

시스템 프로그래밍 실습

[FTP1-2]

Class : D

Professor : 최상호 교수님

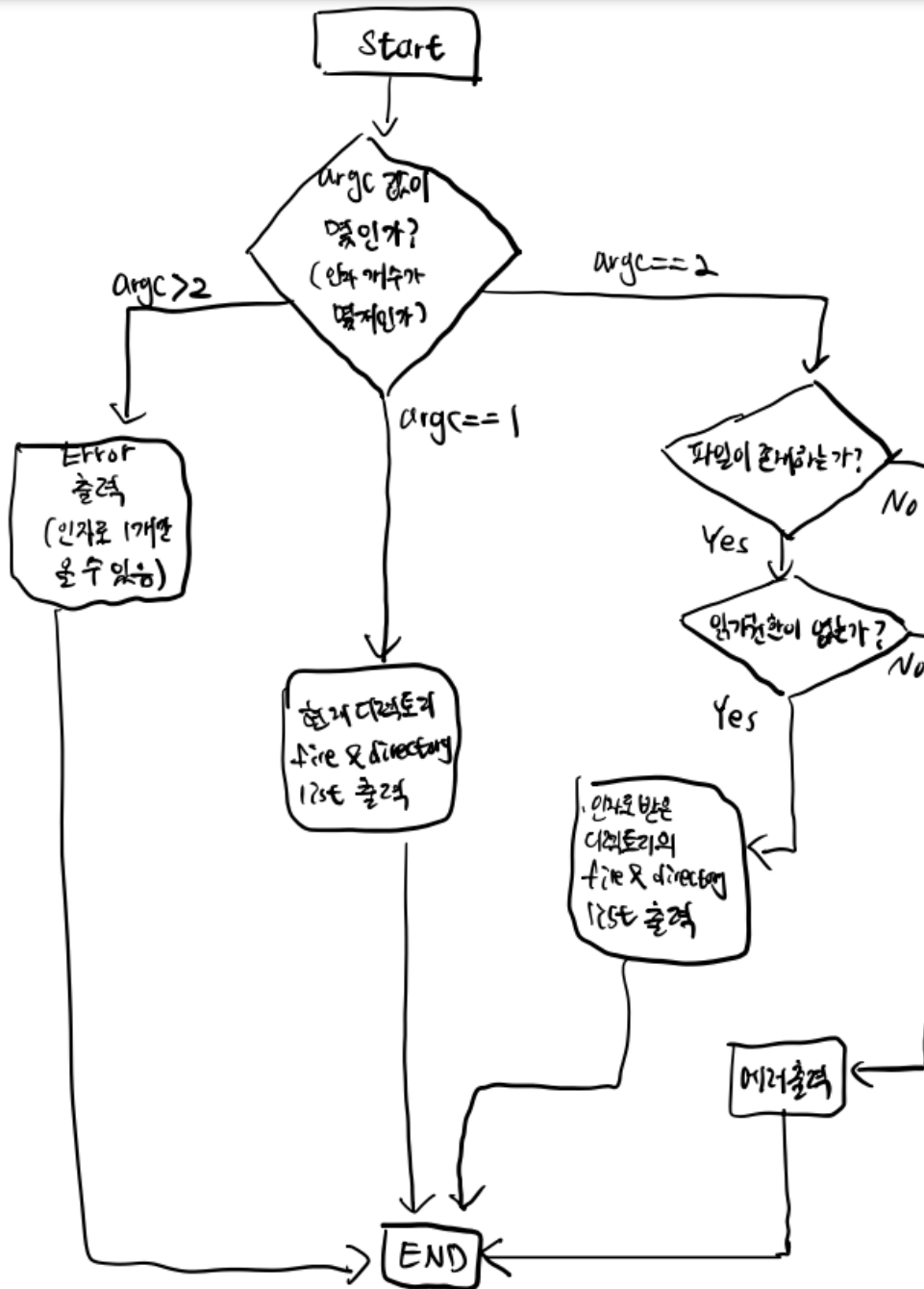
Student ID : 2020202092

Name : 강현민

Introduction

이번 [FTP 1-2](#) 과제는 FTP 서버를 구현하는 것을 최종 목표로 하는 프로젝트 중 첫 번째 단계인 FTP 명령어 구현 중 두 번째 과정으로, ls 명령어 구현을 위한 file system 을 학습하고, 실제로 ls 명령어를 구현하는 것을 목표로 한다. 먼저 터미널에 ./명령어를 입력한 경우 현재 디렉토리의 file list 를 출력하고, ./명령어 뒤에 인자로 디렉토리명을 입력할 경우 해당 디렉토리 내의 파일 리스트를 출력한다. 또한, 인자로 입력한 디렉토리가 접근할 수 없는 디렉토리이거나, 존재하지 않는 디렉토리명일 경우 그에 맞는 에러 문구를 출력한다. 그리고 명령어의 인자가 두개 이상 입력될 경우 이에 맞는 에러 메시지를 출력하여 예외 상황을 처리한다.

