

시스템프로그래밍 과제 보고서

Basic

담당 교수: 김태석

학과: 컴퓨터정보공학부

학번: 2023202070

이름: 최현진


제출일: 2025.03.27

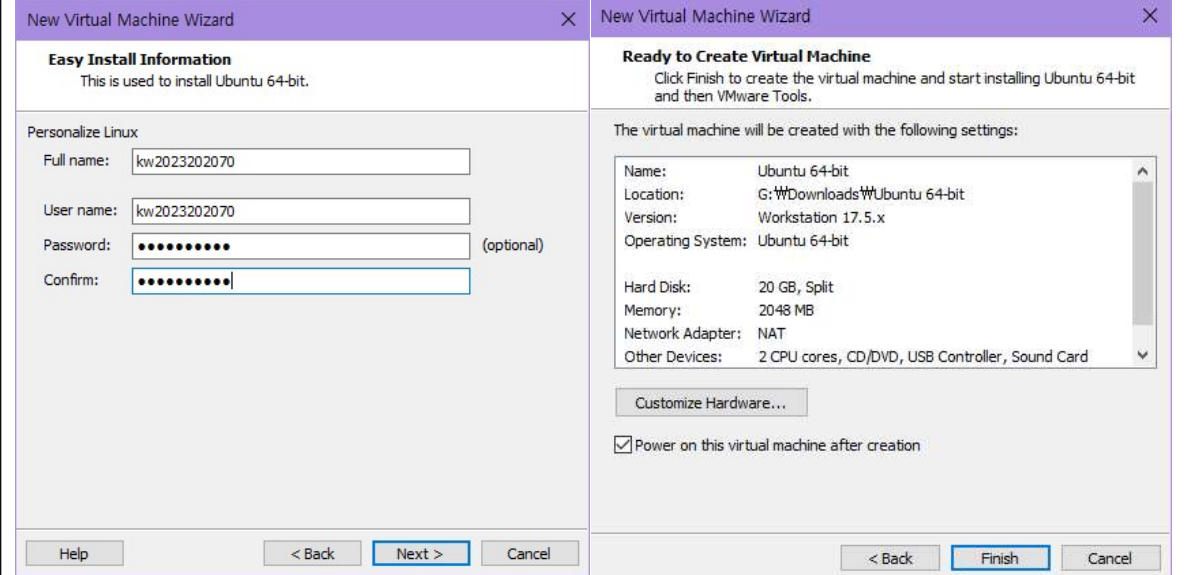
Introduction

이번 Basic 과제는 Ubuntu 환경에서의 기본적인 시스템 프로그래밍 실습 과정을 포함한다. 과제는 크게 세 가지 주제로 나뉘는데, 먼저 특정 버전의 우분투 iso 파일을 다운로드하고 가상 머신을 툴을 사용해 설치하는 과제가 있다. 이 때, 계정 이름은 kw + 본인의 학번으로 설정한다. 다음으로 다양한 리눅스 명령어의 핵심 설명과 직접 실행해본 결과를 캡처 사진과 함께 작성하는 것을 포함한다. 특정 명령어들은 사용해야 할 옵션이 지정되어 있다. 마지막으로 리눅스 기본 편집기인 vi 편집기의 각 모드와 옵션을 실습한다. 또한 Makefile을 이용하여 c 파일을 컴파일하고 실행하는 과정을 설명하고 결과 화면을 첨부하는 과제가 있다.

결과 화면

Basic-1. Ubuntu Installation

 ubuntu-20.04.6-desktop-amd64.iso	2025-03-21 오전 12:13	디스크 이미지 파일	4,249,476...
Ubuntu 20.04.6 Desktop 64bit 운영체제를 설치하기 위해 ISO file을 다운로드했다.			



VMWare를 통해 ID는 학번으로 설정하여 가상 머신을 생성하고 우분투를 설치했다.

```
kw2023202070@ubuntu: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
kw2023202070@ubuntu:~$ uname -r  
5.15.0-134-generic  
kw2023202070@ubuntu:~$
```

우분투를 설치하고 `uname -r` 명령어 실행 결과 사용 중인 리눅스 커널 버전을 확인했다.

Basic-2. Linux Commands

1. man

man [option] name

입력한 name(명령어나 함수,...)에 대한 온라인 매뉴얼 페이지를 표시한다.

Section Number: 섹션 번호를 지정하면 해당 섹션에서만 검색한다.

(1) 실행 가능한 프로그램 또는 셸 명령어

(2) 시스템 콜

(3) C 라이브러리 함수

...

