Network IPC : Sockets

System Programming

1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(서버 프로그램)

- 아래와 같이 TCP소켓 echo 서버 프로그램 작성
- tcp_server1.c 파일로 작성(gedit tcp_server1.c명령 사용)

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

#define BUFSIZE 1024

void error_handling(char * message);

int main(int argc, char ** argv)
{
        int serv_sock;
        int clnt_sock;
        char message[BUFSIZE];
        int str_len;

        struct sockaddr_in serv_addr;
        struct sockaddr_in clnt_addr;
        int clnt_addr_size;

        serv_sock = socket(PF_INET, SOCK_STREAM, 0);
        if(serv_sock == -1)
                error_handling("socket() error");
```

(계속)

1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(서버 프로그램)

❑ 아래와 같이 TCP소켓 echo 서버 프로그램 작성

```
        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family=AF_INET;
        serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
        serv_addr.sin_port=htons(4000);

        if(bind(serv_sock, (struct sockaddr*) &serv_addr, sizeof(serv_addr)) == -1)
                error_handling("bind() error");

        if(listen(serv_sock, 5) == -1)
                error_handling("listen() error");

        clnt_addr_size = sizeof(clnt_addr);
        clnt_sock=accept(serv_sock, (struct sockaddr*) &clnt_addr, &clnt_addr_size);

        if(clnt_sock == -1)
                error_handling("accept() error");

        while((str_len = read(clnt_sock, message, BUFSIZE)) != 0) {
                write(clnt_sock, message, str_len);
                write(1, message, str_len);
        }
        close(clnt_sock);

        return 0;
}

void error_handling(char * message)
{
        fputs(message, stderr);
        fputc('\n', stderr);
        exit(1);
}
```

(끝)

1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(클라이언트 프로그램)

- 아래와 같이 TCP소켓 echo 클라이언트 프로그램 작성
- tcp_client1.c 파일로 작성

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

#define BUFSIZE 1024

void error_handling(char * message);

int main(int argc, char ** argv)
{
        int sock;
        char message[BUFSIZE];
        int str_len;
        struct sockaddr_in serv_addr;

        sock = socket(PF_INET, SOCK_STREAM, 0);
        if(sock == -1)
                error_handling("socket() error");

        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family=AF_INET;
        serv_addr.sin_addr.s_addr=inet_addr("127.0.0.1");
        serv_addr.sin_port=htons(4000);
```

(계속)

TCP 소켓 프로그램 작성(클라이언트 프로그램)

