2. TCP Client Server in C

Server.c

```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr

// Function designed for chat between client and server.
void func(int connfd)
{
      char buff[MAX];
      int n;
      // infinite loop for chat
      for (;;) {
            bzero(buff, MAX);

            // read the message from client and copy it in buffer
            read(connfd, buff, sizeof(buff));
            // print buffer which contains the client contents
            printf("From client: %s\t To client : ", buff);
            bzero(buff, MAX);
            n = 0;
            // copy server message in the buffer
            while ((buff[n++] = getchar()) != '\n')
                  ;

            // and send that buffer to client
            write(connfd, buff, sizeof(buff));

            // if msg contains "Exit" then server exit and chat ended.
            if (strncmp("exit", buff, 4) == 0) {
                  printf("Server Exit...\n");
                  break;
            }
      }
}

// Driver function
int main()
```

```
{
      int sockfd, connfd, len;
      struct sockaddr_in servaddr, cli;

      // socket create and verification
      sockfd = socket(AF_INET, SOCK_STREAM, 0);
      if (sockfd == -1) {
            printf("socket creation failed...\n");
            exit(0);
      }
      else
            printf("Socket successfully created..\n");
      bzero(&servaddr, sizeof(servaddr));

      // assign IP, PORT
      servaddr.sin_family = AF_INET;
      servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
      servaddr.sin_port = htons(PORT);

      // Binding newly created socket to given IP and verification
      if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
            printf("socket bind failed...\n");
            exit(0);
      }
      else
            printf("Socket successfully binded..\n");

      // Now server is ready to listen and verification
      if ((listen(sockfd, 5)) != 0) {
            printf("Listen failed...\n");
            exit(0);
      }
      else
            printf("Server listening..\n");
      len = sizeof(cli);

      // Accept the data packet from client and verification
      connfd = accept(sockfd, (SA*)&cli, &len);
      if (connfd < 0) {
            printf("server accept failed...\n");
            exit(0);
      }
      else
            printf("server accept the client...\n");

      // Function for chatting between client and server
      func(connfd);
```

```
	// After chatting close the socket
	close(sockfd);
}
```

Client.c

```
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr

void func(int sockfd)
{
	char buff[MAX];
	int n;
	for (;;) {
		bzero(buff, sizeof(buff));
		printf("\tTo server : ");
		n = 0;
		while ((buff[n++] = getchar()) != '\n')
			;
		write(sockfd, buff, sizeof(buff));
		bzero(buff, sizeof(buff));
		read(sockfd, buff, sizeof(buff));
		printf("From Server : %s", buff);
		if ((strncmp(buff, "exit", 4)) == 0) {
			printf("Client Exit...\n");
			break;
		}
	}
}

int main()
{
	int sockfd, connfd;
	struct sockaddr_in servaddr, cli;

	// socket create and varification
	sockfd = socket(AF_INET, SOCK_STREAM, 0);
	if (sockfd == -1) {
		printf("socket creation failed...\n");
		exit(0);
```

```
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));

    // assign IP, PORT
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);

    // connect the client socket to server socket
    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");

    // function for chat
    func(sockfd);

    // close the socket
    close(sockfd);
}
```

3. UDP Client Server in C

Server.c

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

void main(int argc, char ** argv){
    if(argc != 2){
        printf("Usage: %s <port>\n", argv[0]);
        exit(0);
    }
    int port = atoi(argv[1]);
    int sockfd;
    struct sockaddr_in si_me, si_other;
    char buffer[1024];
    socklen_t addr_size;
    sockfd=socket(AF_INET, SOCK_DGRAM, 0);
    memset(&si_me, '\0', sizeof(si_me));
    si_me.sin_family=AF_INET;
    si_me.sin_port=htons(port);
    si_me.sin_addr.s_addr=inet_addr("127.0.0.1");
    bind(sockfd, (struct sockaddr*)&si_me, sizeof(si_me));
    addr_size=sizeof(si_other);
    recvfrom(sockfd,buffer,1024,0,(struct sockaddr*)&si_other,
&addr_size);
    printf("[+]Data Recieved: %s", buffer);
}
```

Client.c

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

void main(int argc, char ** argv){
    if(argc != 2){
        printf("Usage: %s <port>\n", argv[0]);
        exit(0);
    }
    int port = atoi(argv[1]);
    int sockfd;
    struct sockaddr_in serverAddr;
    char buffer[1024];
    socklen_t addr_size;
    sockfd=socket(AF_INET, SOCK_DGRAM, 0);
    memset(&serverAddr, '\0', sizeof(serverAddr));
    serverAddr.sin_family=AF_INET;
    serverAddr.sin_port=htons(port);
    serverAddr.sin_addr.s_addr=inet_addr("127.0.0.1");
    strcpy(buffer, "Hello server\n");
    sendto(sockfd,buffer,1024,0,(struct
sockaddr*)&serverAddr,sizeof(serverAddr));
    printf("[+]Data Sent: %s", buffer);
}
```

