chapter 04

grep 패턴 검색

4.1 | grep

4.1.1 grep란?

grep 명령어는 입력되는 파일에서 주어진 패턴 목록과 매칭되는 라인을 검색한 다음 표준 출력으로 검색된 라인을 복사해서 출력해 준다. 또한 정렬 관련 옵션을 사용하면 정렬하여 출력할 수도 있다.

grep의 검색 범위는 메모리 제한을 넘어가지 않는 범위에서 입력 라인의 제한이 없다. 또한 하나의 라인 안의 전체적인 문자들도 매칭할 수 있다. 입력 파일의 마지막 바이트가 newline이 아니라면 grep는 작업을 수행한다. newline은 패턴 목록을 분리하기 때문에 텍스트에서 newline 문자열을 매칭할 방법은 없다.

[grep 형식]

```
grep [옵션] [패턴] [파일명]
```

[grep 옵션]

표 4-1 • grep 일반 옵션

옵션	설명
-b	검색된 라인에 블록 번호를 붙여서 출력한다. [root@localhost shell]# grep -b root /etc/passwd **0:**root:x:0:0:root:/root:/bin/bash **416:**operator:x:11:0:operator:/root:/sbin/nologin [root@localhost shell]#
-c	매칭된 라인을 디스플레이하지 않고 매칭된 라인의 수를 출력한다. [root@localhost shell]# grep -c root /etc/passwd 2 [root@localhost shell]#
-h	파일명은 출력하지 않는다.

(계속)

표 4-1 • grep 일반 옵션(계속)

옵션	설명
-i	패턴에 사용되는 문자열의 대소문자를 무시하고 검색한다. 즉, 대문자, 소문자를 모두 검색하고 출력해 준다. [root@localhost shell]# grep -i Root /etc/passwd root:x:0:0:root:/root:/bin/bash operator:x:11:0:operator:/root:/sbin/nologin [root@localhost shell]#
-l	패턴에 의해 매칭된 라인이 하나라도 있는 파일의 이름만 출력한다. 출력 시 각 파일명들은 newline으로 분리된다. [root@localhost shell]# grep -l root /etc/passwd /etc/hosts /etc/services /etc/passwd /etc/services [root@localhost shell]#
-n	매칭된 라인을 출력할 때 파일상의 라인 번호를 함께 출력한다. [root@localhost shell]# grep -n root /etc/passwd **1:**root:x:0:0:root:/root:/bin/bash **12:**operator:x:11:0:operator:/root:/sbin/nologin [root@localhost shell]#
-s	조용히 진행한다. 즉, 에러 메시지를 출력하지 않는다. [root@localhost shell]# grep root /etc/passwdpasswd grep: /etc/passwdpasswd: No such file or directory [root@localhost shell]# grep -s root /etc/passwdpasswd [root@localhost shell]#
-v	패턴과 매칭되지 않는 라인만 출력한다. [root@localhost shell]# grep -v nologin /etc/passwd root:x:0:0:root:/root:/bin/bash sync:x:5:0:sync:/sbin:/bin/sync shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown halt:x:7:0:halt:/sbin:/sbin/halt news:x:9:13:news:/etc/news: multi:x:500:500:multi:/home/multi:/bin/bash linux:x:501:500::/home/linux:/bin/bash [root@localhost shell]#
-w	\<과 \>로 둘러싸인 하나의 단어로 표현식을 검색한다.

