

시스템 프로그래밍 실습

Assignment2-2

Class : 금 1, 2 분반
Professor : 최상호 교수님
Student ID : 2020202031
Name : 김재현

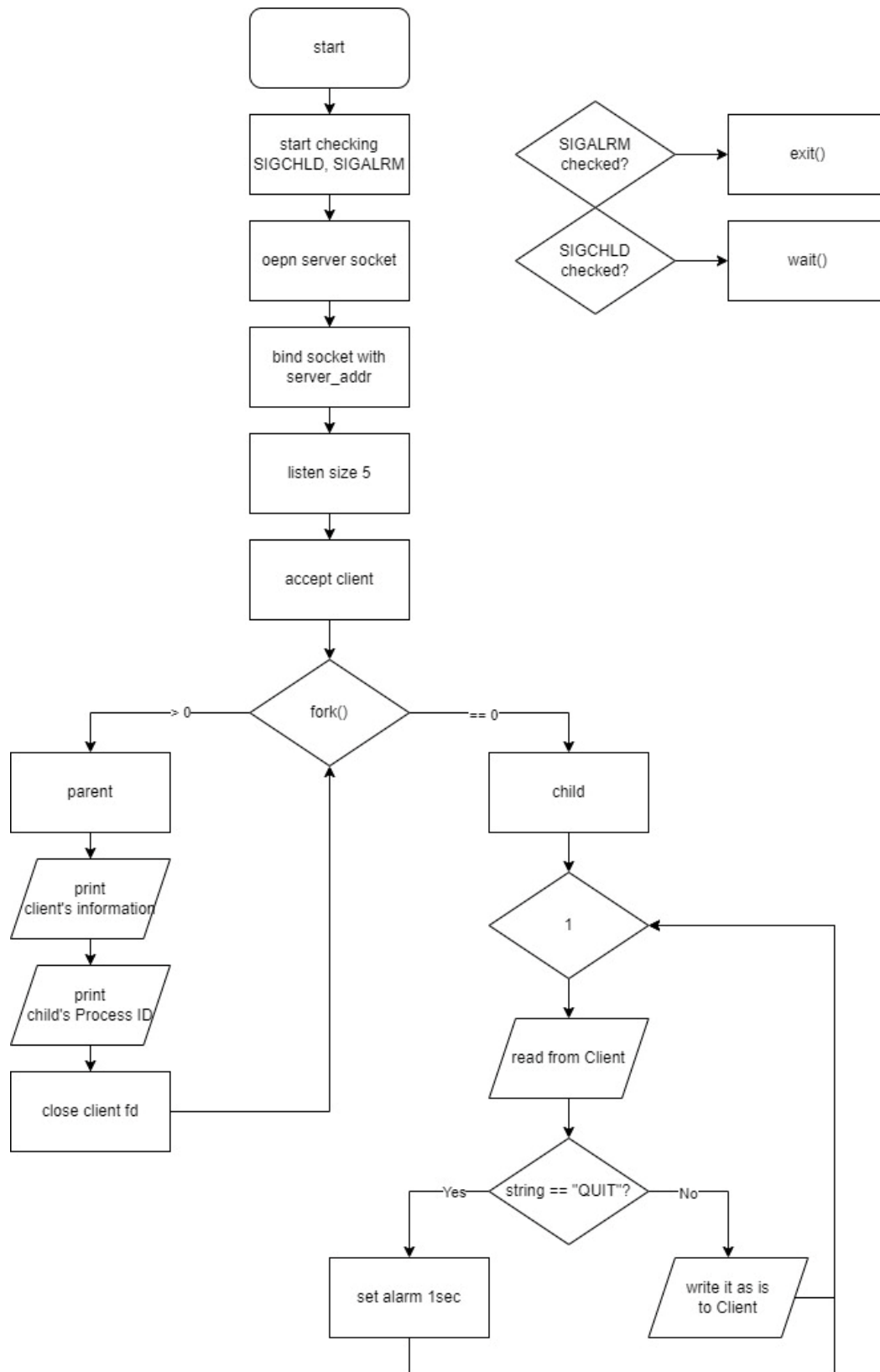
Introduction

과제 2-1에서는 Socket 을 활용한 Client & Server 간 데이터 통신을 실습했습니다.

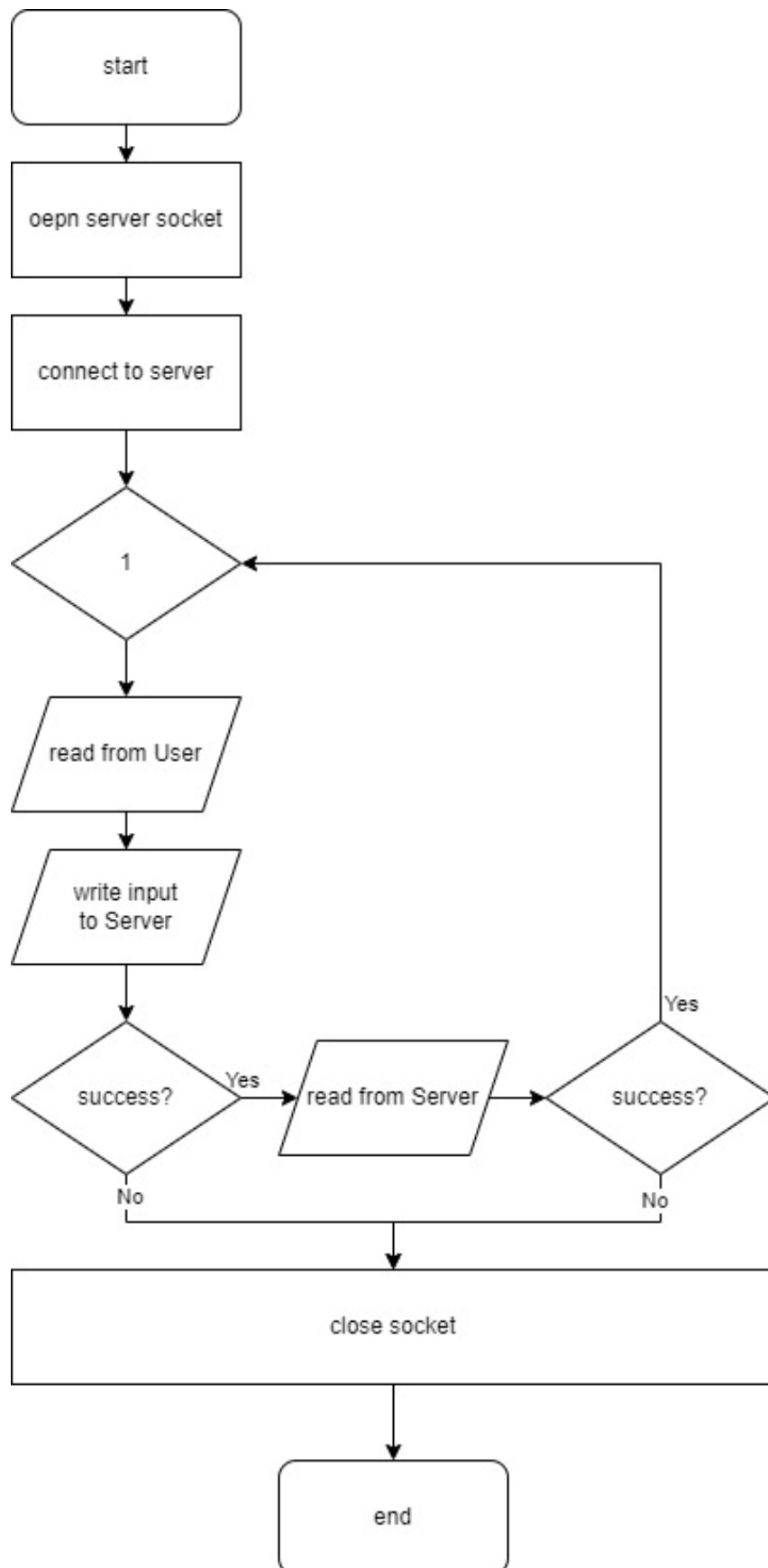
이번 과제에서는 이러한 Socket Programming 을 활용하여, Client 에서 User 로부터 문자열을 입력 받아 Server 로 보내고, Server 에서 Client 로부터 입력 받은 문자열을 그대로 다시 Client 로 보내는 기능을 구현합니다. 그러나 fork()함수를 활용하여 동시에 다수의 Client 와 통신하는 Server 를 구현합니다.

