本稿は Linux Japan 誌 1999 年 12 月号に掲載された記事に補筆修正したものです.

貧相コマンドライン生活

LJ を含めて雑誌の付録 CD-ROM から数種類の Linux をインストールすると，まあ簡単になって，しかも最初から日本語 OK だし，普及は順調に進んでいますね．実用的な X クライアントも揃ってきたし，コマンドラインなんかを知らなくても困らない状況になりつつあるようです．さて，家では奥さんがメールをやりとりするのに PostPet なんぞを使って（悔しけどこれが結構面白いんですね）インターネットができると自慢気でおりますし，かくゆう私も将棋や囲碁をしたりして『変な人』と呼ばれたりしています．Linux もそのように使えるべきでもありますが，そればかりでもつまらないですね．Linux を趣味とする筆者にとっては，簡単なプログラムが書けて，ネットワークが組めてという部分こそ大きな魅力なのです．そんなユーザーにとっては，/bin 以下のコマンドやシェルが自由に使えることこそ大きな喜びとなるに相違ありません．それに，初心者がちょっと設定につまずいているとき，キーボードからコマンドをシャカシャカ（その際，目にも止まらぬ速さであれば申し分なし（^^;）と打ち込んで，問題を解決し，さっと立ち去る（蘊蓄は嫌われるので『直しといたよ』の一言くらいがいいようです）自分の姿を想像すると，やる気が起きてきます（ホンマかいな）.

という訳で，中上級を目指すならコマンドラインの習得は必須です．また，良く言われることですが，UNIX ではある作業を行うのに，単機能コマンドの組合せとして実現していくスタイルが古くから好まれています．今回は『貧相コマンドライン生活』というタイトルですが，もちろんこれはアイロニーでして，テキストを中心に扱うユーザーにとっては，実は GUI オンリーでは味わえない豊かな生活が待っていると言いたいのです．さて，あまり大風呂敷を広げてばかりではまとまりがつきません．筆者の経験を振り返って，テキストを扱う日常的なコマンドとそれにまつわる話を紹介します．ただし，sed，awk 最近では perl，ruby などのツールはそれだけで書籍一冊が出来てしまう内容なので，またの機会ということで勘弁してください．ここでは，man を読めばおおよそ全容がつかめるような，もう少し規模の小さいツールやコマンドを扱います．

コマンドラインの生活を享受しようというならこれだという本を紹介しておきます．

> B.W.Kernighan, Rob Pike 著， 石田晴久 監訳 『UNIX プログラミング環境』（アスキー出版局）

が定番の一冊です．随所に問題も出されてますから，それを解きながら読むのは秋の夜長にはピッタリでしょう．

テキストを表示するコマンド

cat と リダイレクト

ファイルの中身を誤って変更してしまわないように，つまりただ見るためだけに vi を `-r` オプション付きで起動したり，emacs から `C-x C-r` でファイルを呼んできたりするユーザーはあまりいないでしょう．一般には read-only のツールを使います．less はその代表ですが，使えない場合もあります．実際 /bin にはないようです．`cat` は大概の場合に使うことが可能ですから，知っていると結構重宝します．man で確認して欲しいのですが，面白そうなオプション，

- `-n` 行番号を付ける
- `-b` 空白行を除いて行番号を付ける
- `-s` 複数の空白行を一行にまとめる
- `-v` `LFD` と `TAB` を除いた制御文字を表示する
- `-T` `TAB` を ‘^I’ と表示する

があります．MS-DOS のファイルには平テキストのはずなのに（PC9801 等の）外字が埋め込まれている（しかも，それが空白で表示される）場合などがあって，TEX で処理できなかった経験がありますが，制御文字をあぶり出せれば何とか対処できます．`-b` オプションは，プログラミングのソースに番号を振って表示させたい場合に最適ですね．LJ 3 月号の「コマン道」にて紹介された `nl(number lines)` は行番号付けに特化したコマンドで機能が豊富です．例えば，検索文字列を含む行のみ番号付けするなんて芸当ができます.

cat はエディタの代用品にも使えます．昔，Linux のインストールがまだ大変だった頃，emacs はおろか vi も起動できない状況でファイルの書き換えをしなければならない事態に陥ることがあったとします．そんなときには，

```
cat > filename
```

として，慎重に一行ずつコマンドを入力し最後に `C-d` を押して，目的のファイル filename をでっちあげ，急場を凌ぐことが出来たのでした．もっとも,『そういう場合，普通 `ed` でしょ』とおっしゃる方には申し上げることはございません.