Flow chart



`./kw2020202092_ls` 명령어를 입력하면, 먼저 `argc` 변수를 통해 인자의 개수를 확인한다. 이 때 `argc` 가 2 보다 크면, 명령어의 인자로 두개 이상 입력된 경우이기 때문에 출력하고자 하는 디렉토리를 특정할 수 없다. 따라서 에러 문구를 출력한 후 종료한다. 그리고 `argc` 가 1 인 경우에는 현재 디렉토리 내에 있는 파일들의 이름을 모두 출력한 후 종료한다. 마지막으로 `argc` 가 2 인 경우에는 명령어의 인자로 한 개가 입력된 경우인데, 이 때는 `access` 함수를 통해 파일이 존재하는지, 그리고 접근 권한이 있는지

확인한 후, 하나라도 없을 경우 그에 맞는 에러 문구를 출력한 후 종료한다. 그리고 파일이 존재하고 접근 권한이 있는 경우에는 인자로 받은 디렉토리의 파일들의 이름을 모두 출력한 후 프로그램을 종료한다.

Pseudo code

```
int main(int argc, char* argv[]) {  
  
    declare directory stream dp;  
  
    declare directory entry pointer dirp;  
  
    declare char* dirname to store directory name;  
  
    if (argc is greater than 2) {  
        print error (more than 2 arguments for ls command);  
        exit;  
    }  
  
    else if(argc is equal to 2) {  
        store argv[1] into dirname;  
  
        if (cannot open directory of name of dirname) {  
            print "testls: caannot access 'dirname' : "  
  
            if (directory of name of dirname does not exists) {  
                print error  
                exit;  
            }  
  
            else if (the access to read the directory is denied) {  
                print error  
                exit;  
            }  
        }  
    }  
}
```

```

        }

    }

}

else {

    store "." Into dirname;

    if (there is no directory to open)

        exit

}

while (there is no directories to read) {

    print the name of directories in the current directory;

}

Close the directory stream 에||

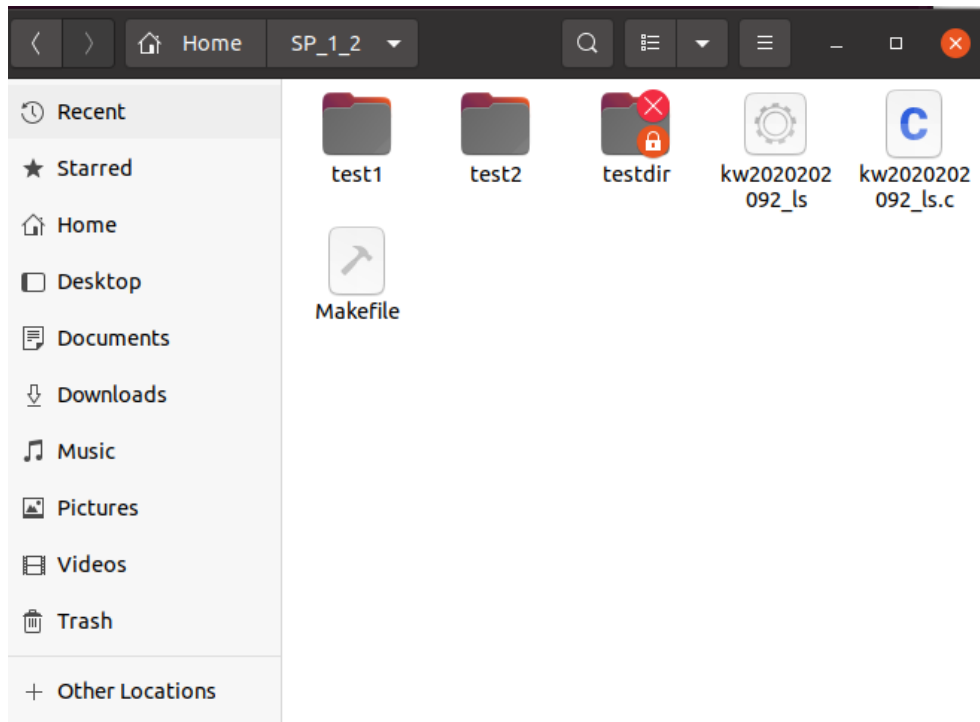
Return 0;

}

