Shikshana Prasaraka Mandali’s

SIR PARASHURAMBHAU COLLEGE

(Autonomous)

Department of Computer Science

Lab-Book

M. Sc. - I

(Computer Science)

Lab Course – MCS12814

(From Academic year 2020-21)

Name: Vinayak Palve

College Name : Sir Parshurambhau College

Roll No. : 12062

Academic Year: 2021 - 2022

Shikshana Prasaraka Mandali's

SIR PARASHURAMBHAU COLLEGE

(Autonomous)

TILAK ROAD, PUNE – 411 030

Department of Computer Science

Subject - Computer Science

Certificate

This is to certify that Mr./Ms.Vinayak Hanumant Palve of M.Sc. (Computer Science) – I has completed _____ practicals out of _____ in the subject ***Computer Science*** *Semester – I/II during the academic year 2020- 2021*

Teacher In-Charge *Head*

Date: *Department of Computer Science*

Examiner 1 *Examiner 2*

Assignment Completion Sheet

Assignments based on Principles of Programming Language

Sr. No.	**Assignment Name**	**Start Date**	**End Date**	**Marks**
1	Networking Assignment 4			

Assignments based on Advanced Networking and Network Programming				
Sr. No.	Assignment Name	Start Date	End Date	Marks

Assignment No : 4

Assignment Name: Network Programming Assignment 4

1) **Write a client server program using UDP Sockets in ‘C’.**

```
//client.c
#include "unp.h"
int main(int argc, char **argv)
{
   int sockfd;
   struct sockaddr_in servaddr, cliaddr;


   bzero(&servaddr, sizeof(servaddr));
   bzero(&cliaddr, sizeof(cliaddr));

   servaddr.sin_family = AF_INET;
   servaddr.sin_port = htons(SERV_PORT);
   servaddr.sin_addr.s_addr = INADDR_ANY;



   sockfd = socket(AF_INET, SOCK_DGRAM, 0);


   char str[100];
   strcpy(str,"Hello, how are you?");
   printf("\n %s", str);

   int len = sizeof(servaddr);
   sendto(sockfd, str, sizeof(str), 0, (SA*)&servaddr, (socklen_t)len);


   int len1 = sizeof(cliaddr);
   bzero (str, 100);
   recvfrom(sockfd,str, 100, 0, (SA*)&cliaddr, (socklen_t *)&len1);
   printf("\n client side : %s", str);
   exit(0);
}

//server.c
```

```
#include "unp.h"

int main(int argc, char **argv)
{
  int sockfd;
  struct sockaddr_in servaddr, cliaddr;
  sockfd = socket(AF_INET, SOCK_DGRAM, 0);

  bzero(&servaddr, sizeof(servaddr));
  bzero(&cliaddr, sizeof(cliaddr));

  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servaddr.sin_port = htons(SERV_PORT);
  bind(sockfd, (SA *) &servaddr, sizeof(servaddr));

  char str[100];
  int len = sizeof(cliaddr);
  recvfrom(sockfd,str, 100, 0, (SA*)&cliaddr, (socklen_t *)&len);


  printf("\n server side %s", str);



  char msg[100];
  strcpy(msg, "I am fine");
  sendto(sockfd, msg, sizeof(msg), 0, (SA*)&cliaddr, (socklen_t)len);
}
```

2) Write a UDP Echo Client and UDP Echo Server in ‘C’.

```
//client.c
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<arpa/inet.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdio.h>
#define MAXLINE 1024
int main(int argc,char* argv[])
{
int sockfd;
int n;
socklen_t len;
```

```
char sendline[1024],recvline[1024];
struct sockaddr_in servaddr;



sockfd=socket(AF_INET,SOCK_DGRAM,0);

bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=htons(5035);
while(1)
{

printf("\n Enter the message : ");
scanf("%s",sendline);
len=sizeof(servaddr);
sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr*)&servaddr,len);

n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);
printf("\n Server's Echo : %s\n\n",recvline);

}
return 0;
}
//server.c
    #include<sys/types.h>
    #include<sys/socket.h>
    #include<netinet/in.h>
    #include<unistd.h>
    #include<netdb.h>
    #include<stdio.h>
    #include<string.h>
    #include<arpa/inet.h>
    #define MAXLINE 1024
    int main(int argc,char **argv)
    {
    int sockfd;
    int n;
    socklen_t len;
    char msg[1024];
    struct sockaddr_in servaddr,cliaddr;

    sockfd=socket(AF_INET,SOCK_DGRAM,0);
    bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=INADDR_ANY;
    servaddr.sin_port=htons(5035);
    printf("\n\n Binded");

    bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

for(;;)
{

    printf("\n ");
    len=sizeof(cliaddr);
    n=recvfrom(sockfd,msg,MAXLINE,0,(struct
sockaddr*)&cliaddr,&len);
    printf("\n Client's Message : %s\n",msg);
    sendto(sockfd,msg,n,0,(struct sockaddr*)&cliaddr,len);
}
    return 0;

}
```