[grep 옵션 도움말 보기 – man grep 또는 grep --help]

```
[root@localhost shell]# grep --help
Usage: grep [OPTION]... PATTERN [FILE] ...
Search for PATTERN in each FILE or standard input.
Example: grep -i 'hello world' menu.h main.c

Regexp selection and interpretation:
  -E, --extended-regexp              PATTERN is an extended regular expression
  -F, --fixed-strings                PATTERN is a set of newline-separated strings
  -G, --basic-regexp                 PATTERN is a basic regular expression
  -P, --perl-regexp                  PATTERN is a Perl regular expression
  -e, --regexp=PATTERN               use PATTERN as a regular expression
  -f, --file=FILE                    obtain PATTERN from FILE
  -i, --ignore-case                  ignore case distinctions
  -w, --word-regexp                  force PATTERN to match only whole words
  -x, --line-regexp                  force PATTERN to match only whole lines
  -z, --null-data                    a data line ends in 0 byte, not newline

Miscellaneous:
  -s, --no-messages                  suppress error messages
  -v, --invert-match                 select non-matching lines
  -V, --version                      print version information and exit
      --help                         display this help and exit
      --mmap                         use memory-mapped input if possible

Output control:
  -m, --max-count=NUM                stop after NUM matches
  -b, --byte-offset                  print the byte offset with output lines
  -n, --line-number                  print line number with output lines
      --line-buffered                flush output on every line
  -H, --with-filename                print the filename for each match
  -h, --no-filename                  suppress the prefixing filename on output
      --label=LABEL                  print LABEL as filename for standard input
  -o, --only-matching                show only the part of a line matching PATTERN
  -q, --quiet, --silent              suppress all normal output
```

(계속)

```
      --binary-files=TYPE              assume that binary files are TYPE
                                       TYPE is 'binary', 'text', or 'without-match'
   -a, --text                          equivalent to --binary-files=text
   -I                                  equivalent to --binary-files=without-match
   -d, --directories=ACTION            how to handle directories
                                       ACTION is 'read', 'recurse', or 'skip'
   -D, --devices=ACTION                how to handle devices, FIFOs and sockets
                                       ACTION is 'read' or 'skip'
   -R, -r, --recursive                 equivalent to --directories=recurse
      --include=PATTERN                files that match PATTERN will be examined
      --exclude=PATTERN                files that match PATTERN will be skipped.
      --exclude-from=FILE              files that match PATTERN in FILE will be skipped.
   -L, --files-without-match only print FILE names containing no match
   -l, --files-with-matches            only print FILE names containing matches
   -c, --count                         only print a count of matching lines per FILE
   -Z, --null                          print 0 byte after FILE name

Context control:
   -B, --before-context=NUM            print NUM lines of leading context
   -A, --after-context=NUM             print NUM lines of trailing context
   -C, --context=NUM                   print NUM lines of output context
   -NUM                                same as --context=NUM
      --color[=WHEN],
      --colour[=WHEN]                  use markers to distinguish the matching string
                                       WHEN may be 'always', 'never' or 'auto'.
   -U, --binary                        do not strip CR characters at EOL (MSDOS)
   -u, --unix-byte-offsets             report offsets as if CRs were not there (MSDOS)

'egrep' means 'grep -E'.               'fgrep' means 'grep -F'.
With no FILE, or when FILE is -, read standard input.  If less than
two FILEs given, assume -h.            Exit status is 0 if match, 1 if no match,
and 2 if trouble.

Report bugs to <bug-grep@gnu.org>.
[root@localhost shell]#
```

표 4-2 • grep 옵션 포맷

grep 옵션 포맷	설명
grep 'pattern' filename(s)	기본 정규표현식 메타문자 패턴 사용
grep -G 'pattern' filename(s)	위와 동일
grep -E 'pattern' filename(s)	확장 정규표현식 메타문자 패턴 사용(egrep)
grep -F 'pattern' filename	정규표현식 메타문자 패턴을 사용하지 않음(fgrep)
grep -P 'pattern' filename	펄 정규표현식 패턴 사용

```
[root@localhost ~]# mkdir grep
[root@localhost ~]# cd grep
[root@localhost grep]# useradd test -s /bin/false
[root@localhost grep]# passwd test
Changing password for user test.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@localhost grep]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
```

