시스템 프로그래밍 실습

[Assignment #3-3: log file]

Class : [금요일 1,2 교시]

Professor : [최상호 교수님]

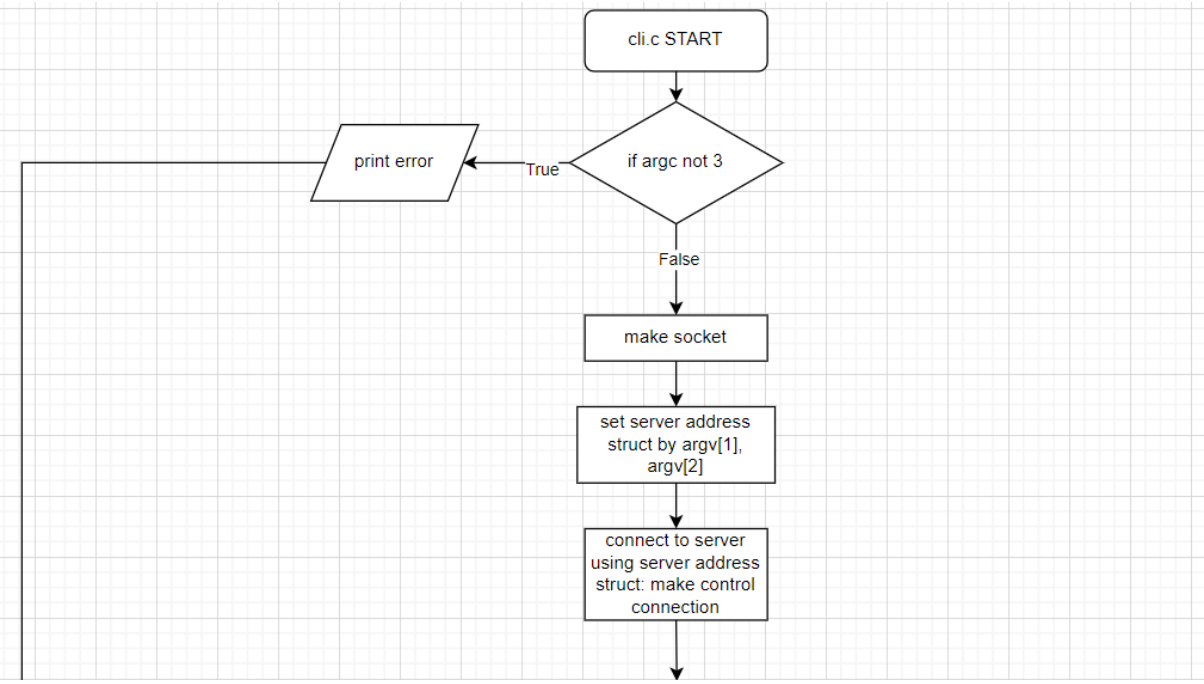
Student ID : [2020202034]

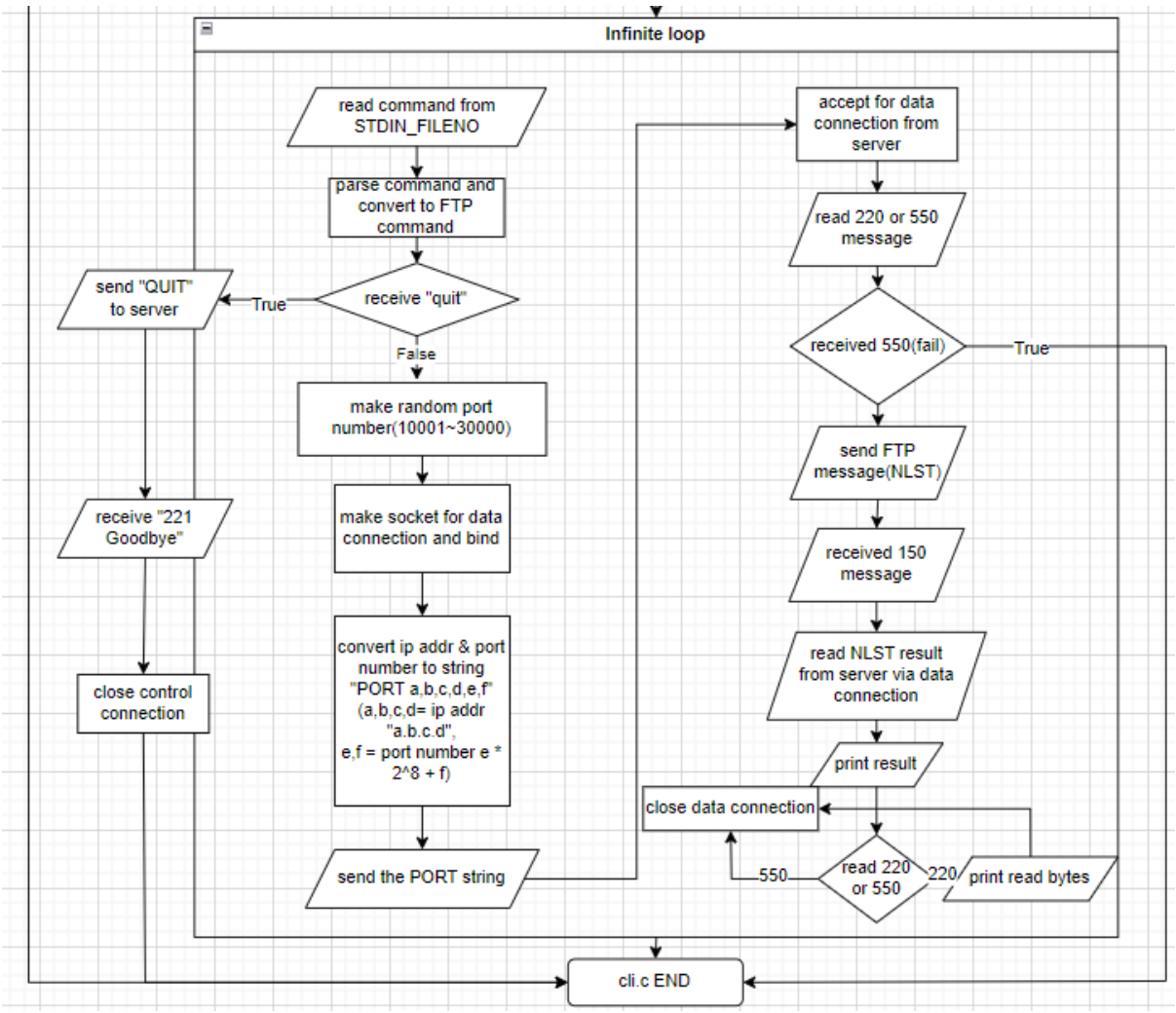
Name : [김태완]

