Win=65280 Len=10
     7 6.849463    127.0.0.15   127.0.0.1    TCP     44 14000 → 52410 [ACK] Seq=51 Ack=11 Win=65280 Len=0
     8 6.849614    127.0.0.15   127.0.0.1    TCP     76 14000 → 52410 [PSH, ACK] Seq=51 Ack=11 Win=65280 Len=32
     9 6.849670    127.0.0.1    127.0.0.15   TCP     44 52410 → 14000 [ACK] Seq=11 Ack=83 Win=65280 Len=0
    10 11.484426   127.0.0.1    127.0.0.15   TCP     47 52410 → 14000 [PSH, ACK] Seq=11 Ack=83 Win=65280 Len=3
    11 11.484492   127.0.0.15   127.0.0.1    TCP     44 14000 → 52410 [ACK] Seq=83 Ack=14 Win=65280 Len=0
    12 11.484684   127.0.0.15   127.0.0.1    TCP     54 14000 → 52410 [PSH, ACK] Seq=83 Ack=14 Win=65280 Len=10
    13 11.484739   127.0.0.1    127.0.0.15   TCP     44 52410 → 14000 [ACK] Seq=14 Ack=93 Win=65280 Len=0
    14 11.484813   127.0.0.15   127.0.0.1    TCP     44 14000 → 52410 [FIN, ACK] Seq=93 Ack=14 Win=65280 Len=0
    15 11.484866   127.0.0.1    127.0.0.15   TCP     44 52410 → 14000 [ACK] Seq=14 Ack=94 Win=65280 Len=0
    16 11.484934   127.0.0.1    127.0.0.15   TCP     44 52410 → 14000 [FIN, ACK] Seq=14 Ack=94 Win=65280 Len=0
    17 11.485037   127.0.0.15   127.0.0.1    TCP     44 14000 → 52410 [ACK] Seq=94 Ack=15 Win=65280 Len=0
    18 47.464530   127.0.0.1    127.0.0.15   TCP     56 52415 → 14000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
    19 47.464696   127.0.0.15   127.0.0.1    TCP     56 14000 → 52415 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
    20 47.464789   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [ACK] Seq=1 Ack=1 Win=65280 Len=0
    21 47.465252   127.0.0.15   127.0.0.1    TCP     94 14000 → 52415 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=50
    22 47.465323   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [ACK] Seq=1 Ack=51 Win=65280 Len=0
    23 53.712332   127.0.0.1    127.0.0.15   TCP     57 52415 → 14000 [PSH, ACK] Seq=1 Ack=51 Win=65280 Len=13
    24 53.712406   127.0.0.15   127.0.0.1    TCP     44 14000 → 52415 [ACK] Seq=51 Ack=14 Win=65280 Len=0
    25 53.712487   127.0.0.15   127.0.0.1    TCP     85 14000 → 52415 [PSH, ACK] Seq=51 Ack=14 Win=65280 Len=41
    26 53.712518   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [ACK] Seq=14 Ack=92 Win=65280 Len=0
    44 67.729214   127.0.0.1    127.0.0.15   TCP     50 52415 → 14000 [PSH, ACK] Seq=14 Ack=92 Win=65280 Len=6
    45 67.729279   127.0.0.15   127.0.0.1    TCP     44 14000 → 52415 [ACK] Seq=92 Ack=20 Win=65280 Len=0
    46 67.729491   127.0.0.15   127.0.0.1    TCP     70 14000 → 52415 [PSH, ACK] Seq=92 Ack=20 Win=65280 Len=26
    47 67.729566   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [ACK] Seq=20 Ack=118 Win=65280 Len=0
    48 67.729636   127.0.0.15   127.0.0.1    TCP     44 14000 → 52415 [FIN, ACK] Seq=118 Ack=20 Win=65280 Len=0
    49 67.729684   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [ACK] Seq=20 Ack=119 Win=65280 Len=0
    50 67.729821   127.0.0.1    127.0.0.15   TCP     44 52415 → 14000 [FIN, ACK] Seq=20 Ack=119 Win=65280 Len=0
    51 67.729920   127.0.0.15   127.0.0.1    TCP     44 14000 → 52415 [ACK] Seq=119 Ack=21 Win=65280 Len=0
```