```

결과화면

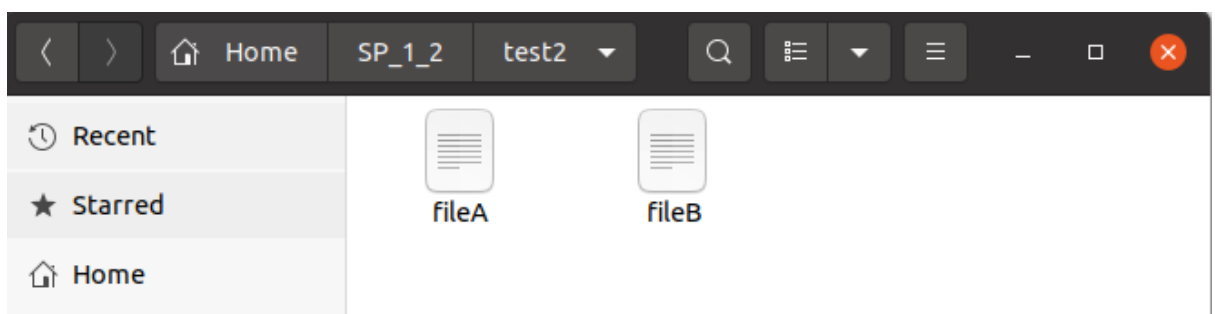
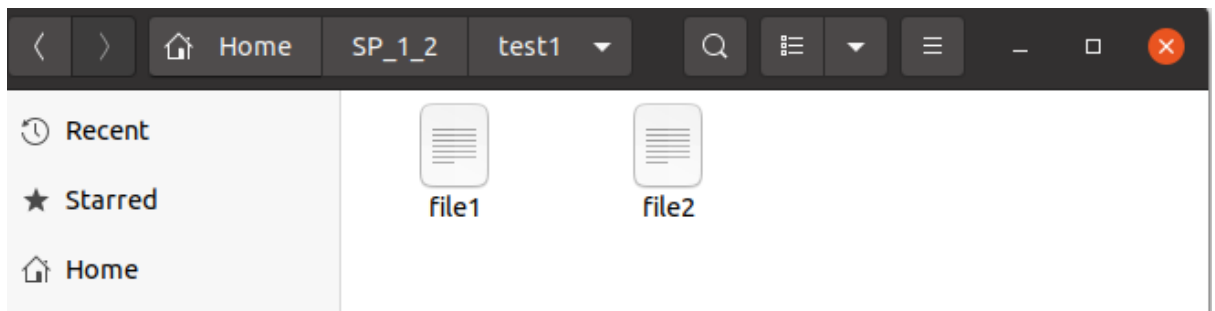
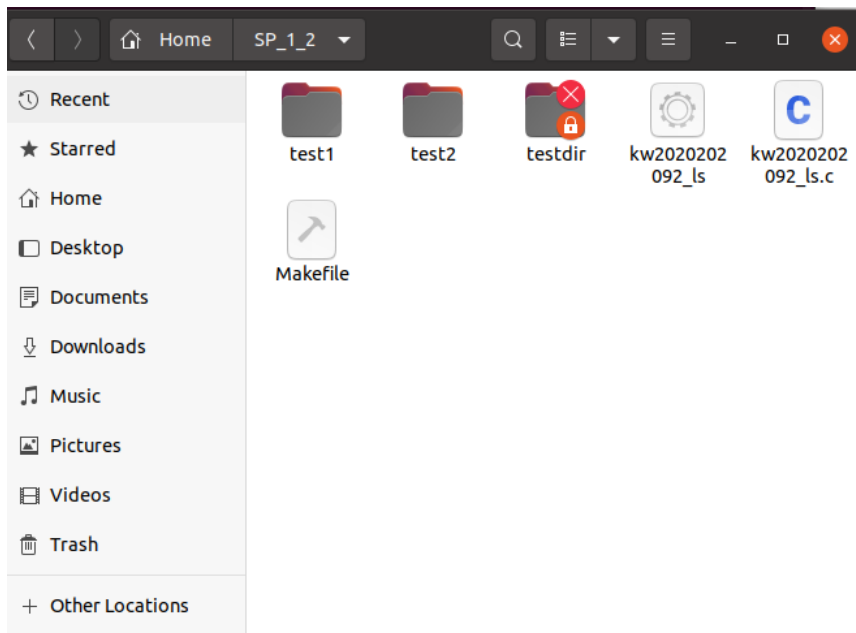
1) 인자를 쓰지 않는 경우 (argc == 1)



```
kw2020202092@ubuntu: ~/SP_1_2
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls
kw2020202092_ls
Makefile
testdir
kw2020202092_ls.c
..
.
test1
test2
kw2020202092@ubuntu:~/SP_1_2$ ls -al
total 48
drwxrwxr-x  5 kw2020202092 kw2020202092 4096 Apr  9 20:14 .
drwxr-xr-x 21 kw2020202092 kw2020202092 4096 Apr  6 18:39 ..
-rwxrwxr-x  1 kw2020202092 kw2020202092 16976 Apr  9 20:14 kw2020202092_ls
-rw-rw-r--  1 kw2020202092 kw2020202092 3311 Apr  9 20:12 kw2020202092_ls.c
-rw-rw-r--  1 kw2020202092 kw2020202092   77 Apr  9 20:14 Makefile
drwxrwxr-x  2 kw2020202092 kw2020202092 4096 Apr  7 05:37 test1
drwxrwxr-x  2 kw2020202092 kw2020202092 4096 Apr  7 05:38 test2
d-----  2 kw2020202092 kw2020202092 4096 Apr  8 23:36 testdir
kw2020202092@ubuntu:~/SP_1_2$
```

