

## ■ 실습보고서 1

교과목명	시스템프로그래밍		분반	001
제출일자	2021-10-14		교수자명	장희숙
팀명	이름	학번	이름	학번
초코파이썬칩	김기태	20173152		
	안대현	20173217		

### ■ 실습 주제

실습 1. 명령어 사용 및 셸 프로그래밍

### ■ 실습 결과

1. 교재 등에 나오는 각종 리눅스 명령어를 사용하여 본다.

기본 명령어

man : 각종 시스템 명령어들의 도움말 또는 메뉴얼을 출력해 준다.

```
adh@adh:~$ 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 specified.

  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
```

ls : 파일 시스템 상의 파일 목록을 보여준다.

```
adh@adh:~$ pwd
/home/adh
adh@adh:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
adh@adh:~$ ls -a
.          .cache      .gnupg     .pkt       .sudo_as_admin_successful
..         .config     .local     .profile   Templates
.bash_history  Desktop    .mozilla   Public     Videos
.bash_logout  Documents  Music      snap
.bashrc       Downloads  Pictures   .ssh
adh@adh:~$ ls -l
total 36
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Desktop
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Documents
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Downloads
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Music
drwxr-xr-x 2 adh adh 4096 9월 12 16:23 Pictures
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Public
drwxr-xr-x 3 adh adh 4096 7월 30 17:56 snap
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Templates
drwxr-xr-x 2 adh adh 4096 7월 30 17:49 Videos
```

cp : 파일을 현재의 위치나 다른 디렉토리로 복사한다.

```
adh@adh:~$ cd Pictures
adh@adh:~/Pictures$ ls
ls.png  man1.png  man2.png  ubuntuVersiom.png
adh@adh:~/Pictures$ cp ls.png ls2.png
adh@adh:~/Pictures$ ls
ls2.png  ls.png  man1.png  man2.png  ubuntuVersiom.png
```

rm : 파일을 삭제한다.

```
adh@adh:~/Pictures$ ls
cp.png  ls2.png  ls.png  man1.png  man2.png  ubuntuVersiom.png
adh@adh:~/Pictures$ rm ls2.png
adh@adh:~/Pictures$ ls
cp.png  ls.png  man1.png  man2.png  ubuntuVersiom.png
```

mv : 파일을 이동하거나 이름을 변경하는 명령이다.

```
adh@adh:~/Pictures$ ls
cp.png  ls.png  man1.png  man2.png  rm.png  ubuntuVersiom.png
adh@adh:~/Pictures$ mv ls.png newls.png
adh@adh:~/Pictures$ ls
cp.png  man1.png  man2.png  newls.png  rm.png  ubuntuVersiom.png
```

mkdir : 디렉토리를 생성한다.

```
adh@adh:~/Pictures$ mkdir myfile
adh@adh:~/Pictures$ ls
cp.png  ls.png  man1.png  man2.png  mv.png  myfile  rm.png  ubuntuVersiom.png
```

rmdir : 디렉토리를 삭제한다.

```
adh@adh:~/Pictures$ ls
cp.png  man1.png  mkdir.png  myfile  ubuntuVersiom.png
ls.png  man2.png  mv.png      rm.png
adh@adh:~/Pictures$ rmdir myfile
adh@adh:~/Pictures$ ls
cp.png  man1.png  mkdir.png  rm.png
ls.png  man2.png  mv.png     ubuntuVersiom.png
```

chmod : 파일의 속성 모드를 변경한다.

```
adh@adh:~/Pictures$ ls -l
total 316
-rw-rw-r-- 1 adh adh 20265  9월 12 16:28 cp.png
-rw-rw-r-- 1 adh adh 82033  9월 12 16:26 ls.png
-rw-rw-r-- 1 adh adh  2457  9월 12 16:23 man1.png
-rw-rw-r-- 1 adh adh 37607  9월 12 16:22 man2.png
-rw-rw-r-- 1 adh adh 12445  9월 12 16:33 mkdir.png
-rw-rw-r-- 1 adh adh 18363  9월 12 16:32 mv.png
-rw-rw-r-- 1 adh adh 22849  9월 12 16:34 rmdir.png
-rw-rw-r-- 1 adh adh 17544  9월 12 16:29 rm.png
-rw-rw-r-- 1 adh adh 86710  9월 12 16:17 ubuntuVersiom.png
adh@adh:~/Pictures$ chmod 777 cp.png
adh@adh:~/Pictures$ ls -l
total 316
-rwxrwxrwx 1 adh adh 20265  9월 12 16:28 cp.png
-rw-rw-r-- 1 adh adh 82033  9월 12 16:26 ls.png
-rw-rw-r-- 1 adh adh  2457  9월 12 16:23 man1.png
-rw-rw-r-- 1 adh adh 37607  9월 12 16:22 man2.png
-rw-rw-r-- 1 adh adh 12445  9월 12 16:33 mkdir.png
-rw-rw-r-- 1 adh adh 18363  9월 12 16:32 mv.png
-rw-rw-r-- 1 adh adh 22849  9월 12 16:34 rmdir.png
-rw-rw-r-- 1 adh adh 17544  9월 12 16:29 rm.png
-rw-rw-r-- 1 adh adh 86710  9월 12 16:17 ubuntuVersiom.png
```

