시스템 프로그래밍 실습

[Assignmen3-3]

Class : D 반(실습 2 금 56)

Professor : 최상호 교수님

Student ID : 2022202104

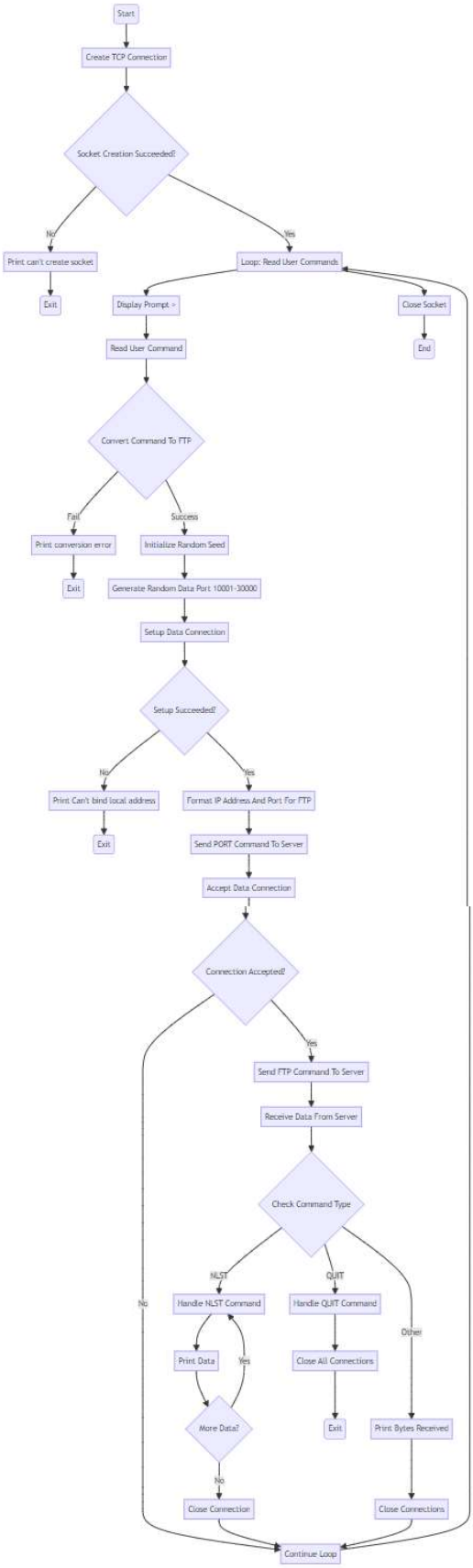
Name : 김유찬

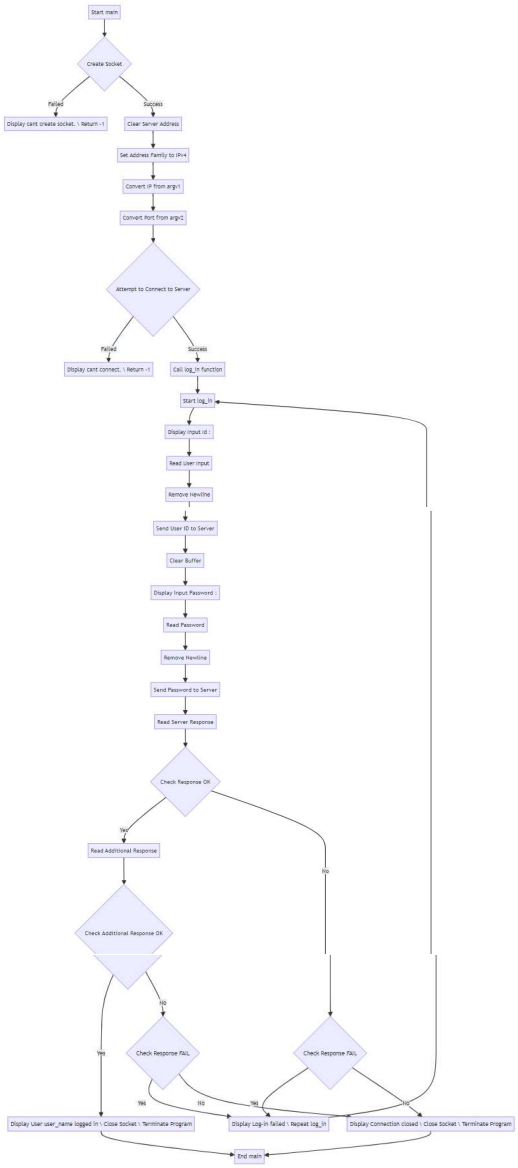
Introduction

지금까지 socket 통신을 c 언어로 코딩을 했다. user, password 를 입력해서 server 에 접속하고 그 전에 허용된 IP 인지도 확인하고 실행결과에 따라 메세지를 보낼 수 있어야 한다. 지금까지 했던 것을 모두 합쳐서 구현해보는 시간을 가질 것이다. 그리고 그 전에 구현하지 하지 않았던 파일 다운로드를 구현할 것이다. 이 때 리눅스 명령어는 get, put 인데 이를 구현해 볼 것이다. 또한 log 파일 같이 실제로 리눅스 파일에 있는 것도 만들고 이 파일을 열어 실행될 수 있도록 구현해볼 것이다.