인자를 쓰지 않고 명령어만 입력하는 경우에는 현재 디렉토리에 있는 모든 파일과 디렉토리의 리스트를 출력하는데, 위의 캡처를 살펴보면 ls 명령어를 사용했을 때와 kw2020202092_ls 를 사용했을 때 동일하게 모든 파일들의 리스트가 출력되는 것을 확인할 수 있다.

2) 인자를 한 개 쓰는 경우 (argc == 2)

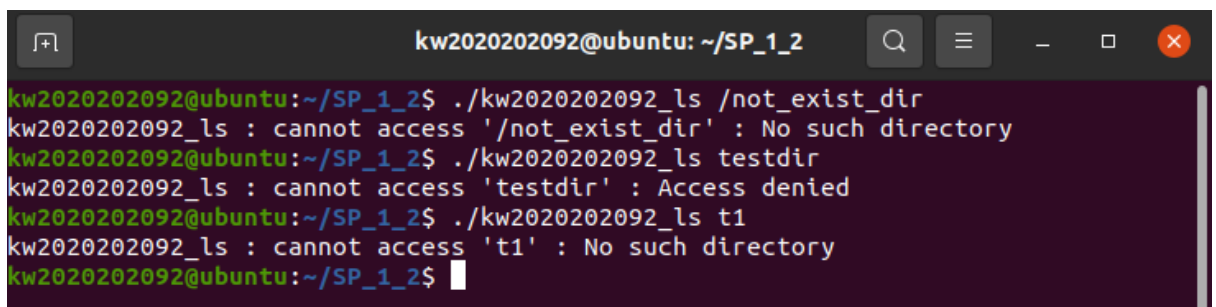



```

kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls
kw2020202092_ls
Makefile
testdir
kw2020202092_ls.c
..
.
test1
test2
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls test1
file2
file1
..
.
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls test2
fileB
fileA
..
.
kw2020202092@ubuntu:~/SP_1_2$

```

위와 같이 인자로 test1 을 쓰는 경우 현재 디렉토리에 있는 test1 디렉토리 내의 파일들의 리스트를 출력하는데, 위와 같이 file1 과 file2 가 출력되는 것을 확인할 수 있고, test2 디렉토리의 경우 fileB 와 fileA 가 정상적으로 출력되는 것을 확인할 수 있다.



```

kw2020202092@ubuntu: ~/SP_1_2
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls /not_exist_dir
kw2020202092_ls : cannot access '/not_exist_dir' : No such directory
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls testdir
kw2020202092_ls : cannot access 'testdir' : Access denied
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls t1
kw2020202092_ls : cannot access 't1' : No such directory
kw2020202092@ubuntu:~/SP_1_2$

```

또한 위와 같이 존재하지 않는 디렉토리를 인자로 입력할 경우 No such directory 에러를 출력하고, testdir 의 경우 read 권한이 없기 때문에 Access denied 에러가 의도대로 잘 출력되는 것을 확인할 수 있다.

