Name – Pranav Tiwari
Batch – Cse
Roll no. - 201127

Network Laboratory

Programming Assignment-7

Ques:
Implementing TCP timeserver in C and Java

C Implementation

Source Code:

time-server.c

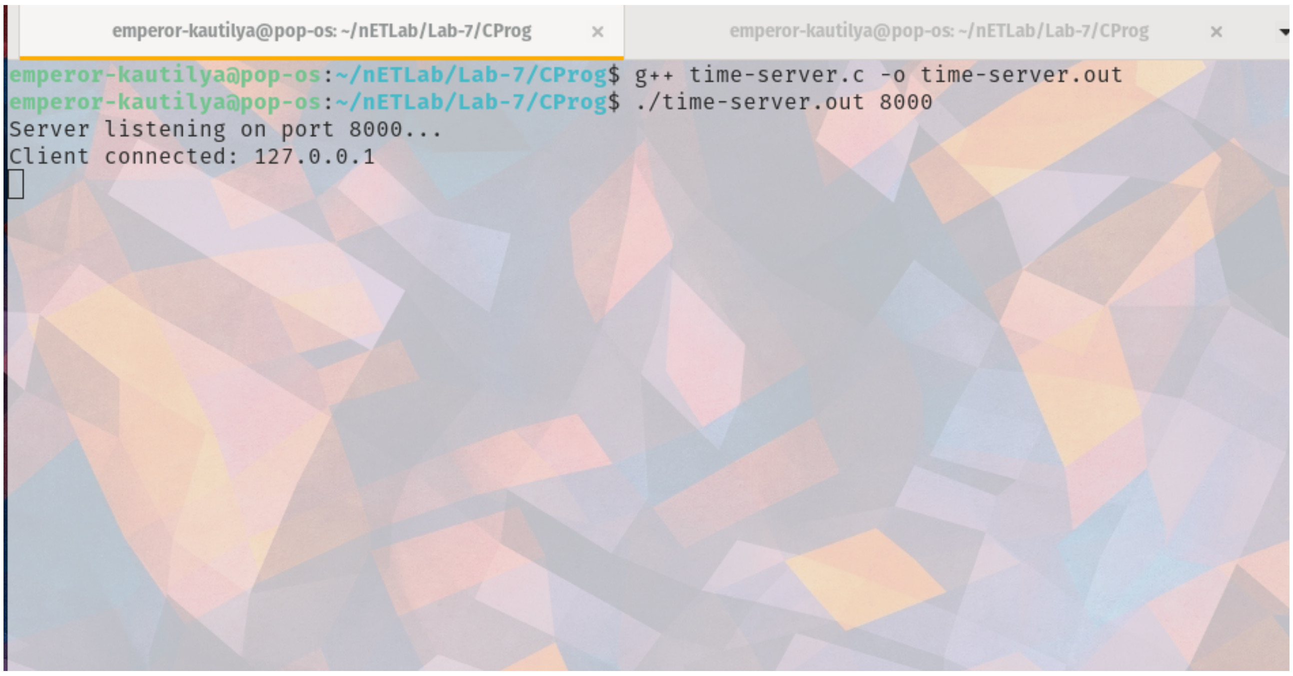
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>

#define MAX_PENDING 5
#define BUF_SIZE 32

int main(int argc, char *argv[]) {
    int server_socket, client_socket;
    struct sockaddr_in server_addr, client_addr;
    char buffer[BUF_SIZE];
    time_t current_time;

    // create server socket
    if ((server_socket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
        perror("socket creation failed");
        return EXIT_FAILURE;
    }

    // set server address
    memset(&server_addr, 0, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    server_addr.sin_port = htons(atoi(argv[1]));

    // bind server socket to address
    if (bind(server_socket, (struct sockaddr *)&server_addr, sizeof(server_addr)) < 0) {
        perror("bind failed");
        return EXIT_FAILURE;
    }

    // listen for incoming connections
    if (listen(server_socket, MAX_PENDING) < 0) {
        perror("listen failed");
        return EXIT_FAILURE;
    }

    printf("Server listening on port %d...\n", atoi(argv[1]));

    while (1) {
        socklen_t client_len = sizeof(client_addr);

        // accept incoming connection
        if ((client_socket = accept(server_socket, (struct sockaddr *)&client_addr, &client_len)) < 0) {
            perror("accept failed");
            return EXIT_FAILURE;
        }

        printf("Client connected: %s\n", inet_ntoa(client_addr.sin_addr));

        // get current time and format as string
        current_time = time(NULL);
        snprintf(buffer, BUF_SIZE, "%.24s\n", ctime(&current_time));