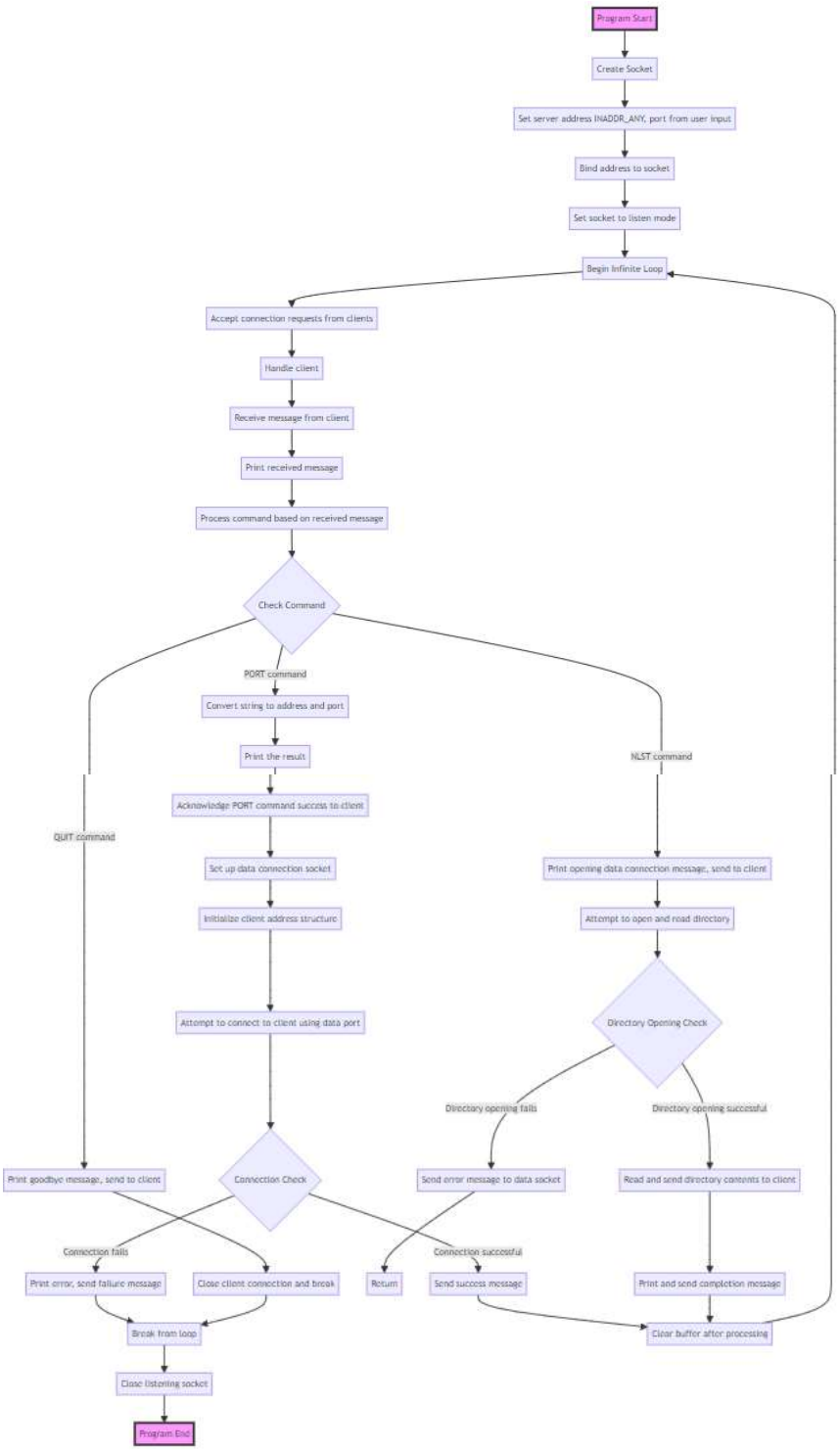
Flow chart

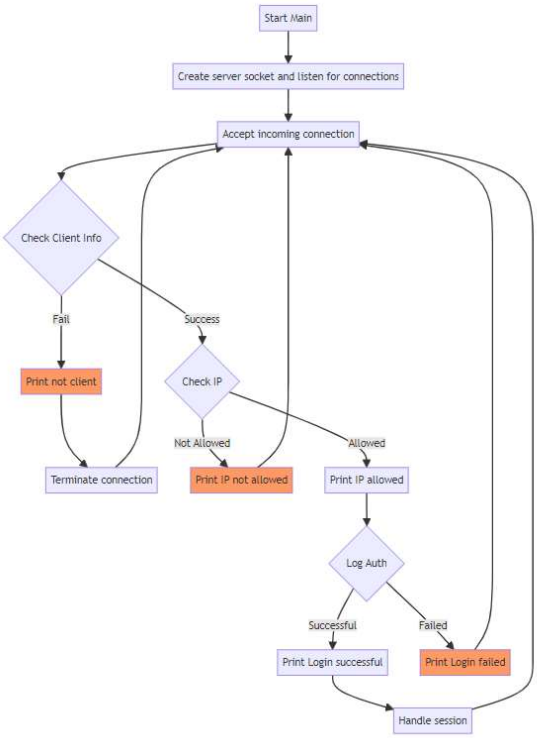
1) client


Start
Create TCP Connection
Socket Creation Succeeded?
No
Yes
Print can't create socket
Exit
Loop: Read User Commands
Display Prompt >
Read User Command
Close Socket
End
Convert Command To FTP
Fail
Success
Print conversion error
Exit
Initialize Random Seed
Generate Random Data Port 10001-30000
Setup Data Connection
Setup Succeeded?
No
Yes
Print Can't bind local address
Exit
Format IP Address And Port For FTP
Send PORT Command To Server
Accept Data Connection
Connection Accepted?
No
Yes
Send FTP Command To Server
Receive Data From Server
Check Command Type
NLST
QUIT
Other
Handle NLST Command
Print Data
Yes
More Data?
No
Close Connection
Handle QUIT Command
Close All Connections
Exit
Print Bytes Received
Close Connections
Continue Loop


Start main
Create Socket
Failed
Success
Display cant create socket. \ Return -1
Clear Server Address
Set Address Family to IPv4
Convert IP from argv1
Convert Port from argv2
Attempt to Connect to Server
Failed
Success
Display cant connect. \ Return -1
Call log_in function
Start log_in
Display input id :
Read user input
Remove Newline
Send User ID to Server
Clear Buffer
Display Input Password :
Read Password
Remove Newline
Send Password to Server
Read Server Response
Check Response OK
Yes
No
Read Additional Response
Check Additional Response OK
Yes
No
Check Response FAIL
Check Response FAIL
Yes
No
Yes
No
Display User user_name logged In \ Close Socket \ Terminate Program
Display Log-in failed \ Repeat log_in
Display Connection closed \ Close Socket \ Terminate Program
End main

2) server


Program Start
Create Socket
Set server address INADDR_ANY, port from user input
Bind address to socket
Set socket to listen mode
Begin Infinite Loop
Accept connection requests from clients
Handle client
Receive message from client
Print received message
Process command based on received message
Check Command
PORT command
Convert string to address and port
Print the result
Acknowledge PORT command success to client
QUIT command
Set up data connection socket
Initialize client address structure
Attempt to connect to client using data port
NLST command
Print opening data connection message, send to client
Attempt to open and read directory
Directory Opening Check
Directory opening fails
Directory opening successful
Print goodbye message, send to client
Connection Check
Send error message to data socket
Read and send directory contents to client
Connection fails
Connection successful
Print error, send failure message
Close client connection and break
Return
Send success message
Print and send completion message
Break from loop
Clear buffer after processing
Close listening socket
Program End



Pseudo code

1) client

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define MAX_BUF 1024
#define CONT_PORT 20001

void log_in(int sockfd) {
    repeat {
        clear buffer

        display "Input Id : "
        read user input into buffer
        remove newline character from buffer if present
        send buffer to server

        clear buffer

        display "Input Password : "
        read password into buffer
        remove newline character from buffer if present
        send buffer to server

        read server response into buffer
        if (buffer equals "OK") {
            read additional server response into buffer
            if (buffer equals "OK") {
                display "** User '[user_name]' logged in **"
                close socket
                terminate program
            } else if (buffer equals "FAIL") {
                display "** Log-in failed **"
            } else {
```