4. UDP Client Server in C using connect()

Server.c

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

int main(){
    int c=0;
    char buff1[100], buff2[100];
    struct sockaddr_in serv_addr, cli_addr;
    int servfd=socket(AF_INET, SOCK_DGRAM, 0);
    bzero(&serv_addr, sizeof(serv_addr));
    bzero(&cli_addr, sizeof(cli_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(5001);
    int x=bind(servfd, (struct sockaddr* )&serv_addr, sizeof(serv_addr));
    if(x==-1){
        printf("Bind error");
        return 1;
    }
    int cli_length=sizeof(cli_addr);
    while(1){
        c++;
        printf("Msg %d to server: ",c);
        recvfrom(servfd, &buff1, sizeof(buff1), 0, (struct sockaddr
*)&cli_addr, &cli_length);
        puts(buff1);
        printf("Msg %d from server: ", c);
        fgets(buff2, 100, stdin);
        sendto(servfd, &buff2, sizeof(buff2), 0, (struct sockaddr
*)&cli_addr, sizeof(cli_addr));
        if(c==2){
            break;
        }
    }
    return 0;
}
```

Client.c

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

int main(){
    int c=0;
    char buff1[100], buff2[100];
    struct sockaddr_in serv_addr, cli_addr;
    int servfd=socket(AF_INET, SOCK_DGRAM, 0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(5001);
    int serv_length=sizeof(serv_addr);
    int n=connect(servfd, (struct sockaddr *)&serv_addr,
sizeof(serv_addr));
    if(n==0){
        printf("Connection established\n");
    }
    else{
        printf("Error\n");
    }
    while(1){
        c++;
        bzero(buff1, sizeof(buff1));
        bzero(buff2, sizeof(buff2));
        printf("Msg %d from client: ", c);
        fgets(buff2, 100, stdin);
        sendto(servfd, &buff2, sizeof(buff2), 0, NULL, 0);
        bzero(buff2, sizeof(buff2));
        printf("Msg %d to client: ", c);
        recvfrom(servfd, &buff1, sizeof(buff1), 0, NULL, 0);
        puts(buff1);
        bzero(buff1, sizeof(buff1));
        if(c==2){
            break;
        }

    }

    return 0;
}
```

5. TCP Client Server in C using fork()

Server.c

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

int main(){
    int c=0;
    pid_t pid;
    int servfd, connfd;
    servfd=socket(AF_INET, SOCK_STREAM, 0);
    char buff1[100], buff2[100];
    struct sockaddr_in serv_addr, cli_addr;
    bzero(&serv_addr, sizeof(serv_addr));
    bzero(&cli_addr, sizeof(cli_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(5000);
    int x=bind(servfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr));
    if(x==-1){
        printf("Bind Error");
        return 1;
    }
    int x1=listen(servfd, 5);
    if(x1==-1){
        printf("\nError");
        return 1;
    }
    for(;;){
        int s1=sizeof(cli_addr);
        connfd=accept(servfd, (struct sockaddr *)&cli_addr, &s1);
        bzero(buff1, sizeof(buff1));
        bzero(buff2, sizeof(buff2));
        if((pid=fork())==0){
            close(servfd);
            printf("Message from client to server: ");
            read(connfd, buff1, sizeof(buff1));
            puts(buff1);
            printf("Message from server to client: \n");
            fgets(buff2, 100, stdin);
            write(connfd, buff2, sizeof(buff2));
            close(connfd);
            exit(0);
        }
        close(connfd);
    }
    return 0;
}
```

Client1.c

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

int main(){
    int c=0;
    char buff1[100], buff2[100];
    struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET, SOCK_STREAM, 0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(5000);
    int n=connect(servfd, (struct sockaddr*)&serv_addr,
sizeof(serv_addr));
    if(n==0){
        printf("Connection established\n");
    }
    else{
        printf("Error\n");
    }
    bzero(buff1, sizeof(buff1));
    bzero(buff2, sizeof(buff2));
    printf("Message from client 1: ");
    fgets(buff2, 100, stdin);
    write(servfd, buff2, sizeof(buff2));
    bzero(buff2, sizeof(buff2));
    printf("Message to client: ");
    read(servfd, buff1, sizeof(buff1));
    puts(buff1);
    bzero(buff1, sizeof(buff1));
    printf("Client 1 disconnected\n");

    return 0;
}
```

