Expt.No.

Page No.
Date.

LAB ASSIGNMENT

1) Write a Server program and Client program for UDP connection.

Server.c

```
// Server program for udp connection.
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#define PORT 5000
#define MAXLINE 1000
int main()                    // Driver code.
    {
        char buffer[100];
        char *message = "Hello Client";
        int listenfd, len;
        struct sockaddr_in servaddr, cliaddr;
        bzero (&servaddr, sizeof(servaddr));
    // Create a UDP Socket
        listenfd = socket (AF_INET, SOCK_DGRAM, 0);
        servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
        servaddr.sin_port = htons (PORT);
        servaddr.sin_family = AF_INET;
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No.
Date.

```
// bind server address to socket descripton.
bind (listenfd, (struct sockaddr*) &servaddr, sizeof(servaddr));
// receive the datagram.
len = sizeof(cliaddr);
int n = recvfrom (listenfd, buffer, sizeof(buffer), 0,
        (struct sockaddr*) &cliaddr, &len); //receive message.
                                            from server.
buffer[n] = '\0';
puts(buffer);
// send the response.
sendto(listenfd, message, MAXLINE, 0,
        (struct sockaddr*) &clidder, sizeof(cliaddr));
}
```

client.c

```
//udp client driver program.
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdlib.h>
#define PORT 5000
#define MAXLINE 1000
```

OXFORD

Teacher's Signature .....

```
//Driver code.
int main()
    {
        char buffer[100];
        char *message = "Hello Server";
        int sockfd, n;
        Struct sockaddr_in servaddr;

// clear servaddr
        bzero(&servaddr, sizeof(servaddr));
        servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
        servaddr.sin_port = htons(PORT);
        servaddr.sin_family = AF_INET;

// create datagram socket
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);

// connect to server
    if(connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)
        <0)
    {
        printf("\n Error! Connect Failed\n");
        exit(0);
    }.
```

Page No.
Date.

```
// request to send datagram.
// no need to specify server address in sendto.
// connect stores the peers IP & port
sendto( sockfd, message, MAXLINE, 0, (struct sockaddr*) NULL, sizeof(servaddr));

// waiting for response.
recvfrom( sockfd, buffer, sizeof( buffer), 0, (struct sockaddr*)NULL, NULL);
puts (buffer);
// close the descriptor.
close( sockfd);
}
```

Output

Server.c

Hello Client.

Client.c

Hello Server.

2) Write a server program & client program for UDP connection. Where the server can send a message to the client by taking input from the keyboard.

Server.c

```
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>.
#define PORT 5000
#define MAXLINE 1000.
```

Teacher's Signature .....

```
int main()
{
    char buffer[100];
    char *message;
    int listenfd, len;
    struct sockaddr_in servaddr, cliaddr;
    bzero(&servaddr, sizeof(servaddr));
// Create a UDP socket
listenfd= socket(AF_INET, SOCK_DGRAM, 0);
 servaddr.sin_addr = htonl(INADDR_ANY);
 Servaddr.sin_port = htons(PORT);
 servaddr.sin_family = AF_INET;
//bind server address to socket descriptor:
bind(listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));

// receive the datagram.
len = sizeof(cliaddr);
int n = recvfrom(listenfd, buffer, sizeof(buffer),
        0, (struct sockaddr*)&cliaddr,&len); //receive message
                                              from client.
buffer[n] = '\0';
puts(buffer);
// Take input from the keyboard.
printf("Enter the message to send to client: ");
scanf("%s", message);
```

Teacher's Signature

Page No.
Date.

```
// send the response.
sendto(listenfd, message, MAXLINE, 0,
        (struct sockaddr*) &cliaddr, sizeof(cliaddr));

// receive the output from client
recvfrom(listenfd, buffer, sizeof(buffer), 0, (struct sockaddr*)
        &cliaddr, &len);
buffer[n] = '\0';
    puts(buffer);
}
```

client.c

```
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdlib.h>
#define PORT 5000
#define MAXLINE 1000
int main()
  {
  char buffer[100];
```

Teacher's Signature .....

Page No.
Date.

```
char    message[100];
int     sockfd, n;
struct  sockaddr_in servaddr;

bzero (&servaddr, sizeof(servaddr));
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);
servaddr.sin_family = AF_INET;

Sockfd = socket(AF_INET, SOCK_DGRAM, 0);

if (connect (sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr))<0)
    {
      printf("\n Error: Connect Failed\n");
      exit (0);
    }.
printf(" Enter message to send to the server:");
scanf("%s", message);
sendto ( sockfd, message, MAXLINE, 0, (struct. sockaddr*) NULL,
          sizeof(servaddr));
recvfrom( sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)NULL,
          NULL);
 printf (" Server Response: %s\n", buffer);
 close (sockfd);
 return 0;
}
```

Teacher's Signature .....

Output

server.c Enter message to send to ~~Server~~ Client: Hi!!
Client Response: Hello!!.

client.c Enter message to sent to server: Hello:.
Server Response: Hii!!