give just a basic code for implemention of udo protocol in c

Here's a basic implementation of a UDP client and server in C. This example will illustrate how to send and receive messages over UDP.

UDP Server

c Copy code

```c
#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 sockfd;
    char buffer[BUFFER_SIZE];
    struct sockaddr_in server_addr, client_addr;
    socklen_t addr_len = sizeof(client_addr);

    // Create socket file descriptor
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&server_addr, 0, sizeof(server_addr));
    memset(&client_addr, 0, sizeof(client_addr));

    // Filling server information
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    server_addr.sin_port = htons(PORT);

    // Bind the socket with the server address
    if (bind(sockfd, (const struct sockaddr *)&server_addr, sizeof(server_addr)) <
        perror("Bind failed");
        close(sockfd);
        exit(EXIT_FAILURE);
    }

    while (1) {
        int n = recvfrom(sockfd, (char *)buffer, BUFFER_SIZE, MSG_WAITALL,
                         (struct sockaddr *)&client_addr, &addr_len);
        buffer[n] = '\0';
        printf("Client: %s\n", buffer);

        sendto(sockfd, (const char *)buffer, strlen(buffer), MSG_CONFIRM,
               (const struct sockaddr *)&client_addr, addr_len);
        printf("Message sent.\n");
    }

    close(sockfd);
    return 0;
}
```

UDP Client

c

```c
#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 sockfd;
    char buffer[BUFFER_SIZE];
    char *message = "Hello from client";
    struct sockaddr_in server_addr;

    // Create socket file descriptor
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&server_addr, 0, sizeof(server_addr));

    // Filling server information
    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(PORT);
    server_addr.sin_addr.s_addr = INADDR_ANY;

    int n, len;

    sendto(sockfd, (const char *)message, strlen(message), MSG_CONFIRM,
           (const struct sockaddr *)&server_addr, sizeof(server_addr));
    printf("Message sent.\n");

    n = recvfrom(sockfd, (char *)buffer, BUFFER_SIZE, MSG_WAITALL,
                 (struct sockaddr *)&server_addr, &len);
    buffer[n] = '\0';
    printf("Server: %s\n", buffer);

    close(sockfd);
    return 0;
}
```

Instructions

1. **Compile the server and client programs:**

   sh

   ```sh
   gcc -o udp_server udp_server.c
   gcc -o udp_client udp_client.c
   ```

2. **Run the server:**

   sh

   ```sh
   ./udp_server
   ```

3. **Run the client (in a separate terminal):**

   sh

   ```sh
   ./udp_client
   ```

The client sends a message to the server, which then echoes the message back to the client. This simple example demonstrates the basic UDP communication mechanism.