Client2.c

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

int main(){
    int c=0;
    char buff1[100], buff2[100];
    struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET, SOCK_STREAM, 0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(5000);
    int n=connect(servfd, (struct sockaddr*)&serv_addr,
sizeof(serv_addr));
    if(n==0){
        printf("Connection established\n");
    }
    else{
        printf("Error\n");
    }
    bzero(buff1, sizeof(buff1));
    bzero(buff2, sizeof(buff2));
    printf("Message from client 2: ");
    fgets(buff2, 100, stdin);
    write(servfd, buff2, sizeof(buff2));
    bzero(buff2, sizeof(buff2));
    printf("Message to client: ");
    read(servfd, buff1, sizeof(buff1));
    puts(buff1);
    bzero(buff1, sizeof(buff1));
    printf("Client 2 disconnected\n");

    return 0;
}
```

6. TCP/UDP Client Server in C using select()

Server.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<unistd.h>
#include<errno.h>

int max(int x,int y){
    if(x>y)
        return x;
    else
        return y;
}

void err_sys(const char* x){
    perror(x);
    exit(1);
}

int main(){
    int c=0,c1=1, len,n; pid_t pid;
    fd_set rset;
    int listenfd, connfd, udpfd, nready, maxfdpl;
    char msg [100];
    listenfd=socket(AF_INET, SOCK_STREAM, 0);
    char buff1[100],buff2[100];
    struct sockaddr_in serv_addr,cli_addr;
    bzero(&serv_addr,sizeof(serv_addr));
    bzero(&cli_addr,sizeof(cli_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons (4500);
    bind(listenfd, (struct sockaddr *) &serv_addr,sizeof(serv_addr));
    listen(listenfd,5);
    udpfd=socket (AF_INET, SOCK_DGRAM,0);
    bzero (&serv_addr,sizeof(serv_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(4500);
    bind(udpfd, (struct sockaddr *)&serv_addr,sizeof(serv_addr));
    FD_ZERO (&rset);
    maxfdpl=max(listenfd, udpfd)+1;
    while(1){
        FD_SET(listenfd, &rset);
        FD_SET(udpfd,&rset);
        if(nready=select(maxfdpl,&rset, NULL, NULL, NULL)<0){
            if(errno==EINTR)  continue;
            else err_sys("select error\n");
        }
        if(FD_ISSET(listenfd,&rset)){
            len=sizeof(cli_addr);
            connfd=accept(listenfd, (struct sockaddr*)&cli_addr, &len);
            bzero(buff1,sizeof(buff1));
            bzero(buff2, sizeof(buff2));
            if((pid=fork())==0){
                close(listenfd);
                printf("msg from TCP client to server: ");
                if((read(connfd, buff1,sizeof(buff1))<0)){
                    perror("Error: ");
                    return(-1);
                }
                else puts(buff1);
                printf("msg from server to TCP client: ");
                fgets(buff2, 100,stdin);
                if((write(connfd, buff2, sizeof(buff2))<0)){
                    perror("Error: ");
                    return(-1);
                }
                close(connfd);
                exit(0);
            }
            close(connfd);
        }
        if(FD_ISSET(udpfd,&rset)){len=sizeof(cli_addr);
            printf("msg from UDP client to server: ");
            recvfrom(udpfd,&buff1,sizeof(buff1),0,(struct
sockaddr*)&cli_addr,&len);
            puts(buff1);
            printf("msg from server to UDP client: ");
            fgets(buff2,100,stdin);
            sendto(udpfd,&buff2,sizeof(buff2),0,(struct sockaddr
*)&cli_addr,sizeof(cli_addr));
        }
    }
    return 0;
}
```