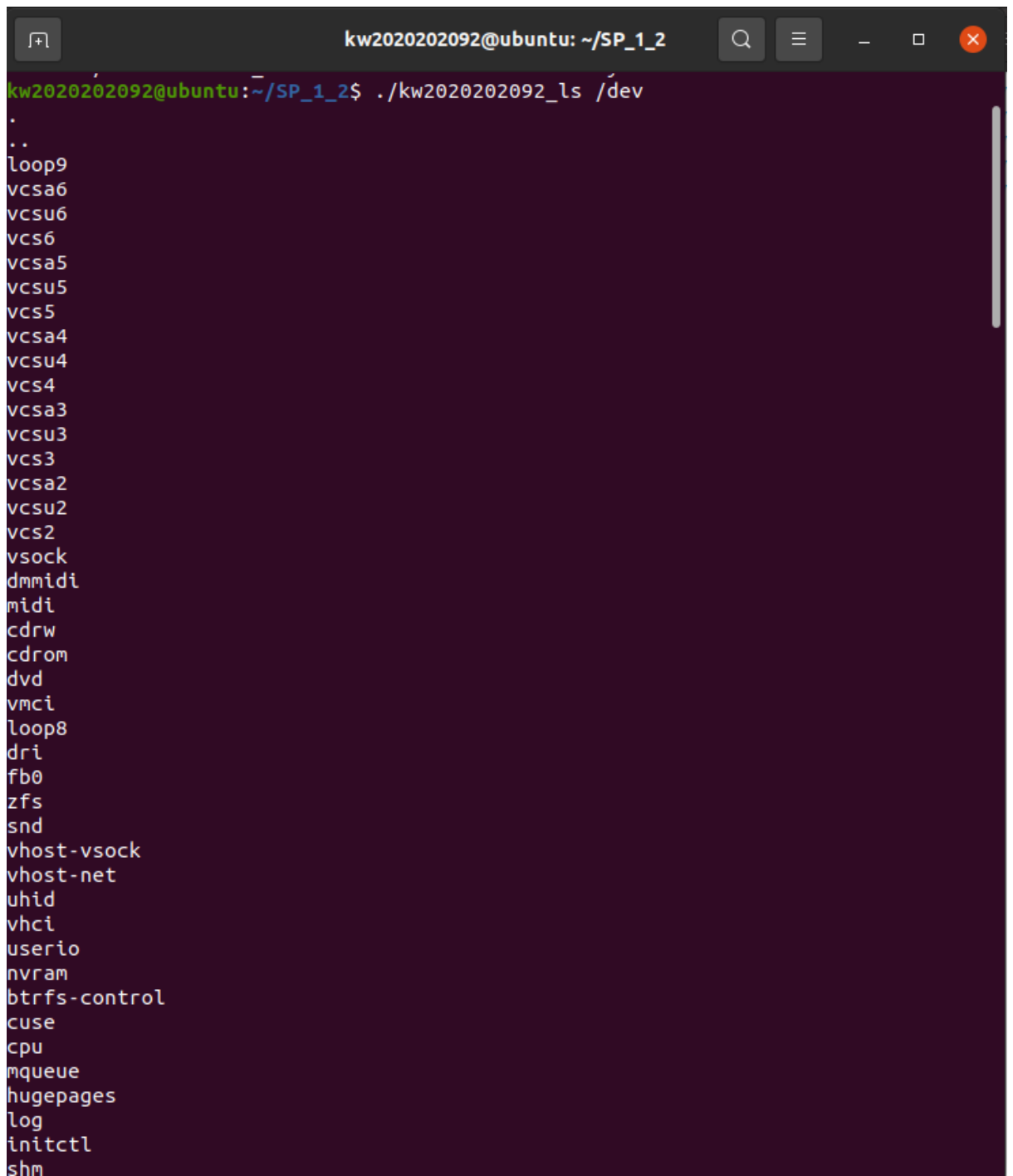
3) 인자가 두개 이상인 경우 (argc > 2)

인자가 두개 이상일 경우 ls 명령어를 수행하고자 하는 디렉토리가 특정되지 않기 때문에 에러문구를 출력해야 한다.

```
kw2020202092@ubuntu: ~/SP_1_2
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls /dir1 /dir2
only one directory path can be processed
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls test1 test2
only one directory path can be processed
kw2020202092@ubuntu:~/SP_1_2$
```

위와 같이 에러 메시지가 잘 출력되는 것을 확인할 수 있다.