## 내부 명령어

pwd : 현재 작업 디렉토리를 출력한다.

```
adh@adh:~$ pwd
/home/adh
```

cd : 작업 디렉토리를 변경하는 명령으로써 자신이 있는 현재 위치에서 절대경로나 상대경로를 이용하여 디렉토리를 이동한다.

```
adh@adh:~$ cd Pictures
adh@adh:~/Pictures$ pwd
/home/adh/Pictures

adh@adh:~$ cd Pictures/basic
adh@adh:~/Pictures/basic$ pwd
/home/adh/Pictures/basic
```

jobs : 실행중인 프로세스는 포그라운드 또는 백그라운드 작업으로 실행되는데, 이 명령은 백그라운드로 실행하고 있는 작업들을 작업 번호와 함께 보여준다.

```
adh@adh:~$ sleep 100 &
[1] 4114
adh@adh:~$ sleep 100 &
[2] 4115
adh@adh:~$ jobs
[1]-  Running                  sleep 100 &
[2]+  Running                  sleep 100 &
```

kill : 현재 수행중인 프로세스에게 시그널을 보낸다. 보통 프로세스를 죽이는 데 사용된다.

```
adh@adh:~$ jobs
[1]-  Running                  sleep 100 &
[2]+  Running                  sleep 100 &
adh@adh:~$ kill 4167
adh@adh:~$ jobs
[1]-  Terminated              sleep 100
[2]+  Running                  sleep 100 &
```

## 텍스트 처리 명령어

touch, cat : 파일을 생성

```
adh@adh:~/Pictures/text$ ls
adh@adh:~/Pictures/text$ touch test.txt
adh@adh:~/Pictures/text$ ls
test.txt
```

```
adh@adh:~/Pictures/text$ cat > test.txt
first line
second line
third line
end~~
```

cat : 파일의 내용을 보여준다

```
adh@adh:~/Pictures/text$ cat -n test.txt
 1 first line
 2 second line
 3 third line
 4 end~~
```

more, less : 파일의 내용을 페이지 단위로 보여준다.

```
adh@adh:~/Pictures/text$ more news.txt
```

```
Photographer Tom Franklin wasn't even supposed to be in the newsroom that morning.

He'd been in the Dominican Republic for a baseball project and had stopped by to talk to his editor at The Record, a newspaper then based in Hackensack, New Jersey, and now part of the USA TODAY Network. They were discussing the assignment when someone interrupted with the news that a plane had hit one of the Twin Towers at the World Trade Center.

Franklin joined the rush of journalists to Jersey City to document history from across the Hudson River. He'd spend most of the day with a clear view of the Manhattan skyline, before boarding a boat to Lower Manhattan.

One of the images he captured that afternoon would become among the most recognizable photos in history.

Tom Franklin, who was working as a photojournalist for The Record on 9/11, poses for a portrait with a poster of the stamp that depicts the image he took on that fateful day.
Wednesday, August 4, 2021
He saw firefighters "fumbling" with an American flag near where the Twin Towers had fallen and fired off a burst of photos.
~More~ (72%)
```

head, tail : 파일의 시작 부분 또는 마지막 부분을 표준출력으로 보여준다.

```
adh@adh:~/Pictures/text$ head -1 news.txt
Photographer Tom Franklin wasn't even supposed to be in the newsroom that morning.
```

```
adh@adh:~/Pictures/text$ tail -1 news.txt
It was even made into a postage stamp.
```

grep : 정규 표현식을 쓰는 다목적 파일 검색 도구이다.

```
adh@adh:~/Pictures/text$ grep '^He' news.txt
He'd been in the Dominican Republic for a baseball project and had stopped by to talk to his editor at The Record,
He'd spend most of the day with a clear view of the Manhattan skyline,
He saw firefighters "fumbling" with an American flag near where the Twin Towers had fallen and fired off a burst of photos.
```

wc : 파일 내의 문자 및 문자열의 개수를 출력한다.

```
adh@adh:~/Pictures/text$ wc -l news.txt
20 news.txt
adh@adh:~/Pictures/text$ wc -w news.txt
250 news.txt
adh@adh:~/Pictures/text$ wc -c news.txt
1455 news.txt
```

diff, cmp, comm 파일을 서로 비교한다.

```
adh@adh:~/Pictures/text$ cat difftest.txt
first line
second line
end line
adh@adh:~/Pictures/text$ diff test.txt difftest.txt
3,4c3
< third line
< end~~
---
> end line
```

```
adh@adh:~/Pictures/text$ cmp test.txt difftest.txt
test.txt difftest.txt differ: byte 24, line 3
```

```
adh@adh:~/Pictures/text$ comm test.txt difftest.txt
      first line
      second line
    end line
comm: file 2 is not in sorted order
third line
comm: file 1 is not in sorted order
end~~
```