TCP client1.c

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>

int main(){
    int c=0;
    char buff1[100],buff2[100];
    struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET,SOCK_STREAM,0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(4500);
    int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
    if(n==0){
        printf("Connection Established\n");
    }
    else{
        printf("error\n");
    }
    bzero(buff1,sizeof(buff1));
    bzero(buff2,sizeof(buff2));
    printf("Message from client 1: ");
    fgets(buff2,100,stdin);
    write(servfd,buff2,sizeof(buff2));
    bzero(buff2,sizeof(buff2));
    printf("Message to client: ");
    read(servfd,buff1,sizeof(buff1));
    puts(buff1);
    bzero(buff1,sizeof(buff1));
    printf("TCP client 1 disconnected !\n");
    return 0;
}
```

TCP Client2.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>

int main(){
    int c=0;
    char buff1[100],buff2[100];
    struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET,SOCK_STREAM,0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(4500);
    int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
    if(n==0){
        printf("Connection Established\n");
    }
    else{
        printf("Error\n");
    }
    bzero(buff1,sizeof(buff1));
    bzero(buff2,sizeof(buff2));
    printf("Message from client 2: ");
    fgets(buff2,100,stdin);
    write(servfd,buff2,sizeof(buff2));
    bzero(buff2,sizeof(buff2));
    printf("Message to client: ");
    read(servfd,buff1,sizeof(buff1));
    puts(buff1);
    bzero(buff1,sizeof(buff1));
    printf("TCP client 2 disconnected !\n");
    return 0;
}
```

UDP Client.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>

int main(){
    int c=0;
    char buff1[100],buff2[100];
    struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET,SOCK_DGRAM,0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(4500);
    int serv_length=sizeof(serv_addr);
    bzero(buff1,sizeof(buff1));
    bzero(buff2,sizeof(buff2));
    printf("Message from client: ");
    fgets(buff2,100,stdin);
    sendto(servfd,&buff2,sizeof(buff2),0,(struct sockaddr
*)&serv_addr,sizeof(serv_addr));
    bzero(buff2,sizeof(buff2));
    printf("Message to client: ");
    recvfrom(servfd,&buff1,sizeof(buff1),0,(struct sockaddr
*)&serv_addr,&serv_length);
    puts(buff1);
    bzero(buff1,sizeof(buff1));
    printf("UDP client disconnected !\n");
    return 0;
}
```

7. Unix Domain Datagram Sockets

server.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<sys/un.h>
#include<unistd.h>

int main(){
    int c=0, c1=1, servfd;
    servfd = socket(AF_LOCAL, SOCK_DGRAM, 0);
    struct sockaddr_un serv_addr, cli_addr;
    unlink("/tmp/unix.str");
    char buff1[100], buff2[100];
    bzero(&serv_addr, sizeof(serv_addr));
    serv_addr.sun_family=AF_LOCAL;
    strcpy(serv_addr.sun_path,"/tmp/unix.str");
    int x=bind(servfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr));
    if(x==-1){
        printf("Bind error\n");
        return 1;
    }
    int cli_len=sizeof(cli_addr);
    bzero(buff1, sizeof(buff1));
    bzero(buff2, sizeof(buff2));
    printf("Msg to server: ");
    recvfrom(servfd, &buff1, sizeof(buff1), 0, (struct sockaddr
*)&cli_addr, &cli_len);
    puts(buff1);
    printf("Msg from server: ");
    fgets(buff2, 100, stdin);
    sendto(servfd, &buff2, sizeof(buff2), 0, (struct sockaddr *)&cli_addr,
sizeof(cli_addr));
    return 0;
}
```