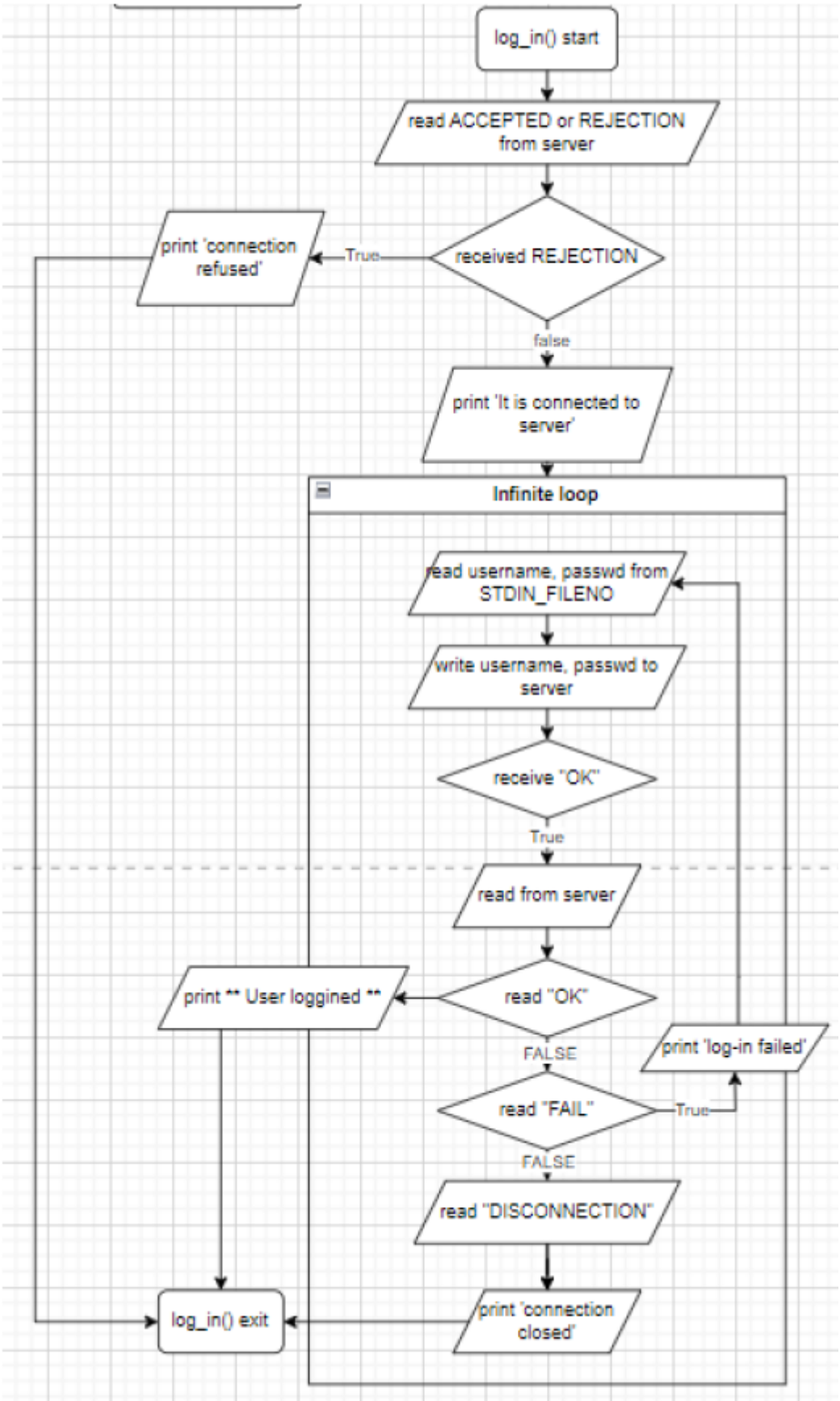
Introduction

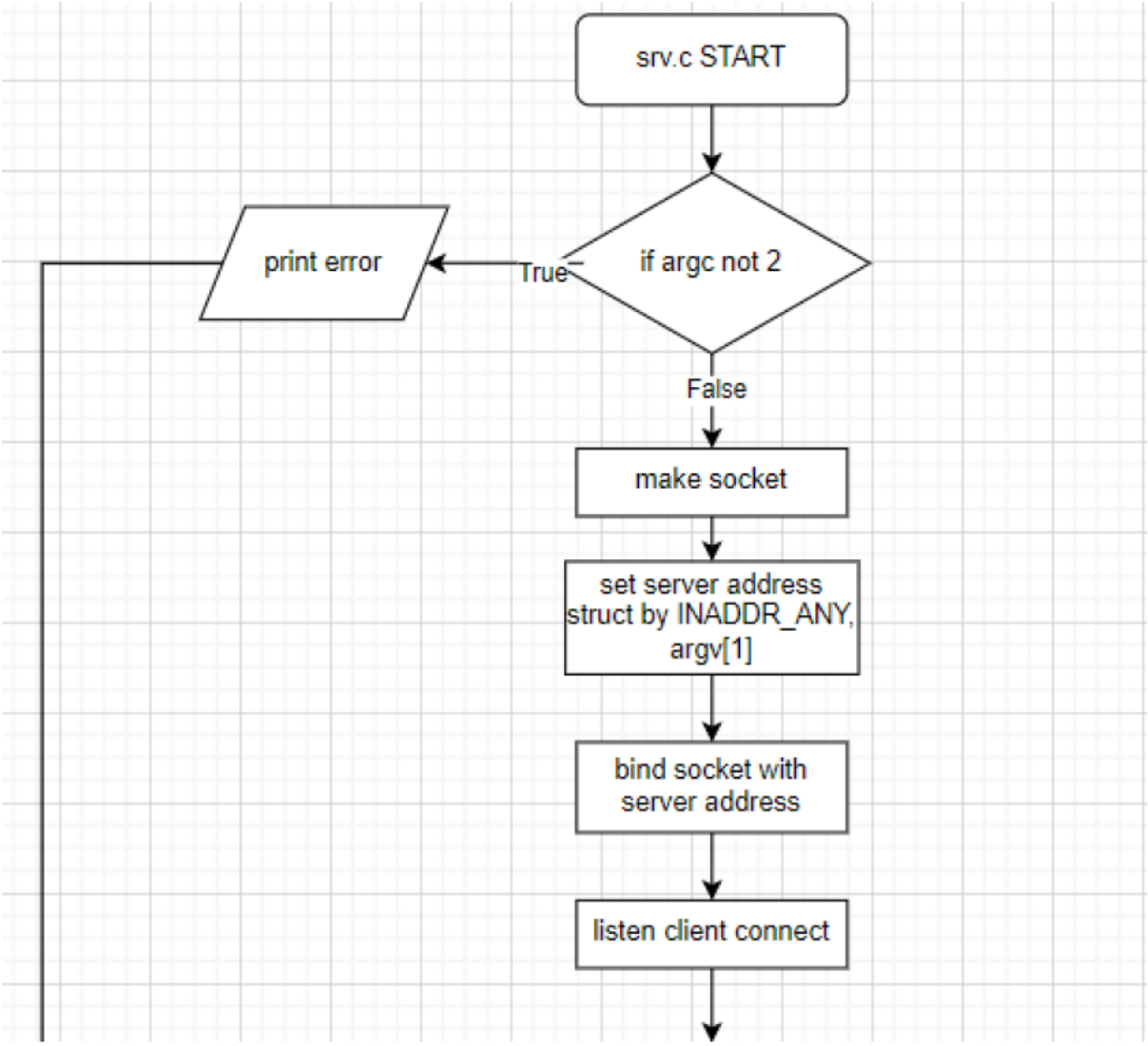
이번 과제는 지금까지 했던 FTP server 와 관련된 모든 내용(FTP command, socket programming, fork, login, data connection)에 log file 을 추가해 실제와 거의 유사한 FTP server program 을 만드는 과제입니다. 지금까지 구현한 코드를 이어 붙인 후, 각 단계의 성공과 실패 여부를 reply code 로 출력하도록 출력부를 전반적으로 교체하면 됩니다. 그리고 get, put 명령어 구현을 위해 control -> data connection 사이에 추가적인 확인 과정을 거쳐야 합니다.

