/*tcp echo client server*/

//tcp echo server

```
#include <iostream>
#include <string>
#include <cstring>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>

#define PORT 8080
#define BUFFER_SIZE 1024

int main() {
    int server_fd, new_socket, valread;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char buffer[BUFFER_SIZE] = {0};
    const char *echo_message = "Echo from server: ";

    // Creating socket file descriptor
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0) {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    // Forcefully attaching socket to the port
    if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT, &opt, sizeof(opt))) {
        perror("setsockopt");
```

```
        exit(EXIT_FAILURE);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(PORT);

    // Forcefully attaching socket to the port
    if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) < 0) {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }
    if (listen(server_fd, 3) < 0) {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if ((new_socket = accept(server_fd, (struct sockaddr *)&address, (socklen_t *)&addrlen)) < 0) {
        perror("accept");
        exit(EXIT_FAILURE);
    }
    while (1) {
        valread = read(new_socket, buffer, BUFFER_SIZE);
        printf("Received: %s\n", buffer);
        send(new_socket, echo_message, strlen(echo_message), 0);
        send(new_socket, buffer, strlen(buffer), 0);
        printf("Echo sent\n");
    }
    return 0;
}
```

```
rajasree@ubuntu-RajasreeVM:~/Desktop/cn$ g++ tcp_echo_server.cpp
rajasree@ubuntu-RajasreeVM:~/Desktop/cn$ ./tcp_echo_server
bash: ./tcp_echo_server: No such file or directory
rajasree@ubuntu-RajasreeVM:~/Desktop/cn$ ./a.out
Received: Hello from client
Echo sent
Received: Hello from client
rajasree@ubuntu-RajasreeVM:~/Desktop/cn$
```

//tcp echo client

```
#include <iostream>
#include <string>
#include <cstring>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define PORT 8080
#define BUFFER_SIZE 1024

int main() {
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char buffer[BUFFER_SIZE] = {0};
    const char *message = "Hello from client";

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        std::cerr << "Socket creation error" << std::endl;
        return -1;
    }

    serv_addr.sin_family = AF_INET;
```

```
    serv_addr.sin_port = htons(PORT);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
        std::cerr << "Invalid address/ Address not supported" << std::endl;
        return -1;
    }

    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        std::cerr << "Connection Failed" << std::endl;
        return -1;
    }

    send(sock, message, strlen(message), 0);
    std::cout << "Message sent: " << message << std::endl;
    valread = read(sock, buffer, BUFFER_SIZE);
    std::cout << "Echo from server: " << buffer << std::endl;
    return 0;
}
```