Aim: To implement a **BMP File Transfer System using TCP Socket Programming** where:

1. The **client** requests a .bmp image from the server.
2. The **server** reads the BMP file and sends it in binary form.
3. The client saves the received file as received.bmp.
4. The **SDL BMP Viewer** program displays the BMP image on the screen.

THEORY

1. TCP Socket Programming

TCP (Transmission Control Protocol) provides **reliable, connection-oriented** communication.
Socket programming uses system calls such as:

- socket()
- bind()
- listen()
- accept()
- connect()
- read() / write()
- close()

2. Client–Server Model

- **Server** waits for client request.
- **Client** connects using IP + Port.
- Client sends filename → Server sends file size → Server sends file contents.

3. Binary File Transfer

Unlike text files, BMP files must be transferred **byte-by-byte** without modification.
The server:

- Opens file in **rb** mode → fread() → sends raw bytes.

The client:

- Receives bytes → stores using fwrite() → reconstructs the exact BMP.

4. SDL (Simple DirectMedia Layer)

Used to display the BMP image:

- SDL_Init()
- SDL_LoadBMP()
- SDL_CreateWindow()
- SDL_RenderCopy()

The viewer program loads a BMP and displays it in a window.

PROCEDURE

A. Server Side (send BMP file)

From the uploaded file —

1. Create a TCP socket using socket().
2. Bind it to port **8080**.
3. Put socket into listening mode using listen().
4. Accept client connection using accept().
5. Receive filename length and filename.
6. Open file in rb mode.

7. Send file size to client.
8. Send file data in chunks of 4096 bytes.
9. Close file and socket.

B. Client Side (receive BMP file)
From the uploaded file —
1. Create socket and connect to server at 127.0.0.1:8080.
2. Ask user to input filename.
3. Send filename length → Send filename.
4. Receive file size.
5. Receive file data in loop until all bytes are received.
6. Save it as **received.bmp**.
7. Close connection.

C. BMP Viewer Program Using SDL
From the uploaded file —
1. Initialize SDL using SDL_Init().
2. Create 800×600 window and renderer.
3. Load BMP from path provided by user argument.
4. Convert it into a texture.
5. Render continuously until user closes the window.
6. Free resources and exit.

D. Compilation Commands
Server
gcc server.c -o server
Client
gcc client.c -o client
SDL Viewer (Linux)
gcc bmp_viewer_sdl.c -o viewer `sdl2-config --cflags --libs`

E. Execution
1️⃣ Start server
./server
2️⃣ Run client (request file)
./client
example.bmp
3️⃣ Run viewer to display received file
./viewer received.bmp

RESULT
- The client successfully connected to the server.
- The client sent the requested filename.
- The server located the BMP file, calculated its size, and sent it in binary chunks.

- The client saved the file perfectly as **received.bmp** without corruption.
- The SDL viewer successfully displayed the transferred BMP image on the screen.

This confirms correct implementation of **binary file transfer and BMP rendering**.

CONCLUSION

The experiment was completed successfully.
We achieved:

✓ TCP server–client communication

✓ File request and binary file transfer

✓ BMP file reconstruction at client side

✓ Real-time display of BMP using SDL

This practical demonstrated:

- How binary files are transferred reliably using TCP
- How to handle filenames, sizes, buffers, and chunks
- How SDL can be used to render images