

Name: Nirjala Naik

Div: B

Roll no: T512022

```
GNU nano 6.2
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#define PORT 8080
#define BUFFER_SIZE 1024

int main() {
    int sock;
    struct sockaddr_in server_addr, client_addr;
    char buffer[BUFFER_SIZE];
    socklen_t addr_len = sizeof(client_addr);
    // Create socket (IPv4, TCP)
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock < 0) {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    // Define server address
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    server_addr.sin_port = htons(PORT);
    // Bind socket to the specified port and IP
    if (bind(sock, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
        perror("bind failed");
        close(sock);
        exit(EXIT_FAILURE);
    }

    printf("UDP Server listening on port %d...\n", PORT);
    // Receive message from client
    ssize_t recv_len = recvfrom(sock, buffer, BUFFER_SIZE - 1, 0,
                                (struct sockaddr*)&client_addr, &addr_len);

    if (recv_len < 0) {
        perror("recvfrom failed");
        close(sock);
        exit(EXIT_FAILURE);
    }

    buffer[recv_len] = '\0'; // Null-terminate the received data
    printf("Client: %s\n", buffer);
    // Send response to client
    char *message = "Hello from UDP Server";
    ssize_t sent_len = sendto(sock, message, strlen(message), 0,
                              (struct sockaddr*)&client_addr, &addr_len);

    if (sent_len < 0) {
        perror("sendto failed");
        close(sock);
        exit(EXIT_FAILURE);
    }

    // Close socket
    close(sock);
    return 0;
}
```

```
GNU nano 6.2
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#define PORT 8080
#define BUFFER_SIZE 1024

int main() {
    int sock;
    struct sockaddr_in server_addr;
    char buffer[BUFFER_SIZE];
    socklen_t addr_len = sizeof(server_addr);
    // Create socket (IPv4, UDP)
    sock = socket(AF_INET, SOCK_DGRAM, 0);
    if (sock < 0) {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    // Define server address
    memset(&server_addr, 0, sizeof(server_addr)); // Zero out structure
    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(PORT);
    server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

    // Send message to server
    char *message = "Hello from UDP client";
    ssize_t sent_len = sendto(sock, message, strlen(message), 0,
                              (struct sockaddr*)&server_addr, sizeof(server_addr));

    if (sent_len < 0) {
        perror("sendto failed");
        close(sock);
        exit(EXIT_FAILURE);
    }

    // Receive response from server
    ssize_t recv_len = recvfrom(sock, buffer, BUFFER_SIZE - 1, 0, NULL, NULL);
    if (recv_len < 0) {
        perror("recvfrom failed");
        close(sock);
        exit(EXIT_FAILURE);
    }

    buffer[recv_len] = '\0'; // Null-terminate the received data
    printf("Server: %s\n", buffer);

    // Close socket
    close(sock);
    return 0;
}
```

```
Activities Terminal Apr 15 23:06
Terminal
\Student@pci:~$ cd T511069
\Student@pci:~/T511069$ gcc udp_server.c
\Student@pci:~/T511069$ gcc udp_client.c
\Student@pci:~/T511069$ gcc udp_server.c -o udp_server
\Student@pci:~/T511069$ gcc udp_client.c -o udp_client
\Student@pci:~/T511069$ ./udp
bash: ./udp: No such file or directory
\Student@pci:~/T511069$ ./udp_server
UDP Server: listening on port 8000...
Client: Hello from UDP client
\Student@pci:~/T511069$
```

```
Activities Terminal Apr 15 23:06
Terminal
\Student@pci:~$ cd T511069
\Student@pci:~/T511069$ ./udp_client
Server: Hello from UDP Server
\Student@pci:~/T511069$
```