Q)2 way multiple time communication using pipe

```
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<stdlib.h>
void main(){
	int fd1[2],fd2[2];
	char message[1024],buffer[1024];
	int pid=0;
	if(pipe(fd1)==-1){
		perror("Pipe Creation Failed \n");
		exit(0);
	}
	if(pipe(fd2)==-1){
		perror("Pipe Creation Failed \n");
		exit(0);
	}
	pid=fork();
	if(pid>0){
		close(fd1[0]);
		close(fd2[1]);
		while(1){
			memset(message,0,sizeof(message));
			printf("Write data for child: ");
			gets(message);
			write(fd1[1],message,1024);
			if(strcmp(message,"exit")==0) break;
			memset(buffer,0,sizeof(buffer));
			read(fd2[0],buffer,1024);
			printf("Recieved data from child: %s \n",buffer);
			if(strcmp(buffer,"exit")==0) break;
```

```
                }
        }
        else{
                close(fd1[1]);
                close(fd2[0]);
                while(1){
                        memset(buffer,0,sizeof(buffer));
                        read(fd1[0],buffer,1024);
                        printf("Received data from parent: %s\n ",buffer);
                        if(strcmp(buffer,"exit")==0) break;
                        memset(message,0,sizeof(message));
                        printf("Write data for parent:");
                        gets(message);
                        write(fd2[1],message,1024);
                        if(strcmp(message,"exit")==0) break;
                }
        }
}
```

Q2) FIFO 2 way

Read-write.c

```
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
void main(){
int fd;
char *myfifo="abc";
char message[1024],buffer[1024];
```

```
mkfifo(myfifo,0666);
while(1)
{
memset(buffer,0,sizeof(buffer));
fd=open(myfifo, O_RDONLY);
read(fd, buffer,1024);
printf("Received data: %s\n",buffer);
if(strcmp(buffer,"exit")==0) break;
close(fd);
memset(message,0,sizeof(message));
fd=open(myfifo, O_WRONLY);
printf("Enter Input string:");
gets(message);
write(fd, message, 1024);
if(strcmp(message,"exit")==0) break;
close(fd);
}
}
```

Write-read.c

```
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
void main(){
int fd;
char *myfifo="abc";
char message[1024],buffer[1024];
mkfifo(myfifo,0666);
while(1)
```

```
{
memset(message,0,sizeof(message));
fd=open(myfifo, O_WRONLY);
printf("Enter Input string:");
gets(message);
write(fd, message, 1024);
if(strcmp(message,"exit")==0) break;
close(fd);
memset(buffer,0,sizeof(buffer));
fd=open(myfifo, O_RDONLY);
read(fd, buffer,1024);
printf("Received data: %s\n",buffer);
if(strcmp(buffer,"exit")==0) break;
close(fd);
}
}
```

Q3)2 way multiple time communication using tcp/ip

TCP SERVER

```
#include<stdio.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#define PORT 8090
void main()
{ int opt=1;
int svrsock_fd,new_conn;
char buffer[1024],message[1024];
struct sockaddr_in address;
socklen_t addrlen=sizeof(struct sockaddr_in);
svrsock_fd=socket(AF_INET,SOCK_STREAM,0);
address.sin_family=AF_INET;
```

```
address.sin_addr.s_addr=INADDR_ANY;
address.sin_port=htons(PORT);
setsockopt(svrsock_fd,SOL_SOCKET,SO_REUSEADDR|SO_REUSEPORT,opt,&opt);
bind(svrsock_fd,(struct sockadddr*)&address,addrlen);
printf("waiting for client\n");
listen(svrsock_fd,3);
new_conn=accept(svrsock_fd,(struct sockaddr*)&address,&addrlen);
while(1)
{
memset(buffer,0,sizeof(buffer));
read(new_conn,buffer,1024);
printf("received data from TCP/IP client:%s\n",buffer);
if(strcmp(buffer,"exit")==0)break;
memset(message,0,sizeof(message));
printf("enter data for TCP/IP client:");
gets(message);
send(new_conn,message,strlen(message),0);
if(strcmp(message,"exit")==0)break;
}
}
```

TCP CLIENT

```
#include<stdio.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#define PORT 8090
void main()
{
int clnsock_fd;
char buffer[1024],message[1024];
struct sockaddr_in svraddr;
socklen_t svraddrlen=sizeof(struct sockaddr_in);
```

```
clnsock_fd=socket(AF_INET,SOCK_STREAM,0);
svraddr.sin_family=AF_INET;
svraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
svraddr.sin_port=htons(PORT);
connect(clnsock_fd,(struct sockadddr*)&svraddr,svraddrlen);
while(1)
{
memset(message,0,sizeof(message));
printf("enter data for TCP/IP server:");
gets(message);
send(clnsock_fd,message,strlen(message),0);
if(strcmp(message,"exit")==0)break;
memset(buffer,0,sizeof(buffer));
read(clnsock_fd,buffer,1024);
printf("received data from TCP/IP server:%s\n",buffer);
if(strcmp(buffer,"exit")==0)break;

}
}
```

Q4)UDP CONNECTION 2 way

UDP SERVER

```
#include<stdio.h>
#include<unistd.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#define PORT 8080
void main()
{char buffer[1024],message[1024];
int svrsock_fd;
struct sockaddr_in svraddr,clnaddr;
socklen_t svraddrlen=sizeof(struct sockaddr_in);
```

```
socklen_t clnaddrlen=sizeof(struct sockaddr_in);
svrsock_fd=socket(AF_INET,SOCK_DGRAM,0);
svraddr.sin_family=AF_INET;
svraddr.sin_addr.s_addr=INADDR_ANY;
svraddr.sin_port=htons(PORT);
bind(svrsock_fd,(struct sockaddr *)&svraddr,svraddrlen);
printf("WAITING FOR UDP/IP CLIENT\n");
while(1)
{
memset(buffer,0,sizeof(buffer));
recvfrom(svrsock_fd,buffer,sizeof(buffer),0,&clnaddr,&clnaddrlen);
printf("received data from UDP/IP CLIENT:%s\n",buffer);
if(strcmp(buffer,"exit")==0)break;
memset(message,0,sizeof(message));
printf("enter data for UDP/IP client:");
gets(message);
sendto(svrsock_fd,message,sizeof(message),0,&clnaddr,clnaddrlen);
if(strcmp(message,"exit")==0)break;
}
}
```

UDP CLIENT

```
#include<stdio.h>
#include<unistd.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#define PORT 8080
void main()
{char message[1024],buffer[1024];
int clnsock_fd;
struct sockaddr_in svraddr,clnaddr;
socklen_t svraddrlen=sizeof(struct sockaddr_in);
socklen_t clnaddrlen=sizeof(struct sockaddr_in);
clnsock_fd=socket(AF_INET,SOCK_DGRAM,0);
```

```
svraddr.sin_family=AF_INET;
svraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
svraddr.sin_port=htons(PORT);
while(1)
{
memset(message,0,sizeof(message));
printf("enter data for UDP/IP server:");
gets(message);
sendto(clnsock_fd,message,sizeof(message),0,&svraddr,svraddrlen);
if(strcmp(message,"exit")==0)break;
memset(buffer,0,sizeof(buffer));
recvfrom(clnsock_fd,buffer,sizeof(buffer),0,&svraddr,&svraddrlen);
printf("received data from UDP/IP server:%s\n",buffer);
if(strcmp(buffer,"exit")==0)break;
}
}
```