Client.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<sys/un.h>

int main(){
    int c=0;
    char buff1[100], buff2[100];
    struct sockaddr_un serv_addr, cli_addr;
    int servfd=socket(AF_LOCAL, SOCK_DGRAM, 0);
    bzero(&cli_addr, sizeof(cli_addr));
    cli_addr.sun_family=AF_LOCAL;
    strcpy(cli_addr.sun_path, tmpnam(NULL));
    int x=bind(servfd, (struct sockaddr *)&cli_addr, sizeof(cli_addr));
    if(x==-1){
        printf("Bind error\n");
        return 1;
    }
    bzero(&serv_addr, sizeof(serv_addr));
    serv_addr.sun_family=AF_LOCAL;
    strcpy(serv_addr.sun_path, "/tmp/unix.str");
    int serv_len=sizeof(serv_addr);
    bzero(buff1, sizeof(buff1));
    bzero(buff2, sizeof(buff2));
    printf("Msg from client: ");
    fgets(buff2, 100, stdin);
    sendto(servfd, &buff2, sizeof(buff2), 0, (struct sockaddr
*)&serv_addr, sizeof(serv_addr));
    bzero(buff2, sizeof(buff2));
    printf("Msg to client: ");
    recvfrom(servfd, &buff1, sizeof(buff1), 0, (struct sockaddr
*)&serv_addr, &serv_len);
    puts(buff1);
    bzero(buff1, sizeof(buff1));
    return 0;
}
```

8. Client/server implementation with alarm()

Server.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<unistd.h>

int main(){
    int c=0;
    char buff1[100],buff2[100];
    struct sockaddr_in serv_addr,cli_addr;
    int servfd=socket(AF_INET,SOCK_STREAM,0);
    bzero(&serv_addr,sizeof(serv_addr));
    bzero(&cli_addr,sizeof(cli_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(7000);
    int x=bind(servfd,(struct sockaddr *)&serv_addr, sizeof(serv_addr));
    if(x==-1){
        printf("bind error");
        return 1;
    }
    int x1=listen(servfd,5);
    if(x1==-1){
        printf("\nerror");
        return 1;
    }
    int s1=sizeof(cli_addr);
    int x2=accept(servfd,(struct sockaddr *)&cli_addr,&s1);
    bzero(buff1,sizeof(buff1));
    bzero(buff2,sizeof(buff2));
    printf("msgto server:");
    read(x2,buff1,sizeof(buff1));
    puts(buff1);
    bzero(buff1,sizeof(buff1));
    printf("msg from server: ");
    fgets(buff2,100,stdin);
    write(x2,buff2,sizeof(buff2));
    bzero(buff2,sizeof(buff2));
    return 0;
}
```

Client.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<signal.h>
#include<unistd.h>

void siga(int signo){
    printf("Read Error\n");
    exit(1);
}

int main(){
    int c=0;
    char buff1[100],buff2[100]; struct sockaddr_in serv_addr;
    int servfd=socket(AF_INET,SOCK_STREAM,0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(7000);
    int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
    if(n==0) printf("Connection Established\n");
    else printf("Error\n");
    signal(SIGALRM, siga);
    bzero(buff1,sizeof(buff1));
    bzero(buff2,sizeof(buff2));
    printf("msg from client: ");
    fgets(buff2,100,stdin); write(servfd,buff2,sizeof(buff2));
    bzero(buff2,sizeof(buff2));
    printf("msg to client: ");
    alarm(10);
    sleep(5);
    read(servfd,buff1,sizeof(buff1));
    puts(buff1);
    bzero(buff1,sizeof(buff1));
    return 0;
}
```

9. UDP Client server program to verify received response

Server.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<unistd.h>
#include <arpa/inet.h>
int main()
{
int c=0;
char buff1[100],buff2[100];
struct sockaddr_in serv_addr,cli_addr;
int servfd=socket(AF_INET,SOCK_DGRAM,0);
bzero(&serv_addr,sizeof(serv_addr)); bzero(&cli_addr,sizeof(cli_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=INADDR_ANY;
serv_addr.sin_port=htons(5000);
int x=bind(servfd,(struct sockaddr *)&serv_addr, sizeof(serv_addr));
if(x==-1)
{
printf("bind error"); return 1;
}
int
cli_length=sizeof(cli_addr);
printf("msg to server: ");
recvfrom(servfd, &buff1, sizeof(buff1), 0, (struct sockaddr *)&cli_addr,
&cli_length);
puts(buff1);
printf("msg from server:");
fgets(buff2,100,stdin);
sendto(servfd, &buff2, sizeof(buff2), 0, (struct sockaddr *)&cli_addr,
sizeof(cli_addr));
return 0;
}
```