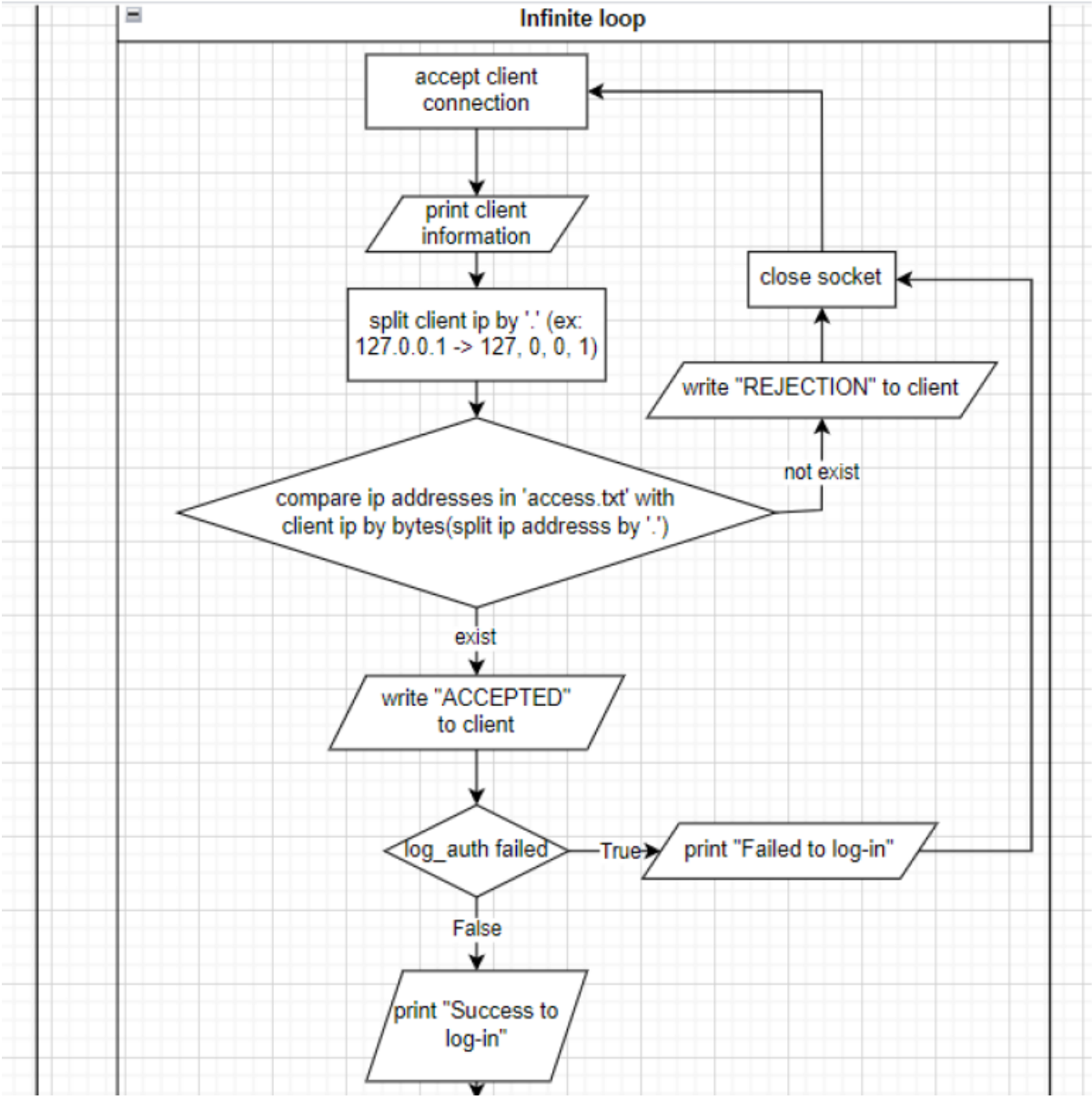
Flow chart

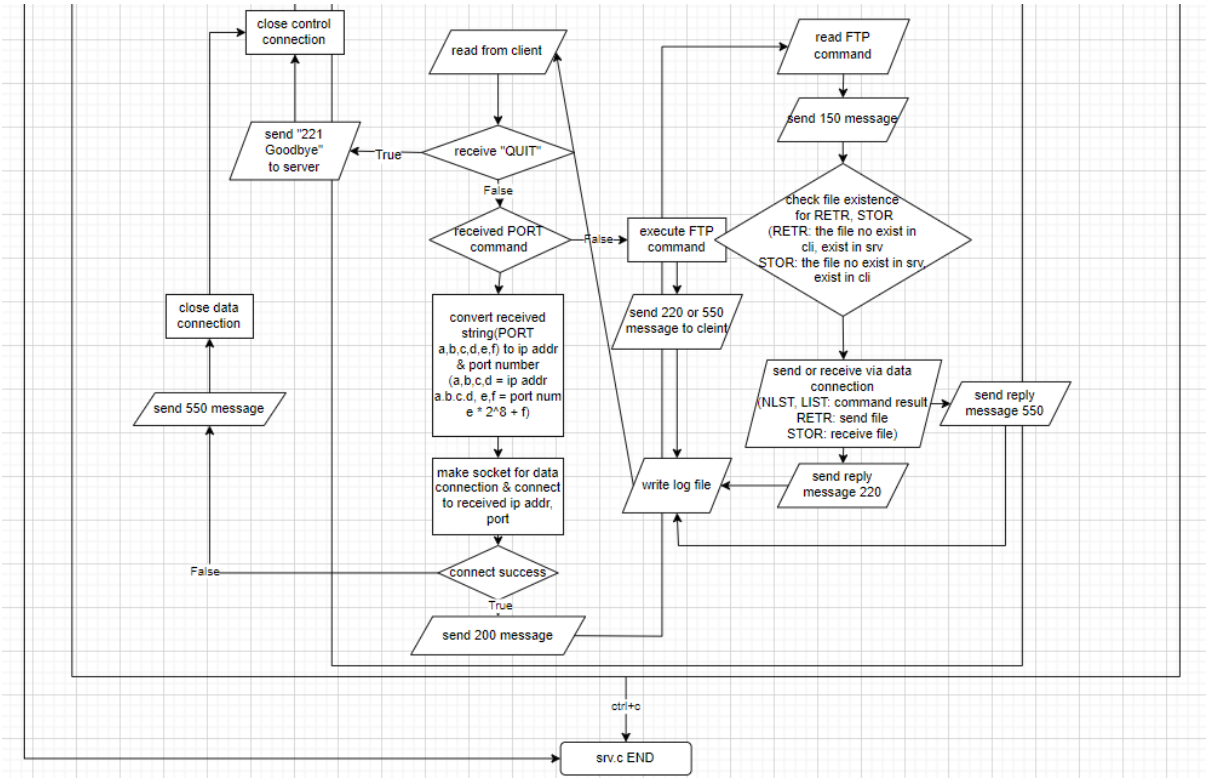



Infinite loop
read command from STDIN_FILENO
parse command and convert to FTP command
receive "quit"
True
send "QUIT" to server
receive "221 Goodbye"
close control connection
False
make random port number(10001~30000)
make socket for data connection and bind
convert ip addr & port number to string "PORT a,b,c,d,e,f" (a,b,c,d= ip addr "a.b.c.d", e,f = port number e * 2^8 + f)
send the PORT string
accept for data connection from server
read 220 or 550 message
received 550(fail)
True
send FTP message(NLST)
received 150 message
read NLST result from server via data connection
print result
close data connection
read 220 or 550
550
220
print read bytes
cli.c END

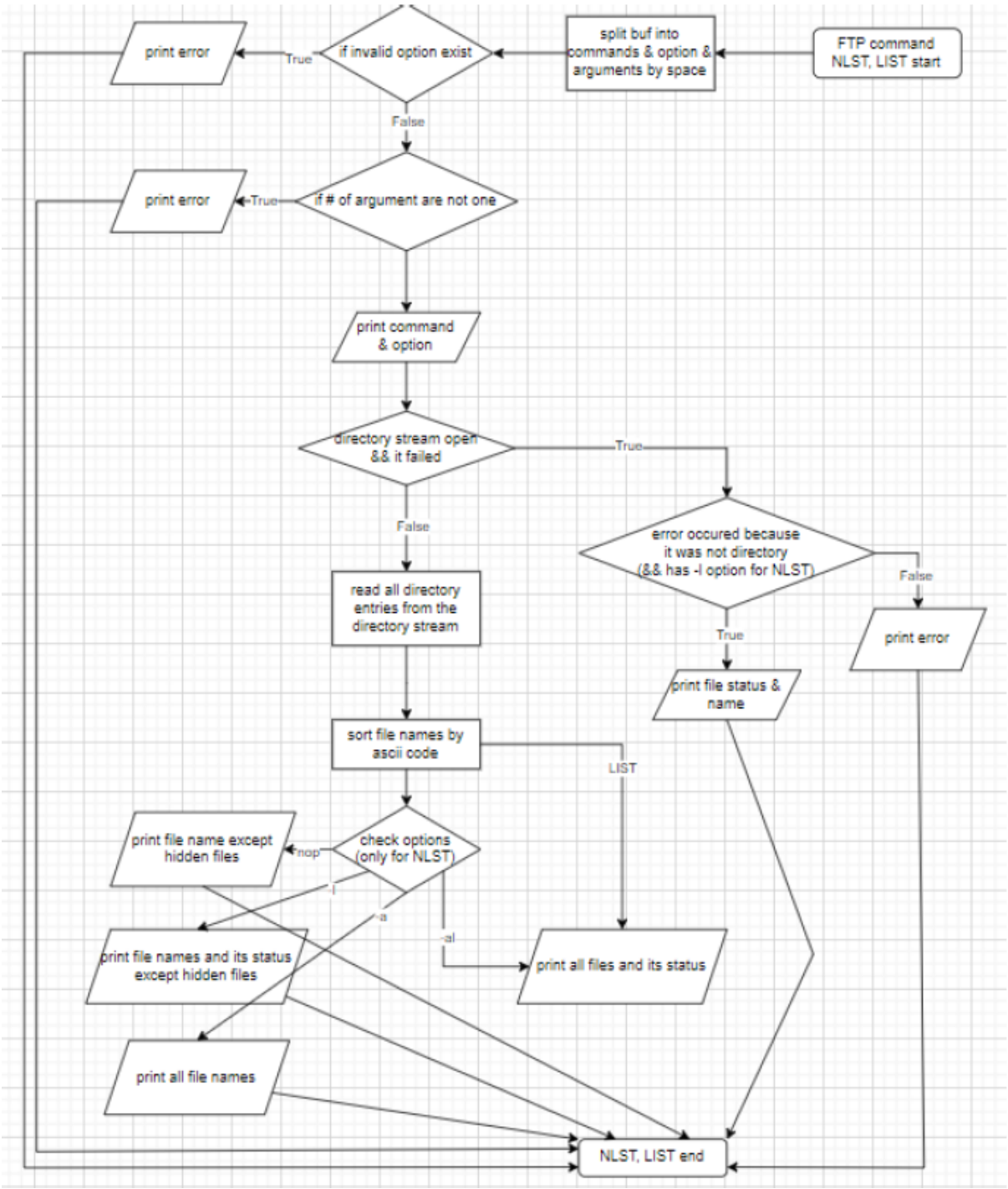

log_in() start
read ACCEPTED or REJECTION from server
received REJECTION
True
print 'connection refused'
false
print 'It is connected to server'
Infinite loop
read username, passwd from STDIN_FILENO
write username, passwd to server
receive "OK"
True
read from server
read "OK"
print ** User loggined **
FALSE
print 'log-in failed'
read "FAIL"
True
FALSE
read "DISCONNECTION"
print 'connection closed'
log_in() exit

