시스템 프로그래밍 실습

Assignment3-1

Class : 금 1, 2 분반

Professor : 최상호 교수님

Student ID : 2020202031

Name : 김재현

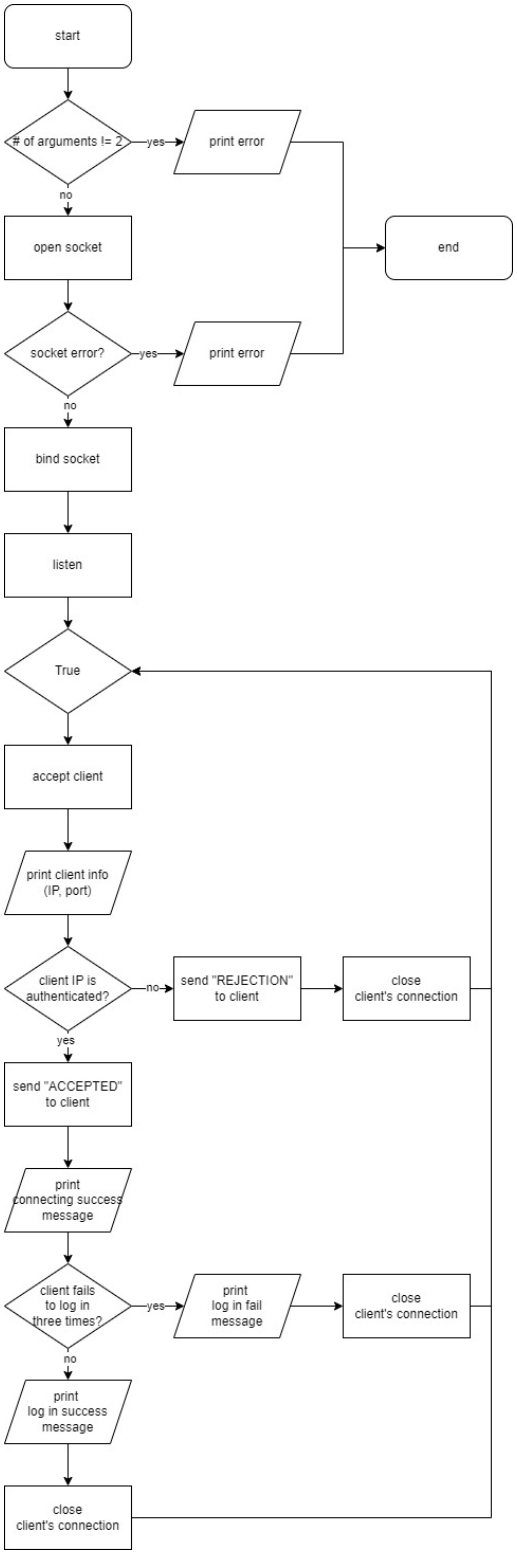
Introduction

이전 실습까지는 server 가 client 의 connect 요청을 무조건적으로 accept 하도록 설계했습니다. 하지만 이번 #3 과제부터는 server 가 connect 를 요청하는 client 의 ip 가 access.txt 에 존재하는 ip 인지, 그렇다면 client 가 입력하는 ID 와 Password 가 passwd 에 존재하는지를 검사하고 login 기능까지 구현합니다.