- 아래와 같이 TCP소켓 echo 클라이언트 프로그램 작성
- tcp_client1.c 파일로 작성

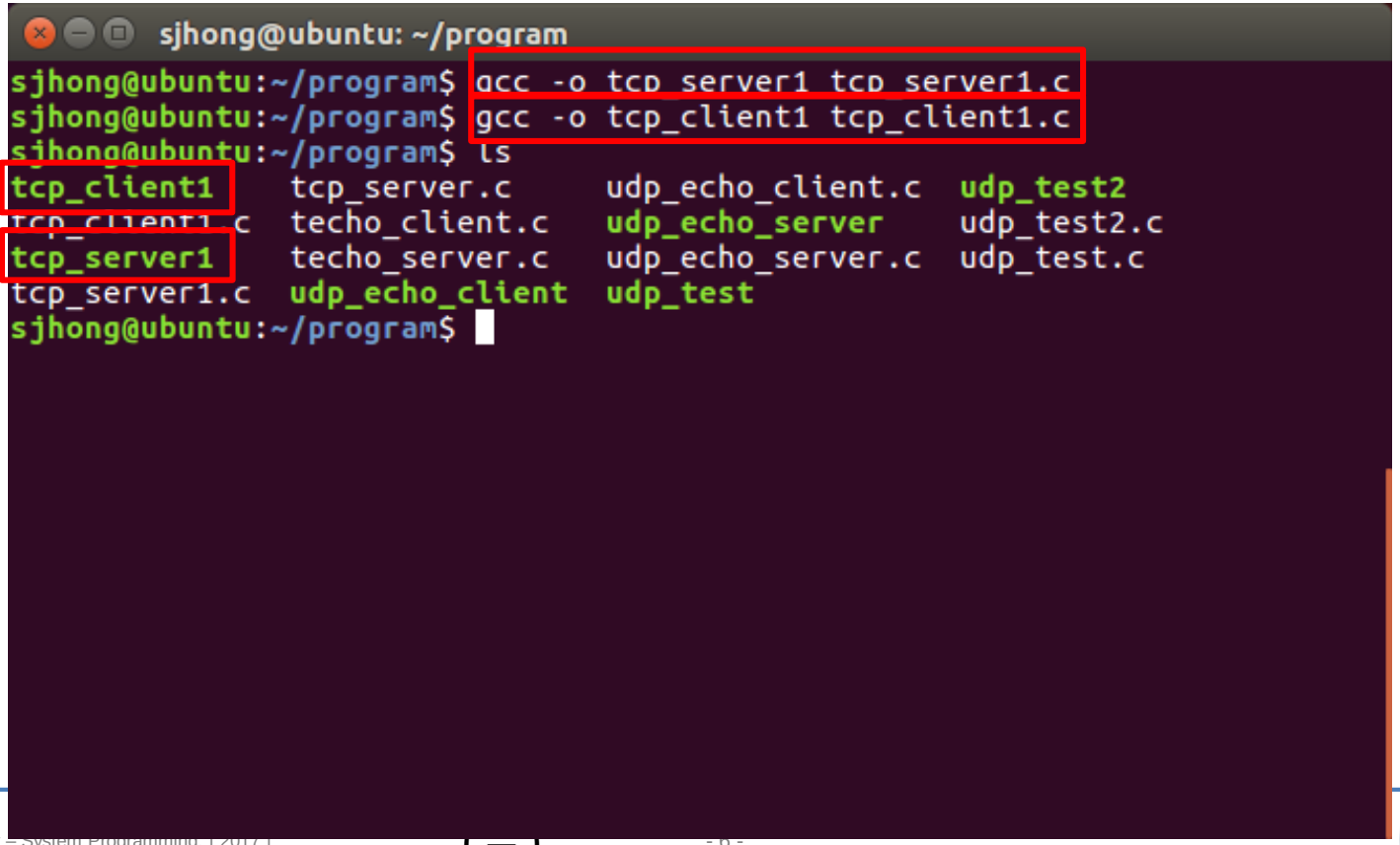
```
        if(connect(sock, (struct sockaddr*)&serv_addr, sizeof(serv_addr)) == -1)
                error_handling("connect() error");

        while(1) {
                fputs("Input message (q to quit) : ", stdout);
                fgets(message, BUFSIZE, stdin);

                if(!strcmp(message, "q\n")) break;
                write(sock, message, strlen(message));

                str_len=read(sock, message, BUFSIZE -1);
                message[str_len]=0;
                printf("Message from server :%s \n", message);

        }

        close(sock);

        return 0;
}

void error_handling(char * message)
{
        fputs(message, stderr);
        fputc('\n', stderr);
        exit(1);
}
```

(끝)