tar : 파일 묶음에 사용한다.

```
adh@adh:~/Pictures/text$ tar czvf test.tar.gz *
cat1.png
cat2.png
cmp.png
comm.png
diff.png
difftest.txt
grep.png
head.png
more1.png
more2.png
news.txt
new.txt
Screenshot from 2021-09-12 17-12-52.png
test.txt
touch.png
wc.png
adh@adh:~/Pictures/text$ ls -l
total 428
-rw-rw-r-- 1 adh adh 7837 9월 12 17:01 cat1.png
-rw-rw-r-- 1 adh adh 8575 9월 12 17:01 cat2.png
-rw-rw-r-- 1 adh adh 7574 9월 12 17:23 cmp.png
-rw-rw-r-- 1 adh adh 15872 9월 12 17:23 comm.png
-rw-rw-r-- 1 adh adh 17649 9월 12 17:22 diff.png
-rw-rw-r-- 1 adh adh 32 9월 12 17:21 difftest.txt
-rw-rw-r-- 1 adh adh 26114 9월 12 17:19 grep.png
-rw-rw-r-- 1 adh adh 10211 9월 12 17:11 head.png
-rw-rw-r-- 1 adh adh 4958 9월 12 17:05 more1.png
-rw-rw-r-- 1 adh adh 74331 9월 12 17:09 more2.png
-rw-rw-r-- 1 adh adh 1455 9월 12 17:12 news.txt
-rw-rw-r-- 1 adh adh 1455 9월 12 17:09 new.txt
-rw-rw-r-- 1 adh adh 7502 9월 12 17:12 'Screenshot from 2021-09-12 17-12-52.png'
-rw-rw-r-- 1 adh adh 194156 9월 12 17:24 test.tar.gz
-rw-rw-r-- 1 adh adh 40 9월 12 17:00 test.txt
-rw-rw-r-- 1 adh adh 8089 9월 12 17:00 touch.png
-rw-rw-r-- 1 adh adh 15975 9월 12 17:19 wc.png
```

gzip : 파일 압축 및 해제에 사용한다.

```
adh@adh:~/Pictures$ ls
basic inner tar.png text
adh@adh:~/Pictures$ gzip tar.png
adh@adh:~/Pictures$ ls
basic inner tar.png.gz text
adh@adh:~/Pictures$ gzip -d tar.png.gz
adh@adh:~/Pictures$ ls
basic inner tar.png text
```

## 시스템과 관리자용 명령어

du : 지정된 특정 디렉토리나 파일들이 차지하는 공간을 보고한다.

```
adh@adh:~/Pictures$ ls
basic inner system text
adh@adh:~/Pictures$ du basic
508      basic
adh@adh:~/Pictures$ du
4        ./system
508      ./basic
392      ./text
60       ./inner
980      .
```

df :디스크의 사용/여유 공간을 검사하여 보고한다.

```
adh@adh:~/Pictures$ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev            8123220         0   8123220   0% /dev
tmpfs           1630568       1776   1628792   1% /run
/dev/sda6       479179464 10792864 443975876   3% /
tmpfs           8152836      154700   7998136   2% /dev/shm
tmpfs           5120          4        5116   1% /run/lock
tmpfs           8152836         0   8152836   0% /sys/fs/cgroup
/dev/loop3      10496         10496         0 100% /snap/canonical-livepatch/105
/dev/loop1      56832         56832         0 100% /snap/core18/2074
/dev/loop2     101760        101760         0 100% /snap/core/11606
/dev/loop6      33152         33152         0 100% /snap/snapd/12883
/dev/loop5      33152         33152         0 100% /snap/snapd/12704
/dev/loop0      56832         56832         0 100% /snap/core18/2128
/dev/loop7     101888        101888         0 100% /snap/core/11420
/dev/loop8      10496         10496         0 100% /snap/canonical-livepatch/104
/dev/loop4      66432         66432         0 100% /snap/gtk-common-themes/1514
/dev/loop13     224256        224256         0 100% /snap/gnome-3-34-1804/72
/dev/loop12     66688         66688         0 100% /snap/gtk-common-themes/1515
/dev/loop10     52224         52224         0 100% /snap/snap-store/547
/dev/loop9      224256        224256         0 100% /snap/gnome-3-34-1804/66
/dev/loop11     52352         52352         0 100% /snap/snap-store/518
/dev/sda2       98304         33627        64677   35% /boot/efi
tmpfs          1630564         40   1630524   1% /run/user/1000
```

ps : 프로세스 통계를 나타낸다. 현재 실행중인 프로세스들을 사용자와 PID에 의해서 보여준다.

```
adh@adh:~$ ps -u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
adh      1557   0.0   0.0 173952  6808 tty2      Ssl+  16:11   0:00 /usr/lib/gdm3/gd
adh      1559   3.0   0.4 840132 73864 tty2      Sl+   16:11   2:30 /usr/lib/xorg/Xo
adh      1581   0.0   0.0 198156 14024 tty2      Sl+   16:11   0:00 /usr/libexec/gno
adh      2623   0.0   0.0  21200  5908 pts/0     Ss    16:14   0:00 bash
adh      8037   0.0   0.0  21432  3532 pts/0     R+    17:32   0:00 ps -u

adh@adh:~$ ps -m
  PID TTY          TIME CMD
 2623 pts/0    00:00:00 bash
    -  -         00:00:00 -
 8039 pts/0    00:00:00 ps
```

## 2. 정규표현식을 사용하는 유닉스/리눅스 유틸리티(grep, find 등)들을 사용하여 본다.

grep은 파일 탐색을 위한 명령어로 다음과 같은 옵션을 지원한다.