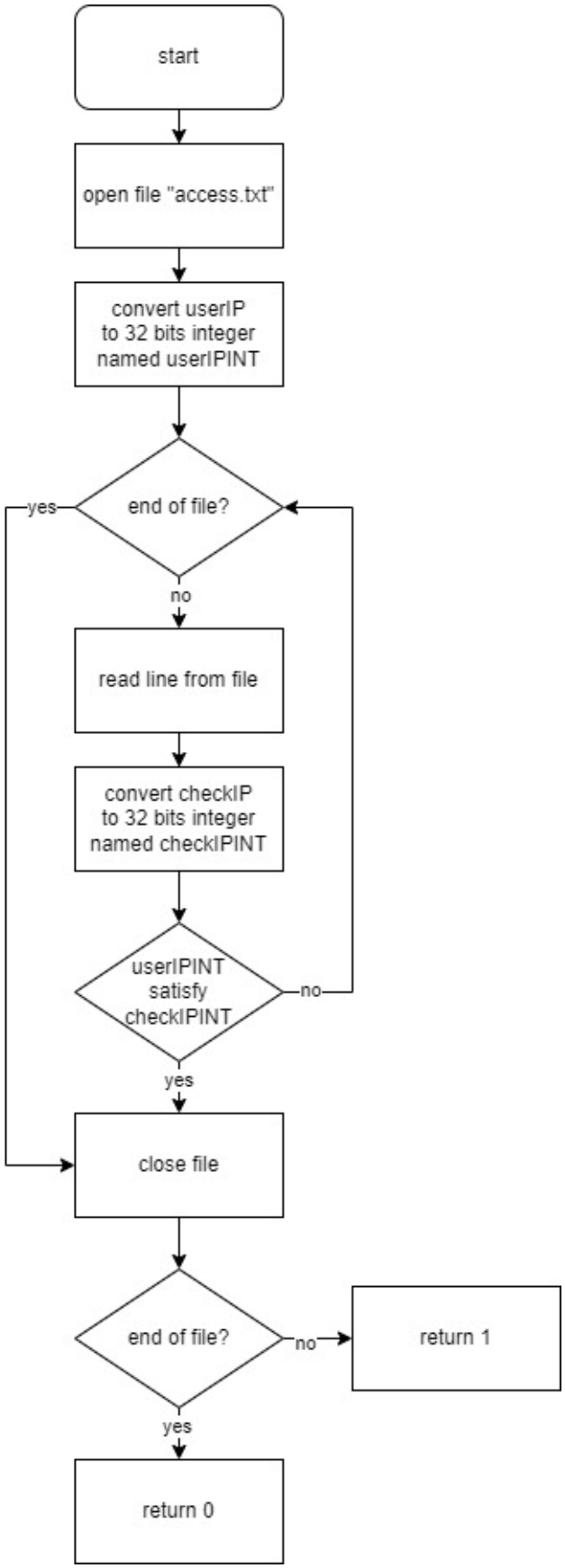
Flow chart

srv.c

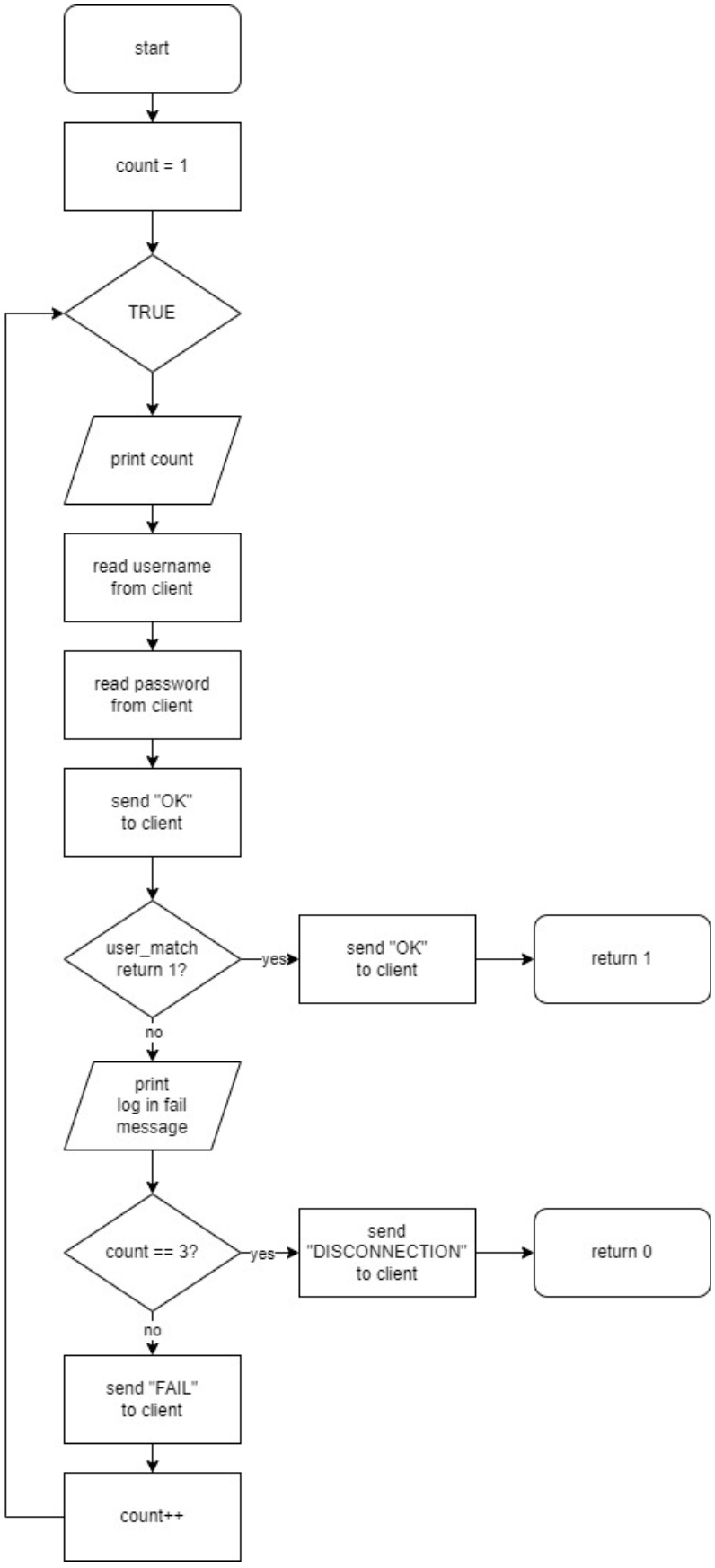
main



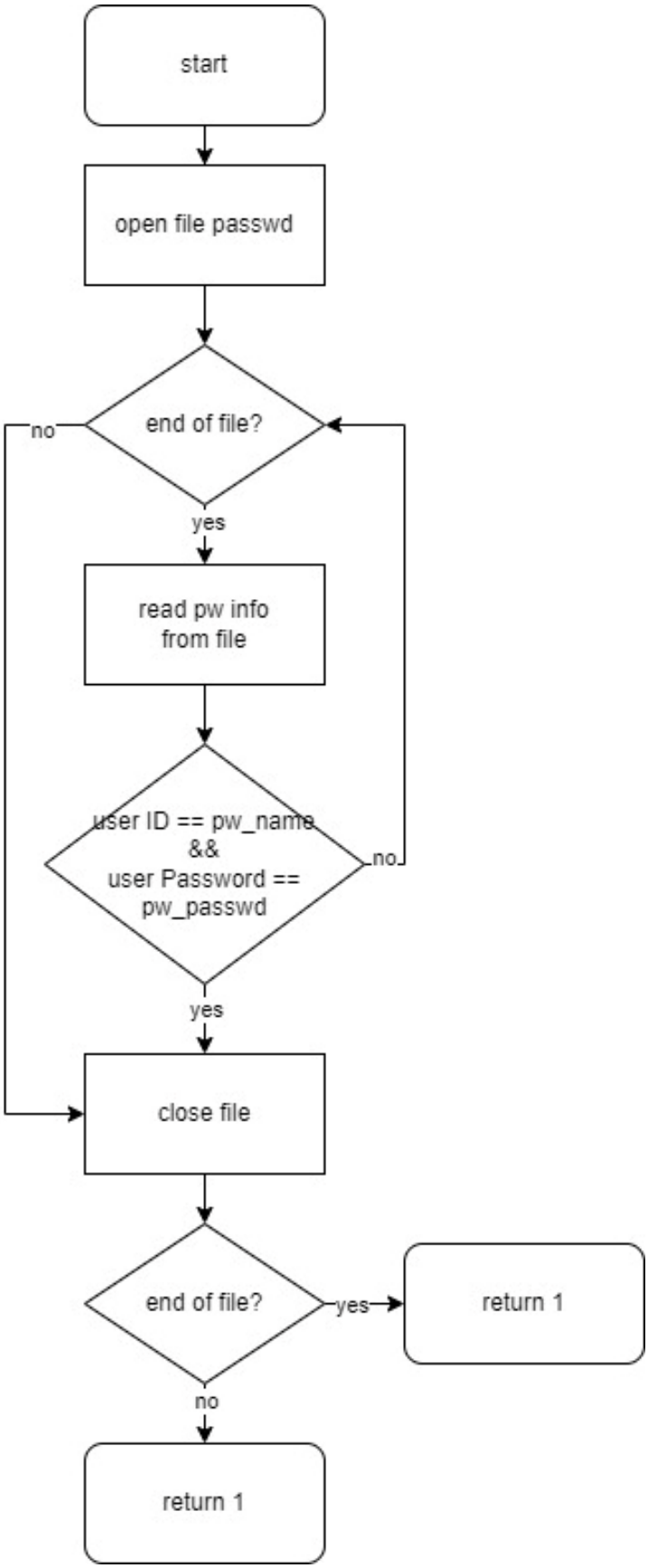
check_ip


start
open file "access.txt"
convert userIP to 32 bits integer named userIPINT
end of file?
yes
no
read line from file
convert checkIP to 32 bits integer named checkIPINT
userIPINT satisfy checkIPINT
no
yes
close file
end of file?
no
return 1
yes
return 0

