PROGRAM:-

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 5000

#define BUFFER\_SIZE 1024

int main() {

int server\_fd, new\_socket;

struct sockaddr\_in address;

int addrlen = sizeof(address);

char buffer[BUFFER\_SIZE];

// Create TCP socket

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

perror("Socket failed");

exit(EXIT\_FAILURE);

}

// Define server address

address.sin\_family = AF\_INET;

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

address.sin\_port = htons(PORT);

// Bind socket to the defined address

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

perror("Bind failed");

exit(EXIT\_FAILURE);

}

// Start listening for client connections

if (listen(server\_fd, 5) < 0) {

perror("Listen failed");

exit(EXIT\_FAILURE);

}

printf("Echo Server is running on port %d...\n", PORT);

while (1) {

// Accept a client connection

if ((new\_socket = accept(server\_fd, (struct sockaddr\*)&address, (socklen\_t\*)&addrlen)) < 0) {

perror("Accept failed");

exit(EXIT\_FAILURE);

}

printf("Client connected...\n");

// Receive data from client

int bytes\_received = recv(new\_socket, buffer, BUFFER\_SIZE, 0);

if (bytes\_received < 0) {

perror("Receive failed");

close(new\_socket);

continue;

}

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

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

// Send the same message back (echo)

send(new\_socket, buffer, bytes\_received, 0);

printf("Echoed: %s\n", buffer);

close(new\_socket); // Close client connection

}

return 0;

}

OUTPUT:-

Echo Server is running on port 5000...

Client connected...

Received: Hello, Server!

Echoed: Hello, Server!