```
                display "** Connection closed **"
                close socket
                terminate program
            }
        }
    } until false
}

int main() {
    sockfd = create socket(ipv4, stream)
    if (sockfd < 0) {
        display "can't create socket."
        return -1
    }

    clear servaddr

    servaddr.family = ipv4
    servaddr.ip = convert ip argv[1]
    servaddr.port = convert port argv[2]

    if (connect socket to server fails) {
        display "can't connect."
        return -1
    }

    call log_in(sockfd)

    close sockfd
    return 0
}
```

```
FTPClientSession(IP address, port number)
{
    sockfd = CreateTCPConnection(IP address, port number);
    if (socket creation fails) {
        Print "can't create socket";
        return -1;
    }

    while (true) {
        Display prompt "> ";
        Read user command into buffer;

        if (ConvertCommandToFTP(buffer, cmd_buff) fails) {
            Print "conversion error";
            Exit;
        }

        Initialize random seed;
        data_port = Generate a random port between 10001 and 30000;

        listenfd = SetupDataConnection(data_port);
        if (SetupDataConnection fails) {
            Print "Can't bind local address";
            return 1;
        }

        hostport = FormatIPAddressAndPortForFTP(temp.sin_addr.s_addr, servaddr.sin_port);
        Print "converting to ", hostport;

        SendPORTCommandToServer(sockfd, hostport);

        connfd = AcceptDataConnection(listenfd);
        if (connection acceptance fails) {
            Continue;
        }

        SendFTPCommandToServer(sockfd, cmd_buff);
        count = ReceiveDataFromServer(sockfd, buffer);

        if (Command equals "NLST") {
            do {
                count += ReceiveDataFromServer(sockfd, buffer);
                Print buffer;
            } while (additional data available);
        } else if (Command equals "QUIT") {
            HandleQuitCommand(sockfd, connfd, listenfd);
            return 0;
        }

        Print "OK. ", count, " bytes is received";
        CloseConnection(connfd, listenfd);
    }
    Close sockfd;
}
```

2) server

```
Program Start

Create Socket
Set server address (INADDR_ANY, port from user input)
Bind address to socket
Set socket to listen mode

Begin Infinite Loop
    Accept connection requests from clients
    Handle client:
        Repeat
            Receive message from client
            Print received message

            Process command based on received message:
                If ("PORT" command) {
                    Convert string to address and port, print the result
                    Acknowledge PORT command success to client
                    Set up data connection socket
                    Initialize client address structure
                    Attempt to connect to client using data port
                    If (connection fails) {
                        Print error, send failure message, and break
                    }
                    If (connection successful) {
                        Send success message
                    }
                }

                If ("NLST" command) {
                    Print opening data connection message, send to client
                    Attempt to open and read directory
                    If (directory opening fails) {
                        Send error message to data socket
                        Return
                    }
                    Read and send directory contents to client
                    Print and send completion message
                }

                If ("QUIT" command) {
                    Print goodbye message, send to client
                    Close client connection and break
                }

            Clear buffer after processing

Close listening socket
Program End

Function convert_str_to_addr {
    Allocate memory for address storage
    Extract IP and port parts using sscanf
    If (format is incorrect) {
        Print error and return NULL
    }
    Format IP address into dot notation
    Calculate port number (p1 * 256 + p2)
    Return formatted IP address
}
```

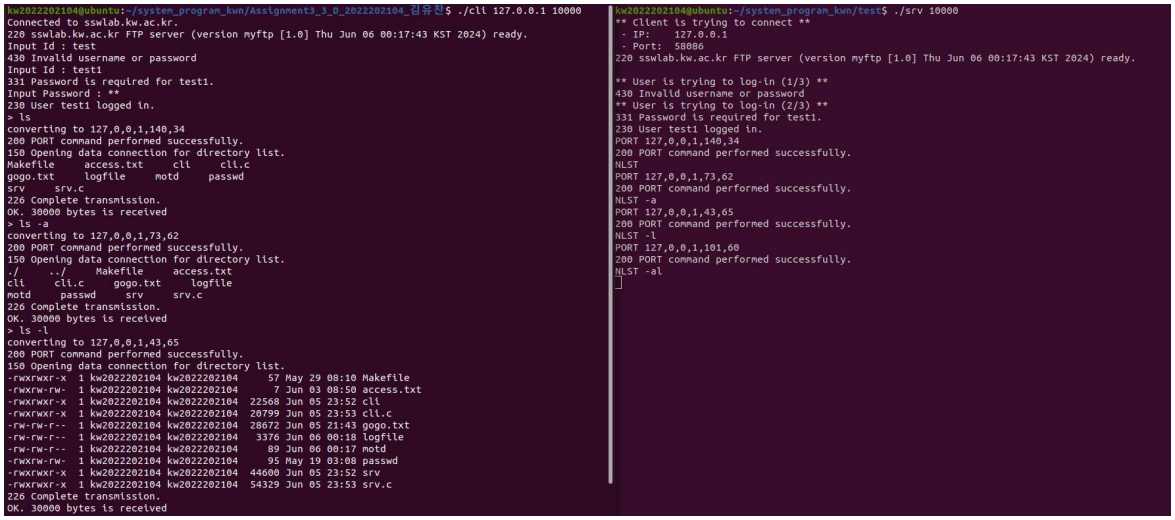
```c
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <stdlib.h>
#include <dirent.h>
#include <pwd.h>
#include <grp.h>
#include <unistd.h>
#include <sys/stat.h>
#include <time.h>

