情報処理　第3回「コマンドとファイル」
2000年4月26日

目 次

1　本日の目標 **1**

2　本日の用語 **1**

2.1　ファイル 1

2.1.1　バイナリファイルとテキストファイル 1

2.1.2　ファイル名 1

2.2　コマンド 1

3　いろいろなコマンド **2**

3.1　ファイルを操作するコマンド 2

3.2　実際にファイルに対して, データ処理をするコマンド 6

4　パニックに陥ったら.... **7**

1　本日の目標

- コマンド を使ってファイルを扱うという事を知る.

2　本日の用語

2.1　ファイル

最初の時間で述べた通り, コンピュータの仕事はデータ処理です. また, von Neumann 型のコンピュータでは, 処理手順もデータとなります.

それらのデータは, 整理された形でコンピュータの内部になければデータの処理が大変であるという事は, 容易に想像が付きます. データを整理する際の最小の単位が, ファイルであると考えて下さい.

ユーザにファイルを使いやすい恰好で維持管理するのは, オペレーティングシステムの基本的な仕事です.

2.1.1　バイナリファイルとテキストファイル

ファイルの中のデータは, 全て 2 進法に変換されコンピュータの 2 次記憶装置 (ハードディスクといわれる磁気円盤) に記録されています. たとえば, アルファベットなどの文字は, 2 進法で 8 桁 (= 8 bit = 1 byte) に変換されますし, 漢字やひらがなは, 16 桁 (= 16 bit = 2 byte) に変換されます. ただし, これら全ての 2 進法の数が文字に割り当てられているわけではなく, 一部分だけです.

ファイルのうち, 文字に割り当てられているデータからなるファイルの事を, テキストファイルといいます. この講義では, ほとんどテキストファイルを使います. そうではなくて, 文字に割り当てられていないデータを含むファイルの事を, バイナリファイルといいます. 厳密に考えると, 全てのファイルはバイナリファイルなのですが, 特にテキストファイルと区別する意味で, この様な言い方をします.

2.1.2　ファイル名

ファイルには全て名前がついており, それをファイル名といいます. ファイル名は, 通常の文字列でユーザはファイル名を用いてそのファイルの内容を利用します. あるいは, ファイル名でそのデータを区別します.

ファイル名には, 通常の文字列が使われます. ファイル名として使えない文字は /(スラッシュ) だけです. ファイル名の長さは, 今では実質上の制限はありません. ただし, 使わない方が良いと思えるファイル名がいくつか有ります. 例えば, RMAIL という名前のファイルは, mule でメールを読んだ時に作成されますから, そのファイル名で別の内容の物を作ると困った事が起きます. 他にもありますが, それらは講義の中で追い追い説明します.

2.2　コマンド

コマンド とは, コンピュータに対する命令を記述したファイルの事であるとここでは考えて下さい. それらが, どこに有るかという事等は, 後の講義で触れます. 以前にも述べましたが, UNIX ではコマンド 名をタイプして ENTER キーを押す事により, コンピュータに命令を与える事ができます. コマンド を用いてデータを入力したり, データを加工したり, あるいは, データ処理の環境を整備したりします.

コマンド は, ほとんどの場合次の形で入力されます.

コマンド 名　[オプション]　[ファイル名または文字列]　[ファイル名または文字列]　$\cdots$

ここで, オプションはコマンドの振舞を少し変化させる時に用いる文字 (文字列) の事で, 多くのコマンドでは, オプションの文字列の前にオプションである事を明示するために – (ハイフン) が付きます. 特にオプションを指定しない時を, デフォルト (**default**) といいます.

ファイル名または文字列の部分は, そのコマンドで加工するファイルまたは文字列が通常入ります. これらをコマンドの引数といいます.

3 いろいろなコマンド

これからいくつかのコマンドについて説明しますが, それらはもっとも頻繁に使われるであろうという事柄についてだけです. より詳しい説明は, **man** コマンド で確認するようにして下さい. どのようなオプションが有り, そのオプション指定をした時のコマンドの振舞の変化を知るには, man コマンドが最も便利です.

説明してあるコマンドは, 全部実際に試してみてください.