매뉴얼은 NAME(명령어 또는 함수의 이름과 간단한 설명), SYNOPSIS(사용법), DESCRIPTION(명령어 또는 함수의 상세 설명), EXAMPLE, SEE ALSO(관련 명령어나 함수) 외에도 OPTIONS, EXIT STATUS, RETURN VALUE, ERRORS, ENVIRONMENT, FILES, VERSIONS, CONFORMING TO, NOTES, BUGS, AUTHORS를 포함한다.

e.g. **\$ man ls**

```
LS(1)                                User Commands                                LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
    fied.

    Mandatory arguments to long options are mandatory for short options
    too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
        print C-style escapes for nongraphic characters

Manual page ls(1) line 1 (press h for help or q to quit)
```

2. ls

ls [OPTION]... [FILE]...

디렉토리의 내용들을 나열한다.

OPTION:

-a: 숨겨진 파일('.')으로 시작하는 파일)도 표시

-F: 파일 종류 표시(/, *)

-l: 파일 정보를 자세하게 출력

e.g. \$ ls

```
kw2023202070@ubuntu:~$ ls
Desktop  Downloads  Pictures  snap      Templates  work
Documents Music      Public    splab_commands  Videos
```

\$ ls -a

```
kw2023202070@ubuntu:~$ ls -a
.      .cache      Downloads  Music      snap      work
..     .config     .gnupg     Pictures   splab_commands
.bash_logout Desktop     .local     .profile   Templates
.bashrc Documents   .mozilla   Public     Videos
```

\$ ls -F

```
kw2023202070@ubuntu:~$ ls -F
Desktop/  Downloads/  Pictures/  snap/      Templates/  work/
Documents/ Music/      Public/    splab_commands*  Videos/
```

\$ ls -l

```
kw2023202070@ubuntu:~$ ls -l
total 44
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Desktop
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Documents
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Downloads
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Music
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Pictures
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Public
drwx----- 3 kw2023202070 kw2023202070 4096 Mar 21 07:48 snap
-rwxrwxrwx 1 kw2023202070 kw2023202070 2690 Mar 20 02:56 splab_commands
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Templates
drwxr-xr-x 2 kw2023202070 kw2023202070 4096 Mar 21 07:47 Videos
drwxrwxr-x 4 kw2023202070 kw2023202070 4096 Mar 26 08:56 work
kw2023202070@ubuntu:~$
```

3. pwd

`pwd [OPTION]`

현재 작업 중인 디렉토리의 절대 경로를 출력한다.

e.g. `$ pwd`

```
kw2023202070@ubuntu:~$ pwd
/home/kw2023202070
```

4. cd

`cd [-L|-P] [dir]`

현재 셸의 작업 디렉토리를 변경한다.

- 인자가 없을 경우, 홈 디렉토리로 이동한다.
- `cd ~`: 홈 디렉토리로 이동한다.
- `cd -`: 이전 디렉토리로 이동한다.
- `cd ..`: 현재 디렉토리의 상위 디렉토리로 이동한다.

e.g.

```
kw2023202070@ubuntu:~$ pwd
/home/kw2023202070
kw2023202070@ubuntu:~$ ls
Desktop    Downloads  Pictures   snap        Templates  work
Documents  Music      Public     splab_commands  Videos
kw2023202070@ubuntu:~$ cd work
kw2023202070@ubuntu:~/work$ pwd
/home/kw2023202070/work
kw2023202070@ubuntu:~/work$ cd .
kw2023202070@ubuntu:~/work$ cd ..
kw2023202070@ubuntu:~$ cd work
kw2023202070@ubuntu:~/work$ cd ~
kw2023202070@ubuntu:~$ cd -
/home/kw2023202070/work
```

5. cat

`cat [OPTION] [FILE]...`

하나 이상의 파일들을 연결(concatenate)하고 표준 출력(standard output)에 출력한다.

e.g.

```
kw2023202070@ubuntu:~/work$ cat file1.txt
Hello This is file 1
kw2023202070@ubuntu:~/work$ cat file2.txt
Hello This is file 2
kw2023202070@ubuntu:~/work$ cat file1.txt file2.txt
Hello This is file 1
Hello This is file 2
kw2023202070@ubuntu:~/work$
```


6. chmod

파일 또는 디렉토리의 권한을 변경한다.

chmod [OPTION]... MODE[,MODE]... FILE...

Symbolic mode: [ugoa...][[-+=][perms...]]

대상: u(user=owner), g(group), o(others), a(all, 기본값)

연산자: +(권한 추가), -(권한 제거), =(특정 권한만 설정)

권한: r(read), w(write), x(execute), s(setuid/setgid 설정), t(sticky bit 설정)

chmod [OPTION]... OCTAL-MODE FILE...