int main() {
    create server socket and listen for connections
    for each incoming connection {
        if (client_info fails) {
            print "not client"
            terminate connection
        }

        if (not allowed IP) {
            print "IP not allowed"
            continue to next iteration
        } else {
            print "IP allowed"
        }

        if (log_auth successful) {
            print "Login successful"
        } else {
            print "Login failed"
            continue to next iteration
```

```c
        }
    }
}

int client_info(pointer to client address) {
    print "client trying to connect"
    if (family is not IPv4) {
        return failure
    }
    print IP and port
    return success
}

int user_match(username, password) {
    open and read password file
    if (file not open) {
        return failure
    }
    for each entry in password file {
        if (username and password match) {
            return success
        }
    }
    return failure
}

int log_auth(connection file descriptor) {
    for up to 3 attempts {
        read username and password from connection
        if (user_match is successful) {
            send "OK"
            return success
        } else {
```

```c
                send "FAIL"
                if (3 attempts exceeded) {
                    send "FAILFAIL"
                    return failure
                }
            }
        }
        return success
    }
```

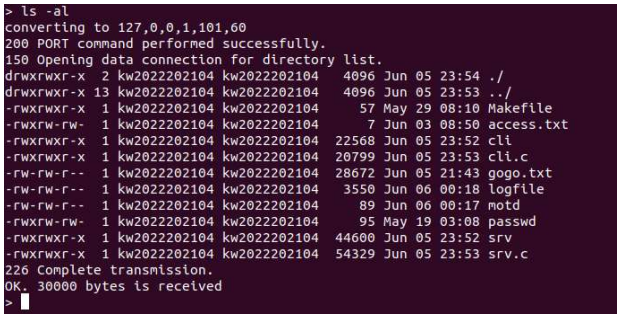
결과화면

- 왼쪽이 client 이고 오른쪽이 server 이다.

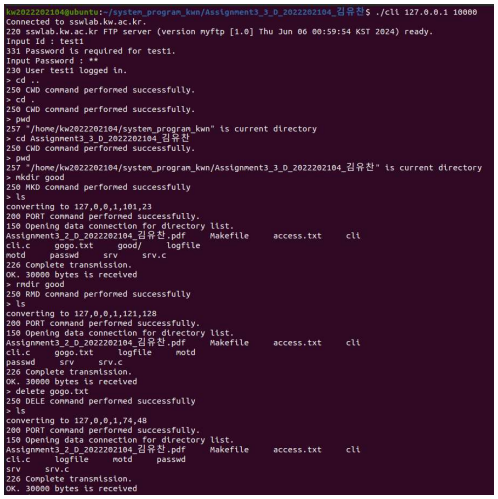