log_auth

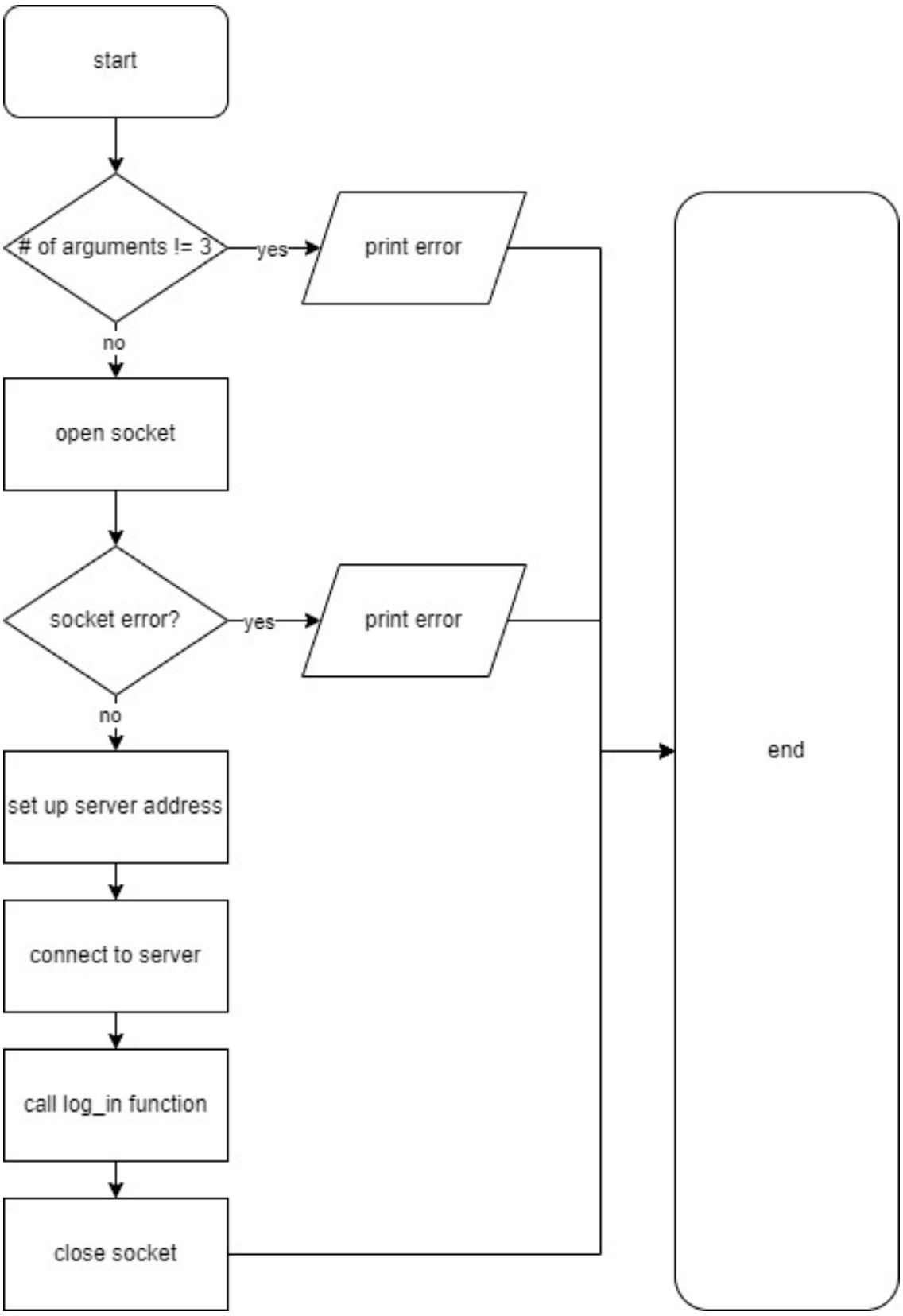

start
count = 1
TRUE
print count
read username
from client
read password
from client
send "OK"
to client
user_match
return 1?
yes
send "OK"
to client
return 1
no
print
log in fail
message
count == 3?
yes
send
"DISCONNECTION"
to client
return 0
no
send "FAIL"
to client
count++

user_match

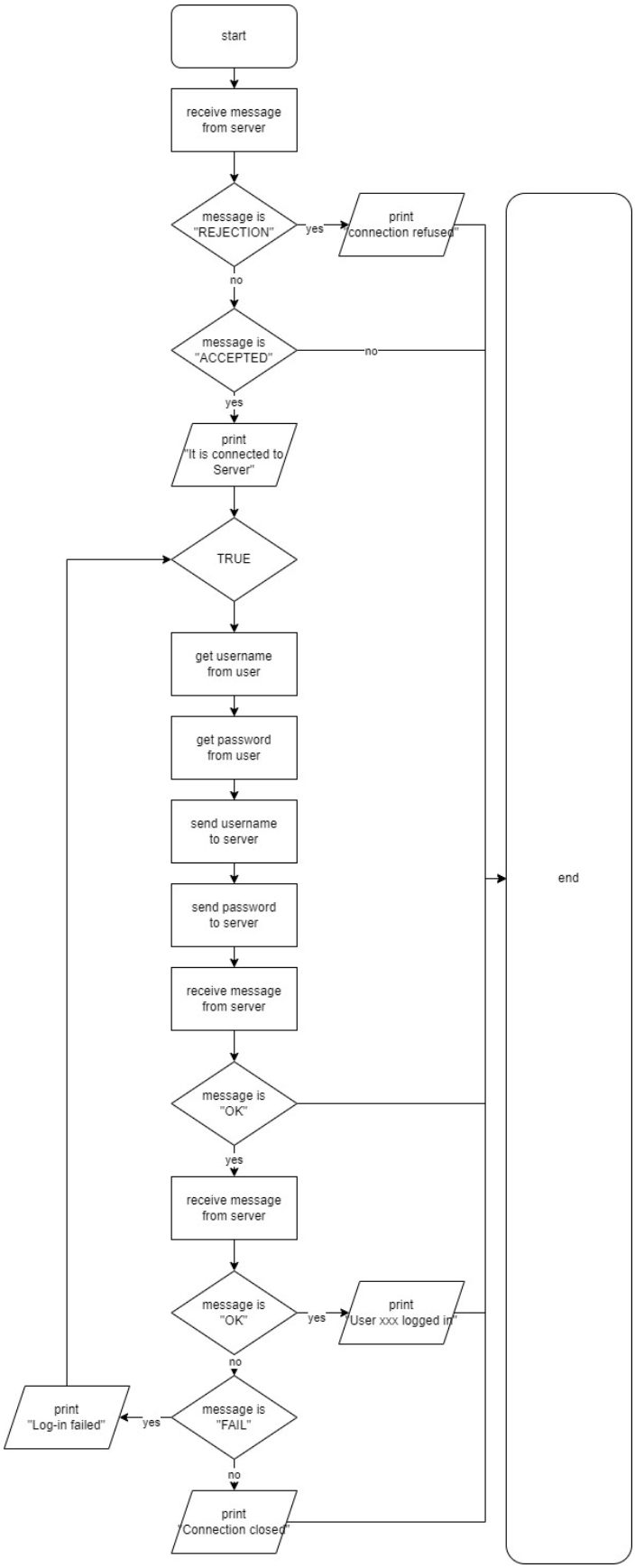

start
open file passwd
end of file?
no
yes
read pw info
from file
user ID == pw_name
&&
user Password ==
pw_passwd
no
yes
close file
end of file?
yes
return 1
no
return 1

