UDP SERVER PROGRAM IN C

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#define MAX_MSG 100

int main(int argc, char *argv[])
{
    int sd, cliLen, newSd, n;
    struct sockaddr_in servAddr, cliAddr;
    char buffer[MAX_MSG];

    if (argc < 3) // name of program, ip address, port number
    {
        printf("Input error\n");
        exit(0);
    }

    servAddr.sin_family = AF_INET;
    servAddr.sin_addr.s_addr = inet_addr(argv[1]); // ined_addr -> converts ip address
into 32 bit binary number
    servAddr.sin_port = htons(atoi(argv[2])); // atoi -> string to integer. // ->
htones -> works only on int, (host to network short) converts into 16-bit binary
number
    memset(&(servAddr.sin_zero), '\0', 8); // zero the rest of the struct.sin_zero

    sd = socket(AF_INET, SOCK_DGRAM, 0);

    if (sd > 0)
        printf("Successfully created stream socket.\n");

    /*bind local port number*/

    bind(sd, (struct sockaddr *)&servAddr, sizeof(servAddr));
    printf("Successfully bound to local address\n");

    printf("Waiting for data on port UDP %u\n", atoi(argv[2]));

    /*send data to server*/
    while (1)
    {
        memset(buffer, 0, MAX_MSG);

        /*recieve data from client*/
        cliLen = sizeof(cliAddr);
        n = recvfrom(sd, buffer, MAX_MSG, 0, (struct sockaddr*)&cliAddr, &cliLen);

        printf("from %s: UDP port %u : %s \n", inet_ntoa(cliAddr.sin_addr),
ntohs(cliAddr.sin_port), buffer);
    }


    printf("closing connection with the server\n");
    close(sd);

    return 0;
}
```

UDP CLIENT PROGRAM IN C

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#define MAX_MSG 100

int main(int argc, char *argv[])
{
      int sd, cliLen, newSd, n;
      struct sockaddr_in servAddr, cliAddr;
      char buffer[MAX_MSG];

      if (argc < 3) // name of program, ip address, port number
      {
             printf("Input error\n");
             exit(0);
      }

      servAddr.sin_family = AF_INET;
      servAddr.sin_addr.s_addr = inet_addr(argv[1]); // ined_addr -> converts ip
address into 32 bit binary number
      servAddr.sin_port = htons(atoi(argv[2])); // atoi -> string to integer. // ->
htones -> works only on int, (host to network short) converts into 16-bit binary
number
      memset(&(servAddr.sin_zero), '\0', 8); // zero the rest of the struct.sin_zero

    sd = socket(AF_INET,SOCK_DGRAM,0);
    printf("Successfully created datagram socket\n");

    do{
        printf("Enter data to send: ");
        scanf("%s",buffer);
        sendto(sd, buffer, strlen(buffer)+1,0,(struct sockaddr *)&servAddr,
sizeof(servAddr));

    }while(strcmp(buffer,"quit"));
      close(sd);
}
```