        // send time to client
        if (send(client_socket, buffer, strlen(buffer), 0) != strlen(buffer)) {
            perror("send failed");
            return EXIT_FAILURE;
        }

        // close client socket
        close(client_socket);
    }

    // close server socket
    close(server_socket);

    return EXIT_SUCCESS;
}
```

client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>

#define BUF_SIZE 32

int main(int argc, char *argv[]) {
    int client_socket;
    struct sockaddr_in server_addr;
    char buffer[BUF_SIZE];

    // create client socket
    if ((client_socket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
        perror("socket creation failed");
        return EXIT_FAILURE;
    }

    // set server address
    memset(&server_addr, 0, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = inet_addr(argv[1]);
    server_addr.sin_port = htons(atoi(argv[2]));

    // connect to server
    if (connect(client_socket, (struct sockaddr *)&server_addr, sizeof(server_addr)) < 0) {
        perror("connection failed");
        return EXIT_FAILURE;
    }

    // receive time from server
    memset(buffer, 0, BUF_SIZE);
    if (recv(client_socket, buffer, BUF_SIZE, 0) < 0) {
        perror("receive failed");
        return EXIT_FAILURE;
    }

    printf("Current time: %s", buffer);

    // close client socket
    close(client_socket);

    return EXIT_SUCCESS;
}
```

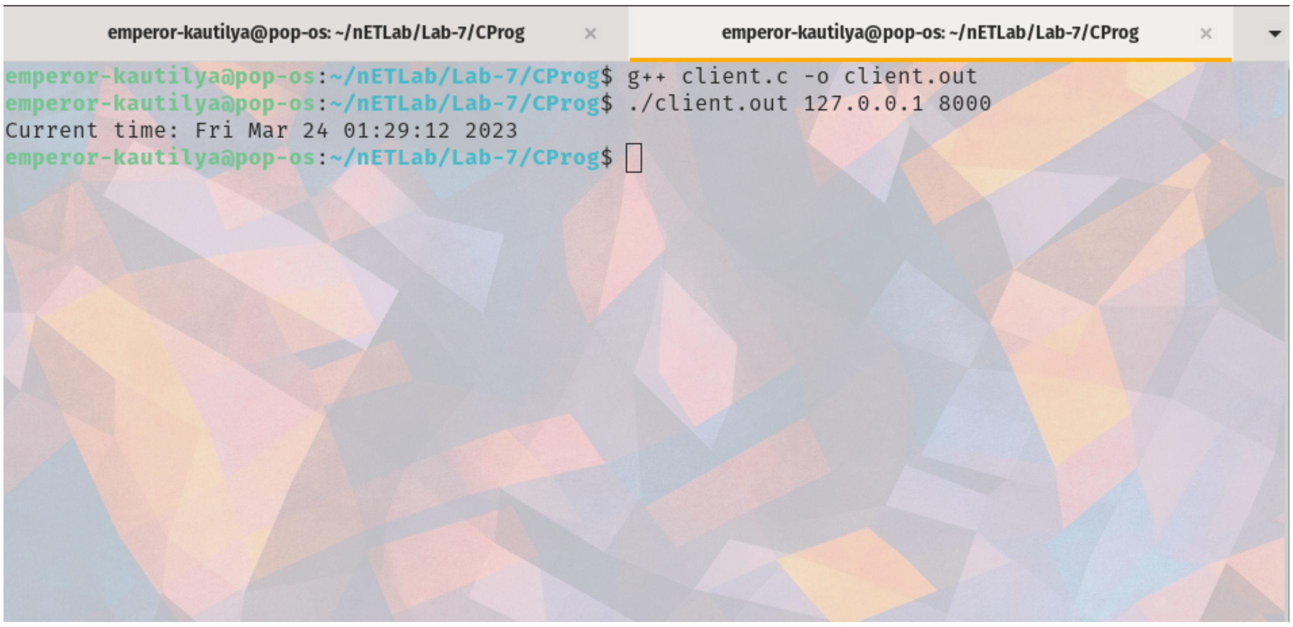
Output:

Server program:


emperor-kautilya@pop-os: ~/nETLab/Lab-7/CProg
emperor-kautilya@pop-os: ~/nETLab/Lab-7/CProg

```
emperor-kautilya@pop-os:~/nETLab/Lab-7/CProg$ g++ time-server.c -o time-server.out
emperor-kautilya@pop-os:~/nETLab/Lab-7/CProg$ ./time-server.out 8000
Server listening on port 8000...
Client connected: 127.0.0.1
```