Octal mode:

각 대상(user, group, others)에 대한 권한을 더하여 8진수 세 개로 표현한다.

r=4, w=2, x=1

e.g. **\$ chmod u-w,g-w,o-r hello.txt**

: hello.txt: user와 group에 쓰기 권한 제거, others에 읽기 권한 제거

```
kw2023202070@ubuntu:~/work$ ls -al
total 28
drwxrwxr-x 3 kw2023202070 kw2023202070 4096 Mar 24 09:03 .
drwxr-xr-x 17 kw2023202070 kw2023202070 4096 Mar 24 09:03 ..
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ chmod u-w,g-w,o-r hello.txt
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-r--r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
```

\$ chmod 644 hello.txt

: hello.txt user에 6, 즉 읽기와 쓰기 권한으로 변경, group과 others에 4, 즉 읽기 권한으로 변경

```
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-r--r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ chmod 644 hello.txt
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
```

7. mkdir

mkdir [OPTION] DIRECTORY...

새로운 디렉토리를 생성한다.

OPTION:

-m: chmod와 같은 방식으로 생성할 디렉토리에 특정 권한을 적용할 수 있다.

-v: 디렉토리를 만들 때마다 생성되었다는 메시지를 출력한다.

-p: 부모 디렉토리가 존재하지 않을 경우 자동 생성한다.

e.g. **\$ mkdir SP_lecture**

```
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ mkdir SP_lecture
kw2023202070@ubuntu:~/work$ ls -l
total 24
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 25 08:19 SP_lecture
```


8. rmdir

rmdir [OPTION]... DIRECTORY...

비어 있는 디렉토리를 삭제한다..

OPTION:

-v: 디렉토리를 삭제할 때마다 메시지를 출력한다.

-p: 부모 디렉토리까지 삭제한다.

e.g. **\$ rmdir SP_lecture/**

```
kw2023202070@ubuntu:~/work$ ls -l
total 24
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 25 08:19 SP_lecture
kw2023202070@ubuntu:~/work$ rmdir SP_lecture/
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
```

9. rm

rm [OPTION]... FILE...

파일 또는 디렉토리를 삭제한다.

OPTION:

-r, -R, --recursive: 디렉토리나 그 내부 파일들까지 재귀적으로 삭제한다.

-i: 삭제할 때마다 그 전에 확인한다.

e.g. **\$ rm -r LINUX: LINUX** 디렉토리가 삭제되었다.

```
kw2023202070@ubuntu:~/work$ ls -l
total 24
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 26 07:11 LINUX
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ rm -r LINUX
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$
```

10. cp

cp [OPTION]... SOURCE DEST: SOURCE를 DEST에 복사

cp [OPTION]... SOURCE... DIRECTORY: 여러 개의 SOURCE를 DIRECTORY에 복사

파일 및 디렉토리를 복사한다.

e.g. \$ cp hello.txt hello.copy.txt: hello.txt가 hello.copy.txt에 복사되었다.

```
kw2023202070@ubuntu:~/work$ ls -l
total 20
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ cp hello.txt hello_copy.txt
kw2023202070@ubuntu:~/work$ ls -l
total 24
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 26 07:17 hello_copy.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
```

\$ cp SP_lab/* .: SP_lab 디렉토리 안에 있는 모든 파일이 현재 디렉토리에 복사되었다.

```
kw2023202070@ubuntu:~/work$ ls -l
total 24
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 26 07:17 hello_copy.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
kw2023202070@ubuntu:~/work$ cp SP_lab/* .
kw2023202070@ubuntu:~/work$ ls -l
total 32
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 15 Mar 26 07:18 fileA.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 15 Mar 26 07:18 fileC.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 26 07:17 hello_copy.txt
-rw-r--r-- 1 kw2023202070 kw2023202070 41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 24 09:03 SP_lab
```


11. mv

mv [OPTION]... SOURCE DEST

파일 또는 디렉토리를 이동하거나 그 이름을 변경한다.

e.g. **\$ mv hello_copy.txt /home/kw2023202070/work/ex**

: hello_copy.txt가 ex 디렉토리로 이동되었다.

\$ mv ex LINUX: ex 디렉토리의 이름이 LINUX로 변경되었다.

```
kw2023202070@ubuntu:~/work$ ls
ex      file2.txt  fileA.txt  hello_copy.txt  SP_lab
file1.txt  file3.txt  fileC.txt  hello.txt
kw2023202070@ubuntu:~/work$ mv hello_copy.txt /home/kw2023202070/work/ex
kw2023202070@ubuntu:~/work$ ls
ex  file1.txt  file2.txt  file3.txt  fileA.txt  fileC.txt  hello.txt  SP_lab
kw2023202070@ubuntu:~/work$ cd ex
kw2023202070@ubuntu:~/work/ex$ ls
hello_copy.txt
kw2023202070@ubuntu:~/work/ex$ cd ..
kw2023202070@ubuntu:~/work$ mv ex LINUX
kw2023202070@ubuntu:~/work$ ls
file1.txt  file3.txt  fileC.txt  LINUX
file2.txt  fileA.txt  hello.txt  SP_lab
kw2023202070@ubuntu:~/work$
```

12. ln

파일 간 링크를 생성한다. 기본적으로 하드 링크를 생성하며, -s 옵션을 사용하여 심볼릭(소프트) 링크를 생성할 수 있다.

ln [OPTION]... TARGET [LINK_NAME]: TARGET 파일의 링크를 LINK_NAME으로 생성한다.

Hard Link:

- 존재하는 파일에 대해서만 작성 가능
- 같은 파일 시스템 간에서만 작성 가능
- 원본 파일과 같은 inode

Symbolic Link:

- 존재하지 않는 파일에 대해 작성 가능
- 다른 파일 시스템을 넘나들어 작성 가능
- 원본 파일과 다른 inode

e.g. `$ ln fileA.txt fileB.txt`: fileA.txt의 하드 링크를 fileB.txt로 생성했다.

이후 fileB.txt의 내용을 수정하고 fileA.txt를 확인한 결과 똑같이 수정되었다.

두 파일은 같은 inode를 공유하기 있기 때문이다.

```
kw2023202070@ubuntu:~/work/SP_lab$ ls
fileA.txt  fileB.txt  fileC.txt
kw2023202070@ubuntu:~/work/SP_lab$ ls
fileA.txt  fileC.txt
kw2023202070@ubuntu:~/work/SP_lab$ cat fileA.txt
This is file A
kw2023202070@ubuntu:~/work/SP_lab$ ln fileA.txt fileB.txt
kw2023202070@ubuntu:~/work/SP_lab$ cat fileB.txt
This is file A
kw2023202070@ubuntu:~/work/SP_lab$ vi fileB.txt
kw2023202070@ubuntu:~/work/SP_lab$ cat fileA.txt
This is file B after the change.
kw2023202070@ubuntu:~/work/SP_lab$ cat fileB.txt
This is file B after the change.
kw2023202070@ubuntu:~/work/SP_lab$
```

`$ ln -s fileC.txt fileE.txt`

: -s 옵션을 사용하여 fileC.txt를 참조하는 심볼릭 링크 fileE.txt를 생성했다.

fileE.txt를 확인한 결과 fileC.txt의 내용을 보여준다.

이후 원본 파일 fileC.txt를 삭제했기 때문에 fileE.txt의 확인에 실패했다.

```
kw2023202070@ubuntu:~/work/SP_lab$ cat fileC.txt
This is file C
kw2023202070@ubuntu:~/work/SP_lab$ ln -s fileC.txt fileE.txt
kw2023202070@ubuntu:~/work/SP_lab$ ls -l
total 12
-rw-rw-r-- 2 kw2023202070 kw2023202070 33 Mar 26 08:22 fileA.txt
-rw-rw-r-- 2 kw2023202070 kw2023202070 33 Mar 26 08:22 fileB.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 15 Mar 26 07:18 fileC.txt
lrwxrwxrwx 1 kw2023202070 kw2023202070  9 Mar 26 08:25 fileE.txt -> fileC.txt
kw2023202070@ubuntu:~/work/SP_lab$ cat fileE.txt
This is file C
kw2023202070@ubuntu:~/work/SP_lab$ rm fileC.txt
kw2023202070@ubuntu:~/work/SP_lab$ cat fileE.txt
cat: fileE.txt: No such file or directory
kw2023202070@ubuntu:~/work/SP_lab$ ls -l
total 8
-rw-rw-r-- 2 kw2023202070 kw2023202070 33 Mar 26 08:22 fileA.txt
-rw-rw-r-- 2 kw2023202070 kw2023202070 33 Mar 26 08:22 fileB.txt
lrwxrwxrwx 1 kw2023202070 kw2023202070  9 Mar 26 08:25 fileE.txt -> fileC.txt
kw2023202070@ubuntu:~/work/SP_lab$
```


13. touch

`touch [OPTION]... FILE...`

파일의 타임스탬프를 변경하거나 새 파일을 생성한다.

e.g. `$ touch empty.txt`

: 처음 실행 시, 새로운 empty.txt 파일이 생성되었다.

이후 실행 시, empty.txt의 atime과 mtime이 갱신되었다.

```
kw2023202070@ubuntu:~/work$ ls
file1.txt  file3.txt  fileC.txt  LINUX
file2.txt  fileA.txt  hello.txt  SP_lab
kw2023202070@ubuntu:~/work$ touch empty.txt
kw2023202070@ubuntu:~/work$ ls
empty.txt  file2.txt  fileA.txt  hello.txt  SP_lab
file1.txt  file3.txt  fileC.txt  LINUX
kw2023202070@ubuntu:~/work$ ls -l
total 32
-rw-rw-r-- 1 kw2023202070 kw2023202070  0 Mar 26 08:56 empty.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  15 Mar 26 07:18 fileA.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  15 Mar 26 07:18 fileC.txt
-rw-r--r-- 1 kw2023202070 kw2023202070  41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 26 07:28 LINUX
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 26 08:25 SP_lab
kw2023202070@ubuntu:~/work$ touch empty.txt
kw2023202070@ubuntu:~/work$ ls -l
total 32
-rw-rw-r-- 1 kw2023202070 kw2023202070  0 Mar 26 08:57 empty.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file1.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 21 Mar 24 09:03 file2.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  15 Mar 26 07:18 fileA.txt
-rw-rw-r-- 1 kw2023202070 kw2023202070  15 Mar 26 07:18 fileC.txt
-rw-r--r-- 1 kw2023202070 kw2023202070  41 Mar 24 09:03 hello.txt
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 26 07:28 LINUX
drwxrwxr-x 2 kw2023202070 kw2023202070 4096 Mar 26 08:25 SP_lab
kw2023202070@ubuntu:~/work$
```


14. ps

ps [options]

현재 실행 중인 프로세스 목록을 출력한다.

옵션 없이 사용하면, 현재 셸에서 실행 중인 프로세스만 출력한다.

options:

-e: 전체 시스템 프로세스 표시

-f: 풀 포맷으로 표시

e.g. \$ ps

\$ ps -ef

```
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
 2203 pts/0        00:00:00 bash
 3278 pts/0        00:00:00 ps
kw2023202070@ubuntu:~$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1         0  0  08:16 ?        00:00:01 /sbin/init auto noprompt
root           2         0  0  08:16 ?        00:00:00 [kthreadd]
root           3         2  0  08:16 ?        00:00:00 [rcu_gp]
root           4         2  0  08:16 ?        00:00:00 [rcu_par_gp]
root           5         2  0  08:16 ?        00:00:00 [slub_flushwq]
root           6         2  0  08:16 ?        00:00:00 [netns]
root           8         2  0  08:16 ?        00:00:00 [kworker/0:0H-events_highp]
root          10         2  0  08:16 ?        00:00:00 [mm_percpu_wq]
root          11         2  0  08:16 ?        00:00:00 [rcu_tasks_rude_]
```

15. exit

exit

현재 셸을 종료한다.

e.g. \$ exit

csh(C shell)를 설치하고 실행하여 새로운 셸을 시작했다.

이 때의 프로세스 목록은 bash, csh, ps이었다.

이후 csh를 종료하고 다시 ps를 확인한 결과 csh가 사라졌다.

```
kw2023202070@ubuntu:~$ sudo apt-get install csh
[sudo] password for kw2023202070:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  csh
```

```
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
  2203 pts/0        00:00:00 bash
  3875 pts/0        00:00:00 ps
kw2023202070@ubuntu:~$ csh
% ps
  PID TTY          TIME CMD
  2203 pts/0        00:00:00 bash
  3876 pts/0        00:00:00 csh
  3878 pts/0        00:00:00 ps
% exit
% exit
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
  2203 pts/0        00:00:00 bash
  3880 pts/0        00:00:00 ps
kw2023202070@ubuntu:~$
```

```
16. kill  
kill [ -s signal | -p ] [ -a ] [ -- ] pid ...  
프로세스에 신호(signal)를 보낸다.  
기본적으로 SIGTERM (종료 요청) 신호를 보낸다.
```

```
kill [ -s signal | -p ] [ -a ] [ -- ] pid ...
```

프로세스에 신호(signal)를 보낸다.

기본적으로 SIGTERM (종료 요청) 신호를 보낸다.

프로세스에 신호(signal)를 보낸다.

기본적으로 SIGTERM (종료 요청) 신호를 보낸다.

기본적으로 SIGTERM (종료 요청) 신호를 보낸다.

e.g. \$ kill 3972

yes 명령어를 통해 "my name"이 무한 반복 출력되고 있었다.

실행 중인 전체 프로세스 목록(ps -e)의 끝 부분(tail)만을 출력한 결과, yes 명령어를 실행한 프로세스의 pid는 3975이었다.

이후 kill 명령을 통해 yes 프로세스를 종료했기 때문에 해당 셀에 Terminated를 확인했다.

```
kw2023202070@ubuntu:~/work$ yes my name
```

```

my name kw2023202070@ubuntu:~$ ps -e | tail
my name      3283 ?      00:00:00 kworker/0:2-cgr
my name      oup_destroy
my name      3874 ?      00:00:00 kworker/u256:2-
my name      events_unbound
my name      3893 ?      00:00:00 kworker/1:0-eve
my name      nts
my name      3912 ?      00:00:00 kworker/u256:1-
my name      events_unbound
my name      3925 pts/1    00:00:00 bash
my name      3972 pts/1    00:00:03 yes
my name      3973 ?      00:00:02 kworker/u256:3-
my name      events_unbound
my name      3974 ?      00:00:00 kworker/u256:4-
my name      events_unbound
my name      3975 pts/0    00:00:00 ps
my name      3976 pts/0    00:00:00 tail
my name      kw2023202070@ubuntu:~$ kill 3972
my name      kw2023202070@ubuntu:~$ ps -e | tail
my name      3280 ?      00:00:00 kworker/u256:0-
my name      events_unbound
my name      3283 ?      00:00:00 kworker/0:2-cgr
my name      oup_destroy
my name      3874 ?      00:00:00 kworker/u256:2-
my name      events_unbound
my name      3893 ?      00:00:00 kworker/1:0-eve
my name      nts
my name      3912 ?      00:00:01 kworker/u256:1-
my name      events_unbound
my name      3925 pts/1    00:00:00 bash
my name      3973 ?      00:00:03 kworker/u256:3-
my name      events_unbound
my name      3974 ?      00:00:00 kworker/u256:4-
my name      events_unbound
my name      3977 pts/0    00:00:00 ps
my name      3978 pts/0    00:00:00 tail
Terminated
kw2023202070@ubuntu:~/work$ █
kw2023202070@ubuntu:~$ █

```

`$ kill -9`

vi 실행 후 ctrl+z를 눌러 vi를 중단시켰다.

이후 ps를 확인한 결과 vi가 중단된 상태로 실행중이었다. (pid 4070)

-9(SIGKILL) 신호를 보내 강제 종료시켰다.

```
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
 2203 pts/0        00:00:00 bash
 4069 pts/0        00:00:00 ps
kw2023202070@ubuntu:~$ vi hello

[1]+  Stopped                  vi hello
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
 2203 pts/0        00:00:00 bash
 4070 pts/0        00:00:00 vi
 4071 pts/0        00:00:00 ps
kw2023202070@ubuntu:~$ kill -9 4070
kw2023202070@ubuntu:~$ ps
  PID TTY          TIME CMD
 2203 pts/0        00:00:00 bash
 4072 pts/0        00:00:00 ps
[1]+  Killed                  vi hello
kw2023202070@ubuntu:~$
```

17. passwd

`passwd [options]`

사용자의 비밀번호를 변경한다.

루트 사용자는 모든 계정의 비밀번호를 변경할 수 있다.

e.g. `$ passwd`

기존 비밀번호와 새 비밀번호를 입력하여 비밀번호를 변경했다.

```
kw2023202070@ubuntu:~$ passwd
Changing password for kw2023202070.
Current password:
New password:
Retype new password:
passwd: password updated successfully
```

18. uname

uname [options]

시스템 정보를 출력한다. 옵션 없이 실행하면 -s 옵션과 동일하게 커널 이름을 출력한다.

options:

-r: 커널 릴리즈

-m: 머신 하드웨어 이름

-a: 모든 정보

e.g.

```
kw2023202070@ubuntu:~$ uname
Linux
kw2023202070@ubuntu:~$ uname -r
5.15.0-134-generic
kw2023202070@ubuntu:~$ uname -m
x86_64
kw2023202070@ubuntu:~$ uname -a
Linux ubuntu 5.15.0-134-generic #145~20.04.1-Ubuntu SMP Mon Feb 17 13:27:16 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
kw2023202070@ubuntu:~$
```

19. wc

wc [options]... [FILE]...

각 파일의 줄 수(newline), 단어 수, 바이트 수를 출력한다.

e.g. **\$ wc hello.txt**

```
kw2023202070@ubuntu:~/work$ cat hello.txt
hello world
My Name is N~~~
How are you?
kw2023202070@ubuntu:~/work$ wc hello.txt
 3  9 41 hello.txt
kw2023202070@ubuntu:~/work$
```

20. echo

echo [OPTION]... [STRING]...

문자열(STRING)을 표준 출력에 출력한다.

e.g.

\$HOME은 환경 변수로 정의된 홈 디렉토리 경로이며, ~는 홈 디렉토리를 뜻하는 셸 기호이다.

```
kw2023202070@ubuntu:~$ echo helloworld
helloworld
kw2023202070@ubuntu:~$ echo $HOME
/home/kw2023202070
kw2023202070@ubuntu:~$ echo ~
/home/kw2023202070
kw2023202070@ubuntu:~$
```


21. alias

단어를 다른 문자열(명령어)로 대체한다.

`alias [name='value'] ...`

`alias`: 현재 설정된 `alias` 목록 출력

e.g. `$ alias myls='ls -al'`

`mysls`라는 명령어가 정의되지 않아 오류가 발생했다.

이후 `ls -al` 명령어를 대체하는 `mysls`를 정의하고 실행할 수 있었다.

정의된 `alias` 목록에서 `mysls`를 확인했다.

```
kw2023202070@ubuntu:~/work$ myls
```

```
Command 'mysls' not found, did you mean:
```

```
command 'tyls' from deb terminology (1.6.0-2)
```

```
command 'mmls' from deb sleuthkit (4.6.7-1build1)
```

```
Try: sudo apt install <deb name>
```

```
kw2023202070@ubuntu:~/work$ alias myls='ls -al'
```

```
kw2023202070@ubuntu:~/work$ myls
```

```
total 40
```

```
drwxrwxr-x  4 kw2023202070 kw2023202070 4096 Mar 26 08:56 .
drwxr-xr-x 17 kw2023202070 kw2023202070 4096 Mar 26 09:24 ..
-rw-rw-r--  1 kw2023202070 kw2023202070    0 Mar 26 08:57 empty.txt
-rw-rw-r--  1 kw2023202070 kw2023202070   21 Mar 24 09:03 file1.txt
-rw-rw-r--  1 kw2023202070 kw2023202070   21 Mar 24 09:03 file2.txt
-rw-rw-r--  1 kw2023202070 kw2023202070 2001 Mar 24 09:03 file3.txt
-rw-rw-r--  1 kw2023202070 kw2023202070   15 Mar 26 07:18 fileA.txt
-rw-rw-r--  1 kw2023202070 kw2023202070   15 Mar 26 07:18 fileC.txt
-rw-r--r--  1 kw2023202070 kw2023202070   41 Mar 24 09:03 hello.txt
drwxrwxr-x  2 kw2023202070 kw2023202070 4096 Mar 26 07:28 LINUX
drwxrwxr-x  2 kw2023202070 kw2023202070 4096 Mar 26 08:25 SP_lab
```

```
kw2023202070@ubuntu:~/work$ alias
```

```
alias alert='notify-send --urgency=low -i "${[ $? = 0 ]} && echo terminal || echo error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[\;|]\s*alert$//'\`')"
```

```
alias egrep='egrep --color=auto'
```

```
alias fgrep='fgrep --color=auto'
```

```
alias grep='grep --color=auto'
```

```
alias l='ls -CF'
```

```
alias la='ls -A'
```

```
alias ll='ls -alF'
```

```
alias ls='ls --color=auto'
```

```
alias myls='ls -al'
```

```
kw2023202070@ubuntu:~/work$
```


22. grep

`grep [options] [PATTERN] [FILE...]`

파일에서 일치하는 PATTERN(문자열 또는 정규 표현식)을 찾아 해당 줄 전체를 출력한다.

e.g. `$ grep hello hello.txt`

hello.txt에서 "hello"가 포함된 줄만 출력됐다.

```
kw2023202070@ubuntu:~/work$ cat hello.txt
hello world
My Name is N~~~
How are you?
kw2023202070@ubuntu:~/work$ grep hello hello.txt
hello world
kw2023202070@ubuntu:~/work$
```

Basic-3. Linux based Programing

1. Vi editor

a. 1라인: 본인 학번, 2라인: 본인 이름, 3라인: Kwangwoon University 입력

\$ vi

Vi editor의 명령 모드에 진입했다.

A screenshot of a Linux terminal window titled "kw2023202070@ubuntu: ~". The terminal displays the output of the command `vim --help`, which shows the following text:

VIM - Vi IMproved

version 8.1.1847
by Bram Moolenaar et al.
Modified by team+vim@tracker.debian.org
Vim is open source and freely distributable

Help poor children in Uganda!

type :help iccf<Enter> for information

type :q<Enter> to exit

type :help<Enter> or <F1> for on-line help

type :help version8<Enter> for version info

The bottom right corner of the terminal shows the cursor position at column 0, row 0-1.