Client.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include<sys/types.h>
int main(){
struct sockaddr_in serv_addr, allegedserv;
int sockfd=socket(AF_INET,SOCK_DGRAM,0);
bzero(&serv_addr,sizeof(serv_addr));
bzero(&allegedserv,sizeof(allegedserv));
serv_addr.sin_family = AF_INET;
serv_addr.sin_port = htons(5000);
serv_addr.sin_addr.s_addr=INADDR_ANY;
char read_buff[10],write_buff[10];
bzero(read_buff,sizeof(read_buff));
bzero(write_buff,sizeof(write_buff));
printf("msg from client: ");
fgets(write_buff,10,stdin);
int s=
sizeof(serv_addr); int
alen;
char str[30]={0},str1[30]={0};
sendto(sockfd,write_buff,sizeof(write_buff),0,(struct
sockaddr*)&serv_addr,s); inet_ntop(AF_INET,&serv_addr.sin_addr,str1, s);
bzero(write_buff,sizeof(write_buff));
printf("msg to client: ");
recvfrom(sockfd,read_buff,sizeof(read_buff),0,(struct
sockaddr*)&allegedserv,&alen);
inet_ntop(AF_INET,&allegedserv.sin_addr,str, alen);
if(s!=alen|| memcmp(&str,&str1,sizeof(str))!=0)
printf("reply came from different server %s \n",str); else
printf("it came from the same server %s \n",str);
puts(read_buff);
bzero(read_buff,sizeof(read_buff));
return 0;
}
```

10. TCP Client server program on different systems

Server.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
int main()
{
int c=0;
char buff1[100],buff2[100];
struct sockaddr_in serv_addr,cli_addr;
int servfd=socket(AF_INET,SOCK_STREAM,0);
bzero(&serv_addr,sizeof(serv_addr)); bzero(&cli_addr,sizeof(cli_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=inet_addr("0.0.0.0");
serv_addr.sin_port=htons(3000);
int x=bind(servfd,(struct sockaddr *)&serv_addr, sizeof(serv_addr));
if(x==-1)
{
printf("bind error");
return 1;
}
int x1=listen(servfd,5);
if(x1==-1)
{
printf("\nerror");
return 1;
}
int s1=sizeof(cli_addr);
int x2=accept(servfd,(struct sockaddr *)&cli_addr,&s1);

getpeername(x2,(struct sockaddr *)&cli_addr,&s1); char
*ch=malloc(100);
inet_ntop(AF_INET,&cli_addr.sin_addr,ch,100); printf("Client IP Address=
%s\n",ch);
bzero(buff1,sizeof(buff1));
bzero(buff2,sizeof(buff2));
printf("msg %d to server: ",c);
read(x2,buff1,sizeof(buff1));
puts(buff1);
bzero(buff1,sizeof(buff1));
printf("msg %d from server: ",c);
fgets(buff2,100,stdin);
write(x2,buff2,sizeof(buff2));
bzero(buff2,sizeof(buff2));
return 0;
}
```

Client.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/socket.h>
#include<sys/types.h>
int main()
{
int c=0;
char buff1[100],buff2[100]; struct
sockaddr_in serv_addr;
int servfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=inet_addr("0.0.0.0");
serv_addr.sin_port=htons(3000);
int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
if(n==0)
{
printf("connection established\n");
}
else
printf("error\n");
bzero(buff1,sizeof(buff1));
bzero(buff2,sizeof(buff2));