- v : 검색 문자열을 포함하는 행을 제외한 모든 라인 출력
- n : 검색 문자열을 포함하는 행들의 번호 표시
- c : 검색 문자열을 포함하는 행을 출력하지 않도록 함
- l : 특정한 패턴을 찾기 위해 여러 개의 파일 검색
- y : 대문자, 소문자 구분을 하지 않음
- w : 완전한 하나의 단어만을 찾고 싶을 때 지정

정규 표현식 : 필요에 따라 약간 더 복잡한 검색 명령어를 사용할 필요로 하는 것

정규식 패턴	의미	사용 예시	설명	[a-z]	a부터 z 사이의 문자 구간에 일치(영어 소문자)	[b-f]	b, c, d, e, f 중 하나
~	줄(Line)의 시작에서 일치	^abc	abc로 시작하는	[A-Z]	A부터 Z 사이의 문자 구간에 일치(영어 대문자)	[A-C]	A, B, C 중 하나
\$	줄(Line)의 끝에서 일치	abc\$	abc로 끝나는	[0-9]	0부터 9 사이의 문자 구간에 일치(숫자)	[1-6]	1, 2, 3, 4, 5, 6 중 하나
.	임의의 한 문자와 일치	...	임의의 문자 3개	[가-힣]	가부터 힣 사이의 문자 구간에 일치(한글)	[다-바]	다부터 바 사이의 문자 구간에 일치(다, 라, 마, 바)
ab	a 또는 b와 일치, 인덱스가 작은 것을 우선 반환	a b	a 또는 b와 일치하는	[^abc]	a 또는 b 또는 c가 아닌 나머지 문자에 일치(부절)		
*	0회 이상 연속으로 반복되는 문자와 가능한 많이 일치	a*	a가 0~n개, {0,}와 동일	\w	이스케이프 문자	/W.W?W/W\$W?/	정규식내 특수 의미 문자를 일반문자화시킬
+	1회 이상 연속으로 반복되는 문자와 가능한 많이 일치	a+	a가 1~n개, {1,}와 동일	\wb	83개 문자가 아닌 나머지 문자에 일치하는 경계(boundary)		대소문자 52개 + 숫자 10개 + _(underscore)
++	1회 이상 연속으로 반복되는 문자와 가능한 적게 일치(lazy)	a++	a가 1~n개, {1,}와 동일	\WB	83개 문자에 일치하는 경계		
?	없거나 1회 가능한 많이 일치	a?	a가 0 또는 1개일 경우	\wd	숫자(Digit)에 일치		
??	없거나 1회 가능한 적게 일치(lazy)	a??		\WD	숫자가 아닌 문자에 일치		
{3}	3(숫자)개 연속 일치	a{3}	a가 3번	\Ws	공백(Space, Tab 등)에 일치		
{3,}	3개 이상 연속 일치	a{3,}	a가 최소 3번	\WS	공백이 아닌 문자에 일치		
{3,5}	3개 이상 5개 이하(3~5개) 연속 일치	a{3,5}	a가 3~5회 즉, aaa,aaaa,aaaaa	\Ww	83개 문자(Word, 영문 대소문자 52개 + 숫자 10개 + _)에 일치		
{3,5}?	3개 이상 5개 이하(3~5개) 연속 중 가능한 적은 3개 연속 일치	a{3,5}?	a가 3~5회 즉, aaa,aaaa,aaaaa	\WW	83개 문자가 아닌 나머지 문자에 일치		
()	캡처(Capture)할 그룹		추출할 문자열을 괄호로 묶는다고 생각하면됨	\Wx	16진수 문자에 일치	/Wx:61/	a에 일치
[abc]	a, b, c 중 1개와 일치		점(.)이나 별표(*) 같은 특수 문자는 [] 안에서 특수 문자가 아님. /W.[,]/	\WO	8진수 문자에 일치	/W141/	a에 일치
				\Wu	유니코드(Unicode) 문자에 일치	/Wu0061/	a에 일치
				\Wc	제어(Control) 문자에 일치		
				\Wf	폼 피드(FF, U+000C) 문자에 일치		
				\Wn	줄 바꿈(LF, U+000A) 문자에 일치		

1) grep [0-9] test.txt -- 0부터 9까지 숫자 구간에 일치

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep [0-9] test.txt
NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years of
collective, scientific and drug-development experience to bring improved health
to patients.
Its investigational product, ZYESAMIcã (aviptadil) for patients with COVID-19, h
as been granted Fast Track designation by the US Food and Drug Administration (F
DA).
And is currently undergoing Phase 3 trials funded by the US National Institutes
of Health, the Biomedical Advanced Research and Development Authority, a part of
the US Department of Health and Human Services, and the Medical Countermeasures
program, part of the US Department of Defense.
The FDA has additionally granted Breakthrough Therapy Designation, a Special Pro
tocol Agreement, and a Biomarker Letter of Support to NRx for NRX-101, an invest
igational medicine to treat suicidal bipolar depression.
NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
NRx also has the BriLifecã vaccine for COVID-19 in clinical trials and holds the
exclusive worldwide license to commercialize the vaccine.
And General H.R. McMaster, Ph.D. (US Army, Ret.) the 26th United States National
Security Advisor.
```

2) `grep -n [abc] test.txt` -- a, b, c중 1개와 일치하는 열의 번호까지 함께 출력

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep -n [abc] test.txt
1:a aa aaa aaaa bbbbbb aabbbb abaaabab
3:About NRx Pharmaceuticals
5:NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years o
f collective, scientific and drug-development experience to bring improved healt
h to patients.
6:Its investigational product, ZYESAMIcã (aviptadil) for patients with COVID-19,
has been granted Fast Track designation by the US Food and Drug Administration
(FDA).
7:And is currently undergoing Phase 3 trials funded by the US National Institute
s of Health, the Biomedical Advanced Research and Development Authority, a part
of the US Department of Health and Human Services, and the Medical Countermeasur
es program, part of the US Department of Defense.
8:The FDA has additionally granted Breakthrough Therapy Designation, a Special P
rotocol Agreement, and a Biomarker Letter of Support to NRx for NRX-101, an inve
stigational medicine to treat suicidal bipolar depression.
9:NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
10:NRx also has the BriLifecã vaccine for COVID-19 in clinical trials and holds
the exclusive worldwide license to commercialize the vaccine.
```

3) `grep '.r' test.txt` — 임의의 한 문자와 r로 이루어진 문자 출력

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep '.r' test.txt
About NRx Pharmaceuticals
NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years of
collective, scientific and drug-development experience to bring improved health
to patients.
Its investigational product, ZYESAMIcã (aviptadil) for patients with COVID-19, h
as been granted Fast Track designation by the US Food and Drug Administration (F
DA).
And is currently undergoing Phase 3 trials funded by the US National Institutes
of Health, the Biomedical Advanced Research and Development Authority, a part of
the US Department of Health and Human Services, and the Medical Countermeasures
program, part of the US Department of Defense.
The FDA has additionally granted Breakthrough Therapy Designation, a Special Pro
tocol Agreement, and a Biomarker Letter of Support to NRx for NRX-101, an invest
igational medicine to treat suicidal bipolar depression.
NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
NRx also has the BriLifecã vaccine for COVID-19 in clinical trials and holds the
exclusive worldwide license to commercialize the vaccine.
The BriLife vaccine was first developed by the Israel Institute for Biological R
```

4) `grep [a-c] test.txt` — a부터 c까지 문자 구간에 일치



```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep [a-c] test.txt
a aa aaa aaaa bbbbbb aaabbb abaabab
About NRx Pharmaceuticals
NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years of
collective, scientific and drug-development experience to bring improved health
to patients.
Its investigational product, ZYESAMIcã (aviptadil) for patients with COVID-19, h
as been granted Fast Track designation by the US Food and Drug Administration (F
DA).
And is currently undergoing Phase 3 trials funded by the US National Institutes
of Health, the Biomedical Advanced Research and Development Authority, a part of
the US Department of Health and Human Services, and the Medical Countermeasures
program, part of the US Department of Defense.
The FDA has additionally granted Breakthrough Therapy Designation, a Special Pro
tocol Agreement, and a Biomarker Letter of Support to NRx for NRX-101, an invest
igational medicine to treat suicidal bipolar depression.
NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
NRx also has the BriLifeçã vaccine for COVID-19 in clinical trials and holds the
exclusive worldwide license to commercialize the vaccine.
```

5) grep '^N' test.txt — N으로 시작하는 문장 출력

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep '^N' test.txt
NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years of
collective, scientific and drug-development experience to bring improved health
to patients.
NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
NRx also has the BriLifeçã vaccine for COVID-19 in clinical trials and holds the
exclusive worldwide license to commercialize the vaccine.
NRx is led by executives who have held senior roles at Allergan, J&J, Lilly, Nov
artis, Pfizer, and the US FDA.
NRx is chaired by Prof Jonathan Javitt, MD, MPH, who has held leadership roles i
n six biotechnology startup companies with public exits and been appointed to ad
visory roles in four US Presidential administrations.
root@9eb578df892c:~/Test#
```

6) grep -n '^N' test.txt — N으로 시작하는 줄과 줄 번호 출력

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep -n '^N' test.txt
3:NRx Pharmaceuticals (NRx) (www.nrxpharma.com) draws upon more than 300 years o
f collective, scientific and drug-development experience to bring improved healt
h to patients.
7:NRX-101 is currently in Phase 3 trials, with readouts expected in 2022.
8:NRx also has the BriLifeçã vaccine for COVID-19 in clinical trials and holds t
he exclusive worldwide license to commercialize the vaccine.
11:NRx is led by executives who have held senior roles at Allergan, J&J, Lilly,
Novartis, Pfizer, and the US FDA.
12:NRx is chaired by Prof Jonathan Javitt, MD, MPH, who has held leadership role
s in six biotechnology startup companies with public exits and been appointed to
advisory roles in four US Presidential administrations.
root@9eb578df892c:~/Test#
```

7) grep 'aW{3W}' test.txt — a가 3개인 문자 출력



```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep 'a\{3\}' test.txt
a aa aaa aaaa bbbbbb aaabbb abaabab
root@9eb578df892c:~/Test#
```

8) `grep -n 'W.$' test.txt` — 문자 '.' 가 줄의 끝인 문장 및 문장 번호 출력

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep -n '\.$' test.txt
6:Its investigational product, ZYESAMI (aviptadil) for patients with COVID-19,
has been granted Fast Track designation by the US Food and Drug Administration
(FDA).
11:The BriLife vaccine was first developed by the Israel Institute for Biological
Research.
15:The NRx board includes Dr. Sherry Glied, former US Assistant Secretary for Health
(ASPE), Daniel E. Troy, JD, former Chief Counsel of the US FDA, Chaim Hurvitz,
former director of Teva and President of the Teva International Group.
16:And General H.R. McMaster, Ph.D. (US Army, Ret.) the 26th United States National
Security Advisor.
```

9) `grep 'aaaW?' test.txt` && `grep 'aaaW+' test.txt` — aaa와 가능한 적게 일치 && 많이 일치

```
root@9eb578df892c: ~/Test
File Edit Tabs Help
root@9eb578df892c:~/Test# grep 'aaa?*' test.txt
a aa aaa aaaa bbbbbb aaabbb abaabab
root@9eb578df892c:~/Test# grep 'aaa+*' test.txt
a aa aaa aaaa bbbbbb aaabbb abaabab
root@9eb578df892c:~/Test#
```