```
kw2022202104@ubuntu:~/system_program_kwn/Assignment3_3_D_2022202104_김유찬$ ./cli 127.0.0.1 10000
Connected to sswlab.kw.ac.kr.
220 sswlab.kw.ac.kr FTP server (version myftp [1.0] Thu Jun 06 00:17:43 KST 2024) ready.
Input Id : test
430 Invalid username or password
Input Id : test1
331 Password is required for test1.
Input Password : **
230 User test1 logged in.
> ls
converting to 127,0,0,1,140,34
200 PORT command performed successfully.
150 Opening data connection for directory list.
Makefile     access.txt     cli     cli.c
gogo.txt     logfile     motd     passwd
srv     srv.c
226 Complete transmission.
OK. 30000 bytes is received
> ls -a
converting to 127,0,0,1,73,62
200 PORT command performed successfully.
150 Opening data connection for directory list.
./     ../     Makefile     access.txt
cli     cli.c     gogo.txt     logfile
motd     passwd     srv     srv.c
226 Complete transmission.
OK. 30000 bytes is received
> ls -l
converting to 127,0,0,1,43,65
200 PORT command performed successfully.
150 Opening data connection for directory list.
-rwxrwxr-x  1 kw2022202104 kw2022202104     57 May 29 08:10 Makefile
-rwxrw-rw-  1 kw2022202104 kw2022202104      7 Jun 03 08:50 access.txt
-rwxrwxr-x  1 kw2022202104 kw2022202104  22568 Jun 05 23:52 cli
-rwxrwxr-x  1 kw2022202104 kw2022202104  20799 Jun 05 23:53 cli.c
-rw-rw-r--  1 kw2022202104 kw2022202104  28672 Jun 05 21:43 gogo.txt
-rw-rw-r--  1 kw2022202104 kw2022202104   3376 Jun 06 00:18 logfile
-rw-rw-r--  1 kw2022202104 kw2022202104     89 Jun 06 00:17 motd
-rwxrw-rw-  1 kw2022202104 kw2022202104     95 May 19 03:08 passwd
-rwxrwxr-x  1 kw2022202104 kw2022202104  44600 Jun 05 23:52 srv
-rwxrwxr-x  1 kw2022202104 kw2022202104  54329 Jun 05 23:53 srv.c
226 Complete transmission.
OK. 30000 bytes is received
```

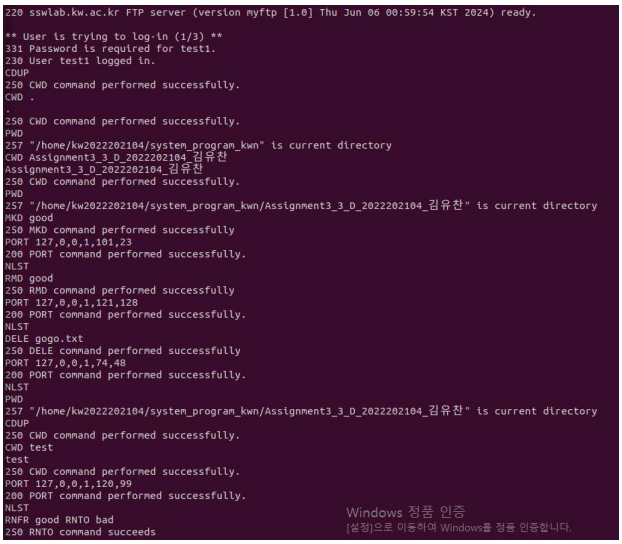
```
kw2022202104@ubuntu:~/system_program_kwn/test$ ./srv 10000
** Client is trying to connect **
 - IP:    127.0.0.1
 - Port:  58086
220 sswlab.kw.ac.kr FTP server (version myftp [1.0] Thu Jun 06 00:17:43 KST 2024) ready.

** User is trying to log-in (1/3) **
430 Invalid username or password
** User is trying to log-in (2/3) **
331 Password is required for test1.
230 User test1 logged in.
PORT 127,0,0,1,140,34
200 PORT command performed successfully.
NLST
PORT 127,0,0,1,73,62
200 PORT command performed successfully.
NLST -a
PORT 127,0,0,1,43,65
200 PORT command performed successfully.
NLST -l
PORT 127,0,0,1,101,60
200 PORT command performed successfully.
NLST -al
```



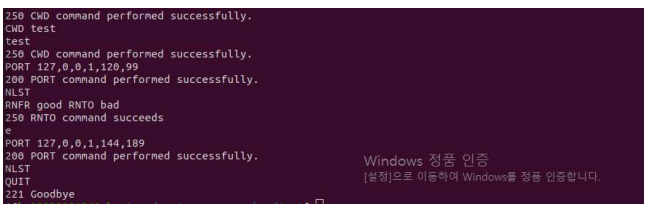
```
> ls -al
converting to 127,0,0,1,101,60
200 PORT command performed successfully.
150 Opening data connection for directory list.