(계속)

```
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
pcap:x:77:77::/var/arpwatch:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
haldaemon:x:68:68:HAL daemon:/:/sbin/nologin
avahi:x:70:70:Avahi daemon:/:/sbin/nologin
hsqldb:x:96:96::/var/lib/hsqldb:/sbin/nologin
avahi-autoipd:x:100:103:avahi-autoipd:/var/lib/avahi-autoipd:/sbin/nologin
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
gdm:x:42:42::/var/gdm:/sbin/nologin
sabayon:x:86:86:Sabayon user:/home/sabayon:/sbin/nologin
multi:x:500:500:multi:/home/multi:/bin/bash
vboxadd:x:101:1::/var/run/vboxadd:/bin/sh
linux:x:501:500::/home/linux:/bin/bash
test:x:502:502::/home/test:/bin/false
```

```
[root@localhost grep]# grep root /etc/passwd
root:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
[root@localhost grep]# grep -n root /etc/passwd
1:root:x:0:0:root:/root:/bin/bash
12:operator:x:11:0:operator:/root:/sbin/nologin
[root@localhost grep]# grep -v bash /etc/passwd | grep -v nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/etc/news:
vboxadd:x:101:1::/var/run/vboxadd:/bin/sh
```

```
test:x:502:502::/home/test:/bin/false
[root@localhost grep]# grep -c false /etc/passwd
1
[root@localhost grep]# grep -i ls ~/.bash* | grep -v history
/root/.bashrc:alias multi='ls'
[root@localhost grep]#
```

위의 예제 중 마지막에 실행한 "grep -i ls ~/.bash* | grep -v history" 문장의 의미는 /root 디렉터리 아래에 .bash로 시작하는 모든 파일들에서 ls라는 문자열을 검색하는데, 이때 대소문자를 구분하지 않고 모두 검색하기 위해 -i 옵션을 사용하고 있다. 그리고 파이프로 연결한 다음 결과값 중 history 문자열을 포함하지 않는 줄을 출력하기 위해 -v 옵션을 사용하였다.

```
[root@localhost grep]# grep korea /etc/passwd
[root@localhost grep]# echo $?
1
[root@localhost grep]#
```

위 예제에서는 grep 결과값으로 검색한 내용이 존재하지 않기 때문에 검색에 실패했으므로 종료상태값은 1로 할당된다.

이번에는 다음과 같이 예제 파일을 만들어두고 간단한 grep 명령어를 사용해 보자.

```
[root@localhost grep]# vim testfile
Seoul           KimLee      50.5        80.5        50.2
Inchon          Hong        91.5        50.3        60.5
Daejun          Bak         30.2        76.4        88.6
Daegu           kim root    80.8        50.6        40.9
Ulsan           Lee         80.6        85.3        56.8
Busan           Kang Hong   85.6        91.7        58.3
```

```
[root@localhost grep]# grep Kim testfile
Seoul           KimLee      50.5        80.5        50.2
[root@localhost grep]# grep [Kk]im test*
Seoul           KimLee      50.5        80.5        50.2
Daegu           kim root    80.8        50.6        40.9
```

```
[root@localhost grep]# grep ^D testfile
Daejun          Bak          30.2          76.4          88.6
Daegu           kim root     80.8          50.6          40.9
[root@localhost grep]# grep '8$' testfile
Ulsan           Lee          80.6          85.3          56.8
[root@localhost grep]# grep Kang Hong testfile
grep: Hong: No such file or directory
testfile:Busan  Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep 'Kang Hong' testfile
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep '5\..' testfile
Ulsan           Lee          80.6          85.3          56.8
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep '\.5' testfile
Seoul           KimLee       50.5          80.5          50.2
Inchon          Hong         91.5          50.3          60.5
[root@localhost grep]# grep '^[SB]' testfile
Seoul           KimLee       50.5          80.5          50.2
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep '[^0-9]' testfile
Seoul           KimLee       50.5          80.5          50.2
Inchon          Hong         91.5          50.3          60.5
Daejun          Bak          30.2          76.4          88.6
Daegu           kim root     80.8          50.6          40.9
Ulsan           Lee          80.6          85.3          56.8
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep '[a-z][a-z] [A-Z]' testfile
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep 'ng*' testfile
Inchon          Hong         91.5          50.3          60.5
Daejun          Bak          30.2          76.4          88.6
Ulsan           Lee          80.6          85.3          56.8
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep '[a-z]\{5\}' testfile
Inchon          Hong         91.5          50.3          60.5
Daejun          Bak          30.2          76.4          88.6
```

```
[root@localhost grep]# grep '\<Dae' testfile
Daejun          Bak          30.2          76.4          88.6
Daegu           kim root     80.8          50.6          40.9
[root@localhost grep]# grep '\<[A-Z].*n\>' testfile
Inchon          Hong         91.5          50.3          60.5
Daejun          Bak          30.2          76.4          88.6
Ulsan           Lee          80.6          85.3          56.8
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]#
```

