



**Motilal Nehru National Institute of Technology Allahabad**  
**Prayagraj-211004 [India]**

**Department of Computer Science & Engineering**

**Programme Name: B.Tech**

**Semester: VII**

**Branch: Computer Science & Engg.**

**Course Code: CS17201**

**Course Name: Distributed Systems (Lab)**

**Lab Assignment 7**

Name-Sugandh Mishra

Reg-20204211

Sec- CSE C

Lab #	Name of Experiments
7	<p>(i) Implement RPC mechanism for a file transfer across a network in 'C'.</p> <p>Server.c----</p> <pre>#include &lt;stdio.h&gt; #include &lt;stdlib.h&gt; #include &lt;string.h&gt; #include &lt;unistd.h&gt; #include &lt;netinet/in.h&gt; #include &lt;arpa/inet.h&gt;  #define PORT 12345  void transfer_file(int client_socket) {     FILE *file = fopen("example.txt", "rb");     if (file == NULL) {         perror("File open error");         exit(1);     }      char buffer[1024];     size_t bytesRead;      while ((bytesRead = fread(buffer, 1, sizeof(buffer), file)) &gt; 0) {         send(client_socket, buffer, bytesRead, 0);     }      fclose(file); }  int main() {     int server_socket, client_socket;     struct sockaddr_in server_addr, client_addr;     socklen_t addr_size;      server_socket = socket(AF_INET, SOCK_STREAM, 0);</pre>

```

    if (server_socket < 0) {
        perror("Socket creation error");
        exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(PORT);
    server_addr.sin_addr.s_addr = INADDR_ANY;

    if (bind(server_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0)
        perror("Binding error");
        exit(1);
    }

    if (listen(server_socket, 10) == 0) {
        printf("Listening...\n");
    } else {
        perror("Listening error");
        exit(1);
    }

    addr_size = sizeof(client_addr);
    client_socket = accept(server_socket, (struct sockaddr*)&client_addr, &addr_size);

    transfer_file(client_socket);
    printf("File sent successfully.\n");

    close(client_socket);
    close(server_socket);
    return 0;
}

```

client.c----

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define SERVER_IP "127.0.0.1" // Change this to the server's IP address
#define PORT 12345

void receive_file(int server_socket) {
    FILE *file = fopen("received_file.txt", "wb");
    if (file == NULL) {
        perror("File create error");
        exit(1);
    }

    char buffer[1024];
    int bytesRead;

    while ((bytesRead = recv(server_socket, buffer, sizeof(buffer), 0)) > 0) {
        fwrite(buffer, 1, bytesRead, file);
    }
}

```

```

    }

    fclose(file);
}

int main() {
    int client_socket;
    struct sockaddr_in server_addr;

    client_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (client_socket < 0) {
        perror("Socket creation error");
        exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(PORT);
    server_addr.sin_addr.s_addr = inet_addr(SERVER_IP);

    if (connect(client_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) <
0) {
        perror("Connection error");
        exit(1);
    }

    receive_file(client_socket);
    printf("File received successfully.\n");

    close(client_socket);
    return 0;
}

```

The screenshot shows a macOS IDE with the following components:

- Code Editor:** Displays `server.c` and `client.c`. The `server.c` file contains the code for the server, and the `client.c` file contains the code for the client.
- Terminal:** Shows the execution of the programs. The client program (`client.c`) is run first, and then the server program (`server.c`) is run. The server receives a file from the client.
- Output:** Shows the output of the programs. The client program outputs "hello i am sugandh mishra 20204211". The server program outputs "File received successfully."

```

Avinashs-MacBook-Air:as7 anurag$ cd q1
Avinashs-MacBook-Air:q1 anurag$ ls
client.c      example.txt   server.c
Avinashs-MacBook-Air:q1 anurag$ gcc client.c -o cli
ent
Avinashs-MacBook-Air:q1 anurag$ ./client
Connection error: Connection refused
Avinashs-MacBook-Air:q1 anurag$ ./client
File received successfully.
Avinashs-MacBook-Air:q1 anurag$

Avinashs-MacBook-Air:q1 anurag$ gcc server.c -o serv
er
Avinashs-MacBook-Air:q1 anurag$ ./server
Listening...
File sent successfully.
Avinashs-MacBook-Air:q1 anurag$

```

ii) Implement 'Java RMI' mechanism for accessing methods of remote systems.

Server.c---

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>

int add(int a, int b) {
    return a + b;
}

int main() {
    int server_socket, client_socket;
    struct sockaddr_in server_addr, client_addr;
    socklen_t addr_size;

    server_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (server_socket < 0) {
        perror("Socket creation error");
        exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(12345);
    server_addr.sin_addr.s_addr = INADDR_ANY;

    if (bind(server_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0)
    {
        perror("Binding error");
        exit(1);
    }

    if (listen(server_socket, 10) == 0) {
        printf("Listening...\n");
    } else {
        perror("Listening error");
        exit(1);
    }

    addr_size = sizeof(client_addr);
    client_socket = accept(server_socket, (struct sockaddr*)&client_addr,
&addr_size);

    int a, b, result;
    recv(client_socket, &a, sizeof(a), 0);
    recv(client_socket, &b, sizeof(b), 0);

    result = add(a, b);
    send(client_socket, &result, sizeof(result), 0);
}
```

```

        close(client_socket);
        close(server_socket);
        return 0;
    }

```

client.c-----

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>

int main() {
    int client_socket;
    struct sockaddr_in server_addr;

    client_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (client_socket < 0) {
        perror("Socket creation error");
        exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(12345);
    server_addr.sin_addr.s_addr = INADDR_ANY;

    if (connect(client_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) <
0) {
        perror("Connection error");
        exit(1);
    }

    int a = 5, b = 3, result;

    send(client_socket, &a, sizeof(a), 0);
    send(client_socket, &b, sizeof(b), 0);

    recv(client_socket, &result, sizeof(result), 0);

    printf("Result: %d\n", result);

    close(client_socket);
    return 0;
}

```

CodeFileEditSelectionViewGoRunTerminalWindowHelp

server.c — dis sys lab

client.c

as7 > q2 > C client.c

```
20
21     if (connect(client_socket, (struct
22         perror("Connection error");
23         exit(1);
24     }
25
26     int a = 5, b = 3, result;
27
28     send(client_socket, &a, sizeof(a),
29     send(client_socket, &b, sizeof(b),
30
31     recv(client_socket, &result, sizeof
32
33     printf("Result: %d\n", result);
```

server.c

as7 > q2 > C server.c

```
37
38     addr_size = sizeof(client_addr);
39     client_socket = accept(server_socke
40
41     int a, b, result;
42     recv(client_socket, &a, sizeof(a),
43     recv(client_socket, &b, sizeof(b),
44
45     result = add(a, b);
46     send(client_socket, &result, sizeof
47
48     close(client_socket);
49     close(server_socket);
```

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

1: bash, bash

vinashs-MacBook-Air:as7 anurag\$ cd q2

Avinashs-MacBook-Air:q2 anurag\$ gcc client.c -o clie

nt

Avinashs-MacBook-Air:q2 anurag\$ ./client

Result: 8

Avinashs-MacBook-Air:q2 anurag\$

Avinashs-MacBook-Air:as7 anurag\$ cd q2

Avinashs-MacBook-Air:q2 anurag\$ gcc server.c -o serv

er

Avinashs-MacBook-Air:q2 anurag\$ ./server

Listening...

Avinashs-MacBook-Air:q2 anurag\$

Ln 51, Col 2 Spaces: 4 UTF-8 LF C Go Live