**Ex:4 b - Computer Network Lab**

**Name:** Athithraja R

**Reg.no:** 2022503702

**4.b UDP**

**Echo\_Server2.c**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 22000

#define BUFFER\_SIZE 1024

int main() {

int server\_fd;

struct sockaddr\_in server\_addr, client\_addr;

socklen\_t addr\_len = sizeof(client\_addr);

char buffer[BUFFER\_SIZE];

// Create socket

server\_fd = socket(AF\_INET, SOCK\_DGRAM, 0);

if (server\_fd < 0) {

perror("Socket creation failed");

exit(EXIT\_FAILURE);

}

// Configure server address

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_addr.s\_addr = INADDR\_ANY;

server\_addr.sin\_port = htons(PORT);

// Bind the socket

if (bind(server\_fd, (struct sockaddr\*)&server\_addr, sizeof(server\_addr)) < 0) {

perror("Bind failed");

close(server\_fd);

exit(EXIT\_FAILURE);

}

printf("UDP Server is listening on port %d\n", PORT);

while (1) {

memset(buffer, 0, BUFFER\_SIZE);

ssize\_t bytes\_received = recvfrom(server\_fd, buffer, BUFFER\_SIZE, 0,

(struct sockaddr\*)&client\_addr, &addr\_len);

if (bytes\_received < 0) {

perror("Receive failed");

continue;

}

printf("Received: %s", buffer);

sendto(server\_fd, buffer, bytes\_received, 0,

(struct sockaddr\*)&client\_addr, addr\_len);

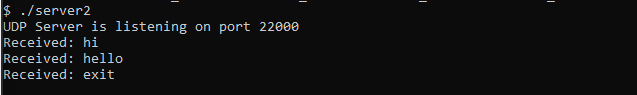
}

close(server\_fd);

return 0;

}

Output:



**Echo\_client2.c**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 22000

#define BUFFER\_SIZE 1024

int main() {

int client\_fd;

struct sockaddr\_in server\_addr;

char buffer[BUFFER\_SIZE];

// Create socket

client\_fd = socket(AF\_INET, SOCK\_DGRAM, 0);

if (client\_fd < 0) {

perror("Socket creation failed");

exit(EXIT\_FAILURE);

}

// Configure server address

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(PORT);

server\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); // Change to server IP if needed

while (1) {

printf("Enter message: ");

fgets(buffer, BUFFER\_SIZE, stdin);

// Send message to server

sendto(client\_fd, buffer, strlen(buffer), 0,

(struct sockaddr\*)&server\_addr, sizeof(server\_addr));

// Receive echoed message from server

ssize\_t bytes\_received = recvfrom(client\_fd, buffer, BUFFER\_SIZE, 0, NULL, NULL);

if (bytes\_received < 0) {

perror("Receive failed");

continue;

}

buffer[bytes\_received] = '\0'; // Null-terminate the received string

printf("Echoed from server: %s", buffer);

}

close(client\_fd);

return 0;

}

Output:

