6. 유틸리티 활용

개요

- 여러 가지 유틸리티 프로그램 소개
  - 유닉스/리눅스를 유용하게 활용하도록 하기 위함

- 파일 정렬
- 파일 비교
- 텍스트 변환
- 정규표현식과 grep
- 스트림 편집기 sed
- 파일 보관 및 압축
- 파일 탐색
- 기타 파일 관련 유틸리티
- 기타 유용한 명령어

6.1 파일 정렬

- sort : 파일 정렬

$ sort file	줄 단위로 정렬(오름차순)
$ sort –r file	내림차순 정렬 (reverse)
$ sort –k2 file	필드2부터 정렬키로 사용
$ sort –b –k2 file	필드2 앞부분의 공백을 제외하여 사용
$ sort –n –k4 file	필드4를 숫자 순서로 정렬
$ sort –b -k3,3 file	필드3
$ sort -n -k4 -k3 file	필드4를 1차 키로, 필드3을 2차 키로 사용
$ sort -t: -n -k2 file	구분자로 문자 :을 사용 (기본 구분자는 공백)

옵션 상세 설명은 p135 표 참조

파일 병합 정렬, 반복 줄 제거

- sort –m : 파일 병합 정렬

$ sort –m file1 file2	**file1, file2가 정렬된 파일**이면 merge 정렬
$ sort file1 file2	보통정렬, -m 옵션을 사용한 것보다 느림
$ sort –u file1	정렬 결과에서 중복된 줄은 한 줄만 출력 (sort와 uniq를 결합한 것과 유사)

- uniq : 반복 줄 제거

$ uniq file	중복된 줄은 한 줄만 출력
$ sort file | uniq	주로 정렬과 결합하여 사용
$ uniq –c file	줄의 반복횟수 출력
$ uniq –f 1 file	처음 1필드를 제외(skip)하고 비교
$ uniq –w 4 –c file	처음 4문자만 비교

6.2 파일 비교

- cmp : 파일 비교

```
$ cmp cdata1 cdata2                    ... 두 파일 비교
cdata1 cdata2 differ: byte 11, line 1  … 다르면 다른 처음 위치 출력
                                       … 종료코드 1
$ cmp cdata1 cdata1                    … 같으면 종료코드 0, 출력없음
$ cmp –s cdata1 cdata2                 … 출력없음(종료코드 만 반환)
```

- diff : 파일 차이점 출력 (줄 단위로)
  - 출력 형태

(1) 첫째 파일에서 둘째 파일로 변환할 때의 변경 목록

(2) C전처리기에 의해서 두 파일 중 하나가 나오도록 두 파일을 merge함.

파일 비교 – diff

```
$ diff file1 file2
     addition:   n1 a n3, n4
                      > (file2 추가 내용)
      deletion:   n1,n2 d n3
                      < (file1 삭제 내용)
      change:    n1,n2 c n3,n4
                      < (file1 삭제 내용)
                      > (file2 추가 내용)
$ diff -Dflag file1 file2              // flag정의 … C전처리기 사용 출력
       #ifndef flag
              (file1 삭제 내용)
       #else flag
              (file2 추가 내용)
       #endif
```

patch – 파일 차이점 적용

- patch :
  - diff 명령어 출력 파일을 원본에 적용하여 수정본으로 갱신

```
$ diff d1.c d2.c > d12.diff
$ cp d1.c d3.c                  … d3.c는 원본과 같음

$ patch d3.c d12.diff           … patch 적용, d3.c는 수정본으로 갱신
```

6.3 텍스트 변환

- tr : 문자 변환
  - **표준입력** 문자를 한 문자 집합에서 다른 문자 집합으로 변환

```
$ tr a-z A-Z                  … 입력: stdin, 출력:stdout
$ tr a-c D-E < file           … 입력: file
$ tr -c a-zA-Z '\n' < file    … 알파벳이 아닌 문자를 개행문자로
$ tr -cs a-zA-Z '\n' < file   … 변환된 문자 반복출력 제거
$ tr -d abc < file            … abc문자 제거
```

- fmt : 파일 포맷 지정 – 줄당 문자 수, 들여쓰기 등

```
$ fmt file                    … 기본 포맷으로 출력 (줄당 최대 75문자)
$ fmt  -w 50 file             … 출당 최대 50문자
$ fmt -s file                 … 짧은 줄은 그대로(긴 줄만 분리)
$ fmt -u file                 … 단어 간격: 한 칸 (한 줄로 합칠 때는 두 칸)
```