cat 自身は標準出力に文字列を送り出すだけであるのに，シェルのリダイレクトを使って，ファイルに振り替えるられるところが組合せの妙というところでしょう．ところで，このリダイレクトを使って 0 バイトのファイルが作れます．訪れた guest に挨拶する意味で，ftp サーバーに Youkoso_Kokohe_Kukku_kuh なんて名前のファイルを置くことを考えます（考えません？）．いろいろ方法があるでしょうが，最も簡単なのは

```
> Youkoso_Kokohe_Kukku_kuh
```

です．touch Youkoso... も正解ですが，キーの数でリダイレクトの勝ち．

ついでに，tac をご存じでしょうか．cat に対して tac，そう，ファイルを末尾から表示するコマンドです．シスログやアクセスログ等の時系列のデータを反転して（最近から遡って）表示するなんて状況は，管理者ならば日常の業務として頻繁に遭遇しますね．

tail と head

cat は最後まで一気に表示してしまうので，途中をじっくり見ることはできません．head はファイルの冒頭から，tail はファイルの末尾から指定された行数分を表示します．この二つを組み合わせれば，ファイルの任意のどの部分も表示できます．例えば，100 行目から 125 行目までを表示したかったら

```
head -125 filename |tail -25
```

とします．指定された箇所が本当に表示されているかどうか確かめるには cat の行番号付加オプションを使って，

```
cat -n filename |head -125 |tail -25
```

としてみましょう．

more OR less

more は 1 ページ毎にスクロールを停止しますからじっくりと内容を見ることができます．しかし，一旦見過ごしてしまうと，逆スクロールできないので最初からやり直しです．この辺りが，ちょっと中途半端な感じがします（ただし，cat と同じく，大概いつでも使えます）．less は言わずと知れた，逆スクロールが可能なファイルビューアで，カーソルキーを押してスクロールの向きを変えます．普段はそれだけでしょうが，二つほど覚えておくといいキーコマンドがあります．一つは “h” で，ヘルプ画面が表示されます（ヘルプ画面を抜ける場合も “q” です）．次に検索指定の “/pattern” です（図 1）．pattern には ed が認識できる正規表現が使えます．検索中は n や N で該当個所を巡ります．目的を持って文書を読むときには検索を旨く使って，効率アップしましょう．

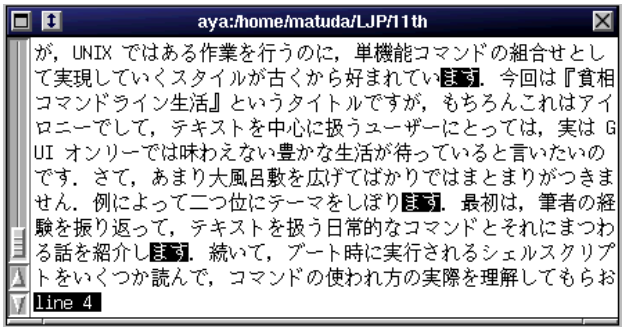

図 1　less で閲覧中に検索をかけた様子．検索文字列 “ます” がハイライト表示されている

strings

rpm や dpkg を使っていると起こらないことでしょうが，Slackware ベースの配布では，ライブラリのバージョンが違っていて動かない場合があります．バイナリ実行ファイルにリンクされているライブラリのバージョンは ldd コマンドで読み出すことができます．例えば /bin/cat にリンクされているライブラリは，

```
$ ldd /bin/cat
        libc.so.5 => /lib/libc.so.5 (0x40003000)
```

さすがに libc.so だけですね．/bin 以下のコマンドはほとんどが libc.so だけで動きますから，大概の場合に動くのです．しかし，大きなアプリケーションになると，ライブラリのバージョンだけではなく，例えば設定ファイルのディレクトリが違っていて動かないなんてこともあります．そういう情報は実行ファイルの中に埋め込まれていますから，それを読み出せればずいぶんと助かりますね．strings はバイナリに含まれる可読文字を取り出して表示してくれます．/bin/cat について実行してみると

```
$ strings /bin/cat |less
/lib/ld-linux.so.1
libc.so.5
...（中略）
GCC: (GNU) 2.7.2.1.2
GCC: (GNU) 2.7.2
...
```

使われたリンカや GCC のバージョンが読み取れます．

od，jhd

バイナリの中の文字列を読み出すには他のツールを利用することで可能です．標準のバイナリダンプ od や，民田雅人氏が作成した日本語対応の jhd などです．滅多に使うことはないでしょうが，JIS，EUC-J，SJIS などの日本語コードや UNIX と MS-DOS の改行コードの違いを確かめてみるという例題が良く教科書などに載ってます．また，od -s では文字列定数だけを表示しますから，strings の代わりにもなります．

