

1. Program Description

This project implements a simple client-server application using TCP sockets. The client reads an integer (between 1 and 100) from the keyboard, sends it along with its name to the server, and then receives a response containing the server's name and its own chosen number. Both client and server then calculate and display the sum of the two numbers. In addition, the server has been implemented as a concurrent server using multi-threading, allowing it to handle multiple client connections simultaneously.

2. User Guide

Python 3 installed

Both 'server.py' and 'client.py' files in the same directory

Running the Server:

1. Open a terminal.
2. Navigate to the directory containing the code.
3. Run the server using: `python3 server.py`
4. The server will start listening on port 5001

Running the Client:

1. Open a terminal.
2. Navigate to the directory containing the code.
3. Run the client using: `python3 client.py`
4. When prompted, enter an integer between 1 and 100.
5. The client will connect to the server, send the data, and then display the response and calculated sum.

3. Choice of Programming Language

The project is implemented in Python 3. Python was chosen due to its simplicity in handling sockets and threading, which makes the development of network applications straightforward and efficient.

4. Program Design and Implementation Details

- **Client:**

1. Prompting for User Input and Formatting the Message

```

4  ✓ def start_client():
5      # Prompt the user for an integer input between 1 and 100
6      user_input = input("Please enter an integer between 1 and 100: ")
7      try:
8          client_number = int(user_input)
9      except ValueError:
10         print("[ERROR] Invalid input. Please enter a valid integer.")
11         return
12
13     # Format the message as "Client of John Q. Smith;[number]"
14     message = f"Client of John Q. Smith;{client_number}"
15

```

2.Establishing a Connection and Sending the Message

```

# Create a TCP/IP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Define the server's IP and port (using localhost for testing)
SERVER_IP = "127.0.0.1"
SERVER_PORT = 5001

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

    # Send the formatted message to the server
    client_socket.send(message.encode("utf-8"))
    print(f"[INFO] Sent message: {message}")

    # Wait for the server's response (up to 1024 bytes)
    response = client_socket.recv(1024).decode("utf-8")
    if not response:
        print("[ERROR] No response received from the server.")
        return

```

3.Receiving and Parsing the Server's Response

```

# Wait for the server's response (up to 1024 bytes)
response = client_socket.recv(1024).decode("utf-8")
if not response:
    print("[ERROR] No response received from the server.")
    return

# Expected response format: "Server of John Q. Smith;[server_number]"
parts = response.strip().split(";")
if len(parts) != 2:
    print("[ERROR] Server response format is incorrect.")
    return

# Extract the server's name and number
server_name = parts[0]
try:
    server_number = int(parts[1])
except ValueError:
    print("[ERROR] The server's number is not a valid integer.")
    return

```

4. Calculating the Sum and Displaying the Information

```

# Calculate the sum of the client's number and the server's number
total_sum = client_number + server_number

# Display the server's information and the sum
print(f"[INFO] Received response from {server_name} with number {server_number}.")
print(f"[INFO] Calculation: {client_number} (client) + {server_number} (server) = {total_sum}")

```

- **Server:**

1. Binding to a Specific Port and Listening for Connections

```

def start_server():
    """
    Starts the server, binds to a port, and listens for incoming connections.
    Each incoming connection is handled in a separate thread.
    """
    # Create a TCP/IP socket
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        # Bind the socket to all available network interfaces on the specified port
        server_socket.bind(("", SERVER_PORT))
    except Exception as e:
        print(f"[ERROR] Could not bind to port {SERVER_PORT}: {e}")
        sys.exit(1)

    # Start listening for incoming connections (allow up to 5 queued connections)
    server_socket.listen(5)
    print(f"[INFO] Server is listening on port {SERVER_PORT}...")

```

2. Handling Each Connection Concurrently

```

while True:
    try:
        # Accept a new client connection
        client_socket, client_address = server_socket.accept()
        # For each connection, create a new thread to handle it concurrently
        client_thread = threading.Thread(target=handle_client, args=(client_socket, client_address))
        client_thread.daemon = True # Daemon threads will automatically close when the main program exits
        client_thread.start()
    except KeyboardInterrupt:
        # Allow the server to be stopped with Ctrl+C
        print("\n[INFO] Keyboard interrupt received. Shutting down the server...")
        server_socket.close()
        break