텍스트 변환 – column, cut

- column : 여러 열로 포맷

  $ column file … 여러 열로 만들어 출력 (세로로 먼저 나열)

  $ column –x file … 가로로 먼저 나열

  (cf) ls 와 ls -C

  $ column –t file … 여러 열로 된 입력을 열 단위로 정렬

- cut : 줄의 지정된 영역 제거, 선택부분 만 출력

  $ cut –c 2-10 … 줄의 2번째 부터 10번째 문자까지 선택

  $ cut –f1-3,5 –d' ' … 필드1-3, 5만 출력(필드 구분자: 빈칸)

텍스트 변환 – paste, iconv

- paste – 줄 단위 병합

  $ paste f1 f2 f3 … 세 파일을 병합(1열:f1, 2열:f2, 3열:f3), 구분자 탭

  $ paste –d: f1 f2 … 구분자 :

- iconv – 문자 인코딩 변환
  - 문자를 한 인코딩에서 다른 인코딩으로 변환. 한글 변환에 유용
  - 한글 완성형: euc-kr, 유니코드: utf-8

  $ iconv –f euc-kr –t utf-8 han > han.out … 완성형을 utf-8로

  $ file han han.out … 파일유형 확인

6.4 정규표현식과 grep

정규표현식(regular expression)

- 문자열 패턴을 나타내는 데 사용되는 표기법
- 셸의 와일드카드와 유사하지만 형식에 차이가 있음
- vi, grep, sed, awk 등의 명령어, perl, python 등의 언어에서 사용

정규표현식의 메타문자

문자	의미	예
.	임의의 한 문자	.
[ ]	대괄호 안의 문자들 중 한 문자	[abx-z]
[^]	대괄호 안의 문자들을 제외한 문자들 중 한 문자	[^abc]
*	앞의 문자가 0번 이상 반복	a*
^	줄의 시작	^abc
$	줄의 끝	xyz$
\	다음의 메타문자의 특수한 의미 제거	*
c	문자 c (비메타문자)	pqr

확장된 정규표현식의 메타문자

문자	의미	예
+	앞의 문자가 1번 이상 반복	a+
?	앞의 문자가 0번 또는 1번 발생	a?
|	앞과 뒤의 두 패턴 중 하나 (or 연산)	unix|linux
( )	뒤에 나오는 메타문자가 괄호 안의 패턴에 적용	(ab)*, (ab|cd)e

정규표현식과 일치하는 문자열 예

- 교과서 p152 참조

grep, fgrep, egrep – 파일 필터

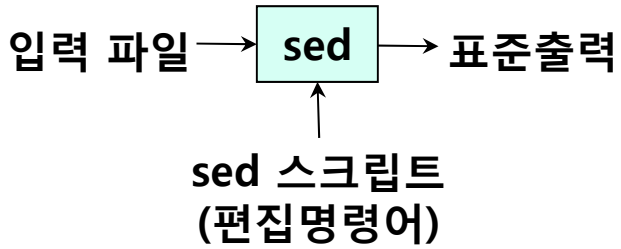
- grep/fgrep/egrep
  - 파일에서 주어진 패턴을 포함한 줄만 출력(필터)
  - 패턴
    - grep : 정규표현식
    - egrep : 확장된 정규표현식 (grep –E와 같음)
    - fgrep : 문자열 (grep –F와 같음)
  - 다른 명령어와 파이프로 결합하여 많이 사용

```
$ grep pattern file …
$ grep 'pattern' file …          셸의 메타문자가 포함된 패턴은 ' ' 사용

$ grep –w pattern file …         단어 전체와 일치하는 줄 출력
$ grep –v pattern file …         패턴을 포함하지 않는 줄 출력
$ grep –n pattern file …         행 번호도 함께 출력
$ grep –l pattern file …         패턴을 포함한 파일이름만 출력
$ grep –e pat1 –e pat2 file …    여러 개의 패턴 제공
```

6.5 스트림 편집기 sed

- sed
  - 미리 준비된 편집 명령어에 따라서 입력파일을 일괄 편집하여 편집된 결과를 출력



  - 사용 형식

	편집명령어 제공 방법
$ sed *script* file1 …	인수
$ sed -e *script* … -e *script* file1 …	여러 개의 인수
$ sed -f *scriptfile* file1 …	파일