2. TCP 소켓 프로그램 컴파일

TCP 소켓 프로그램 컴파일

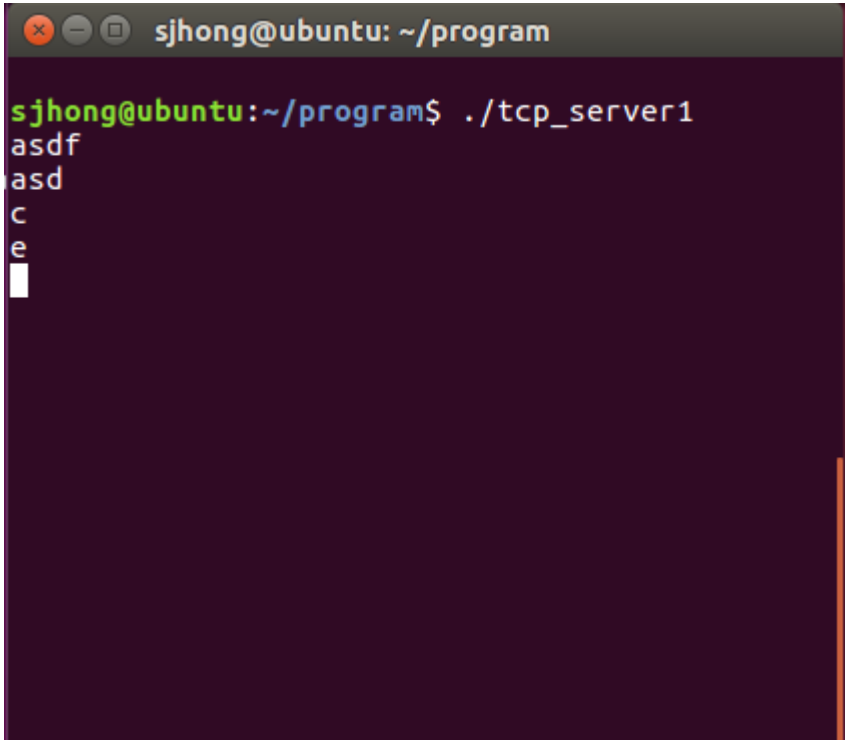
- gcc –o tcp_server1 tcp_server1.c와
- gcc -o tcp_client1 tcp_client1.c 명령어로 tcp프로그램 컴파일
- 실행 파일 생성 확인