```

3. Validating Received Data and Checking Number Range

```
10
11 ✓ def handle_client(client_socket, client_address):
12     """
13     Handles an individual client connection.
14     This function is executed in a separate thread for each client.
15     """
16     print(f"[INFO] Received connection from {client_address}.")
17     try:
18         # Receive data from the client (up to 1024 bytes)
19         data = client_socket.recv(1024).decode("utf-8")
20         if not data:
21             print("[ERROR] No data received from the client.")
22             client_socket.close()
23             return
24
25         # Expected data format: "Client of John Q. Smith;[number]"
26         parts = data.strip().split(";")
27         if len(parts) != 2:
28             print(f"[ERROR] Data format incorrect: {data}")
29             client_socket.close()
30             return
31
32         # Extract the client's name and number from the message
33         client_name = parts[0]
34         try:
35             client_number = int(parts[1])
36         except ValueError:
37             print(f"[ERROR] The client's number is not a valid integer.")
38             client_socket.close()
39             return
40
```

4. Calculating the Sum and Sending the Response

```
# Calculate the sum of the client's number and the server's predetermined number
total_sum = client_number + SERVER_NUM
print(f"[INFO] Calculation: {client_number} (client) + {SERVER_NUM} (server) = {total_sum}")

# Construct the response message in the format: "Server of John Q. Smith;[SERVER_NUM]"
reply = f"{SERVER_NAME};{SERVER_NUM}"
client_socket.send(reply.encode("utf-8"))
print(f"[INFO] Sent reply to client: {reply}")
```

- **Concurrency:**

1. Accepting each incoming connection in a loop.

2. Creating a new thread for each connection

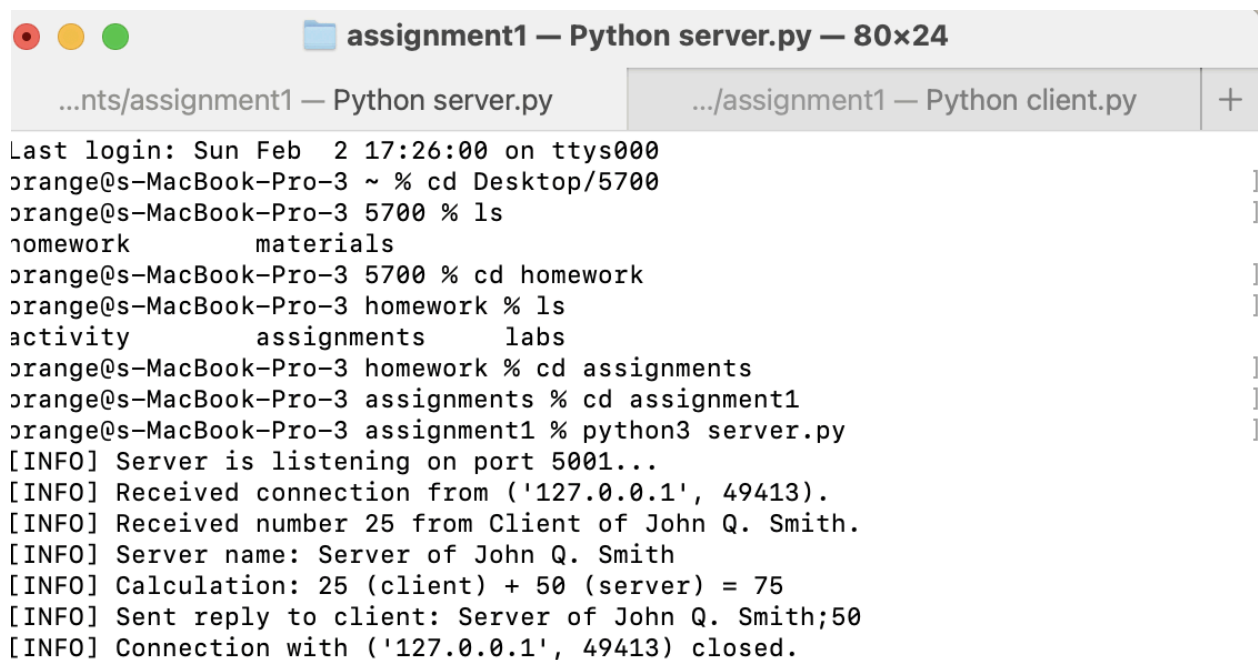
```
while True:
    try:
        # Accept a new client connection
        client_socket, client_address = server_socket.accept()
        # For each connection, create a new thread to handle it concurrently
        client_thread = threading.Thread(target=handle_client, args=(client_socket, client_address))
        client_thread.daemon = True # Daemon threads will automatically close when the main program exit
        client_thread.start()
    except KeyboardInterrupt:
        # Allow the server to be stopped with Ctrl+C
        print("\n[INFO] Keyboard interrupt received. Shutting down the server...")
        server_socket.close()
        break
```