3.1 ファイルを操作するコマンド

ls ファイル名の一覧 (list) を出力するコマンド. これまでの演習をこなして来た方は, 少なくともつぎの 4 つのファイルが存在するはずです.

```
% ls
Mail RMAIL  Wnn6 public_html
%
```

ls -a とか ls -l とかのオプション付のコマンドも実行して出力がどのように変化するかを, 確かめて下さい. ls -l の最後の部分は, 表示されたファイルを最後にいじった時刻です.

file ファイル型を表示するコマンド.

```
% file /usr/bin/ls
/usr/bin/ls:    ELF 32-ビット LSB 実行可能形式  80386 バージョン 1[動的にリンクされています][ストリップされています]
%
```

とすれば, でてきます.

cat 本来は, ファイルを連結する (concatenate) ためのコマンドですが, cat ファイル名でそのファイルの内容を表示する.

```
% cat RMAIL
%
```

cat の引数を二つのファイル名にすると

```
% cat file1 file2
```

ファイル file1 に続いて file2 の内容が出力されます. すなわち, 二つのファイルを連結して出力しているのです. 三つ以上のファイルを引数に与えても同様の結果になります.

cat は, ファイルの中身が長いと, 延々と出力をし続けます. その出力を一時的に止めるには, 標準的な環境では, CTRL + s を押します. (CTRL はキーボードにあるコントロールキーと呼ばれるものです) 適当なキーを押す事により, 止まった出力が再開します. また, cat そのものの動作を中止するには, CTRL + c です.

ファイル名を指定しないとどうなるでしょう.

```
% cat
(カーソル)
```

aiueo ENTER と打ってみます.

```
% cat
aiueo  <- これを入力
aiueo
```

となります. 理由はここでは説明しません. とりあえずここは CTRL+d と 1 回だけ打ちます.

```
% cat
aiueo
aiueo
(CTRL+d)
%
```

注意 むやみに知らないファイル (というかバイナリファイル) を cat してはいけません. 暇なときに

```
% cat /usr/local/bin/mule
```

とでもしてみるとよくわかります. 放っておくと結構長時間かかるので, ctrl+c で止める, 止まった後, 運が良ければ何事もなかったようにもとにもどるが、大抵は画面が意味不明な漢字の嵐になる. 別に壊れたわけではないので , そのウィンドウは無視しておけばよい. 必要ならば, 新しい kterm を起動する事.

```
% man cat
```

でわかるはずです.

od ファイルを 8 進表示するコマンド です. これだとバイナリファイルもちゃんと表示できます. ただし, 表示された物が何なのかは, そのままではわかりません. 次を試して下さい.

```
% od /usr/bin/ls
```

wc デフォルトでは, ファイルの中の行数, 単語数, ファイルの大きさを順に出力します. (word count)

私の場合は,

```
% wc RMAIL
       35      152     1045 RMAIL
%
```

と表示されます.

head ファイルの先頭部分を表示するコマンド.

```
% head RMAIL
```

でファイル RMAIL の先頭十行が出力されます. 出力する行数の指定の仕方は, 各自調べて下さい.

tail ファイルの末尾部分を表示するコマンド.

```
% tail RMAIL
```

でファイル RMAIL の末尾十行が出力されます. 出力する行数の指定の仕方は, head と同じです.

more ファイルの内容を 1 ページ毎に分割して表示するコマンド. man コマンドもこの more を利用して表示を分割しています.

```
% more RMAIL
```

を試してください. 次の規則で移動ができます.

スペースバー	1 ページ進む
ENTER	1 行進む
b	1 ページ戻る
?	使い方の簡単な説明がでる
q	more を終了する

less more の機能拡張版.

```
% less RMAIL
```

を試してください. 次の規則で移動ができます.

スペースバー	1 ページ進む
ENTER または j	1 行進む
k	1 行戻る
q	less を終了する
h	使い方の簡単な説明がでる

touch ファイルをいじります. ファイルをいじると言うのは, 例えばエディタで編集すると言うのもそうですが, この touch は内容に変更を加えず, 単にいじるだけです. もしファイルが存在するなら, ls -l で出力される時刻の部分が, touch した時刻になります. もしファイルが存在しなければ, そのファイル名を持つ空のファイルを作ります.

```
% touch
```

を実行して, ls -l で時刻を確かめて下さい. 次に,

```
% touch hogehoge
```

で, hogehoge と言うファイル名の空のファイルを作り, ls で実際にできている事を確認して下さい.

diff 二つのファイルの相違部分を表示する. 違いが無ければ, 何も出力しません.

```
%touch hogehoge2
```

として新しいファイルを作成し, 先程の hogehoge と比較してみたください.

```
% diff hogehoge hogehoge2
%
```

もし `RMAIL~` という名称のファイルが存在したら,

```
% diff RMAIL RMAIL~
```

を試して下さい.

