**Assignment 4 :**

The provided code implements a simplified version of the Berkeley algorithm for time synchronization. The code combines the server and client functionality into a single program for simplicity. Let's go through the code step by step:

1. `BerkeleyAlgorithm` class:

- This class represents the main class of the program.

- It creates a `ServerSocket` on port 2000 to listen for incoming connections from clients.

- Inside an infinite loop, it accepts client connections and spawns a new thread (`ClientHandler`) to handle each client.

2. `ClientHandler` class:

- This class implements the logic to handle a client request.

- Upon receiving a client connection, it reads the request from the client, which is the client's local time.

- It calculates the current time on the server and sends it back to the client.

- It also calculates the clock difference between the client and the server and establishes a connection to the server itself.

- It sends the clock difference to the server and receives the average clock difference from the server.

- Finally, it adjusts the client's clock by adding the average clock difference and prints the adjusted time.

3. `Client` class:

- This class represents the client part of the program.

- It establishes a connection to the server on localhost and port 2000.

- It retrieves the current time on the client.

- It sends the current time to the server as a request.

- It receives the current time from the server.

- It calculates the clock difference between the server and the client.

- It sends the clock difference to the server.

- It receives the average clock difference from the server.

- It adjusts the client's clock by adding the average clock difference and prints the adjusted time.

- Finally, it closes the connection to the server.

Overall, the code simulates the Berkeley algorithm by exchanging time information between the server and client. The server acts as a central time synchronization point, where it receives time requests from clients, calculates the clock difference, and sends the average clock difference back to the clients. The clients adjust their clocks based on the received clock difference, aiming to achieve time synchronization with the server.