3. This allows multiple clients to be served at the same time without blocking one another.

5. Testing and Screenshots

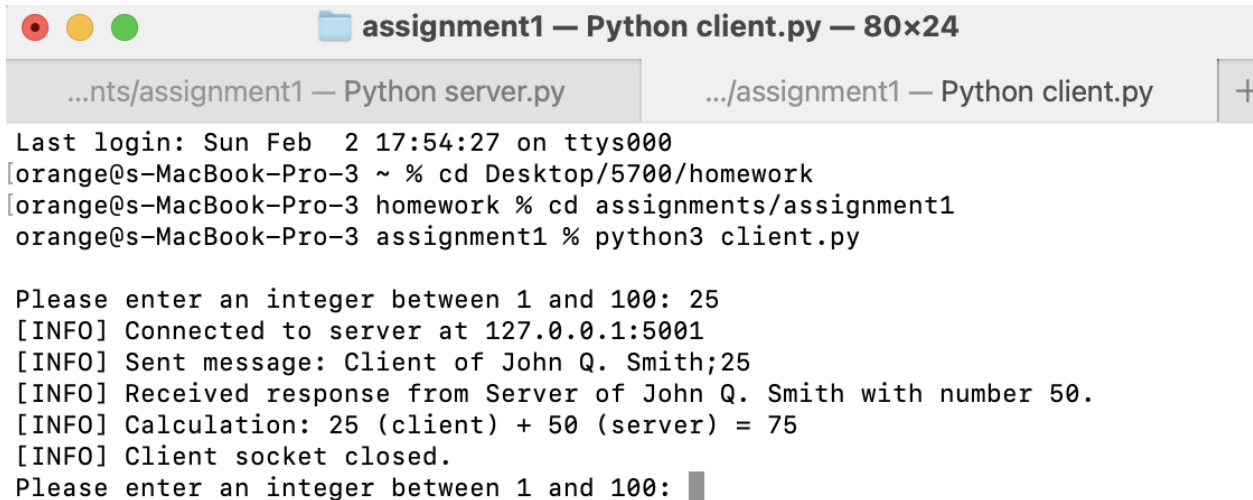
Test Case 1: Valid Input

Server:



```
...nts/assignment1 — Python server.py | .../assignment1 — Python client.py | +
Last login: Sun Feb  2 17:26:00 on ttys000
orange@s-MacBook-Pro-3 ~ % cd Desktop/5700
orange@s-MacBook-Pro-3 5700 % ls
homework      materials
orange@s-MacBook-Pro-3 5700 % cd homework
orange@s-MacBook-Pro-3 homework % ls
activity      assignments   labs
orange@s-MacBook-Pro-3 homework % cd assignments
orange@s-MacBook-Pro-3 assignments % cd assignment1
orange@s-MacBook-Pro-3 assignment1 % python3 server.py
[INFO] Server is listening on port 5001...
[INFO] Received connection from ('127.0.0.1', 49413).
[INFO] Received number 25 from Client of John Q. Smith.
[INFO] Server name: Server of John Q. Smith
[INFO] Calculation: 25 (client) + 50 (server) = 75
[INFO] Sent reply to client: Server of John Q. Smith;50
[INFO] Connection with ('127.0.0.1', 49413) closed.
```

Client:



```
assignment1 — Python client.py — 80x24
...nts/assignment1 — Python server.py
.../assignment1 — Python client.py
Last login: Sun Feb  2 17:54:27 on ttys000
[orange@s-MacBook-Pro-3 ~ % cd Desktop/5700/homework
[orange@s-MacBook-Pro-3 homework % cd assignments/assignment1
orange@s-MacBook-Pro-3 assignment1 % python3 client.py

Please enter an integer between 1 and 100: 25
[INFO] Connected to server at 127.0.0.1:5001
[INFO] Sent message: Client of John Q. Smith;25
[INFO] Received response from Server of John Q. Smith with number 50.
[INFO] Calculation: 25 (client) + 50 (server) = 75
[INFO] Client socket closed.
Please enter an integer between 1 and 100: █
```

Test Case 2: Out-of-Range Input

Server:

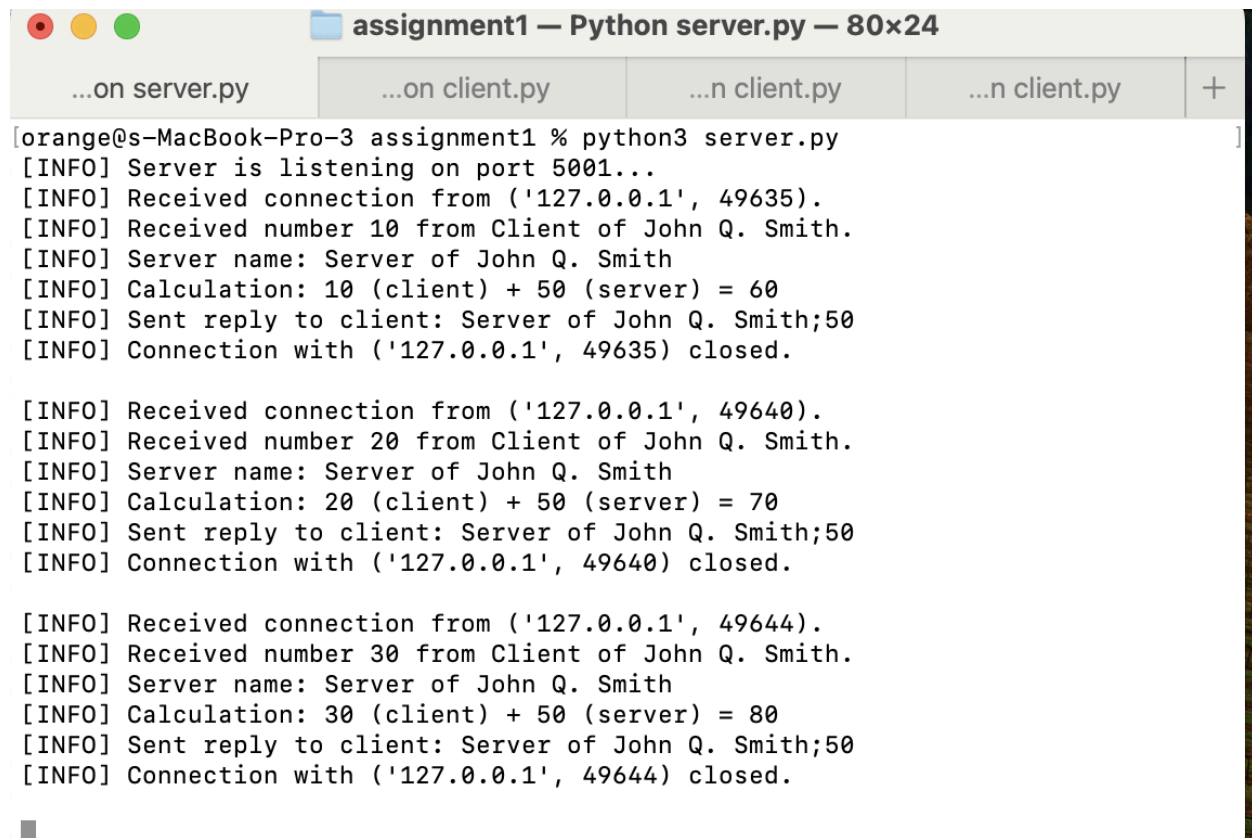
```
[INFO] Received connection from ('127.0.0.1', 49464).
[INFO] Received number 101 from Client of John Q. Smith.
[INFO] Server name: Server of John Q. Smith
[INFO] Received out-of-range number 101. Shutting down the server.
[INFO] Connection with ('127.0.0.1', 49464) closed.
```

Client:

```
Please enter an integer between 1 and 100: 101
[INFO] Connected to server at 127.0.0.1:5001
[INFO] Sent message: Client of John Q. Smith;101
[ERROR] No response received from the server.
[INFO] Client socket closed.
orange@s-MacBook-Pro-3 assignment1 % █
```

Concurrent Connections

Server:

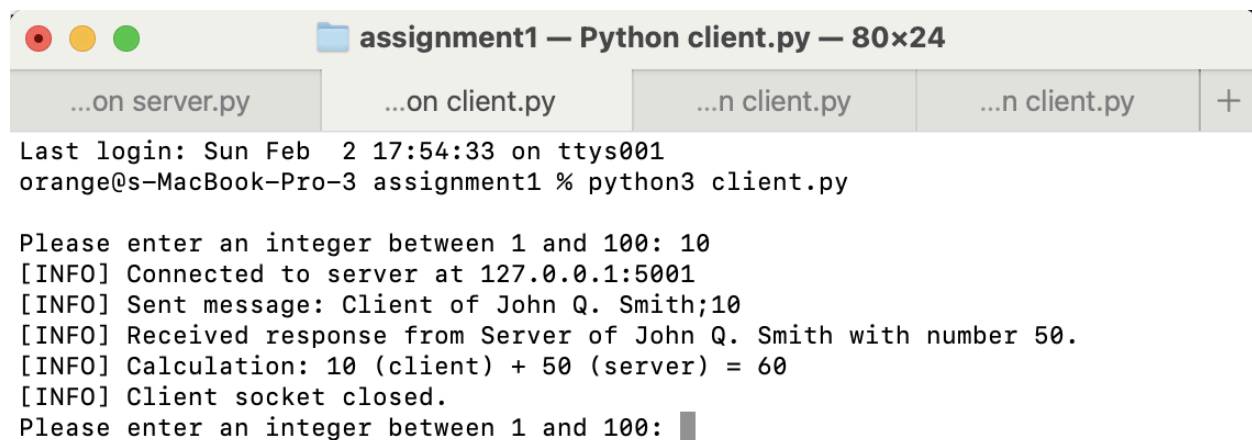


```
[orange@s-MacBook-Pro-3 assignment1 % python3 server.py
[INFO] Server is listening on port 5001...
[INFO] Received connection from ('127.0.0.1', 49635).
[INFO] Received number 10 from Client of John Q. Smith.
[INFO] Server name: Server of John Q. Smith
[INFO] Calculation: 10 (client) + 50 (server) = 60
[INFO] Sent reply to client: Server of John Q. Smith;50
[INFO] Connection with ('127.0.0.1', 49635) closed.

[INFO] Received connection from ('127.0.0.1', 49640).
[INFO] Received number 20 from Client of John Q. Smith.
[INFO] Server name: Server of John Q. Smith
[INFO] Calculation: 20 (client) + 50 (server) = 70
[INFO] Sent reply to client: Server of John Q. Smith;50
[INFO] Connection with ('127.0.0.1', 49640) closed.

[INFO] Received connection from ('127.0.0.1', 49644).
[INFO] Received number 30 from Client of John Q. Smith.
[INFO] Server name: Server of John Q. Smith
[INFO] Calculation: 30 (client) + 50 (server) = 80
[INFO] Sent reply to client: Server of John Q. Smith;50
[INFO] Connection with ('127.0.0.1', 49644) closed.
```

Client1:



```
Last login: Sun Feb  2 17:54:33 on ttys001
orange@s-MacBook-Pro-3 assignment1 % python3 client.py

Please enter an integer between 1 and 100: 10
[INFO] Connected to server at 127.0.0.1:5001
[INFO] Sent message: Client of John Q. Smith;10
[INFO] Received response from Server of John Q. Smith with number 50.
[INFO] Calculation: 10 (client) + 50 (server) = 60
[INFO] Client socket closed.
Please enter an integer between 1 and 100: █
```

Client2:


```
assignment1 — Python client.py — 80x24
...on server.py  ...on client.py  ×  ...n client.py  ...n client.py  +
Last login: Sun Feb  2 18:01:12 on ttys000
orange@s-MacBook-Pro-3 assignment1 % python3 cl: ~/Desktop/5700/homework/assignments/assignment1 — Python client.py

Please enter an integer between 1 and 100: 20
[INFO] Connected to server at 127.0.0.1:5001
[INFO] Sent message: Client of John Q. Smith;20
[INFO] Received response from Server of John Q. Smith with number 50.
[INFO] Calculation: 20 (client) + 50 (server) = 70
[INFO] Client socket closed.
Please enter an integer between 1 and 100: █
```

Client3:

```
assignment1 — Python client.py — 80x24
...on server.py  ...on client.py  ...n client.py  ...n client.py  +
Last login: Sun Feb  2 18:01:31 on ttys002
orange@s-MacBook-Pro-3 assignment1 % python3 client.py

Please enter an integer between 1 and 100: 30
[INFO] Connected to server at 127.0.0.1:5001
[INFO] Sent message: Client of John Q. Smith;30
[INFO] Received response from Server of John Q. Smith with number 50.
[INFO] Calculation: 30 (client) + 50 (server) = 80
[INFO] Client socket closed.
Please enter an integer between 1 and 100: █
```

6. Source Code

<https://github.com/Orange-135/5700/tree/main/homework/assignments/assignment1>