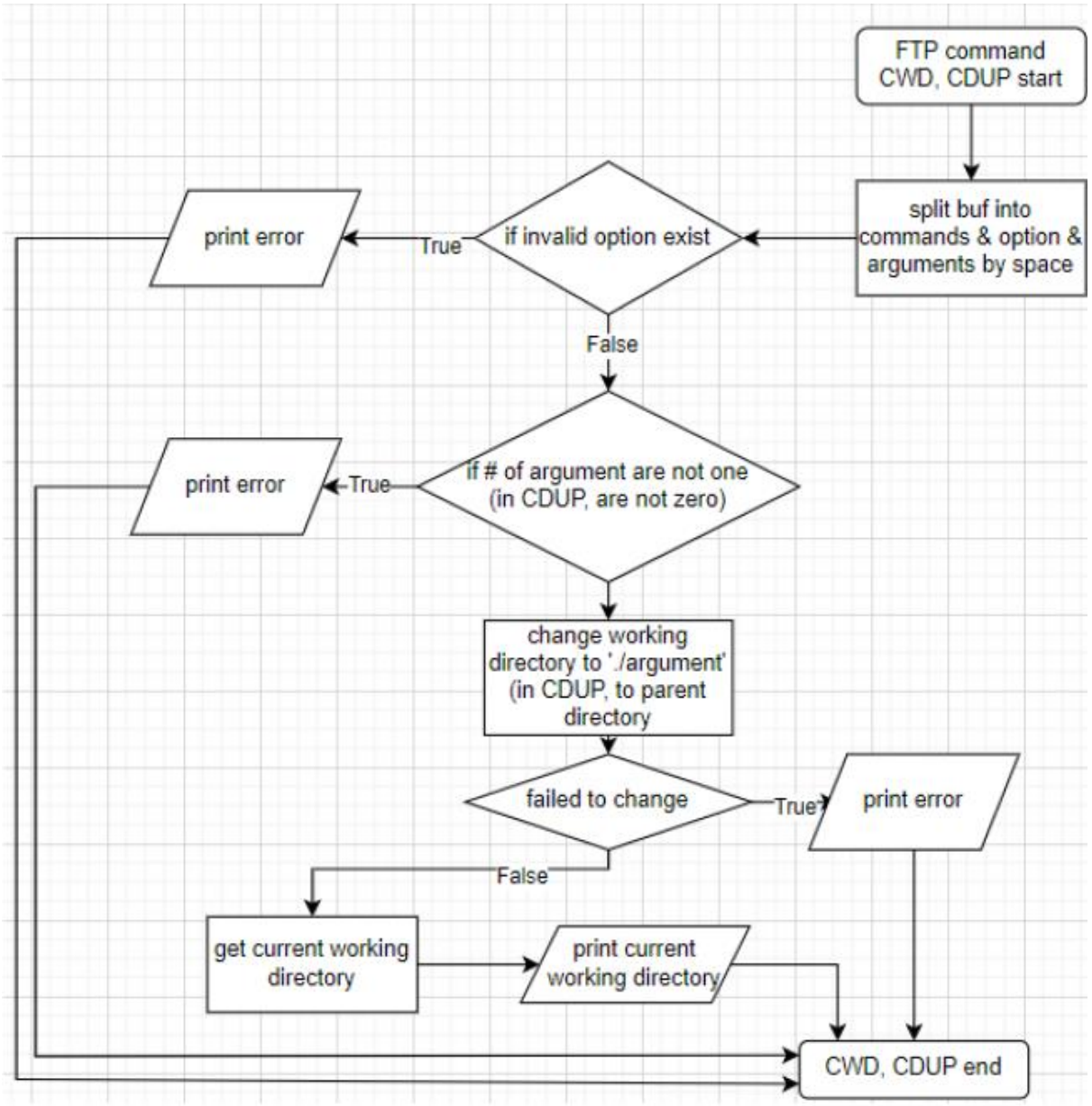

srv.c START
if argc not 2
True
print error
False
make socket
set server address struct by INADDR_ANY, argv[1]
bind socket with server address
listen client connect

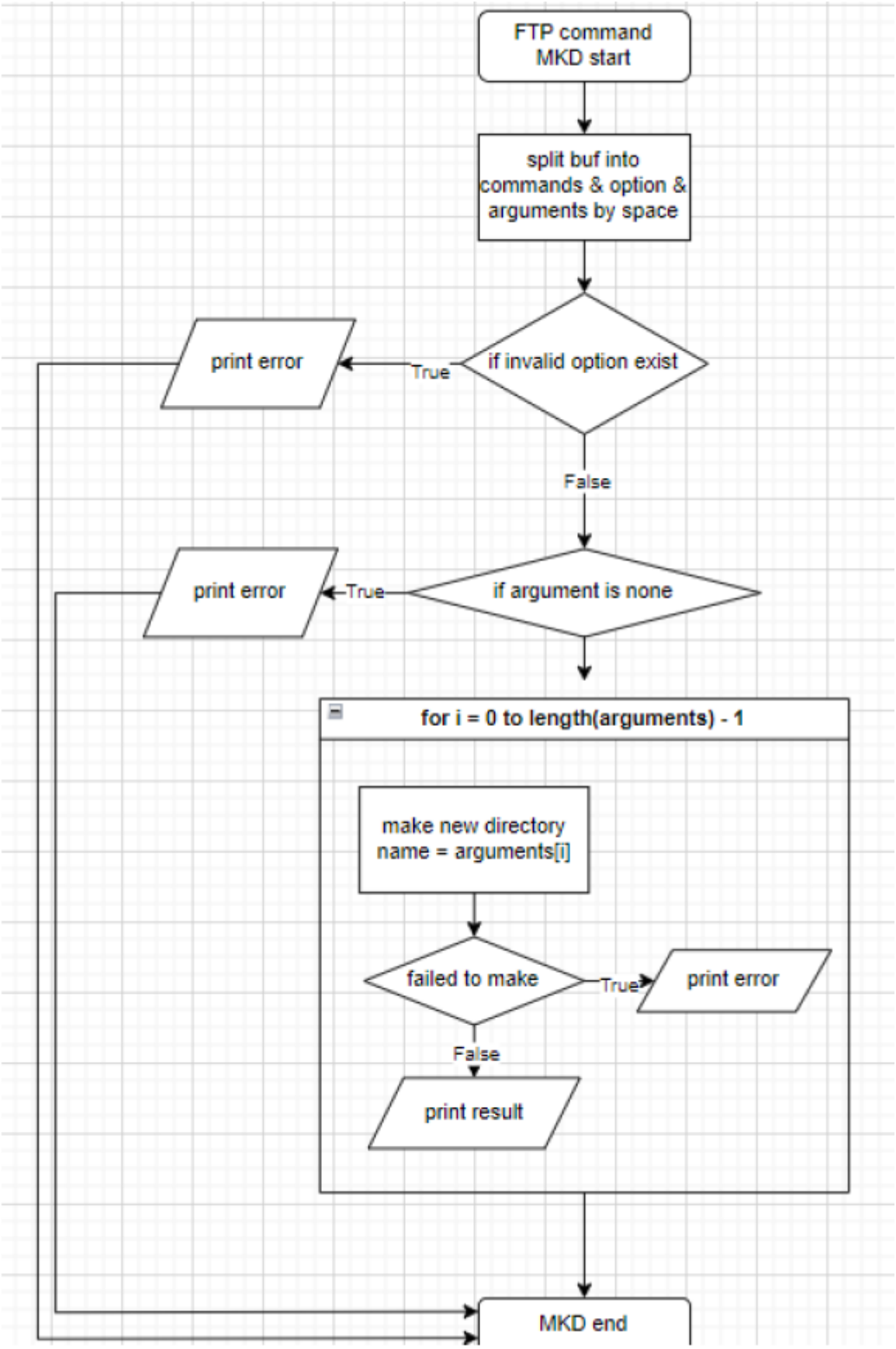

Infinite loop
accept client connection
print client information
split client ip by '.' (ex: 127.0.0.1 -> 127, 0, 0, 1)
compare ip addresses in 'access.txt' with client ip by bytes(split ip addresss by '.')
not exist
write "REJECTION" to client
close socket
exist
write "ACCEPTED" to client
log_auth failed
True
print "Failed to log-in"
False
print "Success to log-in"