drwxrwxr-x  2 kw2022202104 kw2022202104   4096 Jun 05 23:54 ./
drwxrwxr-x 13 kw2022202104 kw2022202104   4096 Jun 05 23:53 ../
-rwxrwxr-x  1 kw2022202104 kw2022202104     57 May 29 08:10 Makefile
-rwxrw-rw-  1 kw2022202104 kw2022202104      7 Jun 03 08:50 access.txt
-rwxrwxr-x  1 kw2022202104 kw2022202104  22568 Jun 05 23:52 cli
-rwxrwxr-x  1 kw2022202104 kw2022202104  20799 Jun 05 23:53 cli.c
-rw-rw-r--  1 kw2022202104 kw2022202104  28672 Jun 05 21:43 gogo.txt
-rw-rw-r--  1 kw2022202104 kw2022202104   3550 Jun 06 00:18 logfile
-rw-rw-r--  1 kw2022202104 kw2022202104     89 Jun 06 00:17 motd
-rwxrw-rw-  1 kw2022202104 kw2022202104     95 May 19 03:08 passwd
-rwxrwxr-x  1 kw2022202104 kw2022202104  44600 Jun 05 23:52 srv
-rwxrwxr-x  1 kw2022202104 kw2022202104  54329 Jun 05 23:53 srv.c
226 Complete transmission.
OK. 30000 bytes is received
>
```



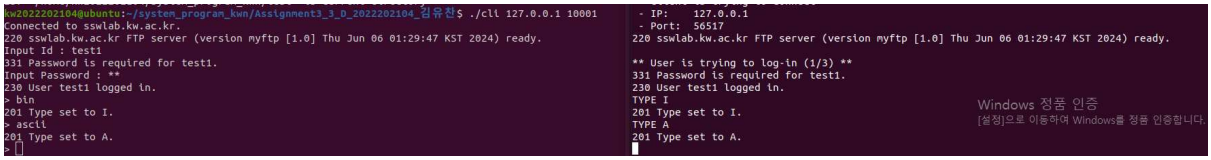
```
kw2022202104@ubuntu:~/system_program_kwn/Assignment3_3_D_2022202104_김유찬$ ./cli 127.0.0.1 10000
Connected to sswlab.kw.ac.kr.
220 sswlab.kw.ac.kr FTP server (version myftp [1.0] Thu Jun 06 00:59:54 KST 2024) ready.
Input Id : test1
331 Password is required for test1.
Input Password : **
230 User test1 logged in.
> cd ..
250 CWD command performed successfully.
> cd .
250 CWD command performed successfully.
> pwd
257 "/home/kw2022202104/system_program_kwn" is current directory
> cd Assignment3_3_D_2022202104_김유찬
250 CWD command performed successfully.
> pwd
257 "/home/kw2022202104/system_program_kwn/Assignment3_3_D_2022202104_김유찬" is current directory
> mkdir good
250 MKD command performed successfully
> ls
converting to 127,0,0,1,101,23
200 PORT command performed successfully.
150 Opening data connection for directory list.
Assignment3_2_D_2022202104_김유찬.pdf     Makefile     access.txt     cli
cli.c     gogo.txt     good/     logfile
motd     passwd     srv     srv.c
226 Complete transmission.
OK. 30000 bytes is received
> rmdir good
250 RMD command performed successfully
> ls
converting to 127,0,0,1,121,128
200 PORT command performed successfully.
150 Opening data connection for directory list.
Assignment3_2_D_2022202104_김유찬.pdf     Makefile     access.txt     cli
cli.c     gogo.txt     logfile     motd
passwd     srv     srv.c
226 Complete transmission.
OK. 30000 bytes is received
> delete gogo.txt
250 DELE command performed successfully
> ls
converting to 127,0,0,1,74,48
200 PORT command performed successfully.
150 Opening data connection for directory list.
Assignment3_2_D_2022202104_김유찬.pdf     Makefile     access.txt     cli
cli.c     logfile     motd     passwd
srv     srv.c
226 Complete transmission.
OK. 30000 bytes is received
```