Flow chart

srv.c



cli.c



Pseudo code

srv.c

main

```
{  
  
    open socket  
  
    bind socket  
  
    listen starts  
  
    while(1)  
    {  
  
        accept client's connect  
  
        pid = fork()  
  
        if(pid >0, that means parent process)  
        {  
  
            print client's information  
  
            print child's process ID  
  
        }  
  
        else if(pid == 0, that means child process)  
        {  
  
            while(1)  
            {  
  
                read string from Client  
  
                if(string is "QUIT")  
                    set alarm 1sec  
  
                else
```

write string to Client

}

}

close client_fd

}

close server_fd

return 0

}

cli.c

main

{

 open socket

 connect to server

 while(1)

 {

 read string from USER

 write string to Server

 if(write fails)

 break

 else

 {

 read from Server

 if(read fails)

 break

 else

 print string

 }

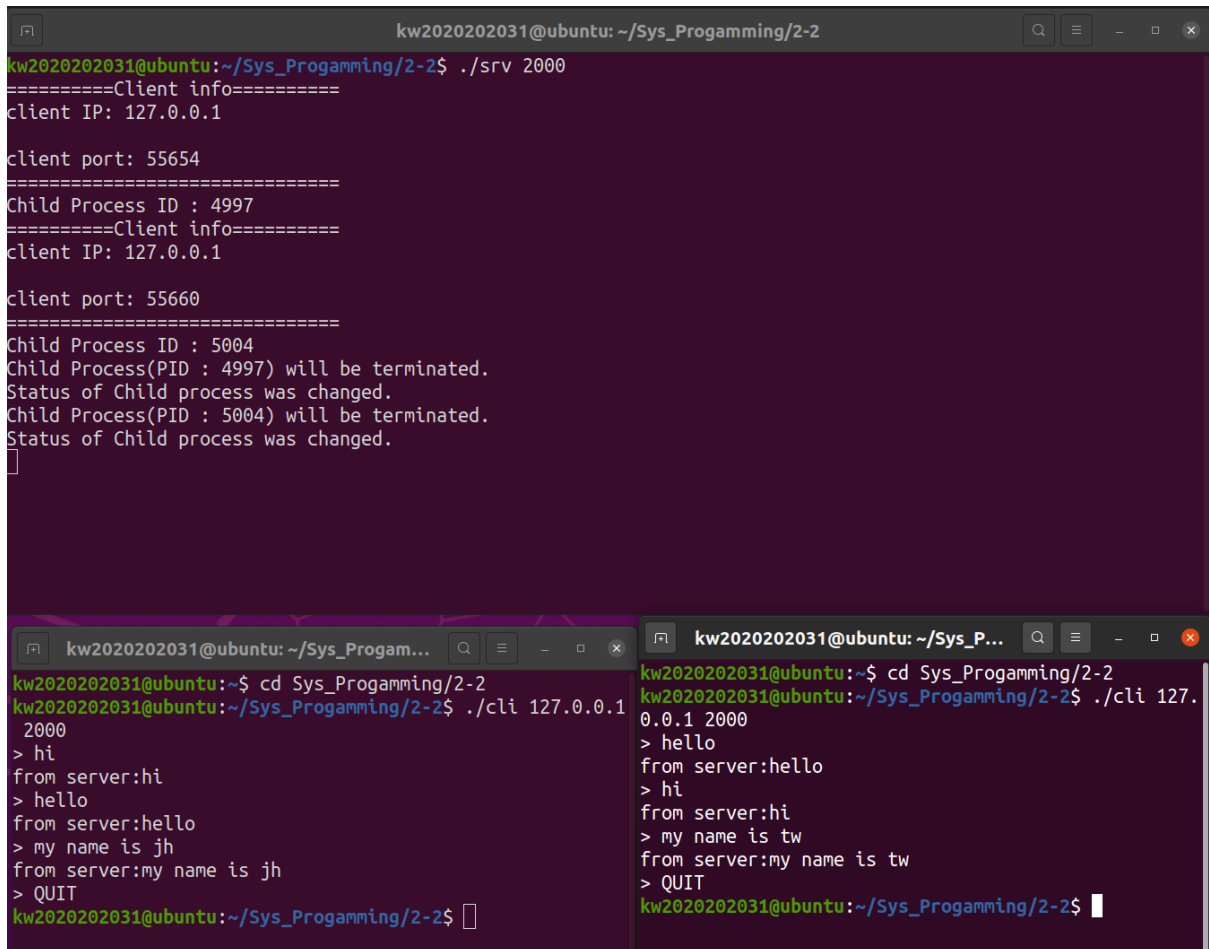
 }

 close server_fd

 return 0

}

결과화면



```
kw2020202031@ubuntu: ~/Sys_Programming/2-2
kw2020202031@ubuntu:~/Sys_Programming/2-2$ ./srv 2000
=====Client info=====
client IP: 127.0.0.1

client port: 55654
=====
Child Process ID : 4997
=====Client info=====
client IP: 127.0.0.1

client port: 55660
=====
Child Process ID : 5004
Child Process(PID : 4997) will be terminated.
Status of Child process was changed.
Child Process(PID : 5004) will be terminated.