$ sed … > outfile 편집된 결과를 파일로 저장

sed 편집 명령어

명령어 구문	의미
range d	(delete) range가 지정하는 행 범위의 텍스트를 삭제
addr a\ *text*	(append) addr가 지정하는 행 뒤에 text를 삽입. text가 여러 줄인 경우에는 마지막 줄을 제외하고 줄 끝에 \를 포함한다.
addr i\ *text*	(insert) addr가 지정하는 행 앞에 text를 삽입 text 형식은 append와 같음
range c\ text	(change) range가 지정하는 행 범위의 텍스트를 text로 변경 text 형식은 append와 같음
addr r *file*	(read) addr가 지정하는 행 뒤에 파일 file의 내용을 삽입
range s/*expr*/*str*/	(substitute) 정규표현식 expr과 일치하는 행의 첫째 문자열을 str로 치환
range s/*expr*/*str*/g	(substitute global) 정규표현식 expr과 일치하는 모든 문자열을 str로 치환
range w *file*	(write) range가 지정하는 행 범위의 텍스트를 file에 저장
range p	(print) range가 지정하는 행 범위의 텍스트를 화면에 출력

sed 편집 명령어(계속)

- 주소와 주소범위 – 편집 범위 지정
  - 없음 : 모든 줄에 대해서 편집 수행
  - 3 : 3행
    /pat/ : 패턴 pat을 포함하는 줄
  - 3, 10 : 3행부터 10행까지
    /begin/,/end/ : 패턴 begin을 포함하는 줄부터 end를 포함하는 줄까지

- 텍스트 치환

```
$ sed 's/line/row/' file
$ sed '1,2/line/row/g' file          … 모든 매칭 문자열 치환
$ sed 's/^/  /' file                 … 정규표현식 패턴, 들여쓰기
$ sed '5s/ *//' file                 … 줄 앞의 빈칸 제거
$ sed '/\<line\>/row/I' file         … 단어 line을 대소문자 구분 없이 치환
```

sed 편집 명령어(계속)

- 텍스트 삭제

```
$ sed '1,3d' file
$ sed '/Line/ d' file
$ sed '/line/I d' file
$ sed '/one/,/three/d' file
```

- 다중 편집 명령어 인수

```
$ sed -e 's/^/<</' -e 's/$/>>/'  file
```
... 줄의 앞뒤에 <<, >>를 삽입

- 파일 삽입

```
$ sed '$r file2'  file
$ sed '10r file2'  file
```
… file의 끝에 file2 삽입

… 10행 뒤에 file2 삽입

sed 편집 명령어(계속)

- 텍스트 삽입 – i (insert), a (append)

"sed1"

```
1i\
Inserted Line 1\
inserted line 2
```

... 1행 앞에 두 줄 삽입

… 마지막 줄을 제외하고 \로 끝남

$ sed -f sed1 file

"sed2"

```
$a\
Appended Line
```

... 마지막 행($) 뒤에 한 줄 삽입(추가)

$ sed -f sed2 file

sed 편집 명령어(계속)

- 텍스트 변경 – c (change)

"sed3"

```
1,2c\
Line 1-2 are removed
4c\
the fourth line
```

$ sed -f sed3 sed.in

- 선택 행 출력 – p (print)

$ sed '1,2p' file … 선택 행 출력 및 편집결과 출력

$ sed –n '1,2p' file … 선택 행만 출력(편집결과 출력 없음)

$ sed –n /pat/p' file … grep 'pat' file과 같은 동작

- 파일 저장

$ sed '1,2w sed.out' file … 1,2행 파일저장 및 편집결과 출력

$ sed –n '1,2w sed.out' file … 1,2행 파일저장(편집결과 출력 없음)

$ sed -n -e '/line/w out1' -e '/Line/w out2' sed.in … 일부 행들을 분류 저장

셸 스크립트에서 작성한 sed 명령어

- sed 편집 명령어를 셸 스크립트에 포함하여 작성 가능

"sed4.sh" ... sed 명령어를 포함한 셸 스크립트

```
#!/bin/sh
sed '
1i\
Inserted Line 1\
inserted line 2' sed.in
```

... **따옴표** 안에 여러 줄의 편집 명령어 작성

```
$ ./sed4.sh
```