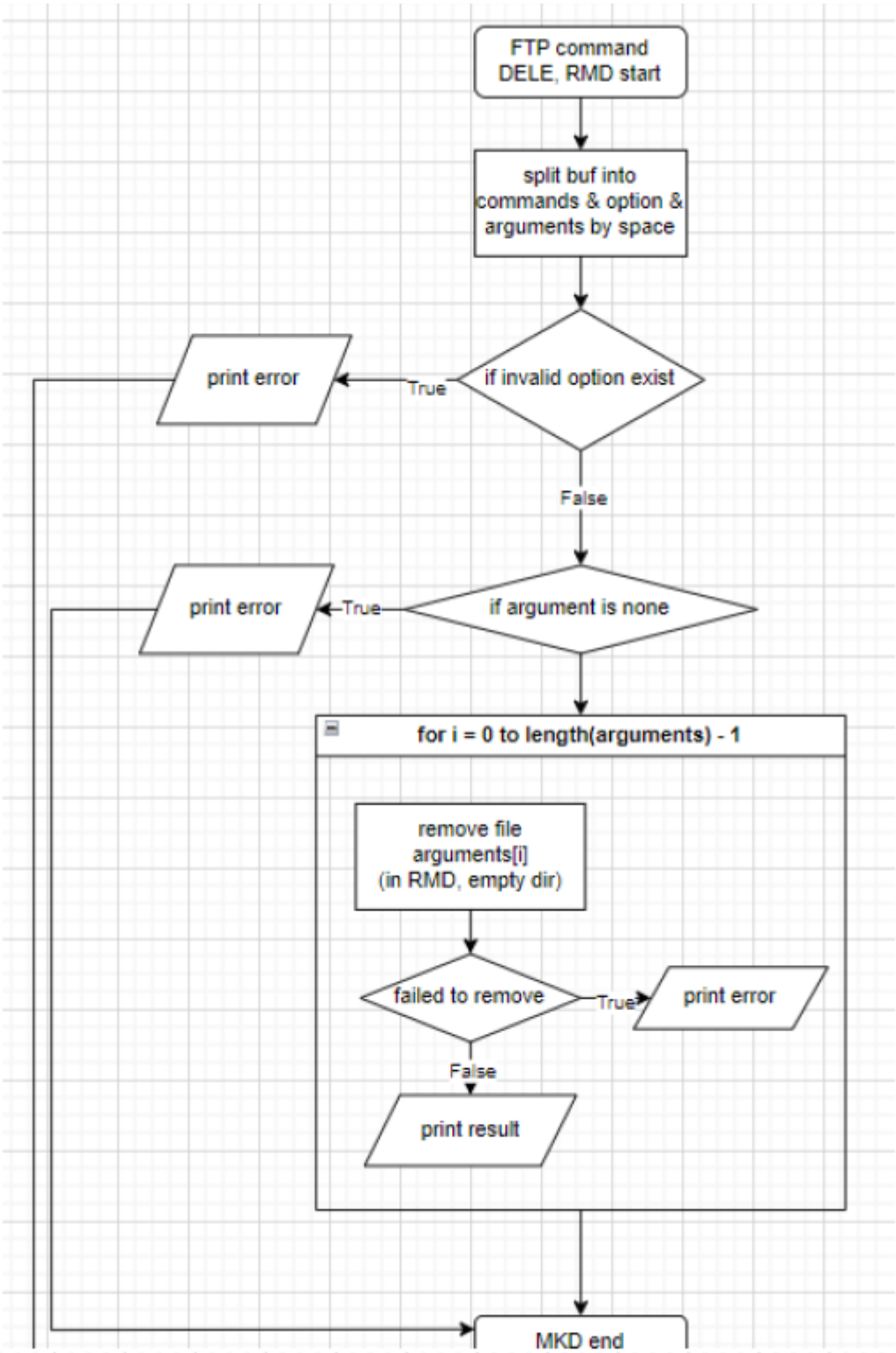

close control connection
read from client
read FTP command
send 150 message
send "221 Goodbye" to server
True
receive "QUIT"
False
received PORT command
False
execute FTP command
check file existence for RETR, STOR (RETR: the file no exist in cli, exist in srv STOR: the file no exist in srv, exist in cli
close data connection
convert received string(PORT a,b,c,d,e,f) to ip addr & port number (a,b,c,d = ip addr a.b.c.d, e,f = port num e * 2^8 + f)
send 220 or 550 message to cleint
send 550 message
send or receive via data connection (NLST, LIST: command result RETR: send file STOR: receive file)
send reply message 550
make socket for data connection & connect to received ip addr, port
write log file
send reply message 220
False
connect success
True
send 200 message
ctrl+c
srv.c END

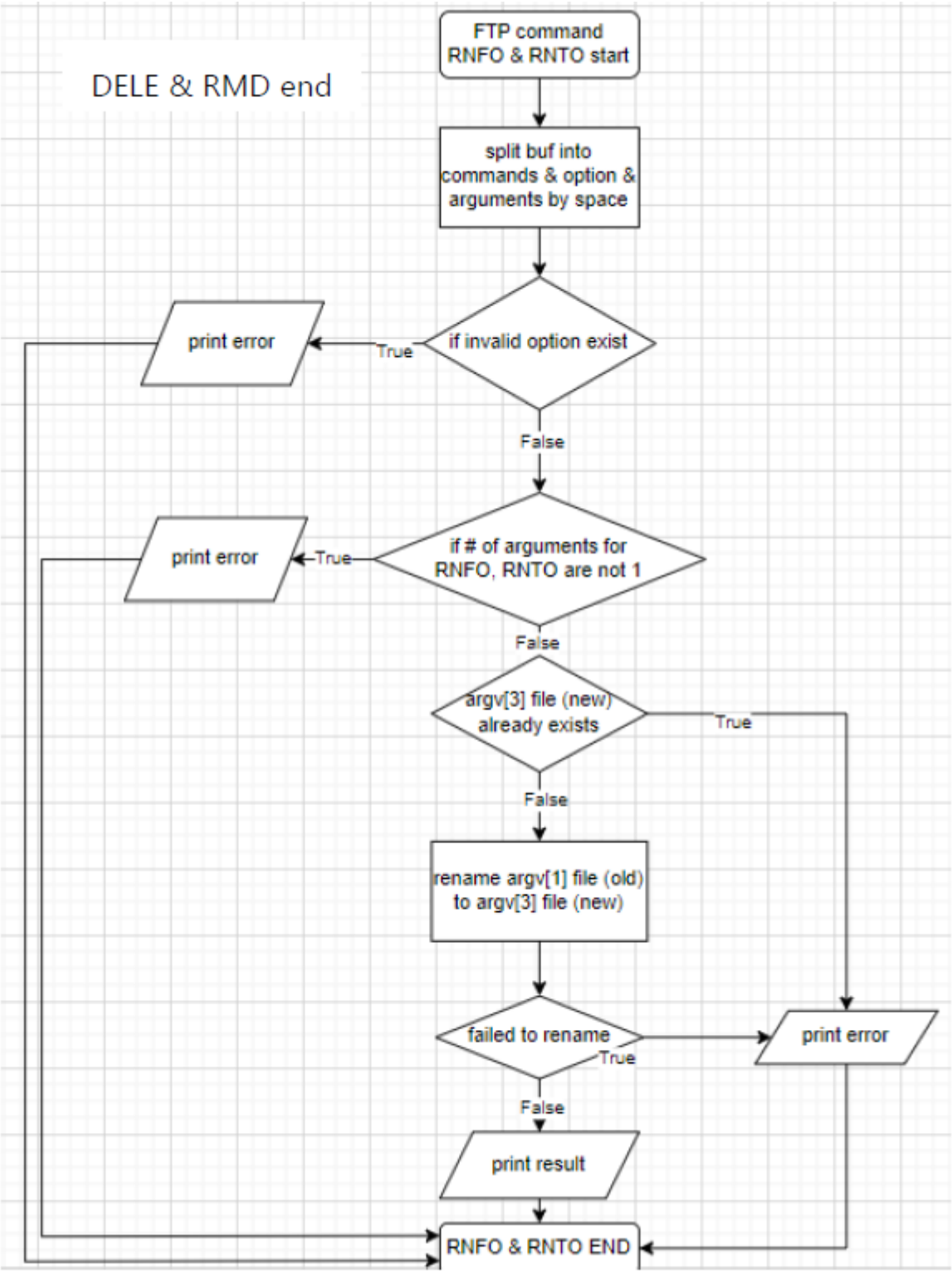

log_auth()
start
Infinite loop
read user name,
passwd from client
write "OK" to client
print log-in try count
received user, passwd is
in the file 'passwd'
(== user_match())
True
write "OK" to client
False
print "log-in failed"
count >= 3
True
write "DISCONNECTION"
to client
false
write "FAIL"
to client
count++
log_auth()
end


FTP command
NLST, LIST start
split buf into
commands & option &
arguments by space
if invalid option exist
True
print error
False
if # of argument are not one
True
print error
print command
& option
directory stream open
&& it failed
True
error occured because
it was not directory
(&& has -l option for NLST)
False
print error
True
print file status &
name
False
read all directory
entries from the
directory stream
sort file names by
ascii code
LIST
check options
(only for NLST)
nop
print file name except
hidden files
l
a
al
print file names and its status
except hidden files
print all files and its status
print all file names
NLST, LIST end