printf("msg %d from client: ",c);
fgets(buff2,100,stdin);
write(servfd,buff2,sizeof(buff2));
bzero(buff2,sizeof(buff2));
printf("msg %d to client: ",c);
read(servfd,buff1,sizeof(buff1));
puts(buff1);
bzero(buff1,sizeof(buff1));
return 0;
}
```

11. TCP Client server program using select without fork()

Server.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
int i,c=0,c1=1,len,n;
int nready, cl[FD_SETSIZE], maxi, maxfd; fd_set
rset,allset;
int listenfd, connfd, sockfd;
listenfd=socket(AF_INET,SOCK_STREAM,0);
char buff1[100],buff2[100];
struct sockaddr_in serv_addr,cli_addr;
bzero(&serv_addr,sizeof(serv_addr));
bzero(&cli_addr,sizeof(cli_addr));
serv_addr.sin_family=AF_INET; serv_addr.sin_addr.s_addr=INADDR_ANY;
serv_addr.sin_port=htons(8000);
int x=bind(listenfd,(struct sockaddr *)&serv_addr, sizeof(serv_addr));
if(x==-1)
{
printf("Bind error");
return 1;
}
int x1=listen(listenfd,5);
if(x1==-1)
{
printf("Error");
return 1;
}
maxfd=listenfd;
maxi=-1;
for(i=0;i<FD_SETSIZE; i++)
cl[i]=-1; FD_ZERO(&allset);
FD_SET(listenfd,&allset);
for(;;){
rset=allset;
nready=select(maxfd+1, &rset, NULL, NULL, NULL);
if(FD_ISSET(listenfd,&rset)){
len=sizeof(cli_addr);
connfd=accept(listenfd,(struct sockaddr *)&cli_addr, &len);
for(i=0;i<FD_SETSIZE;i++)
if(cl[i]<0){
cl[i]=connfd;
break;
}
if(i==FD_SETSIZE)
printf("Too many clients");
FD_SET(connfd,&allset);
if(connfd>maxfd)
maxfd=connfd; if(i>maxi)
maxi=i;
if(--nready<=0) continue;
}
for(i=0;i<=maxi;i++){
if((sockfd=cl[i])<0)
continue;
if(FD_ISSET(sockfd, &rset)){
printf("Msg from client to server: ");
read(sockfd,buff1,100); puts(buff1);
printf("Msg from server to client: ");
fgets(buff2,100,stdin); write(sockfd,buff2,100);

close(sockfd);
FD_CLR(sockfd,&allset);
cl[i]=-1;
if(--nready<=0){
break;}
}
}
}
}
```

Client1.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
int c=0;
char buff1[100],buff2[100]; struct
sockaddr_in serv_addr;
int servfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET; serv_addr.sin_addr.s_addr=INADDR_ANY;
serv_addr.sin_port=htons(8000);
int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
if(n==0)
{
printf("Connection established\n");
}
else printf("Error");
bzero(buff1,sizeof(buff1));
bzero(buff2,sizeof(buff2));
printf("msg from client 1: ");
fgets(buff2,100,stdin);
write(servfd,buff2,sizeof(buff2));
bzero(buff2,sizeof(buff2));
printf("msg to client: ");
read(servfd,buff1,sizeof(buff1));
puts(buff1); bzero(buff1,sizeof(buff1));
printf("Client 1 disconnected");
return 0;
}
```

Client2.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
int c=0;
char buff1[100],buff2[100]; struct
sockaddr_in serv_addr;
int servfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET; serv_addr.sin_addr.s_addr=INADDR_ANY;
serv_addr.sin_port=htons(8000);
int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
if(n==0)
{
printf("Connection established\n");
}
else printf("Error");
bzero(buff1,sizeof(buff1));
bzero(buff2,sizeof(buff2));
printf("msg from client 2: ");
fgets(buff2,100,stdin);
write(servfd,buff2,sizeof(buff2));
bzero(buff2,sizeof(buff2));
printf("msg to client: ");
read(servfd,buff1,sizeof(buff1));
puts(buff1); bzero(buff1,sizeof(buff1));
printf("Client 2 disconnected");
return 0;
}
```

Client3.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
int c=0;
char buff1[100],buff2[100]; struct
sockaddr_in serv_addr;
int servfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET; serv_addr.sin_addr.s_addr=INADDR_ANY;
serv_addr.sin_port=htons(8000);
int n=connect(servfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr));
if(n==0)
{
printf("Connection established\n");
}
else printf("Error");
bzero(buff1,sizeof(buff1));
bzero(buff2,sizeof(buff2));
printf("msg from client 3: ");
fgets(buff2,100,stdin);
write(servfd,buff2,sizeof(buff2));
bzero(buff2,sizeof(buff2));
printf("msg to client: ");
read(servfd,buff1,sizeof(buff1));
puts(buff1); bzero(buff1,sizeof(buff1));
printf("Client 3 disconnected");
return 0;
}
```