**Assignment 1**

1.Write a udp client server program,client writing messages to server program and server

return back the same toggled msg to client

**server**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

#define PORT 8000

#define MAXSZ 100

int main()

{

int sockfd, newsockfd, n;

char msg[MAXSZ];

int clientAddrLen;

socklen\_t client;

struct sockaddr\_in cliaddr, servaddr;

char buf[10000];

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

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

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY); // htonl(inaddrany)

servaddr.sin\_port = htons(PORT); // tcp protocol http port.

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

printf("socket bound to port 8000");

listen(sockfd, 5);

while (1)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*server waiting for new client connection\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

clientAddrLen = sizeof(cliaddr);

newsockfd = accept(sockfd, (struct sockaddr \*)&cliaddr, &clientAddrLen);

while (1)

{

int n;

n = recv(newsockfd, msg, MAXSZ, 0);

if (n == 0)

{

close(newsockfd);

break;

}

msg[n] = 0;

send(newsockfd, msg, n, 0);

printf("received and set %s\n", msg);

}

}

return 0;

}

**client**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

#define PORT 8000

#define MAXSZ 100

int main()

{

char msg1[MAXSZ];

char msg2[MAXSZ];

int sockfd, ret\_val;

int n;

socklen\_t addr\_len;

struct sockaddr\_in servaddr;

sockfd = socket(AF\_INET, SOCK\_DGRAM, 0);

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

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

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY); // htonl(inaddrany)

servaddr.sin\_port = htons(PORT); // tcp protocol http port.

connect(sockfd, (struct sockaddr \*)&servaddr, sizeof(servaddr));

while (1)

{

printf("Enter msg to be sent to srever\n");

fgets(msg1, MAXSZ, stdin);

if (msg1[0] == '#')

break;

n = strlen(msg1) + 1;

send(sockfd, msg1, n, 0);

n = recv(sockfd, msg2, MAXSZ, 0);

printf("recived msg form served:%s\n", msg2);

// close(sockfd);

}

return 0;

}



2.Write a program that implement reliable transmission in TCP protocol, in which server

uses child to handle client request?

**server**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

int main()

{

int listfd, connfd, retval;

socklen\_t client;

struct sockaddr\_in cliaddr, servaddr;

listfd = socket(AF\_INET, SOCK\_STREAM, 0);

if (listfd < 0)

{

perror("sock:");

exit(1);

}

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

servaddr.sin\_port = htons(8000);

retval = bind(listfd, (struct sockaddr \*)&servaddr, sizeof(servaddr));

if (retval < 0)

{

perror("bind:");\

exit(2);

}

listen(listfd, 5);

while (1)

{

char buf[200];

int n;

client = sizeof(cliaddr);

connfd = accept(listfd, (struct sockaddr \*)&cliaddr, &cliaddr);

printf("client connected!!\n");

n = read(connfd, buf, 200);

buf[n] = '\0';

printf("data rec'd from client = %s\n", buf);

write(connfd, "good bye", 0);

return 0;

}

}

**Client**

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<netinet/in.h>

#include<stdio.h>

#include<stdlib.h>

main(){

char buf[200];

char \*serv\_ip = "127.0.0.1";

int n;

int sockfd,ret\_val;

struct sockaddr\_in servaddr;

sockfd = socket(AF\_INET,SOCK\_STREAM,0);

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(8000);

inet\_pton(AF\_INET, serv\_ip,&servaddr.sin\_addr);

ret\_val = connect(sockfd,(struct sockaddr \*)&servaddr,sizeof(servaddr));

if(ret\_val < 0){

perror("connect:");

exit(1);

}

printf("client established connection with server\n");

gets(buf);

write(sockfd,buf,strlen(buf));

n = read(sockfd,buf,200);

buf[n]='\0';

printf("rec'd %s from server \n",buf);

close(sockfd);

}