3. vi 파일 편집 도구를 사용하여 파일을 작성·수정하고, 익숙해지도록 사용해 본다.

```
ads0070@ads0070-virtual-machine:~$ vi vitest2.txt
```

```
ads0070@ads0070-virtual-machine: ~
vi 편집기
a:커서 바로 뒤에서 입력을 시작
i:커서가 위치한 곳부터 입력을 시작
x:커서의 위치에 있는 문자 삭제
:wq!=파일 내용을 저장하고 종료
~
~
~
~
~
:wq!
```

```
ads0070@ads0070-virtual-machine:~$ ls
bitcoin.txt  myfile  공개  문서  비디오  음악
createFile.txt vitest2.txt 다운로드 바탕화면 사진  템플릿
ads0070@ads0070-virtual-machine:~$ cat vitest2.txt

vi 편집기
a:커서 바로 뒤에서 입력을 시작
i:커서가 위치한 곳부터 입력을 시작
x:커서의 위치에 있는 문자 삭제
:wq!=파일 내용을 저장하고 종료
```

- vi 편집기에서 입력 모드를 시작할 때
- a: 커서 바로 뒤에서 입력 시작
  - i : 커서가 위치한 곳부터 입력 시작
  - o : 커서가 위치한 아래 행부터 입력을 시작

- vi 편집기에서 커서를 이동할 때
- h(왼쪽), l(오른쪽), k(위쪽), j(아래쪽)

- 0 : 커서를 현재 행의 처음으로 이동
- \$ : 커서를 현재 행의 마지막으로 이동

vi 편집기에서 화면을 이동할 때

- :숫자 : 파일에서 지정한 숫자번째 행으로 이동

vi 편집기에서 내용 삭제

- x : 커서의 위치에 있는 문자 삭제
- X : 커서의 왼쪽 문자 삭제
- dd : 커서가 위치한 행 삭제

4. 쉘 상에서 각종 쉘 명령어를 익숙해지도록 사용해 본다. 특히, 별명, 쉘 프롬프트, 히스토리 기능의 변경이 포함된 .bashrc와 같은 쉘 설정 파일을 직접 수정하여 본다.

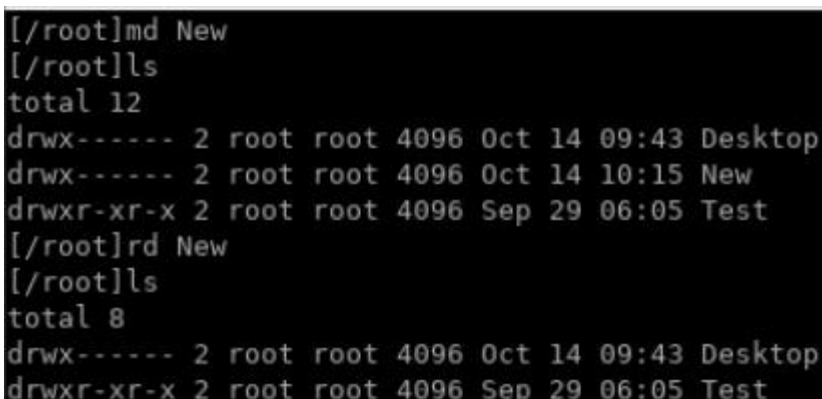
1) 쉘 설정 파일인 .bashrc를 다음과 같이 수정하여 저장한다. 파일 퍼미션은 077, 히스토리 출력개수는 10, 저장개수는 100으로 지정한 뒤, alias를 통해 별명을 지정한다. 마지막으로 export PS1을 통해 터미널의 출력모습을 지정한다.



```

1 umask 077
2 export HISTSIZE=10
3 export HISTFILESIZE=100
4 stty erase ^H
5 export PATH=./bin:/usr/sbin:/usr/bin:/usr/lib:/usr/local/bin
6 #----- Aliases -----
7 alias md='mkdir'
8 alias rd='rmdir'
9 alias ls='ls -l'
10 alias cls='clear'
11 #----- End Alias -----
12 export PS1="[ $PWD]"
  
```

2) source ~/.bashrc를 통해 수정한 .bashrc를 적용한다. 사용자 출력과 별명이 사용됨을 확인할 수 있다.



```

[/root]md New
[/root]ls
total 12
drwx----- 2 root root 4096 Oct 14 09:43 Desktop
drwx----- 2 root root 4096 Oct 14 10:15 New
drwxr-xr-x 2 root root 4096 Sep 29 06:05 Test
[/root]rd New
[/root]ls
total 8
drwx----- 2 root root 4096 Oct 14 09:43 Desktop
drwxr-xr-x 2 root root 4096 Sep 29 06:05 Test
  
```

3) history명령어를 통해 출력개수가 10개인 것을 확인할 수 있다.

```
[/root]history
249  clear
250  cls
251  ls
252  cls
253  md New
254  ls
255  rd New
256  ls
257  cls
258  history
```

5. bash 쉘 스크립트가 익숙해지도록 사용해 보고, 다음 조건을 만족하는 bash 쉘 스크립트를 작성해 본다.