テキストを整形するコマンド

expand

タブストップを使って文字揃えをしている場合，タブの設定位置が異なれば，当然レイアウトも違ったものとなります．従って，どんな場合にも書き手の意図通りにレイアウトしたいならば空白は空白文字で埋める必要があります．expand はタブを複数の空白文字に変換します．ちょっと確かめてみましょう．まず，tab で区切ったデータを

```
echo -e '\t*\t*\t*\t*\t*' > tab.tb
```

としてつくります．これを expand を使って空白（間隔）を変えて表示させます．ひとつだけ指定した場合には，その値の整数倍にタブストップがあるものとして空白文字を埋め込みます．

```
$ expand -t2 tab.tb; expand -t6 tab.tb
  * * * * *
      *     *     *     *     *
+----+----1----+----2----+----3----+----4
```

また，タブストップの位置を任意の位置に指定することも可能です．ただし，0 には設定できません．

```
$ expand -t 1,10,14,17,26 tab.tb
 *         *   *  *         *
+----+----1----+----2----+----3----+----4
```

unexpand

タブ文字によって位置揃えをしたい場合には，unexpand を使います．ただし，expand と違って予想外に表示される場合があります．

```
$ echo -e '\t*******\t::::::\t.....\t@' >tab1.tb
$ expand -8 tab1.tb > tab1.txt
```

という試しファイルを使ってみましょう．本当にどうゆうコードが入っているかはバイナリダンプの ascii モード **od -a** で確かめてください．od の ascii モードでは空白文字は sp，タブ文字は ht で表示されます．タブ文字を含まない tab1.txt を unexpand でタブ文字を含むファイルに変換して表示しますと，

```
$ unexpand -8a tab1.txt; unexpand -6a tab1.txt;
        ******* ::::::  .....   @
          ******* ::::::        .....     @
+----+----1----+----2----+----3----+----4
```

となります．最初は正しいですね，次は？．どうしてこう表示されてしまうのかは，端末のタブストップが 8 桁毎に設定されていることを頭に入れて，unexpand されたコードをみれば納得できます．

```
$ unexpand -6a tab1.txt |od -a -An
 ht sp sp  *  *  *  *  *  *  * sp  :  :  :  :  :
  : ht  .  .  .  .  . ht sp sp  @ nl
```

表示を行わせている端末に無関係に正しく表示したければ，<n> を 2 以上の整数として，

```
$ unexpand -<n>a tab1.txt |expand -<n>
```

と expand によりタブ文字を含まないテキストにしてしまえばいいのです．

column

カラムを揃えて表形式でデータを表示するには column が使えます．次のような内容のデータファイル

```
1,1e-06,1234,after
2,2.3e-01,-345,before
11,-3e-9,12.3,during
```

を用意して（col.dat としましょう），column で整形します．

```
$ column -t -s, column.dat
1   1e-06    1234  after
2   2.3e-01  -345  before
11  -3e-9    12.3  during
```

-t を指定すると表を作成します．-s<chars> はカラムを区切る文字セットを指定するオプションです．

colrm

指定したカラムを取り除くコマンドです．単に各行の先頭から指定範囲の数だけ文字数を取り除くだけのようです．したがって，タブ文字が含まれていると変になります．

```
$ column -t -s, column.dat | colrm 4 6
1  -06    1234  after
2  3e-01  -345  before
11 e-9    12.3  during
```

cut

物理的に何文字と指定するのではなく，区切り文字で挟まれた部分（フィールド）を指定して表示するにはコマンド cut を使います．例えば，先程の column.dat の第 2,4 フィールドを切り出して見るには

```
$ cut -d, -f2,4  column.dat
1e-06,after
2.3e-01,before
-3e-9,during
```

のようにします．-d<char> で区切り文字を指定し，-f<num> でフィールドを指定します．またオプション -c<num> を用いると colrm とは逆に指定部分だけを切り出して表示します．

tr

上記の cut の実行例はちょっと見苦しいので，体裁を整えるには column - t -s，にパイプで渡すことになります．その場合にはタブ文字を全く含まないテキストに変換されます．ところで，この例では “,” をタブ文字に変換すれば済みそうです．そこで，文字を変換・消去するツール tr の登場です．実行例を示します．

```
$ cut -d, -f2,4 column.dat | tr "," "\t"
1e-06   after
2.3e-01 before
-3e-9   during
```

