

CS204 Lab-7 Assignment

Raghava Gatadi(21bcs088)

Server:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
    int sockfd;
    char buffer[MAX_BUFFER_SIZE];
    struct sockaddr_in serverAddr, clientAddr;
    if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0){
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }
    memset(&serverAddr, 0, sizeof(serverAddr));
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_addr.s_addr = INADDR_ANY;
    serverAddr.sin_port = htons(PORT);
    if(bind(sockfd, (struct sockaddr *)&serverAddr, sizeof(serverAddr)) < 0) {
        perror("Binding failed");
        exit(EXIT_FAILURE);
    }
    while(1) {
        memset(buffer, 0, sizeof(buffer));
        socklen_t len = sizeof(clientAddr);
        ssize_t n = recvfrom(sockfd, buffer, MAX_BUFFER_SIZE, 0, (struct sockaddr
*)&clientAddr, &len);
        if(n < 0) {
            perror("Error receiving data");
```

```

        exit(EXIT_FAILURE);
    }
    printf("message from client: %s\n", buffer);
    char response[1024];
    printf("Enter message to respond to client: ");
    fgets(response, 1024, stdin);
    sendto(sockfd, response, strlen(response), 0, (struct sockaddr *)&clientAddr, len);
    }
    close(sockfd);
    return 0;
}

```

Client:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
    int sockfd;
    char buffer[MAX_BUFFER_SIZE];
    struct sockaddr_in serverAddr;
    if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }
    memset(&serverAddr, 0, sizeof(serverAddr));
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(PORT);
    serverAddr.sin_addr.s_addr = INADDR_ANY;
    while(1) {
        printf("Enter a message: ");
        fgets(buffer, MAX_BUFFER_SIZE, stdin);
        sendto(sockfd, (const char *)buffer, strlen(buffer), 0, (const struct sockaddr
*)&serverAddr, sizeof(serverAddr));
        memset(buffer, 0, sizeof(buffer));
        socklen_t len = sizeof(serverAddr);
    }
}

```

```

        ssize_t n = recvfrom(sockfd, buffer, MAX_BUFFER_SIZE, 0, (struct sockaddr
*)&serverAddr, &len);
        if(n < 0) {
            perror("Error receiving data");
            exit(EXIT_FAILURE);
        }
        printf("message from server: %s\n", buffer);
    }
    close(sockfd);
    return 0;
}

```

output:

<pre> iiit@iiit-HP-406-G1-MT:~\$ gcc server.c -o server iiit@iiit-HP-406-G1-MT:~\$./server message from client hey Enter message to respond to client: hi ^C iiit@iiit-HP-406-G1-MT:~\$ gcc server.c -o server iiit@iiit-HP-406-G1-MT:~\$./server message from client: hey Enter message to respond to client: hi message from client: what u doin Enter message to respond to client: ntg bro </pre>	<pre> iiit@iiit-HP-406-G1-MT:~\$ gcc client.c -o client iiit@iiit-HP-406-G1-MT:~\$./client Enter a message: hey message from server hi Enter a message: ^C iiit@iiit-HP-406-G1-MT:~\$ gcc client.c -o client iiit@iiit-HP-406-G1-MT:~\$./client Enter a message: hey message from server: hi Enter a message: what u doin message from server: ntg bro Enter a message: </pre>
--	---