A. "test" 명령을 사용하여 현재 디렉토리에서 일반 파일만 출력하는 쉘 프로그램을 작성하시오.

test 파일 연산 중 -f 파일은 파일이 일반 파일이면 0을 반환하고, 아니면 1을 반환한다.

```
#!/bin/sh
dirname=${PWD##*/}
input='ls'

for i in $input
do
    `test -f /home/adh/$dirname/$i`
    n=`echo $?`
    if [ $n -eq 0 ]
    then
        echo "일반파일 : $i"
    fi
done
```

```
adh@adh: ~/test
adh@adh:~/test$ vi testFile.sh
adh@adh:~/test$ chmod +x testFile.sh
adh@adh:~/test$ ls -al
합계 84
drwxrwxr-x 4 adh adh 4096 10월 14 17:26 .
drwxr-xr-x 23 adh adh 4096 9월 28 19:30 ..
-rw-rw-r-- 1 adh adh 55 9월 28 19:31 a.c
-rwxrwxr-x 1 adh adh 16808 9월 28 19:57 add
-rw-rw-r-- 1 adh adh 12 9월 28 19:31 a.h
-rw-rw-r-- 1 adh adh 1664 9월 28 19:57 a.o
-rw-rw-r-- 1 adh adh 60 9월 28 19:32 b.c
-rw-rw-r-- 1 adh adh 12 9월 28 19:32 b.h
-rw-rw-r-- 1 adh adh 1664 9월 28 19:57 b.o
drwxrwxr-x 2 adh adh 4096 10월 14 17:26 dir1
drwxrwxr-x 2 adh adh 4096 10월 14 17:26 dir2
-rwxrwxr-x 1 adh adh 259 10월 14 16:39 fibo.sh
-rw-rw-r-- 1 adh adh 76 9월 28 19:33 main.c
-rw-rw-r-- 1 adh adh 1608 9월 28 19:57 main.o
-rw-rw-r-- 1 adh adh 148 9월 28 19:57 Makefile
-rwxrwxr-x 1 adh adh 140 10월 14 16:42 printStar.sh
-rwxrwxr-x 1 adh adh 166 10월 14 17:26 testFile.sh
adh@adh:~/test$ ./testFile.sh
일반파일 : a.c
일반파일 : add
일반파일 : a.h
일반파일 : a.o
일반파일 : b.c
일반파일 : b.h
일반파일 : b.o
일반파일 : fibo.sh
일반파일 : main.c
일반파일 : main.o
일반파일 : Makefile
일반파일 : printStar.sh
일반파일 : testFile.sh
adh@adh:~/test$
```

우선 input에 ls 명령어를 사용하여 현재 디렉토리 내에 있는 파일 목록이 입력되도록 한다. 다음으로, for 반복문을 사용하여 input에 입력된 파일 목록이 하나씩 I에 들어가도록 한다. 이때 i를 사용하여 test -f ~/I 명령을 통해 해당 파일이 일반 파일인지 확인한다. echo \$?를 사용해 결과를 n에 대입하고, n이 0이면 해당 파일 \$i를 출력한다.

B. 반복문(while, for, until)을 사용하여 구구단을 표시하는 쉘 스크립트를 작성하시오.

```
ads0070@ads0070-virtual-machine: ~/myfile
#!/bin/bash
for (( i=1; i<=9; i++ ))
do
    echo "----- $i dan -----"
    for (( j=1; j<=9; j++))
    do
        echo "$i X $j = $((i*$j))"
    done
    echo
done

ads0070@ads0070-virtual-machine:~/myfile$ vi multiplicationTable.sh
ads0070@ads0070-virtual-machine:~/myfile$ chmod +x multiplicationTable.sh
ads0070@ads0070-virtual-machine:~/myfile$ ./multiplicationTable.sh
----- 1 dan -----
1 X 1 = 1
1 X 2 = 2
1 X 3 = 3
1 X 4 = 4
1 X 5 = 5
1 X 6 = 6
1 X 7 = 7
1 X 8 = 8
1 X 9 = 9

----- 2 dan -----
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18

----- 3 dan -----
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27

----- 4 dan -----
4 X 1 = 4
4 X 2 = 8
4 X 3 = 12
4 X 4 = 16
4 X 5 = 20
4 X 6 = 24
4 X 7 = 28
4 X 8 = 32
4 X 9 = 36

----- 5 dan -----
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

----- 6 dan -----
6 X 1 = 6
6 X 2 = 12
6 X 3 = 18
6 X 4 = 24
6 X 5 = 30
6 X 6 = 36
6 X 7 = 42
6 X 8 = 48
6 X 9 = 54

----- 7 dan -----
7 X 1 = 7
7 X 2 = 14
7 X 3 = 21
7 X 4 = 28
7 X 5 = 35
7 X 6 = 42
7 X 7 = 49
7 X 8 = 56
7 X 9 = 63

----- 8 dan -----
8 X 1 = 8
8 X 2 = 16
8 X 3 = 24
8 X 4 = 32
8 X 5 = 40
8 X 6 = 48
8 X 7 = 56
8 X 8 = 64
8 X 9 = 72

----- 9 dan -----
9 X 1 = 9
9 X 2 = 18
9 X 3 = 27
9 X 4 = 36
9 X 5 = 45
9 X 6 = 54
9 X 7 = 63
9 X 8 = 72
9 X 9 = 81
```

for 반복문을 이중으로 사용하여 1단부터 9단까지의 반복문을 출력한다.