cli.c

main


start
# of arguments != 3
yes
print error
no
open socket
socket error?
yes
print error
no
set up server address
connect to server
call log_in function
close socket
end

log_in


start
receive message from server
message is "REJECTION"
yes
print "connection refused"
no
message is "ACCEPTED"
no
yes
print "It is connected to Server"
TRUE
get username from user
get password from user
send username to server
send password to server
receive message from server
message is "OK"
yes
receive message from server
message is "OK"
yes
print "User xxx logged in"
no
message is "FAIL"
yes
print "Log-in failed"
no
print "Connection closed"
end

Pseudo code

srv.c

main

```
main
{
    If number of arguments is not 2
        Print error message "enter two arguments!"
        return -1

    Open server socket
    If failed to open socket
        Print error message "Can't open stream socket."
        return -1

    Bind socket to any available IP address and port specified by argument
    Listen for incoming connections on the socket (maximum 5 connections in
    queue)

    While (1)
    {
        Accept incoming client connection
        Print client IP address and port number

        Get client IP address
        If client IP is not authenticated
            Send "REJECTION" message to client
            Close client connection
            Continue to next iteration of loop

        Send "ACCEPTED" message to client
        Print "Client is connected"

        If client fails to log in after 3 attempts
            Print "Fail to log-in"
            Close client connection
            Continue to next iteration of loop

        Print "Success to log-in"
        Close client connection
    }

    return 0
}
```

check_ip

```
check_ip(userIP)
{
    Open file "access.txt" for reading

    Parse userIP into an array of octets (uip)

    While not end of file
    {
        Read line from file into checkIPStr
        Remove newline character from checkIPStr

        cnt = 0
        Parse checkIPStr into tokens using '.' as delimiter
        Repeat
            token = Get next token from checkIPStr
            If token is '*' or token matches uip[cnt]
                Increment cnt
            Else
                Break loop
        Until no more tokens or cnt is 4

        If cnt is 4
            Break outer loop
    }

    Close file

    If cnt is 4
        Return 1 (IP is allowed)
    Else
        Return 0 (IP is not allowed)
}
```

log_auth

```
log_auth(connfd)
{
    authCount = 1

    While (1)
    {
        Print "User is trying to log-in (authCount/3)"

        Receive username from client
```

```
        Receive password from client
        Send "OK" to client

        If user_match(username, password) == 1 (authentication successful)
            Send "OK" to client
            Break loop
        Else If user_match(username, password) == 0 (authentication failed)
            Print "Log-in failed"

            If authCount is 3 (maximum attempts reached)
                Send "DISCONNECTION" to client
                Return 0 (authentication failed)
            Else
                Send "FAIL" to client
                Increment authCount
                Continue loop
    }

    Return 1 (authentication successful)
}
```

user_match

```
user_match(user, passwd)
{
    Open file "passwd" for reading

    While not end of file
        Read user entry from file
        If client's ID matches pw_name && client's password matches pw_passwd
            Break loop

    Close file

    If not end of file
        Return 1 (authentication successful)
    Else
        Return 0 (authentication failed)
}
```