FTP command
CWD, CDUP start
split buf into
commands & option &
arguments by space
if invalid option exist
True
print error
False
if # of argument are not one
(in CDUP, are not zero)
True
print error
change working
directory to './argument'
(in CDUP, to parent
directory
failed to change
True
print error
False
get current working
directory
print current
working directory
CWD, CDUP end


FTP command
MKD start
split buf into
commands & option &
arguments by space
if invalid option exist
True
print error
False
if argument is none
True
print error
for i = 0 to length(arguments) - 1
make new directory
name = arguments[i]
failed to make
True
print error
False
print result
MKD end


FTP command
DELE, RMD start
split buf into
commands & option &
arguments by space
if invalid option exist
True
print error
False
if argument is none
True
print error
for i = 0 to length(arguments) - 1
remove file
arguments[i]
(in RMD, empty dir)
failed to remove
True
print error
False
print result
MKD end



Pseudo code

cli.c

```
Define constants: BUF_SIZE, FLAGS, MODE


Define functions:
    convert_addr_to_str(char* buf, struct sockaddr_in* addr)
    log_in(int sockfd)


main(int argc, char** argv)
    If argc != 3
        Write error message
        Exit


    ctrlfd = socket(AF_INET, SOCK_STREAM, 0)
    temp.sin_family = AF_INET
    temp.sin_addr.s_addr = inet_addr(argv[1])
    temp.sin_port = htons(atoi(argv[2]))


    If connect(ctrlfd, (struct sockaddr*)&temp, sizeof(temp)) < 0
        Print error
        Exit


    log_in(ctrlfd)


    While true
        Clear buff
```

```
Write "ftp> "
n = read(STDIN_FILENO, buff, BUF_SIZE)
If n < 0
    Print error
    Exit
buff[n - 1] = '₩0'

len = 0
For each token in buff
    split[len++] = token
split[len] = NULL

If len == 0
    Continue

Clear cmd

For each table entry
    If split[0] matches table entry
        strcat(cmd, table entry value)
        Break

If split[0] == "rename" and len == 3
    Handle rename command
```

```
        Else if split[0] == "type" and len == 2
            Handle type command
        Else
            For i = 1 to len - 1
                If split[0] == "cd" and split[i] == ".."
                    Modify cmd for CDUP
                Else
                    Append split[i] to cmd

        If cmd == "QUIT"
            Handle quit command
            Return

        If cmd starts with "NLST", "LIST", "RETR", or "STOR"
            Handle data connection commands
        Else
            Write cmd to ctrlfd
            Read response from ctrlfd
            Print response
convert_addr_to_str(char* buf, struct sockaddr_in* addr)
    strcpy(buf, "PORT ")
    strcat(buf, inet_ntoa(addr->sin_addr))
    tmp = buf
    For each ptr = strchr(tmp, '.')
```

```
        *ptr = ','
        tmp = ptr
    sprintf(buf + strlen(buf), ",%d,%d", ntohs(addr->sin_port) >> 8, ntohs(addr->sin_port)
& 0xFF)


log_in(int sockfd)
    n = read(sockfd, buf, BUF_SIZE)
    If n <= 0
        Print error
        Exit
    buf[n] = '₩0'
    If buf starts with "431"
        Write buf
        close(sockfd)
        Exit
    Else
        Write buf


    While true
        Write "Name : "
        n = read(STDIN_FILENO, tmp_buff, 1024)
        If n <= 0
            Exit
        tmp_buff[n - 1] = '₩0'
```

```
sprintf(buf, "USER %s", tmp_buff)
write(sockfd, buf, strlen(buf))

n = read(sockfd, buf, BUF_SIZE)
If n <= 0
    Exit
buf[n] = '₩0'
Write buf
If buf starts with "430"
    Continue
Else if buf starts with "530"
    close(sockfd)
    Exit

passwd = getpass("Passwd : ")
sprintf(buf, "PASS %s", passwd)
write(sockfd, buf, strlen(buf))

n = read(sockfd, buf, BUF_SIZE)
If n <= 0
    Exit
buf[n] = '₩0'
Write buf
If buf starts with "430"
```

Continue

Else if buf starts with "530"

close(sockfd)

Exit

Else

Return

srv.c

Define constants: BUF_SIZE, TMP_SIZE, MAX_BUF, FLAGS, MODE

Define global variables: g_ip, g_port, g_user, g_mode, g_time, log_fd

Define functions:

sh_int(int sig)

convert_str_to_addr(char* str, struct sockaddr_in* addr)

log_auth(int connfd)

user_match(char* user, char* passwd)

MtoS(struct stat* infor, const char* pathname, char* print_buf)

NLST(char* buf, char* print_buf)

LIST(char* buf, char* print_buf)

PWD(char* buf, char* print_buf)

CWD(char* buf, char* print_buf)

CDUP(char* buf, char* print_buf)

MKD(char* buf, char* print_buf)

DELE(char* buf, char* print_buf)

RMD(char* buf, char* print_buf)

```
    RNFR(char* buf, char* name_from)
    RNTO(char* buf, char* name_from)
    convert_ascii(char* file)
    write_log(int fd, char* command, int bytes, int type)

main(int argc, char** argv)
    If argc != 2
        Write error message
        Raise SIGINT

    server_fd = socket(PF_INET, SOCK_STREAM, 0)
    Set server_addr with INADDR_ANY and argv[1]
    bind(server_fd, server_addr)
    listen(server_fd, 5)

    While true
        client_fd = accept(server_fd, &client_addr)
        g_ip = inet_ntoa(client_addr.sin_addr)
        g_port = ntohs(client_addr.sin_port)

        If fork() == 0 (child process)
            Close server_fd
            Open access.txt and check client IP
            If client IP not allowed
```

```
            Write error message
            write_log(log_fd, NULL, 0, ILLEGAL)
            Exit
        Else
            Open motd file
            Write welcome message
            write_log(log_fd, NULL, 0, AUTH)

        If log_auth(client_fd) == 0
            Close client_fd
            Exit

        While true
            Read command from client_fd
            If command == "QUIT"
                Handle quit command
                write_log(log_fd, NULL, 0, DISCONNECT)
                Exit
            Else if command starts with "PWD", "CWD", "CDUP", "DELE", "MKD",
"RMD", "RNFR", "RNTO", or "TYPE"
                Handle corresponding command
            Else (NLST, LIST, RETR, STOR)
                Handle data connection commands
    Else (parent process)
```

```
            Close client_fd

Define sh_int(int sig)
    Wait for all child processes to terminate
    write_log(log_fd, NULL, 0, TERM)
    Close log_fd
    Exit

Define convert_str_to_addr(char* str, struct sockaddr_in* addr)
    Parse str to extract IP address and port number
    Fill addr with parsed IP address and port number

Define log_auth(int connfd)
    While true
        Read USER command from connfd
        If user_match(user, NULL) < 0
            If failed 3 times
                Write error message
                write_log(log_fd, NULL, 0, ILLEGAL)
                Return 0
            Else
                Write error message
                Continue
```

```
        Read PASS command from connfd
        If user_match(user, passwd) < 0
            If failed 3 times
                Write error message
                write_log(log_fd, NULL, 0, ILLEGAL)
                Return 0
            Else
                Write error message
                Continue
        Else
            Write success message
            write_log(log_fd, NULL, 0, AUTH)
            Break
    Return 1


Define user_match(char* user, char* passwd)
    Open passwd file
    For each entry in passwd file
        If user matches and passwd is NULL
            Return 0
        If user and passwd match
            Return 1
    Return -1
```

```
Define MtoS(struct stat* infor, const char* pathname, char* print_buf)
    Determine file type and permissions
    Format file information string
    Write formatted string to print_buf

Define NLST(char* buf, char* print_buf)
    Parse options and arguments from buf
    Open directory specified by argument
    Read directory entries and store filenames
    Sort filenames
    If -l option
        For each filename
            Get file information and format string
            Append formatted string to print_buf
    Else
        For each filename
            Append filename to print_buf
            Append '/' if file is a directory
    Return 0 on success, -1 on failure

Define LIST(char* buf, char* print_buf)
    Similar to NLST, but always prints file information (-l)

Define PWD(char* buf, char* print_buf)
```

Get current working directory

Write current working directory to print_buf

Return 0 on success, -1 on failure

Define CWD(char* buf, char* print_buf)

Parse argument from buf

Change working directory to argument

Return 0 on success, -1 on failure

Define CDUP(char* buf, char* print_buf)

Change working directory to parent directory

Return 0 on success, -1 on failure

Define MKD(char* buf, char* print_buf)

Parse argument from buf

Create directory with argument as name

Return 0 on success, -1 on failure

Define DELE(char* buf, char* print_buf)

Parse argument from buf

Delete file with argument as name

Return 0 on success, -1 on failure

Define RMD(char* buf, char* print_buf)