Client program:


emperor-kautilya@pop-os: ~/nETLab/Lab-7/CProg
emperor-kautilya@pop-os: ~/nETLab/Lab-7/CProg

```
emperor-kautilya@pop-os:~/nETLab/Lab-7/CProg$ g++ client.c -o client.out
emperor-kautilya@pop-os:~/nETLab/Lab-7/CProg$ ./client.out 127.0.0.1 8000
Current time: Fri Mar 24 01:29:12 2023
emperor-kautilya@pop-os:~/nETLab/Lab-7/CProg$
```

Java Implementation

Source Code:

JavaTimeServer.java

```
import java.io.IOException;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class JavaTimeServer {

    public static void main(String[] args) {
        if (args.length != 1) {
            System.err.println("Usage: java TimeServer <port number>");
            System.exit(1);
        }

        int port = Integer.parseInt(args[0]);

        try (ServerSocket serverSocket = new ServerSocket(port)) {
            System.out.println("Time Server listening on port " + port + "...");

            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("Client connected: " +
clientSocket.getInetAddress().getHostAddress());

                OutputStream out = clientSocket.getOutputStream();
                DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss");
                String timeString = LocalDateTime.now().format(formatter);
                out.write(timeString.getBytes());
                out.flush();

                clientSocket.close();
            }
        } catch (IOException e) {
            System.err.println("Exception caught");
            System.err.println(e.getMessage());
        }
    }
}
```

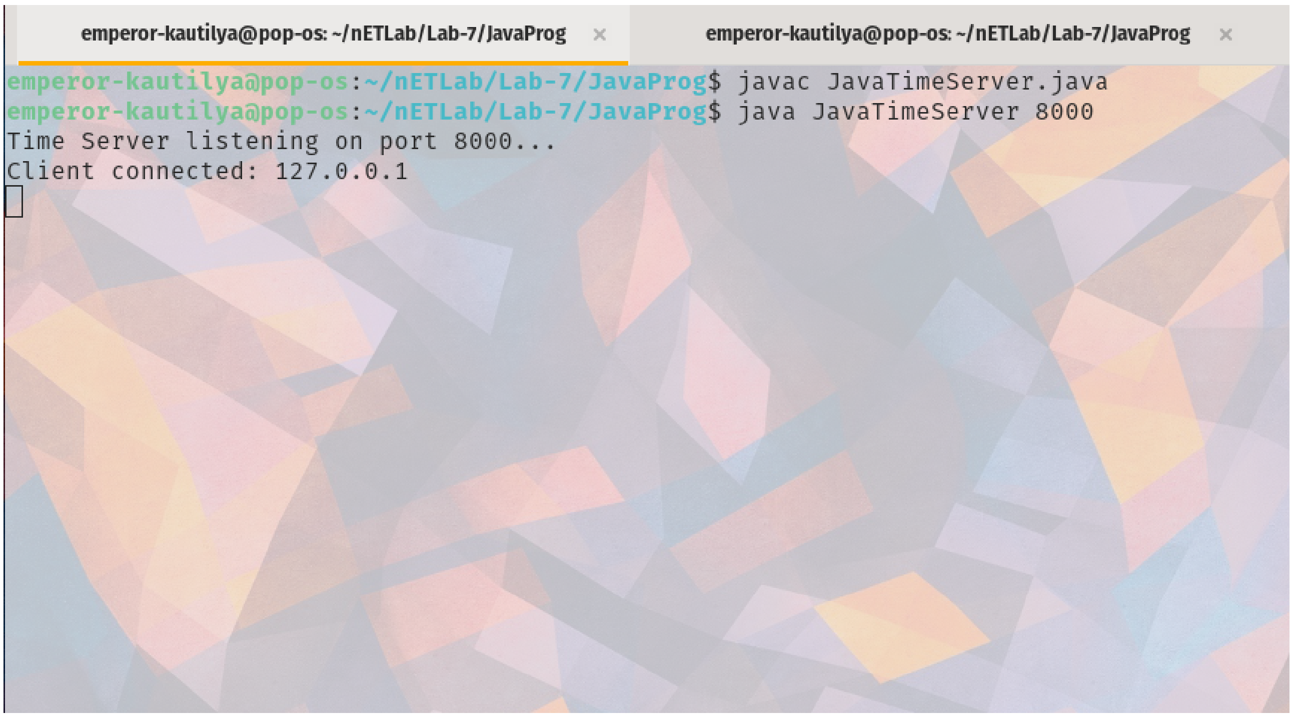
JavaTimeClient.java

```
import java.io.*;
import java.net.*;
```

```
public class JavaTimeClient {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.err.println("Usage: java TimeClient <hostname> <port>");
            System.exit(1);
        }

        String hostname = args[0];
        int port = Integer.parseInt(args[1]);

        try (
            Socket socket = new Socket(hostname, port);
            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
        ) {
            String time = in.readLine();
            System.out.println("Current time: " + time);
        } catch (IOException e) {
            System.err.println("Error communicating with server: " + e.getMessage());
            System.exit(2);
        }
    }
}
```