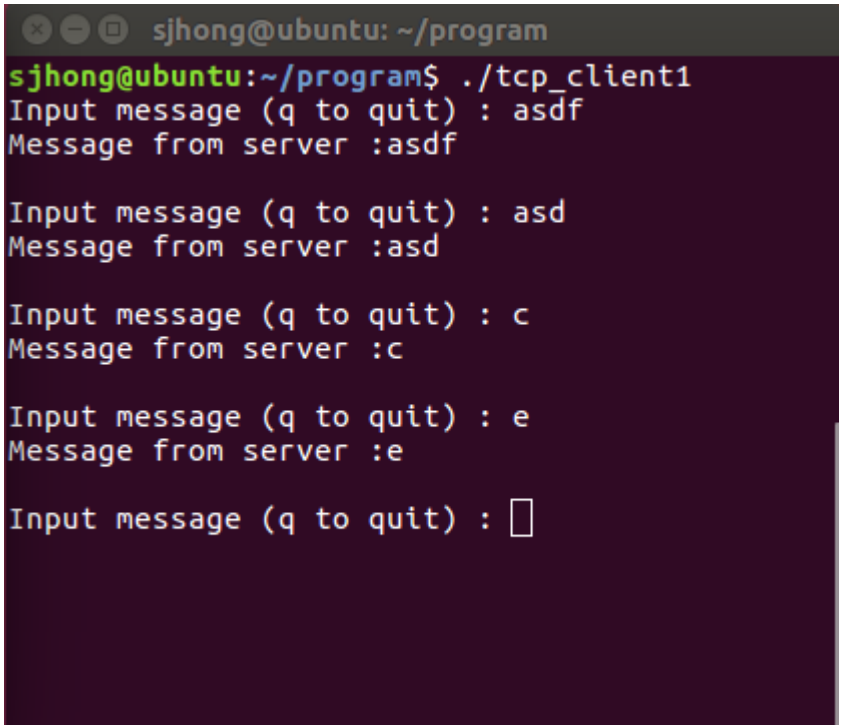


3. 소켓 프로그램(TCP) 실행

TCP 소켓 프로그램 실행

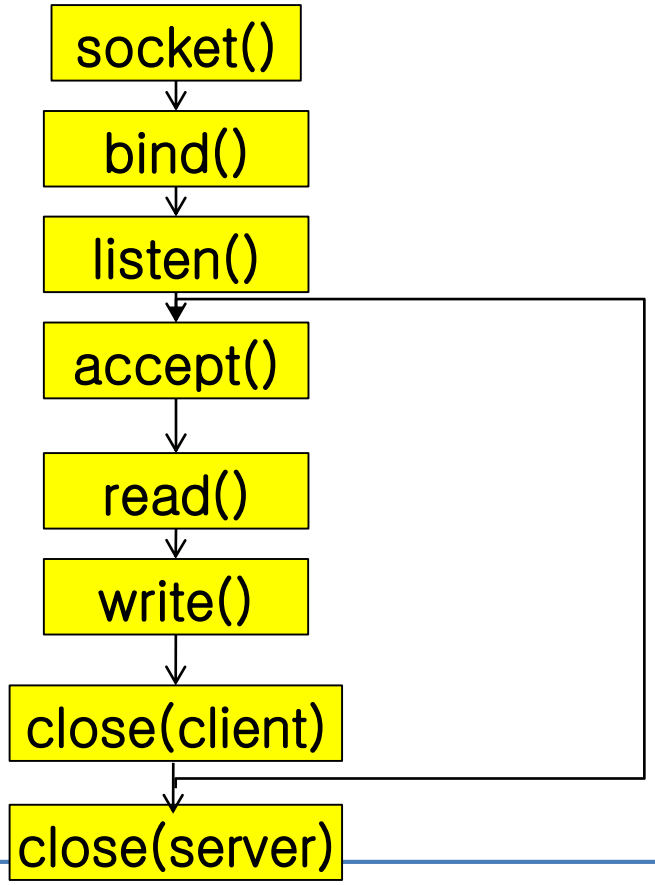
- 2개의 프로그램을 각각 터미널에 띄워서 아래와 같이 동작하는지 확인
- TCP 서버를 먼저 실행해야함





TCP 소켓 프로그램 작성(서버 프로그램)

- 아래 그림과 같이 여러 클라이언트를 수락하는 TCP소켓 echo 서버 프로그램 작성
- tcp_server2.c 파일로 작성(gedit tcp_server2.c명령 사용)



1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(서버 프로그램)

❑ **Hint : tcp_server1.c의 아래 소스 코드 부분이 반복될 수 있도록 수정**

```
        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family=AF_INET;
        serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
        serv_addr.sin_port=htons(4000);

        if(bind(serv_sock, (struct sockaddr*) &serv_addr, sizeof(serv_addr)) == -1)
                error_handling("bind() error");

        if(listen(serv_sock, 5) == -1)
                error_handling("listen() error");

        clnt_addr_size = sizeof(clnt_addr);
        clnt_sock=accept(serv_sock, (struct sockaddr*) &clnt_addr, &clnt_addr_size);

        if(clnt_sock == -1)
                error_handling("accept() error");

        while((str_len = read(clnt_sock, message, BUFSIZE)) != 0) {
                write(clnt_sock, message, str_len);
                write(1, message, str_len);
        }
        close(clnt_sock);

        return 0;
}

void error_handling(char * message)
{
        fputs(message, stderr);
        fputc('\n', stderr);
        exit(1);
}
```

(끝)

TCP 소켓 프로그램 컴파일 및 실행

- gcc –o tcp_server2 tcp_server2.c 컴파일 및 실행 확인
- 아래 그림처럼 실행되는지 확인(클라이언트는 기존 tcp_client1사용)
  - 클라이언트 종료 (q)후에도 서버가 종료되지 않는지를 확인
  - 다른 클라이언트 혹은 재접속을 하는 경우에도 계속 접속 가능한지 확인

```
sjhong@ubuntu: ~/program
^C
sjhong@ubuntu:~/program$ ^C
sjhong@ubuntu:~/program$ ./tcp_server2
^C
sjhong@ubuntu:~/program$ clear

sjhong@ubuntu:~/program$ ./tcp_server2
asdf
asdf
we
v
d
w
df
wer
df
```

```
sjhong@ubuntu: ~/program
sjhong@ubuntu:~/program$ ./tcp_client1
Input message (q to quit) : asdf
Message from server :asdf

Input message (q to quit) : asdf
Message from server :asdf

Input message (q to quit) : we
Message from server :we

Input message (q to quit) : v
Message from server :v

Input message (q to quit) : d
Message from server :d

Input message (q to quit) : w
Message from server :w

Input message (q to quit) : q
sjhong@ubuntu:~/program$ ./tcp_client1
Input message (q to quit) : df
Message from server :df
```

Q and A