Parse argument from buf

Remove directory with argument as name

Return 0 on success, -1 on failure

Define RNFR(char* buf, char* name_from)

Parse argument from buf

Store argument in name_from

Return 0 on success, -1 on failure

Define RNTO(char* buf, char* name_from)

Parse argument from buf

Rename file from name_from to argument

Return 0 on success, -1 on failure

Define convert_ascii(char* file)

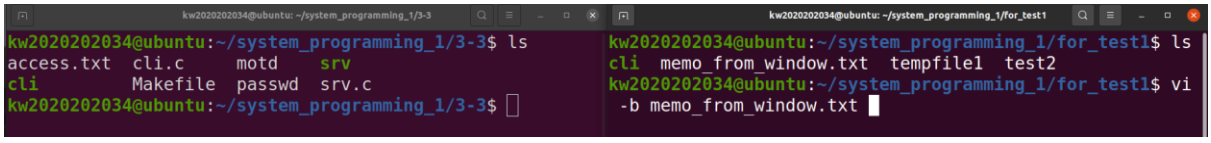
Convert ₩r₩n and ₩n₩r sequences to ₩n in file

Define write_log(int fd, char* command, int bytes, int type)

Format log message based on type

Write formatted log message to fd

결과화면



get, put 의 test 를 위해 cli 와 srv 를 다른 경로에 두고 테스트하였습니다. login 을 위해 access.txt, motd, passwd 는 srv 와 같은 경로에 두었습니다.

```
kw2020202034@ubuntu: ~/system_programming_1/for_test1
memo from window^M
I wanna jong gang^M
I love saeu kkang^M
thank^M
you
~
~
~
~
```

binary, ascii mode transmission 을 보이기 위해 윈도우에서 텍스트 파일을 만들어 cli 와 같은 경로에 두었습니다. vi -b memo_from_window.txt 로 확인한 결과, ₩r₩n 이 개행으로 들어가 있는 걸 확인할 수 있습니다.

```
kw2020202034@ubuntu:~/system_programming_1/3-3$ cat access.txt
*.0.0.0
127.0.0.*
128.0.0.1kw2020202034@ubuntu:~/system_programming_1/3-3$
```

access.txt 는 다음과 같이 구성되어 있어 제 컴퓨터 내의 cli 가 접속하면 ip 가 127.0.0.1 이므로 이에 해당해 접근을 허용할 것입니다.

```
kw2020202034@ubuntu:~/system_programming_1/3-3$ ./srv 30
00
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue J
un 04 16:06:50 KST 2024) ready.
```

```
kw2020202034@ubuntu:~/system_programming_1/for_test1$ ./
cli 127.0.0.1 3000
Connected to sswlab.kw.ac.kr.
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue J
un 04 16:06:50 KST 2024) ready.
Name :
```

보시는 바와 같이 cli 가 잘 접속해 220 reply message 가 나온 것을 볼 수 있습니다.

```
kw2020202034@ubuntu:~/system_programming_1/3-3$ cat passwd
taewan:12:0:0:SPLab1:/home/1:sh1
nono:34:1:0:SPLab2:/home/2:sh2
gang:56:2:1:SPLab3:/home/3:sh3kw2020202034@ubuntu:~/system_pr
```

passwd 는 taewan, nono, gang 3 사용자의 정보가 저장되어 있습니다. 저는 taewan 으로 접속할 건데, 비밀번호가 12 이므로 12 를 password 로 입력할 것입니다.

```
kw2020202034@ubuntu:~/system_programming_1/3-3$ ./srv 30
00
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue J
un 04 16:06:50 KST 2024) ready.
USER taewan
331 Password is required for username.
PASS 12
230 User taewan logged in.
```

```
kw2020202034@ubuntu:~/system_programming_1/for_test1$ ./
cli 127.0.0.1 3000
Connected to sswlab.kw.ac.kr.
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue J
un 04 16:06:50 KST 2024) ready.
Name : taewan
331 Password is required for username.
Passwd :
230 User taewan logged in.
ftp>
```

passwd 에 존재하는 ID, passwd 를 입력하니 접속이 성공한 것을 볼 수 있습니다.

```
kw2020202034@ubuntu:~/system_programming_1/3-3$ ./srv 3000
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue Jun 04 16:06:50 KST 2024) r
eady.
USER taewan
331 Password is required for username.
PASS 12
230 User taewan logged in.
PORT 127,0,0,1,142,16
200 PORT command successful
NLST -la
150 Opening data connection for directory list
226 Complete transmission.
```

```
kw2020202034@ubuntu:~/system_programming_1/for_test1$ ./cli 127.0.0.1 3000
Connected to sswlab.kw.ac.kr.
220 sswlab.kw.ac.kr FTP server (version myftp[1.0] Tue Jun 04 16:06:50 KST 2024) r
eady.
Name : taewan
331 Password is required for username.
Passwd :
230 User taewan logged in.
ftp> ls -la
200 PORT command successful
150 Opening data connection for directory list
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Jun 04 16:06 ./
drwxrwxr-x 14 kw2020202034 kw2020202034   4096 Jun 04 11:54 ../
-rw-rw-r--  1 kw2020202034 kw2020202034     98 Jun 02 20:41 Makefile
-rw-rw-r--  1 kw2020202034 kw2020202034     27 Jun 03 17:46 access.txt
-rwxrwxr-x  1 kw2020202034 kw2020202034  22312 Jun 04 16:06 cli
-rw-rw-r--  1 kw2020202034 kw2020202034  14819 Jun 04 16:06 cli.c
-rwxr-xr-x  1 kw2020202034 kw2020202034    412 Jun 04 16:19 logfile
-rw-rw-r--  1 kw2020202034 kw2020202034     50 Jun 03 21:54 motd
-rw-rw-r--  1 kw2020202034 kw2020202034     94 Jun 03 08:32 passwd
-rwxrwxr-x  1 kw2020202034 kw2020202034  40552 Jun 04 14:39 srv
-rw-rw-r--  1 kw2020202034 kw2020202034  46819 Jun 04 14:39 srv.c
226 Complete transmission.
OK. 727 bytes is received.
ftp>
```

ftp 명령어를 입력하는 부분이 출력되어 ls -la 명령어를 입력하니 PORT 연결, data connection 을 통한 명령어 결과 수신, 바이트 수가 잘 출력됨을 볼 수 있습니다. 이때, 전송받은 reply message, FTP command 는 server 에도 출력됩니다. 다음 장부터는 편한 보기를 위해 client side 만 캡쳐하였습니다.

```
ftp> dir ..
200 PORT command successful
150 Opening data connection for directory list
drwxrwxr-x 14 kw2020202034 kw2020202034   4096 Jun 04 11:54 ./
drwxr-xr-x 30 kw2020202034 kw2020202034   4096 Jun 04 16:07 ../
drwxrwxr-x  8 kw2020202034 kw2020202034   4096 Jun 04 14:42 .git/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Apr 01 17:40 .vscode/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Apr 12 09:38 1-1/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Apr 09 11:36 1-2/
drwxrwxr-x  3 kw2020202034 kw2020202034   4096 May 02 14:42 1-3/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 May 02 14:44 2-1/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 May 09 14:54 2-2/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 May 16 11:34 2-3/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 May 21 14:25 3-1/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 May 29 22:05 3-2/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Jun 04 16:06 3-3/
drwxrwxr-x  2 kw2020202034 kw2020202034   4096 Jun 04 16:06 for_test1/
-rw-rw-r--  1 kw2020202034 kw2020202034   2222 Apr 08 20:06 header.sh
226 Complete transmission.
OK. 988 bytes is received.
ftp>
```

dir 명령어의 결과입니다. 상위 디렉토리의 상세 정보를 잘 출력하였습니다.

```
ftp> pwd
257 "/home/kw2020202034/system_programming_1/3-3" is current directory.
ftp> cd ~
250 CWD command succeeds.
ftp> pwd
257 "/home/kw2020202034" is current directory.
ftp> cd system_programming_1/3-3
250 CWD command succeeds.
ftp> pwd
257 "/home/kw2020202034/system_programming_1/3-3" is current directory.
ftp> cd ..
250 CDUP command succeeds.
ftp> pwd
257 "/home/kw2020202034/system_programming_1" is current directory.
ftp> cd 3-3
250 CWD command succeeds.
ftp> pwd
257 "/home/kw2020202034/system_programming_1/3-3" is current directory.
ftp>
```