cli.c

main

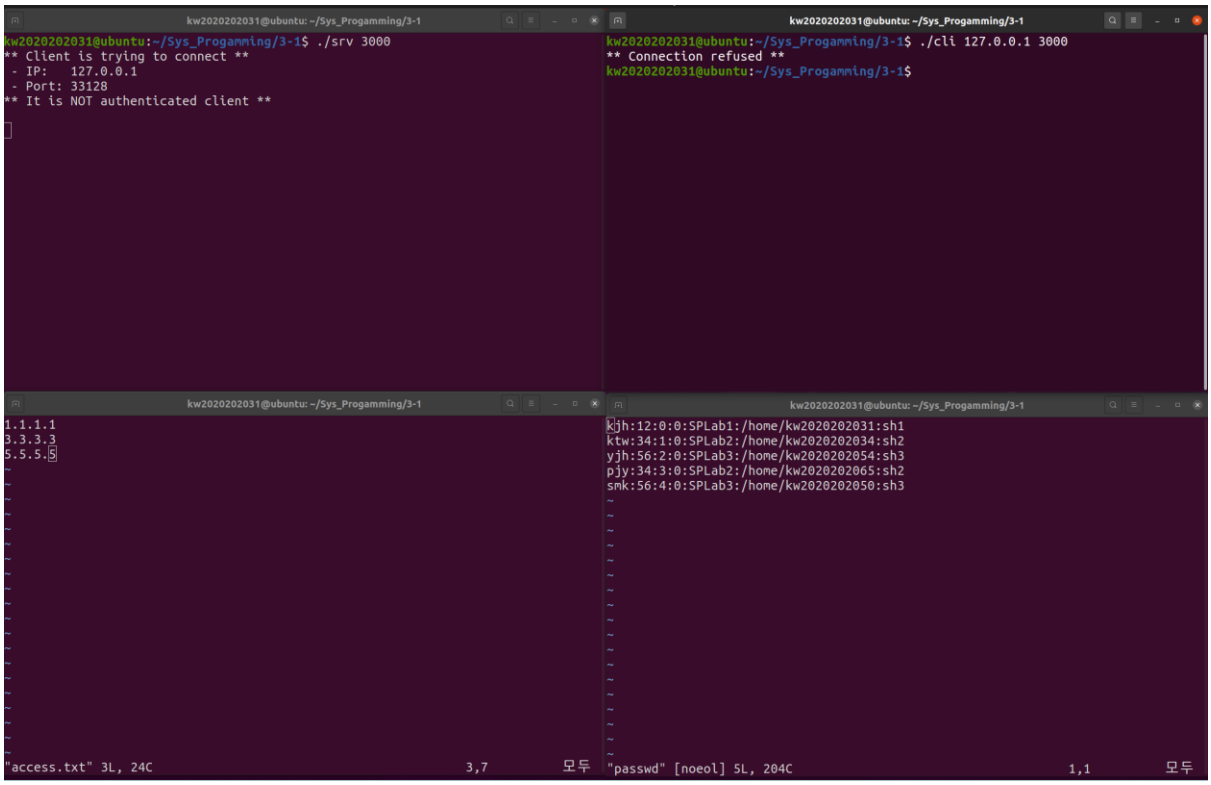
```
main
{
    If the number of arguments is not 3
        Print error message "enter three arguments!"
        Exit program with error code 1

    Open socket for client
    If failed to open socket
        Print error message "Can't open stream socket."
        Exit program with error code 1

    Set up server address with IP and port from arguments
    Connect socket to server address

    Call log_in function with socket file descriptor
    Close socket

    return 0
}
```

log_in

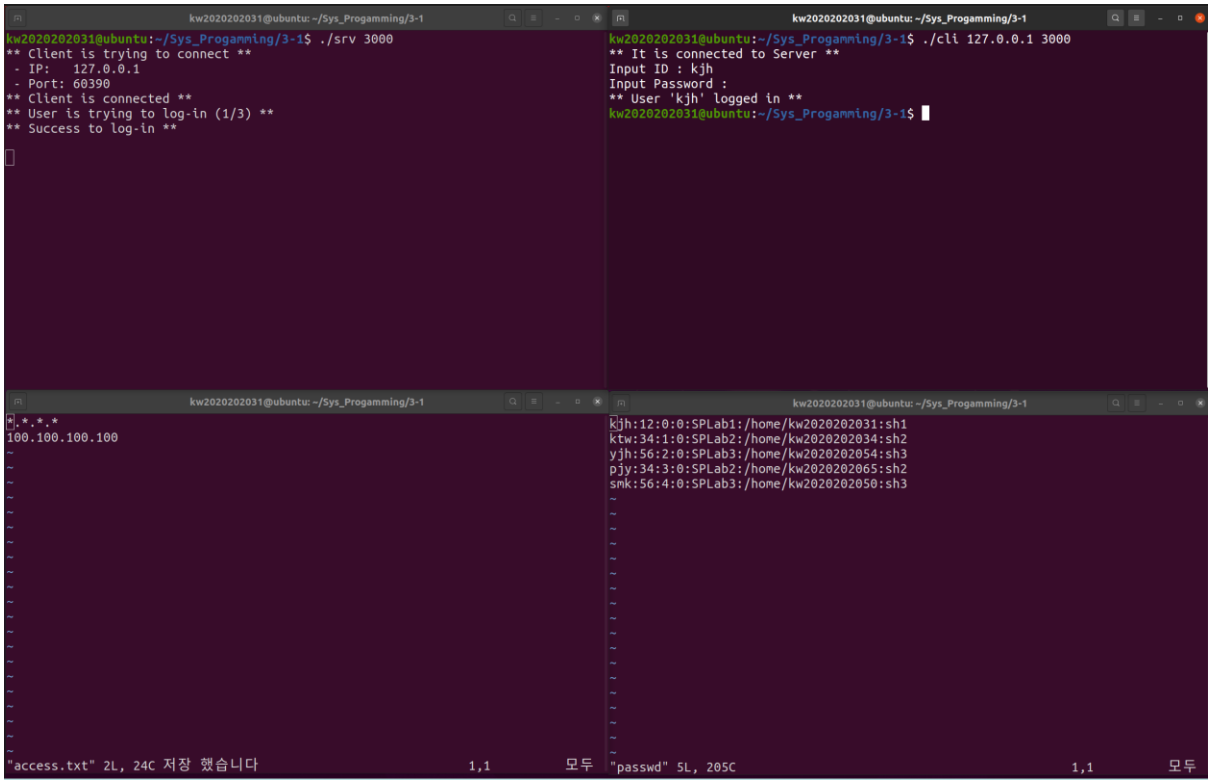
```
log_in(server_fd)
{
    Receive message from server
    If message is "REJECTION"
        Print "Connection refused"
    Else If message is "ACCEPTED"
        Print "It is connected to Server"

        While (1)
            Print "Input ID : "
            Get username from user

            Get password from user

            Send username to server
            Send password to server

            Receive message from server
            If message is "OK"
                Receive another message from server
```

```
                If message is "OK"
                    Print "User 'username' logged in"
                    Break loop
                Else If message is "FAIL"
                    Print "Log-in failed"
                    Continue loop
                Else (message is "DISCONNECTION")
                    Print "Connection closed"
                    Break loop
}
```