>> grep 명령어와 옵션 사용 예제

```
[root@localhost grep]# grep -n '^Dae' testfile
3:Daejun        Bak          30.2          76.4          88.6
4:Daegu         kim root     80.8          50.6          40.9
[root@localhost grep]# grep -i 'kim' testfile
Seoul           KimLee       50.5          80.5          50.2
Daegu           kim root     80.8          50.6          40.9
[root@localhost grep]# grep -v 'kim' testfile
Seoul           KimLee       50.5          80.5          50.2
Inchon          Hong         91.5          50.3          60.5
Daejun          Bak          30.2          76.4          88.6
Ulsan           Lee          80.6          85.3          56.8
Busan           Kang Hong    85.6          91.7          58.3
[root@localhost grep]# grep -l 'Daegu' *
testfile
[root@localhost grep]# grep -c 'Dae' testfile
2
[root@localhost grep]# grep -w 'kim' testfile
Daegu           kim root     80.8          50.6          40.9
[root@localhost grep]# echo $LOGNAME
root
[root@localhost grep]# grep -i "$LOGNAME" testfile
Daegu           kim root     80.8          50.6          40.9
[root@localhost grep]#
```

>> grep 명령어와 파이프 사용 예제

다음의 예제는 ls 명령과 파이프로 grep를 사용하여 디렉터리만 출력하도록 한 것이다. 파일 목록 출력에서 디렉터리는 첫 문자로 d를 가지기 때문에 grep에서 정규표현식 ^d를 검색하면 쉽게 출력해 볼 수 있다.

```
[root@localhost ~]# pwd
/root
[root@localhost ~]# ls
aaa.sh            datefile       install.log    testcommand.sh while.sh
anaconda-ks.cfg   file           install.log.syslog            testfunc.sh
bin               for.sh         plus.sh        test.sh
chapter1          grep           scsrun.log     var1.sh
comm.sh           grouping       shell          var.sh
[root@localhost ~]# ls -l | grep '^d'
drwxr-xr-x 2 root root           4096 2009-07-18 20:24 bin
drwxr-xr-x 8 root root           4096 2009-07-18 16:18 chapter1
drwxr-xr-x 2 root root           4096 2009-07-20 16:08 grep
drwxr-xr-x 5 root root           4096 2009-07-20 14:16 shell
[root@localhost ~]#
```

추가적인 grep 명령 예제는 다음의 표를 참고하자.

표 4-3 • grep 예제

grep 예제	설명
grep '\<Tom\>' file	단어 Tom이 있는 라인을 출력한다.
grep 'Tom Jerry' file	'Tom Jerry'를 포함하는 라인을 출력한다.
grep '^Tommy' file	라인의 시작이 Tommy 문자열로 시작되는 라인을 출력한다.
grep '\.bak$' file	라인의 끝이 .bak으로 끝나는 라인을 출력한다. 여기서 작은따옴표(' ')는 $ 사인을 인터프리터의 해석으로부터 보호한다.
grep '[Pp]hoto' *	현재 작업 디렉터리의 모든 파일명에서 photo 또는 Photo를 포함하고 있는 파일명을 찾은 라인을 출력한다.
grep '[A-Z]' file	대문자를 하나라도 포함하고 있는 라인을 출력한다.
grep '[0-9]' file	숫자를 하나라도 포함하고 있는 라인을 출력한다.
grep '[A-Z]...[0-9]' file	대문자로 시작해서 마지막이 숫자로 끝나는 5개의 문자열 패턴을 가지고 있는 라인을 출력한다.
grep -w '[tT]est' files	Test와 test 단어를 가지고 있는 라인을 출력한다.
grep -s 'TY Kim' file	'TY Kim'을 가지고 있는 라인들을 찾지만 화면에 출력하지는 않는다.
grep -v 'Jerry' file	'Jerry'를 포함하고 있지 않은 모든 라인을 출력한다.
grep -i 'sam' file	대소문자를 무시하고 sam을 포함하는 모든 라인을 출력한다. 예) SAM, sam, SaM, sAm 등
grep -l 'Dear Boss' *	'Dear Boss'를 포함하고 있는 모든 파일 목록을 출력한다.
grep -n 'Tom' file	Tom을 포함하고 있는 라인들을 출력할 때 라인번호도 함께 출력한다.
grep "$name" file	name 변수의 값을 가지고 있는 라인을 모두 출력한다. 변수를 사용할 경우에는 반드시 큰따옴표(" ")를 사용해야 한다.
grep '$5' file	문자 '$5'를 포함하는 라인을 출력한다. 반드시 작은따옴표를 붙여주어야 한다. "안의 $는 문자 그자체로 인식.