```
220 sswlab.kw.ac.kr FTP server (version myftp [1.0] Thu Jun 06 00:59:54 KST 2024) ready.

** User is trying to log-in (1/3) **
331 Password is required for test1.
230 User test1 logged in.
CDUP
250 CWD command performed successfully.
CWD .
.
250 CWD command performed successfully.
PWD
257 "/home/kw2022202104/system_program_kwn" is current directory
CWD Assignment3_3_D_2022202104_김유찬
Assignment3_3_D_2022202104_김유찬
250 CWD command performed successfully.
PWD
257 "/home/kw2022202104/system_program_kwn/Assignment3_3_D_2022202104_김유찬" is current directory
MKD good
250 MKD command performed successfully
PORT 127,0,0,1,101,23
200 PORT command performed successfully.
NLST
RMD good
250 RMD command performed successfully
PORT 127,0,0,1,121,128
200 PORT command performed successfully.
NLST
DELE gogo.txt
250 DELE command performed successfully
PORT 127,0,0,1,74,48
200 PORT command performed successfully.
NLST
PWD
257 "/home/kw2022202104/system_program_kwn/Assignment3_3_D_2022202104_김유찬" is current directory
CDUP
250 CWD command performed successfully.
CWD test
test
250 CWD command performed successfully.
PORT 127,0,0,1,120,99
200 PORT command performed successfully.
NLST
RNFR good RNTO bad
250 RNTO command succeeds
```

```
250 CWD command performed successfully.
CWD test
test
250 CWD command performed successfully.
PORT 127,0,0,1,120,99
200 PORT command performed successfully.
NLST
RNFR good RNTO bad
250 RNTO command succeeds
e
PORT 127,0,0,1,144,189
200 PORT command performed successfully.
NLST
QUIT
221 Goodbye
```



- logfile