결과화면

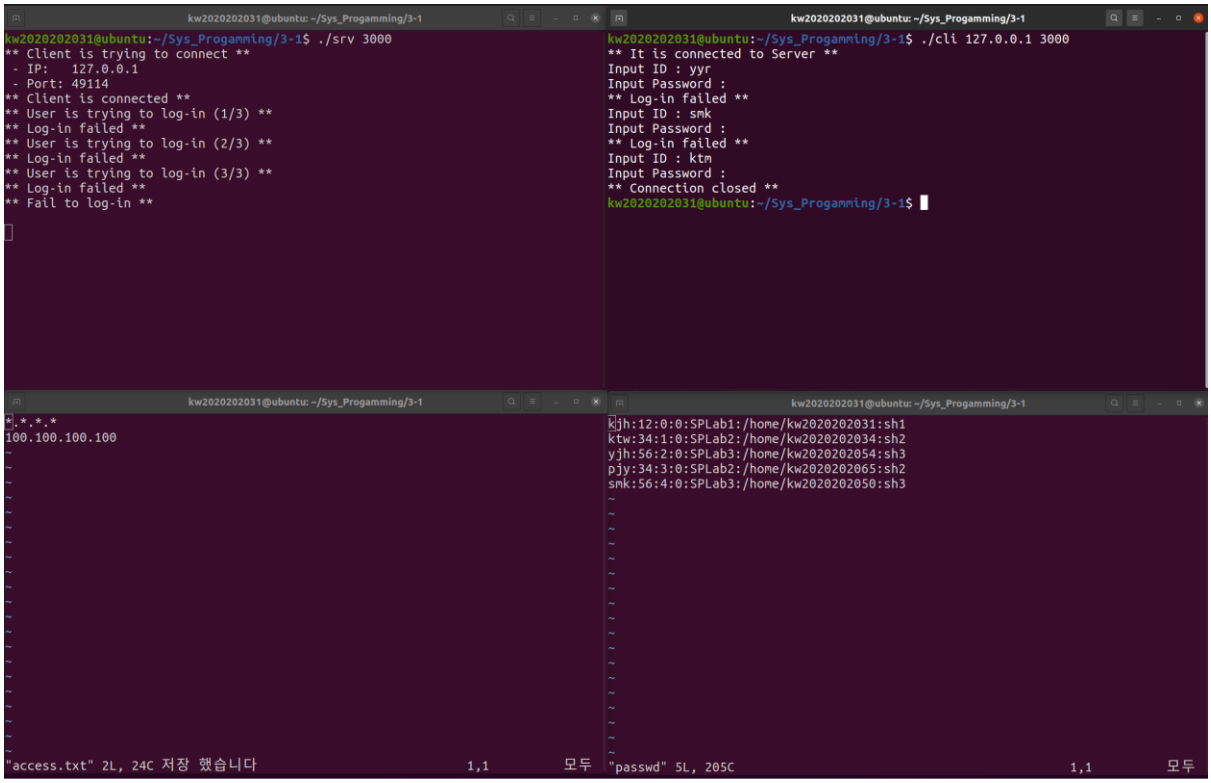


access.txt 에 client ip 주소인 127.0.0.1 이나 192.168.117.128 이 존재하지 않기 때문에, server 가 client 측의 접속 시도를 refuse 한 것을 확인할 수 있습니다.



access.txt 에 와일드카드로 이루어진 ip 주소 *.*.*.*이 존재하므로, server 가 어떠한 ip 주소의 client 의 접속이라도 모두 accept 합니다. 따라서 server 와 client 가 connect 된 것을 확인할 수 있습니다.

connect 된 후에는, client 로부터 ID 와 password 를 입력 받습니다. 위 사진에서는 ID : kjh, Password : 12 를 입력했습니다. 해당 ID 와 password 가 server 의 passwd 파일의 pw 정보 중 하나와 일치하므로 login 을 성공하는 것을 확인할 수 있습니다.



connect 된 후에는, client 로부터 ID 와 password 를 입력 받습니다.