ps aux | grep '^ *multi'	ps aux의 출력을 grep와 파이프하고 라인 앞에 공백을 포함하여 multi가 있는 라인은 모두 출력한다.
	`[root@localhost ~]# ps aux | grep '^ *multi'`
	`multi 2709 0.0 0.6 9980 1664 ? S 11:33 0:01 sshd: multi@pts/0`
	`multi 2710 0.0 0.5 5736 1432 pts/0 Ss 11:33 0:00 -bash`
	`[root@localhost ~]#`

4.2 | egrep

egrep[Extended grep]는 grep의 확장으로서 추가적인 정규표현식 메타문자들을 사용할 수 있다.

표 4-4 • egrep에 추가된 메타문자들

메타문자	역할	예제	설명
^	라인의 시작	'^linux'	linux로 시작하는 모든 라인 검색
$	라인의 끝	'linux$'	linux로 끝나는 모든 라인 검색
.	하나의 문자 매칭	'l…x'	l을 포함하고 3개의 문자가 있고 다음에 x문자가 있는 라인을 검색
*	문자가 없거나 그 이상의 문자들이 매칭	' *linux'	linux 문자 앞에 아무것도 없거나 공백이 존재하는 라인을 검색
[ ]	[ ] 안의 문자 중 하나라도 매칭되는 문자	'[LI]inux'	Linux 또는 linux를 포함하는 라인을 검색
[^]	[ ] 안의 문자 중 하나도 매칭되지 않는 문자	'[^A-KM-Z]inux'	inux 앞의 문자에 A에서 K까지의 문자가 없고 M에서 Z까지의 문자가 없는 라인을 검색
egrep에 추가된 메타문자들			
+	+ 앞의 문자 중 하나 이상이 매칭되는 문자	'[a-z] + inux'	inux 앞에 하나 이상의 소문자를 포함하고 있는 라인을 검색(linux, mylinux, sulinux, ginux, frdoralinux, centoslinux 등)
?	바로 앞의 문자 하나가 없거나 하나가 매칭되는 문자	'lo?ve'	? 바로 앞에 문자 o가 있거나 문자가 없는 문자열을 가지는 라인을 출력(love, lve)
a|b	a 또는 b와 매칭되는 문자(or)	'love|hate'	love 또는 hate를 포함하는 라인을 검색
( )	문자 그룹	'love(able|ly)' '(ov) +'	loveable 또는 lovely 매칭 한 번 이상의 ov 문자 매칭

```
[root@localhost grep]# pwd
/root/grep
[root@localhost ~]# cat testfile
```

```
Seoul           KimLee        50.5          80.5          50.2
Inchon          Hong          91.5          50.3          60.5
Daejun          Bak           30.2          76.4          88.6
Daegu           kim root      80.8          50.6          40.9
Ulsan           Lee           80.6          85.3          56.8
Busan           Kang Hong     85.6          91.7          58.3
```

```
[root@localhost grep]# egrep 'Kim|Kang' testfile
Seoul           KimLee        50.5          80.5          50.2
Busan           Kang Hong     85.6          91.7          58.3
[root@localhost grep]# egrep '9+' testfile
Inchon          Hong          91.5          50.3          60.5
Daegu           kim root      80.8          50.6          40.9
Busan           Kang Hong     85.6          91.7          58.3
[root@localhost grep]# egrep '8\.?[0-9]' testfile
Seoul           KimLee        50.5          80.5          50.2
Daegu           kim root      80.8          50.6          40.9
Ulsan           Lee           80.6          85.3          56.8
[root@localhost grep]# egrep '(Dae)+' testfile
Daejun          Bak           30.2          76.4          88.6
Daegu           kim root      80.8          50.6          40.9
[root@localhost grep]# egrep 'K(i|a)' testfile
Seoul           KimLee        50.5          80.5          50.2
Busan           Kang Hong     85.6          91.7          58.3
[root@localhost grep]# egrep 'sa|u' testfile
Seoul           KimLee        50.5          80.5          50.2
Daejun          Bak           30.2          76.4          88.6
Daegu           kim root      80.8          50.6          40.9
Ulsan           Lee           80.6          85.3          56.8
Busan           Kang Hong     85.6          91.7          58.3
[root@localhost grep]#
```