Status of Child process was changed.
[]

kw2020202031@ubuntu:~/Sys_Programming/2-2$ cd Sys_Programming/2-2
kw2020202031@ubuntu:~/Sys_Programming/2-2$ ./cli 127.0.0.1 2000
> hi
from server:hi
> hello
from server:hello
> my name is jh
from server:my name is jh
> QUIT
kw2020202031@ubuntu:~/Sys_Programming/2-2$

kw2020202031@ubuntu:~/Sys_Programming/2-2$ cd Sys_Programming/2-2
kw2020202031@ubuntu:~/Sys_Programming/2-2$ ./cli 127.0.0.1 2000
> hello
from server:hello
> hi
from server:hi
> my name is tw
from server:my name is tw
> QUIT
kw2020202031@ubuntu:~/Sys_Programming/2-2$
```

srv 동작

server 에 차례로 Client 가 연결되고, Client 가 하나 연결될 때마다 Client 의 IP 주소, port 번호가 출력되고 동시에 Child Process 의 ID 가 출력됩니다. Client 로부터 입력 받은 문자열을 그대로 다시 Client 에게 출력하고, 만일 입력 받은 문자열이 "QUIT"이라면, Child Process 가 1 초뒤 종료됨과 동시에 Client 와의 연결이 끊어집니다.

cli 동작

server 에 연결된 후, USER 로부터 입력 받은 문자열을 server 로 출력합니다. 그리고 server 에서 출력하는 문자열을 USER 에게 출력해줍니다. 또한 server 에게 "QUIT"이라는 문자열을 입력하면, 1 초 뒤에 서버와의 연결이 끊기면서 프로세스가 종료됩니다.

고찰

이론 시간에 process 의 종료 status 와 signal 들에 대해 배웠습니다. 하지만 감이 오지 않았는데, 이번 실습을 진행하면서, signal 함수를 통해 SIGCHLD, SIGALRM 신호를 handling 하는 코드를 직접 작성해 봄으로써 process handling 에 대한 감을 잡을 수 있었습니다.

Reference

시스템프로그래밍 / 광운대학교 / 최상호 교수님 / 5.SP_-_Process_control

시스템프로그래밍 / 광운대학교 / 최상호 교수님 / 6.SP_-_Sockets

시스템프로그래밍 / 광운대학교 / 최상호 교수님 / 2024-1_SPLab_07_FTP2_2_v3

시스템프로그래밍 / 광운대학교 / 최상호 교수님 / 2024-1_SPLab_FTP_Assginment2_2_v3