cp ファイルをコピー (copy) するコマンド.

```
% cp hogehoge hogehoge3
%
```

で, hogehoge と同じ内容を持ったファイル hogehoge3 が作られます.(当然中味はカラ) 上を実行して

```
% cat hogehoge3
```

で実際にそうなっているかを確かめて下さい.

mv 本来は, ファイルの位置 (パス名) を変更する (move する) コマンド. パスについては後の講義で詳しく述べます. ここでは, ファイル名の変更にこのコマンドを使います. ここまできちんとやって来た方は, 次を実行して下さい.

```
% mv hogehoge hogehoge4
% ls
    Mail          Wnn6          public_html   hogehoge2
    RMAIL         hogehoge3     hogehoge4
%
```

というようになるはずです.

rm ファイルを消去する (remove) コマンド. 先程作った hogehoge2 というファイルと hogehoge3, hogehoge 4 を消そうとすると, 次のようになります.

```
% rm hogehoge2 hogehoge3 hogehoge4
% ls
    Mail          RMAIL         Wnn6          public_html
%
```

注意 一旦消したファイルを復活する事はできません.

3.2 実際にファイルに対して, データ処理をするコマンド

latex(jlatex) TEX 処理系のコマンドの一つで LATEX(jLATEX) ファイルを処理して dvi (device independent) ファイルを作成する. ここまでの演習で, mule が 使えるようになっている思いますので, 以下の文章を作成し renshu1.tex としてセーブしてください.

```
\documentstyle{article}
\begin{document}
\section{\LaTeX}
Here's how they look when displayed:
$$\sum_{i=1}^n x_i = \int_0^i f$$
and in the text:
$\sum_{k=1}^n k^2 = (1/6)n(n+1)(2n+1)$.
\end{document}
```

このファイルは LATEX 用のファイルです. 実行してうまくいくと, 以下のようなメッセージがでて, renshu1.tex と言うファイルから, renshu1.aux, renshu1.dvi, renshu1.log と新しく三つのファイルが作成されます. renshu1.dvi はバイナリファイルですが, 他はテキストファイルですので, cat で読む事ができます. 興味のある方は, 読んで下さい.

```
% jlatex renshu1.tex
        This is pTeX, Version p2.1.5, based on TeX, Version 3.14159 (EUC) (Web2c 7.0)
        (renshu1.tex
        LaTeX Version 2.09 <7 Dec 1989>
        (/usr/local/share/texmf/tex/inputs/inputs/article.sty
        Document Style `article' <16 Mar 88>.
        (/usr/local/share/texmf/tex/inputs/inputs/art10.sty))
        No file renshu1.aux.
        [1] (renshu1.aux) )
        Output written on renshu1.dvi (1 page, 876 bytes).
        Transcript written on renshu1.log.
        % ls
        Mail         Wnn6         renshu1.aux  renshu1.dvi  renshu1.tex
        RMAIL        public_html  renshu1.doc  renshu1.log
        %
```

renshu1.tex の内容が間違っていて, jLATEX に失敗する時は, x を押した後, ENTER で jLATEX から抜ける事ができますから, mule で renshu1.tex を修正して再度挑戦して下さい.

xdvi dvi(device independent) ファイルを preview するソフトウェア. 上の, jlatex の実行結果を確かめるコマンドのひとつです.

```
% xdvi renshu1.dvi &
%
```

を実行して見ましょう. renshu1.tex を処理した結果がどうなるかが, わかります.

4　パニックに陥ったら....

以下簡単に UNIX 上でパニックに陥ったときの脱出法について述べます.

(1) キーボードから入力した文字が画面にエコーしなかったら (現れなかったら)
CTRL + q
を押す.

(2) キーボードから入力した文字が画面にエコーするがプロンプト (%) が現れないときは
CTRL + d
を押す.

(3) コマンドが暴走してしまったら,
CTRL + c
を入力する. 大抵はこれでクリアーするが, それでもだめなら
CTRL + z
を押す. これはコマンドを一時的に停止させる働きをもっています. このとき完全にコマンドを停止させたいときは, このあと

```
kill %%
```

をすればよろしいです.

(4) (1), (2), (3) でもだめなときは, 違うマシーンからログインしなおしして, おかしくなった端末のプロセスを殺します. これについてはあとで説明します. (自分で調べよう)