最低限の sed と awk

1 sed

1.1 文字列の置換

- sed 's/cat/dog/' pet.lst　% s コマンド

  ファイル pet.lst の各行で、最初の cat を dog に置換する。

- sed 's/cat/dog/g' pet.lst　% s コマンドと g フラグ

  ファイル pet.lst の各行で、全ての cat を dog に置換する。

- sed -n 's/cat/dog/gp' pet.lst

  ファイル oet.lst の各行で、全ての cat を dog に置換し、置換が行われた行のみを出力する。

  -n はデフォルト出力を止め、/p は置換結果のみを出力する。

- sed -f cattodog.sed pet,lst は sed s/cat/doc/g pet.lst と同じ。ここで cattodogg.sed は次のスクリプト；

  s/cat/dog/g

- sed 'y/adcde/12345/' list.lst　% y コマンド

  ファイル list.lst の a,b,c,d,e をそれぞれ 1,2,3,4,5 で置換する。

1.2 特定の行を出力する

1.2.1 行番号を指定する

- sed -n '5p' price.lst

  ファイル price.lst の第５行を出力する。

- sed '5\!d' price.lst

  ファイル price.lst の第５行以外を消去して出力する。

- sed -e '1,4d;6,$d' price.lst
  - ファイル price.lst の １～４行と６～最終行を消去して出力する。
  - -e；複数のスクリプトに従って処理するように指示するオプション。
  - $；最終行

- sed -e '1,4d' -e '6,$d' price.lst
  - ファイル price.lst の １～４行と６～最終行を消去して出力する。

1.2.2 正規表現を用いる

- sed -n '/abelian/p' integral.lst

  ファイル integral.lst から abelian を含む行を出力する。

- sed -n '/finite$/p' integral.lst

  ファイル integral.lst から finite を行末に含む行を出力する。

- sed -n '/cat/,/dog/p' pets.lst

  ファイル pets.lst で cat を含む行から dog を含む行までを出力する。

- sed -n -f dogstop.sed pets.lst

  ファイル pets.lst で cat を含む行から dog を含む行の最初のものを出力して、そこで止まる。スクリプト dogstop.sed は

  ```
  /cat/,/dog/p
  /dog/q
  ```

1.3 テキストの追加、挿入

- sed -f underline.sed book.tex

  underline.sed はスクリプト

  ```
  /Chap.[0-9][0-9]*[ ]/a\
  ---------------------------------
  ```

  book.tex の各章の下に線を入れる。

- sed -f overline.sed book.tex

  overline.sed はスクリプト

  ```
  /Chap.[0-9][0-9]*[ ]/i\
  ---------------------------------
  ```

  book.tex の各章の上に線を入れる。

- sed -f overunderline.sed book.tex

  overunderline.sed はスクリプト

  ```
  /Chap.[0-9][0-9}*[ ]{
  i\
  ---------------------------------
  a\
  ---------------------------------
  }
  ```

  book.tex の各章の上下に線を入れる。

2 awk, gawk, jawk, jgawk

2.1 awk の入力データ

- awk の基本パターン；awk'*pattern*{*action*}'*file*
- awk の入力データは
  - レコード：レコードセパレータ（既定値は改行文字）で区切られた文字列、
  - フィールド：フィールドセパレータ（既定値はタブ、スペース）で区切られた文字列、
  - フィールドセパレータがタブ、スペースならば複数のフィールドセパレータは１個のフィールドセパレータと解釈される。フィールドセパレータがタブ、スペース以外ならば複数のフィールドセパレータは複数の０文字フィールドと解釈される。

2.2 awk の基本書式

- awk '! /gold/' metal.tex

  ファイル metal.tex から gold を含まない各レコードを出力する。

- awk '/gold/{print FILENAME":"$0}' metal.tex

  ファイル metal.tex から gold を含む各レコードを、ファイル名：と共に出力する。

  * $n：n=0 ならばレコード全体、n>0 ならば第 n フィールド。
  * awk の組み込み関数は

FILENAME	現在の入力ファイル名
RS	レコードセパレータ
FS	フィールドセパレータ
NF	現在読み込んでいるレコードのフィールド数
NR	処理しているレコード番号

- awk 'BEGIN{print "results"} /result/{print $1}END{print "end"}' keisan.tex

  ファイル keisan.tex を処理する前に results とプリントしてから、keisan.tex の result を含むレコードの第１フィールドをプリントし、最後に end とプリントする。