```
Thu Jun 06 00:47:35 2024[127.0.0.1:24811] test1|
Thu Jun 06 00:47:38 2024[0.0.0.0:5415] Server is started
Thu Jun 06 00:47:44 2024[127.0.0.1:10971] test1 LOG_IN
Thu Jun 06 00:47:45 2024[127.0.0.1:10971] test1| PORT 127,0,0,1,117,0
Thu Jun 06 00:48:11 2024[127.0.0.1:10971] test1|
Thu Jun 06 00:48:11 2024 Server is terminated
Thu Jun 06 00:48:13 2024[0.0.0.0:5415] Server is started
Thu Jun 06 00:48:17 2024[127.0.0.1:31955] test1 LOG_IN
Thu Jun 06 00:48:18 2024[127.0.0.1:31955] test1| PORT 127,0,0,1,94,155
200 PORT command performed successfully.
200 PORT command performed successfully.
Thu Jun 06 00:48:18 2024[127.0.0.1:31955] test1| NLST

Thu Jun 06 00:48:20 2024[127.0.0.1:31955] test1| LIST

Thu Jun 06 00:48:21 2024[127.0.0.1:31955] test1| LIST

Thu Jun 06 00:48:22 2024[127.0.0.1:31955] test1| LIST

Thu Jun 06 00:48:23 2024[127.0.0.1:31955] test1| LIST

Thu Jun 06 00:50:29 2024[127.0.0.1:31955] test1|
Thu Jun 06 00:51:31 2024[0.0.0.0:5415] Server is started
Thu Jun 06 00:51:35 2024[127.0.0.1:29339] test1 LOG_IN
Thu Jun 06 00:51:37 2024[127.0.0.1:29339] test1| PORT 127,0,0,1,221,119
200 PORT command performed successfully.

Thu Jun 06 00:59:58 2024[127.0.0.1:34012] test1 LOG_IN
Thu Jun 06 01:00:04 2024[127.0.0.1:34012] test1| CDUP
250 CWD command performed successfully.
Thu Jun 06 01:03:55 2024[127.0.0.1:34012] test1| PORT 127,0,0,1,120,99
200 PORT command performed successfully.
Thu Jun 06 01:03:55 2024[127.0.0.1:34012] test1| NLST

Thu Jun 06 01:04:25 2024[127.0.0.1:34012] test1| RNFR good RNTO bad
250 RNTO command succeeds
Thu Jun 06 01:04:36 2024[127.0.0.1:34012] test1| PORT 127,0,0,1,144,189
200 PORT command performed successfully.
Thu Jun 06 01:04:36 2024[127.0.0.1:34012] test1| NLST

Thu Jun 06 01:06:00 2024[127.0.0.1:34012] test1| QUIT
221 Goodbye
Thu Jun 06 01:06:00 2024[127.0.0.1:34012] test1 LOG_OUT [total service time : 52sec]
Thu Jun 06 01:06:04 2024 Server is terminated
Thu Jun 06 01:10:47 2024[0.0.0.0:4135] Server is started
Thu Jun 06 01:10:50 2024[127.0.0.1:13514] test1 LOG_IN
Thu Jun 06 01:10:51 2024 Server is terminated
```

Client 가 올바른 IP 주소와 3 번의 기회 동안 성공적으로 id, password 를 입력해서 server 에 접속하고 나면 리눅스 명령어를 입력하여 FTP 명령어로 변환하고 그 명령어를 server 에 보내서 server 에서 실행될 수 있도록 한다. 실행 결과는 다시 client 에 전달하여 client 쪽에서 결과가 출력될 수 있도록 한다. 그리고 이런 결과들을 logfile 에 기록할 수 있도록 한다. 그리고 언제 기록되었는지 알 수 있도록 기록을 남기게 한다. 또한 motd 기록된 내용도 client 에 보낼 수 있도록 한다.

get, put, ls 명령어는 control connection, data connection 을 따로 구현해서 데이터를 효율적으로 받아올 수 있도록 하고 몇 byte 의 데이터를 처리했는지 client 에서 출력되게 한다. 하지만 내 코드 put 오류가 난 부분을 해결하지 못해서 put 은 구현하지 못했다.

고찰

지금까지 했던 것을 다 합치고 get, put 을 control connection, data connection 으로 구현해보면 된다. 기능적으로 많이 할 부분은 별로 없었지만 합치는 과정에서 에러가 많이 나왔다. 하지만 지금까지 했던 대로 하나하나 꼼꼼히 에러가 날 부분을 보고 printf 로 디버거를 해보면서 하나하나 해결해 나갔다고 최종적으로 지금까지 했던 것을 다 합쳐서 실행을 할 수 있었다. 이번 과제를 통해 FTP 서버가 리눅스에서 어떻게 구현되는지 알 수 있는 시간이 되었을 뿐만 아니라 socket 프로그래밍에서 어떤 부분에 에러가 많이 나는지 어떤 부분을 조심해서 코딩을 해야 할지 알 수 있는 시간이 되었다.

Reference

시스템프로그래밍 실습 강의자료