[1]를 눌러 명령 모드에서 입력 모드로 진입했다.

```
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
VIM - Vi Improved  
  
version 8.1.1847  
by Bram Moolenaar et al.  
Modified by team+vim@tracker.debian.org  
Vim is open source and freely distributable  
  
Help poor children in Uganda!  
type :help iccf<Enter>      for information  
  
type :q<Enter>              to exit  
type :help<Enter> or <F1>   for on-line help  
type :help version8<Enter> for version info  
  
~  
~  
~  
~  
~  
~  
-- INSERT --
```

0,1 All

입력 모드에서 학번, 이름, Kwangwoon University를 입력했다.

[illegible]

b. Kwangwoon University 복사 후 본인 학번 다음 라인에 복사 붙여 놓기
[esc]를 눌러 입력 모드에서 명령 모드로 진입했다.

yy: 커서가 있는 행(Kwangwoon University가 작성된 행)을 복사했다.

[illegible]

k를 두 번 눌러 학번이 있는 행으로 커서를 이동했다.

[illegible]

p: 커서 아래 복사된 문자열을 붙여 넣었다.

[illegible]

c. 편집기에 라인 표시

:set number: 편집기의 라인을 표시하는 명령어를 실행했다.

```
1 2023202070
2 Kwangwoon University
3 최현진
4 Kwangwoon University
```

:set number 2,1 All

d. 본인 학번을 파일 이름으로 저장

:w 2023202070: 학번을 이름으로 하여 파일을 저장했다.

[illegible]

.wq: 저장 후 종료하는 명령어를 실행했다.

\$ ls: 학번이 이름인 파일이 저장된 것을 확인했다.

```
kw2023202070@ubuntu:~$ ls
2023202070  Documents  Music      Public  splab_commands  Videos
Desktop     Downloads  Pictures   snap    Templates        work
```

a.

```
#include <stdio.h>

void main() {
    printf("2023202070 최현진\n");
}
```

c 파일 컴파일 자동화를 위한 Makefile을 다음과 같이 작성하고 저장했다.

```
hello: kw_hello.c
gcc kw_hello.c -o hello
```

"Makefile" 2L, 43C written

2,1-8 All

c.

\$ **make**: Makefile 제일 첫 번째 타겟에 해당하는 규칙이 수행되어 gcc 컴파일 명령어가 실행됐다.

```
kw2023202070@ubuntu:~$ make  
gcc kw_hello.c -o hello
```

\$ **ls**: 컴파일된 실행 파일 hello가 생성된 것을 확인했다.

\$ **./hello**: 실행 파일 실행 결과, 학번과 이름이 정상적으로 출력되었다.

```
kw2023202070@ubuntu:~$ ls  
2023202070 Documents hello Makefile Pictures snap Templates worl  
Desktop Downloads kw_hello.c Music Public splab_commands Videos  
kw2023202070@ubuntu:~$ ./hello  
2023202070 최현진  
kw2023202070@ubuntu:~$
```

고찰

먼저 이번 Basic 과제는 말 그대로 기본적인 내용을 다루고 있었기 때문에 과제를 진행하는 데에 큰 어려움은 없었다. 무엇보다 2-2학기 데이터구조설계 시간에 진행했던 프로젝트들이 모두 이번 시스템프로그래밍 강좌와 같은 우분투 환경에서 채점되었기 때문에 이미 가상 머신에 우분투를 설치해서 사용해본 경험이 있어서 Basic-1 같은 경우는 진행이 수월했다. Basic-2에서 리눅스 명령어를 공부하며 처음에는 실행할 명령어가 너무 많아 막막했으나, cd나 rm처럼 윈도우 cmd에서 자주 실행해본 명령어들도 있고 강의 자료에 설명이 잘 나와 있어서 생각보다 어렵지 않았다. 그래도 ln를 통해 하드/심볼릭 링크를 만들거나 프로세스를 다루는 명령어들을 학습하면서 평생 써오던 윈도우와 gui와는 다른 리눅스 명령어만의 특징을 이해할 수 있었다. Basic-3에서는 이전에 git bash에서 커밋 메시지를 변경하고 커밋을 삭제하거나 순서를 바꾸는 데 vi를 사용해봤던 경험이 있어서 리눅스 vi editor 또한 이해하기 어렵지 않았다. 그 때 처음 vi를 사용하면서 모드가 구분되어 있고 마우스로 커서를 옮길 수 없다는 것에 혼자 정보를 찾아보며 너무 힘들었던 안 좋은 기억이 있지만, 지금 생각해보니 그 때 고생해봐서 다행이었다. make의 경우 마찬가지로 데구 수업에서 gcc 컴파일을 사용해본 적 있던 터라 쉽게 과제를 마칠 수 있었던 것 같다. 이번 세 가지 주제 모두 전반적인 리눅스 시스템프로그래밍 실습을 할 수 있게 한 것 같아서 구성이 알차다고 생각한다.

Reference

생략합니다.