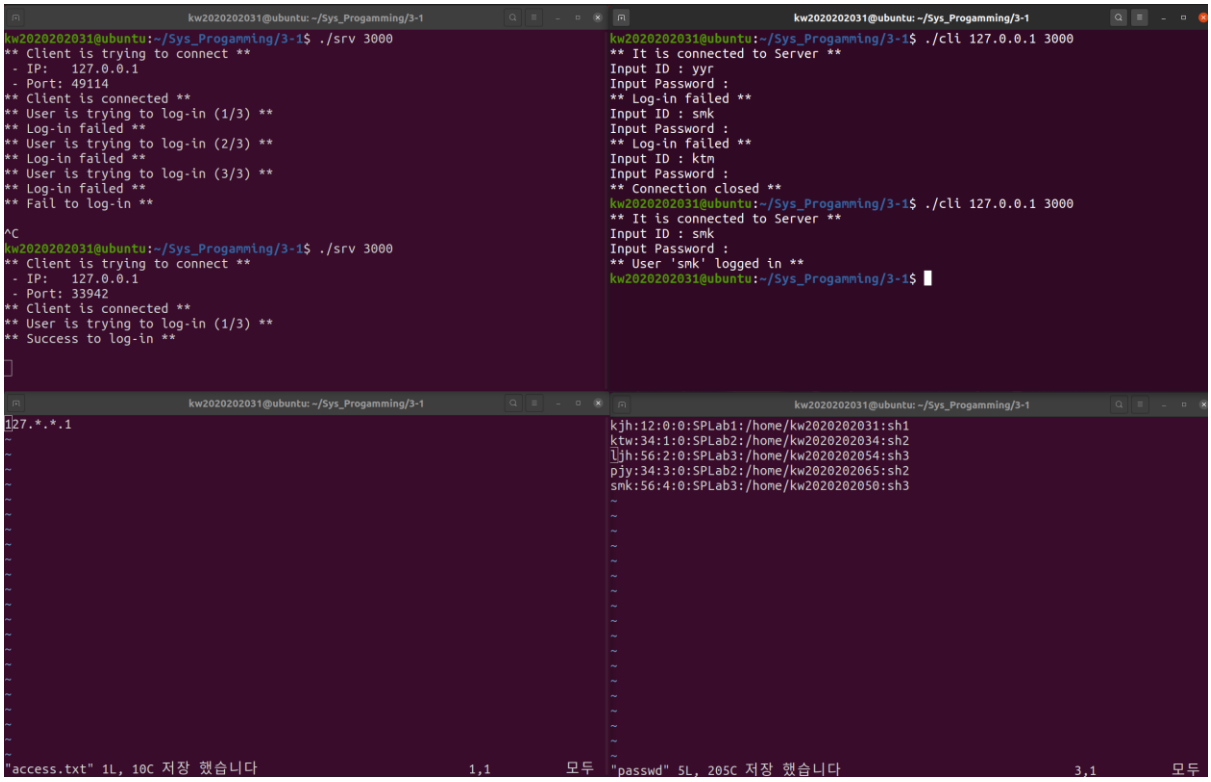
위 사진에서는

첫 번째 로그인 시도로 ID : yyr, Password : 12 를,

두 번째 로그인 시도로 ID : smk, Password : 34 를,

세 번째 로그인 시도로 ID : ktm, Password : 09 를 입력했습니다.

해당 ID 나 password 가 server 의 passwd 파일의 pw 정보와 일치하지 않으면 login 을 실패하는 것을 확인할 수 있습니다.



client 의 ip 는 127.0.0.1 이므로 127.*.*.1 에 해당하여 server 와 connect 되는 것을 확인할 수 있습니다.

고찰

fgets() 함수는 파일에서 문자열을 한 줄씩 읽어들이는 함수입니다. ₩n 문자를 기준으로 한 줄씩 읽어들인다는 사실을 인지해야합니다. 문자열을 정수로 바꿔주는 atoi 의 경우 white space 를 무시하기 때문에 문제가 없습니다만, wildcard 의 경우 ip 주소의 마지막 부분에 위치할 경우, strcmp 로 비교하면, *₩n 이기 때문에 strcmp 로 비교해도 무조건 false 입니다. 따라서 line 을 읽어들인 후 마지막 개행 문자를 ₩0 문자로 바꿔줘야 합니다.

client 의 ip 가 authentical 한지 판별하기 위해 아래와 같은 방식을 사용했습니다.

//

inet_addr() 함수는 struct sockaddr 구조체에서 ip 를 32bit 정수로 반환해주는 함수입니다. 이때 big endian 으로 저장되므로 check ip 를 access.txt 에서 읽어들인 후, '.'으로 parsing 하여, 맨 앞자리 수부터 8*i 만큼 right shift 하여 cip 에 oring 해줍니다.

그 후 cip | uip == uip 를 compare 하면 uip 가 accessible 한 ip 인지를 판단할 수 있습니다.

//

하지만 이는 client 의 ip 가 0 인 부분은 check ip 와 oring 했을 때 무조건 조건을 만족하게 되므로 실패했습니다.

따라서 ip 를 '.'으로 parsing 한 후 strcmp 를 통해 정수로 이루어진 문자열을 비교하는 형식으로 ip 가 authentical 한지 판별했습니다.

Reference

시스템프로그래밍실습 / 광운대학교 / 최상호 교수님 / 2024-1_SPLab_07_FTP3_1_v1
시스템프로그래밍실습 / 광운대학교 / 최상호 교수님 / 2024-1_SPLab_FTP_Assginment3_1_v2