連続する同一文字を一つに圧縮する -s と，指定文字を消去する - d が基本的なオプションです．man で例を見ることができますから，このくらいにしておきます．

```
$ jfold -46 11th.tex
cat 自身は標準出力に文字列を送り出すだけである
のに，シェルのリダイレクトを使って，ファイルに
振り替えるられるところが組合せの妙というところ
でしょう．ところで，このリダイレクトを使って 0
バイトのファイルが作れます．訪
```

図 2　jfold -46（全角 23 文字）の実行例

```
$ jfold -30 11th.tex
cat 自身は標準出力に文字列を送
り出すだけであるのに，シェルの
リダイレクトを使って，ファイル
に振り替えるられるところが組合
せの妙というところでしょう．と
ころで，このリダイレクトを使っ
て 0 バイトのファイルが作れま
す．訪
```

図 3　jfold -30（全角 15 文字）の実行例

(j)fold

fold は指定した桁で行を折り返して整形するコマンドです．ところで，当然ながら 2 バイト文字の途中で折り返しが起こると表示がおかしくなります．そこで，PJE には jfold という 2 バイト対応版があります．図 2，図 3 に整形の様子を示します．

indent

C のソースの体裁を整えてくれるプログラマ必携のツールです．間違いでなければ，体裁は個人の趣味の問題かもしれませんが，空白を適当に使って少なくも物理的に読みやすいソースを作成するように心がけましょう．次のような，かなり見苦しいソースも

```
main(int argc, char **argv){
     int j;
  double im, re;

  for (j=1;      j < argc;  j++){
  if (argv[j][0] == '-') {
  switch(argv[j][1]){
  case 'i':
          im = atof(argv[++j]);
          break;
        default: break;
     }
    }
else {
      printf("%s -r 実部 -i 虚部\n", argv[0]);
      exit(0);
}}}
```

```
indent -kr -i2 ソースファイル名
```

により，次のように綺麗に整形されます．

```
main(int argc, char **argv)
{
  int j;
  double im, re;

  for (j = 1; j < argc; j++) {
    if (argv[j][0] == '-') {
      switch (argv[j][1]) {
      case 'i':
        im = atof(argv[++j]);
        break;
      default:
        break;
      }
    } else {
      printf("%s -r 実部 -i 虚部\n", argv[0]);
      exit(0);
    }
  }
}
```

-i2 はインデントの深さを 2 ずつに設定してます．もちろん，もっと他にも細かな指定できます．しかし，代表的なスタイルを指定するオプションがありますからそれを使いましょう．この例での -kr は Kernighan & Ritchie のスタイルを選んだことになります．-orig は Berkley スタイルで，何もなければ当然 GNU の推奨スタイルです．この 3 種類のスタイルがあることを覚えておくと何かの役に立つかもしれません．

端末画面のリセット

バイナリファイルを cat で無理に表示させると，大抵，端末エミュレータの調子が悪くなります．そんな時は，制御文字列 \033c を出力して画面をリセットしましょう．いくつか方法がありますが（LJ3 月号コマン道にもあります），echo を使うなら，

```
echo -e "\033c"
```

です．画面が異常な状態で上記文字列を打ち込むのは面倒なので，.bash_login に

```
function rst () { echo -e "\033c" }
```

とシェルのユーザー関数 rst を定義しておくといいかもしれません．これなら rst と打てばいいですからね．他にも，コマンド reset や clear（C-l も可）も覚えておくと便利です．また，良くあるのが tty デバイスから端末エミュレータへの文字列の出力を C-s を押して停止させ（cat で長いファイルを読む際に併用することがありました）てしまった場合です．これは慌てず，C-q で解除します．tty の設定情報は stty -a で表示されますが，今だに良く理解してません．tty を普通の状態に戻す stty sane くらいは覚えてます．

次回は

どうでしたか，テキスト表示だけでこんなにいろんな話がありました．この調子でもう一回くらい，コマンド話をしてみようと思います．というのも，Plamo と Vine と Redhat（本当は Debian がいいのでしょうが）を同じホストにインストールしたので，この 3 者の /bin に含まれる共通のコマンドを並べてみたところ，

```
arch ash bash cat chgrp chmod chown cp cpio
csh date dd df dmesg dnsdomainname domainname
echo ed false gawk grep gunzip gzip hostname
kill ln login ls mail mkdir mknod more mount
mt mv netstat nisdomainname ping ps pwd red
rm rmdir sh stty su sync tar tcsh touch true
umount uname ypdomainname zcat
```

となっていて，これはおさえとく必要があるなと感じたからです．しかし，総勢 55 個は結構な数ですね．