- awk '$2<=3 && $4>=5 && $5!=1{print $1}' test.tex

  ファイル test.tex で、第２フィールドが３以下、かつ第４フィールドが５以上、かつ第５フィールドが１でないレコードの第１フィールドをプリントする。

- awk '$2==3 || $4>5{print $1}' test.tex

  ファイル test.tex で、第２フィールドが３に等しいか、または第４フィールドが５より大であるレコードの第１フィールドをプリントする。

- awk '$2 ~/good/ || $3 !~/bad/{print $1}' nice.tex

  ファイル nice.tex で、第２フィールドが good を含むか、または第３フィールドが bad を含まないレコードの第１フィールドをプリントする。

- awk '/June/,/August/ {print $1}' schedule.tex

  ファイル schedule.tex から June を含むレコードから August を含むレコードの間の全てのレコードの第１フィールドをプリントする。

- `awk -F:'$7~/csh/{prine $1}'/etc/passwd`

  フィールドセパレータを：に変更し、/etc/passwd の第7フィールドが csh を含むレコードの第1フィールドをプリントする。

- `awk 'BEGIN{FS=":"}$7~/csh/{prine $1}' /etc/passwd`

  フィールドセパレータを：に変更し、/etc/passwd の第7フィールドが csh を含むレコードの第1フィールドをプリントする。

- `awk -f script text.txt`

  awk プログラム script を読み込んで、ファイル text.tex を処理する。

2.3 awk で使える actions

2.3.1 演算子

二項演算子	
i+1	変数 i に 1 を足す
3-2	3 から 2 を引く
n*4	変数 n に 4 を掛ける
10/5	10 を 5 で割る
10%3	10 を 3 で割った余り
2^3	2 の 3 乗

単項演算子	
i++	変数 i の値を 1 増やす
j--	変数 j の値を 1 減らす

代入演算子	
i=1	i に 1 を代入
i+=1	i に i+1 を代入
i-=(a+1)	i に i-(a+1) を代入
i*=n	i に i*n を代入
i/=5	i に i/5 を代入
i^=n	i に i^n を代入

2.3.2 出力書式を整える；printf

変換指示子（出力は全て右寄せ）	
%20s	20 桁の文字列
%15d	15 桁の 10 進整数
%10e	10 桁の 10 進小数（指数型）
%10.5f	全 10 桁の 10 進小数（小数部 5 桁）
%15g	15 桁で e, f の短くなる方を自動選択
%10o	10 桁の 8 進数
%10x	10 桁の 16 進数
\n	改行文字（これが無いと改行しない）
%%	% を出力

print は出力の終わりに自動的に改行されるが、printf は\n（改行文字）を入れないと改行しない。

- `awk '{printf("%15d%10e\n",$1,$2)}'list.tex`

  ファイル list.tex の第１フィールドを 10 進整数 15 桁、第２フィールドを 10 進小数（指数型）10 桁でプリントして改行する。

- `awk '{printf("%s\n%s \n\n",$1,$2)}' list.tex`

  ファイル list.tex の第１フィールド、第２フィールドを既定値桁の文字列で空白行をはさんで縦に並べてプリントする。

2.3.3 制御文

- if, else

```
{
   if ($2<$3) mark="↑"
   else if ($2>$3) mark="↓"
   else mark="→"

   print $0,mark
}
```

  第３フィールドが第２フィールドより大きい（resp. 小さい、等しい）ならば、mark に↑（resp. ↓、→）を代入し、レコード全体の後に mark の内容をプリントするスクリプト。

- while

```
{
   printf("%30s",$1)
   {
       while($2>=100){
                   printf("*")
                   $2-=100
                       }
   }
   printf("\n")
}
```

  第１フィールドの内容を 30 桁文字列右詰めで出力した後、[$2/100] 個の * をプリントするスクリプト。

- for

  - for(i=0;i<12;i++)；初めに i=1 として処理し、i<12 ならば i を 1 増やして処理を行う。i=12 なら処理を終わる。

```
{
   sum=0
   for(i=1;i<=12;i++){
                sum+=$(i+1)
                     }
   print $1,sum
}
```

    各レコードの第２フィールドから第１３フィールドの合計を sum として、第１フィールドと sum をプリントするスクリプト。

– ファイル book.year のレコードは、本の題名と各月の売り上げである。次のスクリプトを monthsum.awk とする；

```
{
   for(i=1;i<=12;i++){
              sum[i]+=$(i+1)
                     }
   print
}
END{
      printf("合計")
      for(i=1;i<=12;i++){
                printf("%d",sum[i])
                         }
      printf("\n")
    }
```

awk -f monthsum.awk book.year は、ファイル book.year の各レコードを改行しながらプリントし、改行して、合計の後に各月の合計をプリントし、改行する。

2.3.4 パターンに応じて別のファイルに出力する

次のスクリプトを select.awk とする；

```
$1~/UNIX/{print>"UNIX.book"}
$1~/C/{print>"C.book"}
$1~/Shell/{print>"Shell.book"}
```

awk -f select.awk book.list は、ファイル book.list の各レコードの第1フィールドが UNIX（resp. C, Shell）を含めば、そのレコードをファイル UNIX.book（resp. C.book, Shell.book）に書き出す。