Output:

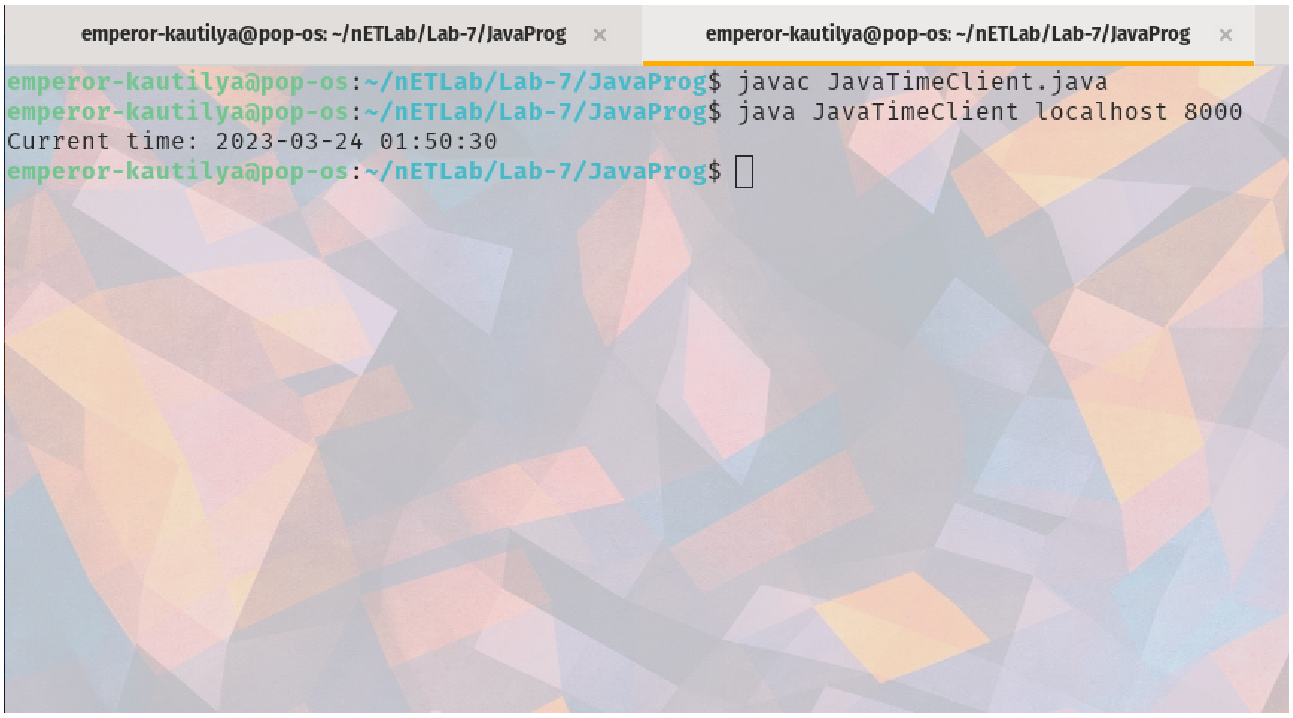
Server program**:**



emperor-kautilya@pop-os: ~/nETLab/Lab-7/JavaProg
emperor-kautilya@pop-os: ~/nETLab/Lab-7/JavaProg

```
emperor-kautilya@pop-os:~/nETLab/Lab-7/JavaProg$ javac JavaTimeServer.java
emperor-kautilya@pop-os:~/nETLab/Lab-7/JavaProg$ java JavaTimeServer 8000
Time Server listening on port 8000...
Client connected: 127.0.0.1
```

Client program**:**



emperor-kautilya@pop-os: ~/nETLab/Lab-7/JavaProg
emperor-kautilya@pop-os: ~/nETLab/Lab-7/JavaProg

```
emperor-kautilya@pop-os:~/nETLab/Lab-7/JavaProg$ javac JavaTimeClient.java
emperor-kautilya@pop-os:~/nETLab/Lab-7/JavaProg$ java JavaTimeClient localhost 8000
Current time: 2023-03-24 01:50:30
emperor-kautilya@pop-os:~/nETLab/Lab-7/JavaProg$
```