C. "let" 또는 "expr" 명령을 사용하여 피보나치 수열을 나타내는 쉘 스크립트를 작성하시오.

```
adh@adh: ~/test
#!/bin/sh
echo "수행할 단계를 입력하세요"
read n
first=0
second=1
i=2
echo "피보나치 시작"
echo "$first"
echo "$second"
while [ $i -lt $n ]
do
    i=`expr $i + 1`
    third=`expr $first + $second`
    echo "$third"
    first=$second
    second=$third
done

adh@adh:~/test$ vi fibo.sh
adh@adh:~/test$ chmod +x fibo.sh
adh@adh:~/test$ ./fibo.sh
수행할 단계를 입력하세요
6
피보나치 시작
0
1
1
2
3
5
adh@adh:~/test$
```

피보나치 수열은 처음 두 수를 0, 1로 설정해둔 뒤,  $n = (n-2) + (n-1)$ 의 형태로 이루어진다. while 반복문을 통해 피보나치 수열 출력이 n번 반복되도록 한다.



D. 사용자로부터 3개의 숫자를 입력받아 가장 큰 수와 가장 작은 수를 출력하는 쉘 스크립트를 작성하시오.

```
ads0070@ads0070-virtual-machine: ~/myfile
#!/bin/bash

echo -e "input nums : c "
read -a nums

max=${nums[0]}
min=${nums[0]}

for i in ${nums[@]}
do
    if [[ $i -gt $max ]]
    then
        max="$i"
    fi

    if [[ $i -lt $min ]]
    then
        min="$i"
    fi
done

echo "max num : $max"
echo "min num : $min"
```

```
ads0070@ads0070-virtual-machine:~/myfile$ vi minmax.sh
ads0070@ads0070-virtual-machine:~/myfile$ chmod +x minmax.sh
ads0070@ads0070-virtual-machine:~/myfile$ ./minmax.sh
input nums : c
30 15 47
max num : 47
min num : 15
```

사용자로부터 3개의 숫자를 입력받고, 해당 숫자를 배열 nums에 저장한다.

반복문과 조건문을 사용하여 nums 배열의 원소 중 max 값과 min 값을 구한다.

수식1 -gt 수식2 : 수식1이 수식2보다 크면 참

수식1 -lt 수식2 : 수식1이 수식2보다 작으면 참

E. Here 자료를 이용하여 문장 내에 "korea"를 검색하는 쉘 스크립트를 작성하시오.

```
ads0070@ads0070-virtual-machine:~/myfile$ vi here.sh
ads0070@ads0070-virtual-machine:~/myfile$ chmod +x here.sh
```

```
ads0070@ads0070-virtual-machine:~/myfile$ cat here.sh
#!/bin/bash
grep $1 << EOF
name korean english math
hong 90 90 90
kim 80 85 75
lee 95 80 90
park 70 75 65
EOF
```

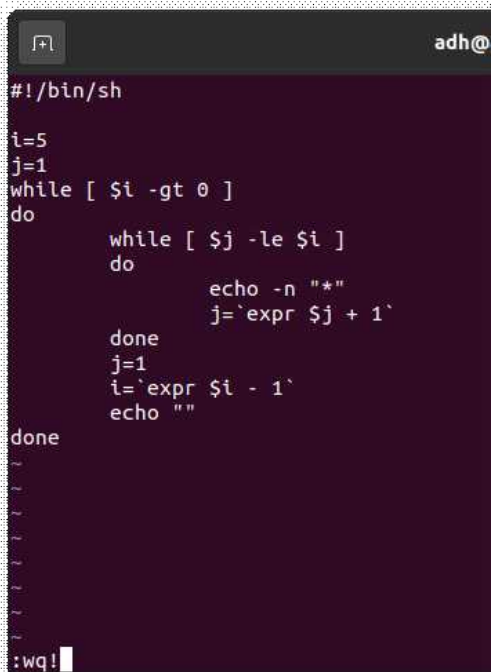
```
ads0070@ads0070-virtual-machine:~/myfile$ ./here.sh korea
name korean english math
```

grep은 입력으로 전달된 파일의 내용에서 특정 문자열을 찾고자할 때 사용된다.

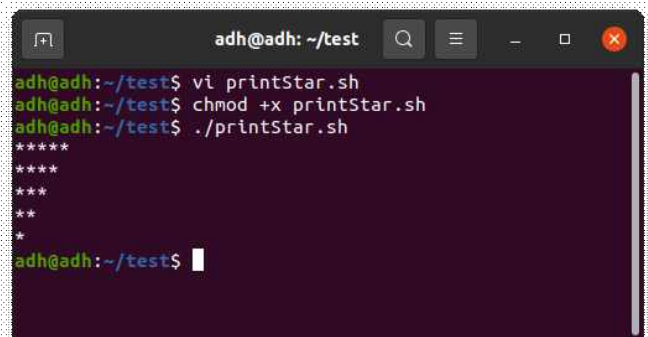
Here 자료는 "<<"로 시작하고, 구분자 "EOF"가 나오는 부분까지 범위가 정해진다.

F. 다음과 같은 문자를 표시하는 셸 스크립트를 작성하시오.

```
*****
****
***
**
*
```



```
#!/bin/sh
i=5
j=1
while [ $i -gt 0 ]
do
    while [ $j -le $i ]
    do
        echo -n "*"
        j=`expr $j + 1`
    done
    j=1
    i=`expr $i - 1`
    echo ""
done
:wq!
```



```
adh@adh: ~/test
adh@adh:~/test$ vi printStar.sh
adh@adh:~/test$ chmod +x printStar.sh
adh@adh:~/test$ ./printStar.sh
*****
****
***
**
*
adh@adh:~/test$
```

while문을 이중으로 사용하여 i를 5부터 1씩 감소시키고, j를 i번 만큼 반복되도록 하여 “\*”을 출력시킨다.