4) `./kw2020202092_ls /dev`

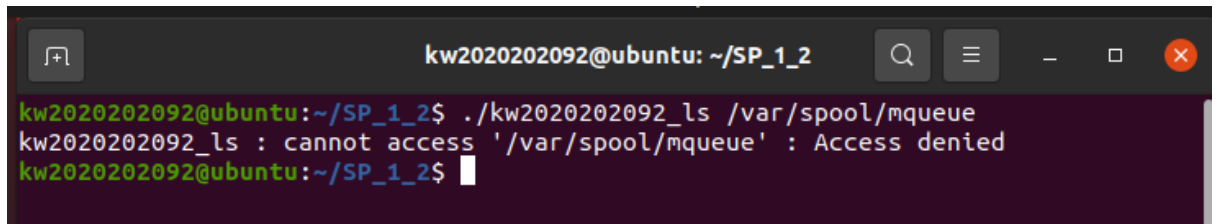


```
kw2020202092@ubuntu: ~/SP_1_2
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls /dev
.
..
loop9
vcsa6
vcsu6
vcs6
vcsa5
vcsu5
vcs5
vcsa4
vcsu4
vcs4
vcsa3
vcsu3
vcs3
vcsa2
vcsu2
vcs2
vsock
dmidi
midi
cdrw
cdrom
dvd
vmci
loop8
dri
fb0
zfs
snd
vhost-vsock
vhost-net
uhid
vhci
userio
nvram
btrfs-control
cuse
cpu
mqueue
hugepages
log
initctl
shm
```

```
kw2020202092@ubuntu: ~/SP_1_2
tty25
tty24
tty23
tty22
tty21
tty20
tty19
tty18
tty17
tty16
tty15
tty14
tty13
tty12
tty11
tty10
tty9
tty8
tty7
tty6
tty5
tty4
tty3
tty2
tty1
vcsa1
vcsu1
vcs1
vcsa
vcsu
vcs
tty0
console
tty
kmsg
urandom
random
full
zero
port
null
mem
rfkill
vga_arbiter
kw2020202092@ubuntu:~/SP_1_2$
```

위와 같이 `./kw2020202092_ls /dev` 를 입력하는 경우, 절대경로를 통해 `dev` 디렉토리에 있는 파일의 리스트가 출력되는 것을 확인할 수 있다.

5) ./kw2020202092_ls /var/spool/mqueue

A terminal window with a dark background. The title bar shows 'kw2020202092@ubuntu: ~/SP_1_2'. The terminal text shows a user prompt 'kw2020202092@ubuntu:~/SP_1_2\$' followed by the command './kw2020202092_ls /var/spool/mqueue'. The next line shows the output 'kw2020202092_ls : cannot access '/var/spool/mqueue' : Access denied'. The prompt 'kw2020202092@ubuntu:~/SP_1_2\$' is shown again on the third line with a cursor.

```
kw2020202092@ubuntu:~/SP_1_2$ ./kw2020202092_ls /var/spool/mqueue
kw2020202092_ls : cannot access '/var/spool/mqueue' : Access denied
kw2020202092@ubuntu:~/SP_1_2$
```

절대경로를 /var/spool/mqueue 로 명령어를 입력할 경우, 접근 권한이 없기 때문에 Access denied 문구가 출력되는 것을 확인할 수 있다.

고찰

이번 과제를 통해 파일 및 디렉토리 관리와 관련된 리눅스 C 언어의 함수나 문법 등을 학습할 수 있었다. 특히, dirent.h 헤더 파일과 관련된 함수인 opendir, closedir, readdir 등의 함수들을 사용하여 디렉토리를 열고 읽는 방법에 대해 새로 학습하고, 이를 응용하여 ls 커맨드를 구현하는 방법에 대해 알 수 있었다. 또한, 파일 및 디렉토리에 대한 접근 권한 및 존재 여부를 확인할 수 있는 access 함수의 사용법에 대해 새로 알 수 있었고, 프로그램의 구조를 통해 예외 상황을 처리하는 과정을 실습하면서 더욱 익숙해질 수 있었다.

Reference

basename 함수 사용법: <https://www.it-note.kr/20>