



Echo Server

Objective

- In this lab, you are required to implement both **TCP and UDP echo servers** using Python socket programming.
- The servers must process client messages according to the rules described below.

Part 1: Echo Server Implementation

- You are required to implement:
 1. A TCP Echo Server (Multi-threaded)
 2. A UDP Echo Server

General Server Behavior (Applies to Both TCP and UDP)

- The server will receive one line from the client at a time.
 - The **first character** of the line will be one of:
 - 'A'
 - 'C'
 - 'D'
 - The rest of the line will be treated as a **single string of letters**.
- After receiving the message, the server must process it as follows:

Message Processing Rules

1. If the first character is 'A' → The server sorts the characters of the string in descending order and sends the ordered string back to the client.
2. If the first character is 'C' → The server sorts the characters of the string in ascending order and sends the ordered string back to the client.
3. If the first character is 'D' → The server converts all letters in the string to uppercase and sends it back to the client.
4. If the first character is anything else → The server returns the exact same message, it received.



Part 2: TCP Server Requirements

- Your TCP implementation must:
 - Use **multi-threading**.
 - Handle multiple clients concurrently.
 - Create a separate thread for each connected client.
 - Continue serving other clients even if one client disconnects.

Part 3: UDP Server Requirements

- Your UDP implementation must:
 - Use connectionless communication.
 - Handle client requests correctly.

Part 4: Wireshark Requirement

- You must:
 - Capture network traffic using **Wireshark**.
 - Provide screenshots showing:
 - TCP communication between client and server.
 - UDP communication between client and server.
 - Clear visibility of request and response packets.

Deliverables

- You must submit:
 1. Source code for:
 - TCP multi-threaded server
 - UDP server
 - Client program(s)
 2. Screenshots of:
 - Successful TCP communication
 - Successful UDP communication
 - Wireshark capture showing packet exchange
 3. Clean and well-documented code



Supplementary Material

1. Socket Programming

- Python Sockets:
<https://docs.python.org/3/howto/sockets.html>
- Python socketserver library:
<https://docs.python.org/3/library/socketserver.html>

2. Wireshark Tutorial

- <https://youtu.be/L3bCS9siEkI>

Policy

- You are required to answer all parts of this lab.
- This is an individual lab.
- No late submissions are allowed.
- Prepare yourself to discuss your work during your lab time.
- If any cheating or copying is detected, you will get zero without discussion.
- Your submission must be clean, readable, and supported with screenshots.