다음의 표에 egrep 명령 예제를 정리해 두었다.

표 4-5 • egrep 예제

egrep 예제	설명
egrep '^ +' file	하나 또는 하나 이상의 공백으로 시작하는 라인을 출력
egrep '^ *' file	라인의 시작에 아무것도 없거나 공백이 있는 라인을 출력
egrep '(Tom|Dan) Jerry' file	Tom Jerry 또는 Dan Jerry를 포함하는 라인을 출력
egrep '(ab) +' file	ab가 한 번 또는 그 이상 나오는 라인을 출력
egrep '^X[0-9]?' file	X 문자로 시작하고 다음에 아무것도 없거나 하나의 숫자만 있는 문자열을 가지는 라인을 출력
egrep 'fun\.$' *	모든 파일에서 fun으로 끝나는 문자열을 가지는 라인을 출력
egrep '[A-Z] +' file	하나 이상의 대문자를 포함하고 있는 라인을 출력
egrep '[0-9]' file	숫자를 포함하고 있는 라인을 출력
egrep '[A-Z]...[0-9]' file	5개의 문자 패턴으로 대문자로 시작하고 3개의 어떤 문자가 와도 되며, 마지막에는 숫자가 오는 문자열을 포함하고 있는 라인을 출력
egrep '[tT]est' files	test 또는 Test를 포함하고 있는 라인을 출력
egrep -v 'Jerry' file	"Jerry"를 포함하고 있지 않은 라인을 모두 출력
egrep -i 'sam' file	대소문자에 상관없이 sam을 포함하는 라인을 모두 출력 예) SAM, sam, SaM, sAm 등
egrep -l 'Dear Boss' *	Dear Boss를 포함하는 모든 파일 목록을 출력
egrep -n 'Tom' file	Tom을 포함하는 각 라인의 번호를 함께 출력
egrep -s "$name" file	name 변수를 포함하는 라인을 찾지만 출력하지 않는다.

4.3 | fgrep

fgrep[Fixed grep or Fast grep] 명령은 grep와 유사하다. 하지만, 정규표현식 메타문자들은 사용할 수 없기 때문에 특수 문자 및 $ 문자들은 문자 그대로 인식한다.

```
[root@localhost grep]# vim fgrep.txt
[A-Z]       $95
B           99
C           66
```

```
[root@localhost grep]# fgrep '[A-Z]' fgrep.txt
[A-Z]       $95
[root@localhost grep]# fgrep '$9' fgrep.txt
[A-Z]       $95
[root@localhost grep]#
```

위 예제의 두 번째 명령에서 사용한 '$9' 패턴은 fgrep 명령에서 $ 문자를 문자 그대로 인식하기 때문에 fgrep.txt 파일의 첫 번째 줄을 검색하여 출력해 준다.