3) Write a UDP daytime Client and UDP daytime server in 'C'.

```
//client.c
#include "unp.h"
#include<time.h>
int main(int argc, char **argv)
{
   int sockfd;
   time_t current_time;
   char strTime[1024];
   current_time = time(NULL);
   char strTime2[1024];
    strcpy(strTime,ctime(&current_time));

    struct sockaddr_in servaddr, cliaddr;


   bzero(&servaddr, sizeof(servaddr));
   bzero(&cliaddr, sizeof(cliaddr));

   servaddr.sin_family = AF_INET;
   servaddr.sin_port = htons(SERV_PORT);
   servaddr.sin_addr.s_addr = INADDR_ANY;

   sockfd = socket(AF_INET, SOCK_DGRAM, 0);

   int len = sizeof(servaddr);
   sendto(sockfd, strTime, sizeof(strTime), 0, (SA*)&servaddr,
(socklen_t)len);


   int len1 = sizeof(cliaddr);
   bzero (strTime2, 1024);
   recvfrom(sockfd,strTime2, 1024, 0, (SA*)&cliaddr, (socklen_t
*)&len1);
   printf("\n client side DayTime recieve from server : %s\n ",
strTime2);

   sleep(20);
   return 0;
}


//server.c
#include "unp.h"
#include<time.h>

int main(int argc, char **argv)
{
   int sockfd;
   struct sockaddr_in servaddr, cliaddr;
   time_t current_time;
   char strTime[1024];
    //time(&tm);
    //char *str=ctime(&tm);
   current_time = time(NULL);
   char strTime1[1024];
    strcpy(strTime,ctime(&current_time));
   //printf("\n Current time : %s",strTime);
   sockfd = socket(AF_INET, SOCK_DGRAM, 0);

   bzero(&servaddr, sizeof(servaddr));
   bzero(&cliaddr, sizeof(cliaddr));

   servaddr.sin_family = AF_INET;
   servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
   servaddr.sin_port = htons(SERV_PORT);
```

```
    bind(sockfd, (SA *) &servaddr, sizeof(servaddr));


    int len = sizeof(cliaddr);
    recvfrom(sockfd,strTime1, 1024, 0, (SA*)&cliaddr, (socklen_t *)&len);
    printf("\n Current time serverside send from client: %s\n",strTime1);
    sendto(sockfd, strTime, sizeof(strTime), 0, (SA*)&cliaddr,
  (socklen_t)len);
    sleep(20);


  }
```

4) **Write a UDP client server program in 'C'. Accept a file name at client side, send it to server. Server will read the file and send back content of file to client and client will display the file content. Use command line argument to accept a file name.**

```
//client.c

#include "unp.h"
int main(int argc, char **argv)
{
   int sockfd;
   struct sockaddr_in servaddr, cliaddr;


   bzero(&servaddr, sizeof(servaddr));
   bzero(&cliaddr, sizeof(cliaddr));

   servaddr.sin_family = AF_INET;
   servaddr.sin_port = htons(SERV_PORT);
   servaddr.sin_addr.s_addr = INADDR_ANY;



   sockfd = socket(AF_INET, SOCK_DGRAM, 0);


   char fname[100];


   printf("\nEnter file name: ");
```

```
        scanf("%s", fname);

    int len = sizeof(servaddr);
    sendto(sockfd, fname, sizeof(fname), 0, (SA*)&servaddr, (socklen_t)len);

        char str[100];
    int len1 = sizeof(cliaddr);
    bzero (str, 100);
    recvfrom(sockfd,str, 100, 0, (SA*)&cliaddr, (socklen_t *)&len1);
    printf("\n client side : %s", str);

    sleep(5);
    exit(0);
}




//server.c
#include "unp.h"

int main(int argc, char **argv)
{
    int sockfd,fd,n;
      char fname[100];
      char buffer[1024];
      char msg[100];
    struct sockaddr_in servaddr, cliaddr;
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);

    bzero(&servaddr, sizeof(servaddr));
    bzero(&cliaddr, sizeof(cliaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(SERV_PORT);
    bind(sockfd, (SA *) &servaddr, sizeof(servaddr));


    int len = sizeof(cliaddr);
    recvfrom(sockfd,fname, 100, 0, (SA*)&cliaddr, (socklen_t *)&len);




                fd = open("E:\\Programs\\Networking\\assig4\\abc.txt",
O_RDONLY);
```

```
              if (fd< 0){

                     strcpy(msg,"file not found");
                            sendto(sockfd, msg, sizeof(msg), 0, (SA*)&cliaddr,
(socklen_t)len);
                      }
              else
                 while ((n = read(fd, buffer, sizeof(buffer))) > 0)

   sendto(sockfd, buffer, sizeof(buffer), 0, (SA*)&cliaddr, (socklen_t)len);
}
```

Date: ____/____/_______

Signature of the instructor

4: Complete ☐

5: Well Done ☐

Assignment Evaluation

0: Not done ☐ 2: Late Complete ☐

1: Incomplete ☐ 3: Needs improvement ☐