National University of Computer and Emerging Sciences

Laboratory Manuals

for

Computer Networks - Lab

(CL -3001)

Department of Computer Science
FAST-NU, Lahore, Pakistan

Lab Manual 05

Objective:

Students should know:

- What a computer network is and what its advantages are.
- What is OSI Model?
- What is a socket?
- Client-Server Model
- TCP Socket Programming.

In-lab Statement 1: [5]

- Write **TCP** client and server that can communicate to each other saying

"Hello I am client and My id is 1" and "Hello I am server. Your received id is 1"

- The ID of the client should be only a **single digit** i.e from 0 to 9
- Run one client and server on same machine
- Your server should be in running state **infinitely** and should not terminate after serving one client only. The clients will keep on coming one by one and server will keep on serving them unless terminated intentionally.
- **<u>Sample Test Bench</u>**
  - Client1 sends : "Hello I am client and My id is 1"
  - Client2 sends: "Hello I am client and My id is 2"
  - Server response on client1: "Hello I am server. Your received id is 1"
  - Server response on client2: "Hello I am server. Your received id is 2"

In-lab Statement 2: [15]

- Write TCP client and server program such that client will send one string to a server and server will display the string with all the words containing one or more vowels in an inverted fashion e.g., computer must be inverted as 'retupmoc'.

- The server will then send the resulting string to client and client as a result will invert all the words containing no vowels and display it on the terminal e.g., dry must be inverted as ‘ryd’.
- Your server should be in running state **infinitely** and should not terminate after serving one client only. The clients will keep on coming one by one and server will keep on serving them unless terminated intentionally.
- Sample Test Bench
  - Client sends to server: “the birds fly in dry sky at night”
  - Server displays the string and returns to client: “eht sdirb fly ni dry sky ta thgin”
  - Client displays the string: “eht sdirb ylf ni yrd yks ta thgin”

Useful Functions for TCP Client-Server Communication

1. **Socket Creation**:
   - **Purpose**: To create a socket for communication.
   - **Description**: This function initializes a socket using `socket.socket()`, specifying the address family (AF_INET for IPv4) and the socket type (SOCK_STREAM for TCP).
2. **Bind Socket**:
   - **Purpose**: To bind the server socket to a specific address and port.
   - **Description**: This function uses `socket.bind()` to associate the socket with a local address, allowing the server to listen for incoming connections.
3. **Listen for Connections**:
   - **Purpose**: To put the server socket in listening mode.
   - **Description**: By calling `socket.listen()`, the server can accept incoming client connections. You can specify a backlog to limit the number of queued connections.
4. **Accept Connection**:
   - **Purpose**: To accept a connection from a client.

- **Description**: The server uses `socket.accept()` to accept an incoming connection. This returns a new socket object for communication with the client and the address of the client.

5. **Send Data**:

- **Purpose**: To send data from the server to the client (or vice versa).
- **Description**: The `socket.send()` function transmits data over the connected socket. Ensure that the data is in bytes format.

6. **Receive Data**:

- **Purpose**: To receive data from the client.
- **Description**: The `socket.recv()` function reads data from the socket. You can specify the maximum amount of data to receive in bytes.

7. **Close Socket**:

- **Purpose**: To close the socket and free up resources.
- **Description**: The `socket.close()` function terminates the socket connection, ensuring that resources are properly released.

8. **Error Handling**:

- **Purpose**: To manage exceptions that may occur during socket operations.
- **Description**: Use try-except blocks to handle common socket errors, such as `socket.error`, to prevent crashes and handle failures gracefully.

9. **Data Serialization**:

- **Purpose**: To convert data into a format suitable for transmission.
- **Description**: Use modules like `pickle` or `json` to serialize complex data structures into byte streams that can be sent over the network.

10. **Logging**:

- **Purpose**: To log connection details and errors.
- **Description**: Implement logging using the `logging` module to keep track of events, errors, and status updates during the server-client communication process.