PWD, CWD, CDUP 의 결과입니다. cd(CWD, CDUP)로 working directory 를 이동하였을 때마다 reply code 250 의 유효성을 pwd 로 current working directory 가 cd 의 인자임을 확인할 수 있습니다.

```
ftp> mkdir temp
250 MKD command performed successfully.
ftp> ls
200 PORT command successful
150 Opening data connection for directory list
Makefile
access.txt
cli
cli.c
logfile
motd
passwd
srv
srv.c
temp/
226 Complete transmission.
OK. 66 bytes is received.
ftp> rmdir temp
250 RMD command performed successfully.
ftp> rmdir not_exist_dir
550 not_exist_dir: Can't remove directory.
ftp> get Makefile
200 PORT command successful
150 Opening binary mode data connection for Makefile.
226 Complete transmission.
OK. 98 bytes is received.
ftp>
```

mkdir, rmdir, get 의 결과입니다. ls 를 통해 mkdir, rmdir 이 성공적으로 이루어졌음을 알 수 있고, get 도 문제 없이 성공적으로 실행되었다는 결과를 확인할 수 있습니다.

```
kw2020202034@ubuntu:~/system_programming_1/for_test1$ ls
cli  Makefile  memo_from_window.txt  tempfile1  test2
kw2020202034@ubuntu:~/system_programming_1/for_test1$ cat Makefile
CC = gcc
NAME = cli srv

all : $(NAME)

%.o : %.c
        $(CC) -o $@ $< -lsocket

clean:
        rm -rf $(NAME)kw2020202034@ubuntu:~/system_programming_1/for
```

실제 검증을 위해 cli 가 있는 디렉토리 for_test1 에 들어가보니, get 으로 받은 Makefile 이 잘 들어있고, 내용을 확인해보면 문제 없이 잘 전달되었음을 볼 수 있습니다.

```
ftp> put memo_from_window.txt
200 PORT command successful
150 Opening binary mode data connection for memo_from_window.txt.
226 Complete transmission.
OK. 66 bytes is sent.
ftp> ls
200 PORT command successful
150 Opening data connection for directory list
Makefile
access.txt
cli
cli.c
logfile
memo_from_window.txt
motd
passwd
srv
srv.c
226 Complete transmission.
OK. 81 bytes is received.
ftp>
```

다음은 put 명령어의 결과입니다. window 에서 작성한 파일을 srv side 로 보냈는데, binary 로 보냈음을 확인할 수 있고, ls 로 확인 결과 파일이 생성되었음을 볼 수 있습니다.

```
memo from window^M
I wanna jong gang^M
I love saeu kkang^M
thank^M
you
~
```

위 파일을 열어보면 '₩r'이 남아있는데, 이는 binary mode 였기 때문에 개행이 바뀌지 않은 것입니다.

```
ftp> type ascii
201 Type set to A.
ftp> put memo_from_window.txt
200 PORT command successful
150 Opening ascii mode data connection for memo_from_window.txt.
550 Failed transmission.
ftp> delete memo_from_window.txt
250 DELE command performed successfully.
ftp> put memo_from_window.txt
200 PORT command successful
150 Opening ascii mode data connection for memo_from_window.txt.
226 Complete transmission.
OK. 66 bytes is sent.
ftp> ls
200 PORT command successful
150 Opening data connection for directory list
Makefile
access.txt
cli
cli.c
logfile
memo_from_window.txt
motd
passwd
srv
srv.c
226 Complete transmission.
OK. 81 bytes is received.
ftp>
```

type 명령어로 ascii mode 로 바꾸었습니다. put 을 다시 하게 되면 아까 memo_from_window.txt 가 아직 존재해 오류가 떴고, delete 명령어로 해당 파일은 지운 후 다시 put 을 하면 잘 받았음을 알 수 있습니다.

```
memo from window
I wanna jong gang
I love saeu kkang
thank
you
~
```

다시 받은 memo_from_window.txt 를 vi -b memo_from_window.txt 로 확인 결과 '₩r₩n'이 일반 개행으로 바뀌었음을 확인할 수 있습니다.

```
ftp> rename memo_from_window.txt window
350 File exists, ready to rename.
250 RNTO command succeeds.
ftp> rename not_exist_file b
550 not_exist_file: Can't find such file or directory.
ftp>
```

다음은 rename test 입니다. 존재하는 파일을 존재하지 않는 파일명으로 바꾸었기에 성공적으로 실행되었고, 존재하지 않는 파일을 바꾸려 하면 위와 같이 550 reply message 를 받게 됩니다.

```
ftp> quit
221 Goodbye.
```

quit 명령어를 입력하면 221 reply message 를 받으며 cli 가 종료됩니다.

```
Tue Jun  4 16:06:43 2024 Server is started

Tue Jun  4 16:08:05 2024 [127.0.0.1:59918] taewan LOG_IN

Tue Jun  4 16:19:24 2024 [127.0.0.1:59918] taewan PORT 127,0,0,1,142,16

Tue Jun  4 16:19:24 2024 [127.0.0.1:59918] taewan 200 PORT command successful

Tue Jun  4 16:19:24 2024 [127.0.0.1:59918] taewan NLST -la

Tue Jun  4 16:19:24 2024 [127.0.0.1:59918] taewan 150 Opening data connection for directory list

Tue Jun  4 16:19:24 2024 [127.0.0.1:59918] taewan 226 Complete transmission. | 727 bytes

Tue Jun  4 16:20:06 2024 [127.0.0.1:59918] taewan PORT 127,0,0,1,90,9

Tue Jun  4 16:20:06 2024 [127.0.0.1:59918] taewan 200 PORT command successful

Tue Jun  4 16:20:06 2024 [127.0.0.1:59918] taewan LIST ..

Tue Jun  4 16:20:06 2024 [127.0.0.1:59918] taewan 150 Opening data connection for directory list

Tue Jun  4 16:20:06 2024 [127.0.0.1:59918] taewan 226 Complete transmission. | 988 bytes

Tue Jun  4 16:20:28 2024 [127.0.0.1:59918] taewan PWD

Tue Jun  4 16:20:28 2024 [127.0.0.1:59918] taewan 257 "/home/kw2020202034/system_programming_1/3-3" is current dir
ectory.
```

다음은 log file 의 일부분들을 보여드리겠습니다. 시작 부분에 서버가 켜질 때, login 에 성공했을 때, PORT, NLST 등의 FTP 명령어가 전송될 때, reply code 들이 전송될 때 log file 에 이들이 잘 기록되었음을 볼 수 있습니다. 추가로, data connection 을 쓰는 NLST, LIST 명령어의 226 reply message 는 뒤에 전송한 bytes 수를 같이 기록함을 볼 수 있습니다.

```
Tue Jun  4 16:25:56 2024 [127.0.0.1:59918] taewan PORT 127,0,0,1,223,148

Tue Jun  4 16:25:56 2024 [127.0.0.1:59918] taewan 200 PORT command successful

Tue Jun  4 16:25:56 2024 [127.0.0.1:59918] taewan RETR Makefile

Tue Jun  4 16:25:56 2024 [127.0.0.1:59918] taewan 150 Opening binary mode data connection for Makefile.

Tue Jun  4 16:25:56 2024 [127.0.0.1:59918] taewan 226 Complete transmission. | 98 bytes

Tue Jun  4 16:27:51 2024 [127.0.0.1:59918] taewan PORT 127,0,0,1,54,24

Tue Jun  4 16:27:51 2024 [127.0.0.1:59918] taewan 200 PORT command successful

Tue Jun  4 16:27:51 2024 [127.0.0.1:59918] taewan STOR memo_from_window.txt

Tue Jun  4 16:27:51 2024 [127.0.0.1:59918] taewan 150 Opening binary mode data connection for memo_from_window.txt
.

Tue Jun  4 16:27:51 2024 [127.0.0.1:59918] taewan 226 Complete transmission. | 66 bytes
```

data connection 을 사용하는 다른 명령어인 RETR, STOR 도 226 reply message 를 보낼 때 bytes 수를 같이 기록함을 볼 수 있습니다.

```
Tue Jun  4 16:33:35 2024 [127.0.0.1:59918] taewan RNFR not_exist_file

Tue Jun  4 16:33:35 2024 [127.0.0.1:59918] taewan 550 not_exist_file: Can't find such file or directory.

Tue Jun  4 16:34:01 2024 [127.0.0.1:59918] taewan QUIT

Tue Jun  4 16:34:01 2024 [127.0.0.1:59918] taewan 221 Goodbye.

Tue Jun  4 16:34:01 2024 [127.0.0.1:59918] taewan LOG_OUT
[total service time : 1556 sec]

Tue Jun  4 16:34:22 2024 Server is terminated
```

quit 을 한 후에 client 와 연결된 srv 의 child process 가 종료되고, 이때 LOG_OUT 되며 child process 가 실행되었던 시간을 같이 기록합니다.

이후, ctrl+c 를 눌러 srv 가 완전 종료되면 이때도 기록합니다.

고찰

그동안 했던 과제를 모두 합치니 연결되는 부분에서 에러도 많이 나고 reply code 등 수정할 부분이 많아 시간을 많이 소모했습니다. 그러나 한 학기 동안 노력한 결과 FTP server 를 제 힘으로 구현했다는 사실이 매우 뿌듯하여 이와 관련된 server 쪽 공부를 방학 기간 동안에 한번 해보고 싶습니다.