第2回UNIXゼミ

川崎 赤塚 堤

1999年4月27日

目 次

第1章　bashの活用

1.1　コマンド ラインの編集

通常，正しく設定されている bash の環境では，コマンド ラインを編集するために，[←]や，[→]でのカーソル移動，[Backspace]による文字削除，[↑]，[↓]による履歴の参照などができます．このため，Windows での文字編集に慣れている方は特に何も覚えなくても簡単な編集は出来ると思います．ここでは，それ以外に覚えておくと便利な bash のキー操作について説明します．なお，Emacs や bash のマニュアルでは通常，キー操作の表示に特殊な記法を使います．また，ここでは，Windows にあわせたキーを表 1.1 書いておきます．

1.1.1　行頭・行末への移動

```
cho hello_ eが足りない
  ↓  C-a  行頭に戻る
cho hello
  ↓  e  eを入力
echo hello
```

逆に行末に移動するにはC-e を使用します．

1.1.2　単語単位の移動

```
echo ello_ hが足りない
  ↓  M-b  一つ前の単語の先頭に戻る
echo ello
```

表 1.1: キー操作の表示

コマンド	操作
M-p	[ESC]を押した後に，[p]を押す．あるいは，[Alt]を押しながら[p]を押す．
C-b	[Ctrl]を押しながら，[b]を押す．
M-C-e	[ESC]を押した後に，[Ctrl]を押しながら，[e]を押す．あるいは，[Alt]と[Ctrl]を押しながら[e]を押す．

```
  ↓  h   h を入力
echo hello
```

一つ先の単語の先頭に移動するには M-f を使用します.

1.1.3 単語の削除

```
echo hello_ hello を bye に直す
  ↓  M-C-h   前の単語を削除
echo _
  ↓  bye   bye を入力
echo bye_
```

後の単語を削除するには M-C-d を使用します.

1.1.4 行単位の削除

```
echo hello, hello   カーソル以下をすべて削除したい
  ↓  C-k   カーソルから行末まで削除
echo _
```

行すべてを削除するには C-u を使用します.

1.1.5 削除した文字列の挿入

M-d,M-C-h,C-k,C-u で削除された文字列は，bash が一時的に記憶しています．C-y で記憶されている文字列をカーソル位置に挿入することが出来ます.

```
echo hello, hello
  ↓  C-k   カーソルから行末まで削除 (記憶される)
echo _
  ↓  C-y   記憶されていた文字列が挿入されるecho hello, hello_
```

1.2 履歴機能

bash はユーザーが入力したコマンドを記憶しています．履歴を表示するには history コマンドを実行します.

```
$ history Enter
1 ls
2 cd
```

```
3 mkdir temp
```

1.2.1 直前のコマンドを実行

単に直前のコマンドを実行するだけならば!!でかまいません．次の例では，直前のコマンドの文字列の一部を置き換えて実行してみます．最初に次のように入力してみます．

```
$ echo hello Enter
hello
```

この状態で，hello の部分 bye に変更したコマンドラインを入力してみる場合，に次のように入力する．

```
$ ^hello^bye^ Enter
echo bye
bye
```

1.2.2 履歴番号での指定

history コマンドで表示した履歴番号を使用して対応したコマンドを実行できます．

```
$ history Enter
1 ls
2 cd
3 mkdir temp
$ !3 Enter
mkdir temp
```

1.2.3 履歴の検索

!の後に文字列を指定すると history リストの中からその文字列で始まるコマンドで最新のものを見つけて実行します．$!echo なら最後に実行した echo コマンドを実行します．

1.3 エイリアス機能

エイリアスとは別名と言う意味ですが，bash にはコマンドに別の名前を付ける機能があります．例えばalias h="history"とすれば以降はh だけでhistory を実行することが出来ます．
一度設定したエイリアスを解除するにはunalias を使用します．先の例でいうとunalias h となります．
良く使われるものとしてはalias ls="ls -F"のように同じコマンド名にオプションを付けたものを

設定することがあります．これをすると以降オプションを付けなくても標準でオプションが有効になります．この場合オプションの無い`ls`を実行したい場合はコマンド名に¥を付けて実行します．
現在設定されているエイリアスをすべてみるには引数なしで`alias`を実行します．

1.4 補完機能

bashの補完機能を使うとファイル名やディレクトリ名をすべて入力しなくても途中で TAB を押すことで補完することが出来ます．
候補が複数ある場合は2回押すと候補一覧が表示されます．

1.5 初期化ファイル

bashではホームディレクトリに`.bashrc`というファイル作り，ログインしたときに環境変数の設定やエイリアスの設定を自動的に行うことが出来ます．

```
# This is an sample .bashrc
alias ls="ls -F"
alias h="history"
```

第2章　Emacs(mule)の基本的な使い方

2.1　Emacs(mule)とは

Emacsはviと並んでUNIXでの標準的なエディターで，高機能でカスタマイズの自由度が高く，今回は取り扱いませんがメーラーとして使用することも出来ます．muleはEmacsの多国語版で，日本語のほかにも中国語なども使用することが出来ます．

2.2　基本用語

Emacsでは様々な動作を`Ctrl`や`ESC`[1]などと他のキーを同時に押すことで行い，コントロールキーを押すことは`C -`で，`ESC`を押すことは`M-`で表します．例えば「`Ctrl`を押しながら`x`を入力する」は`C-x`と表されます．
エディタは一般的に編集をしてもファイルを直接変更するわけでは有りません．Emacsでも開かれたファイルはバッファといったものに格納され編集はこのバッファに対して行うことになります．編集内容をファイルに反映するためには保存しなければなりません．バッファにはそれぞれバッファ名が付き，例えば新規作成したファイル名のついていないバッファは『`*scratch*`』と表示され，ファイルならばファイル名と同じとなります．

2.3　起動

`emacs`と入力すれば起動できます．

`emacs`[ファイル名]`Enter`

2.4　画面構成

図2.1を参照してください．

- **ウィンドウ (バッファ)**

  ファイルの内容などが表示されるところです．

- **モードライン**

  上のバッファの状態が表示されます．

[1] メタキーと呼びます．

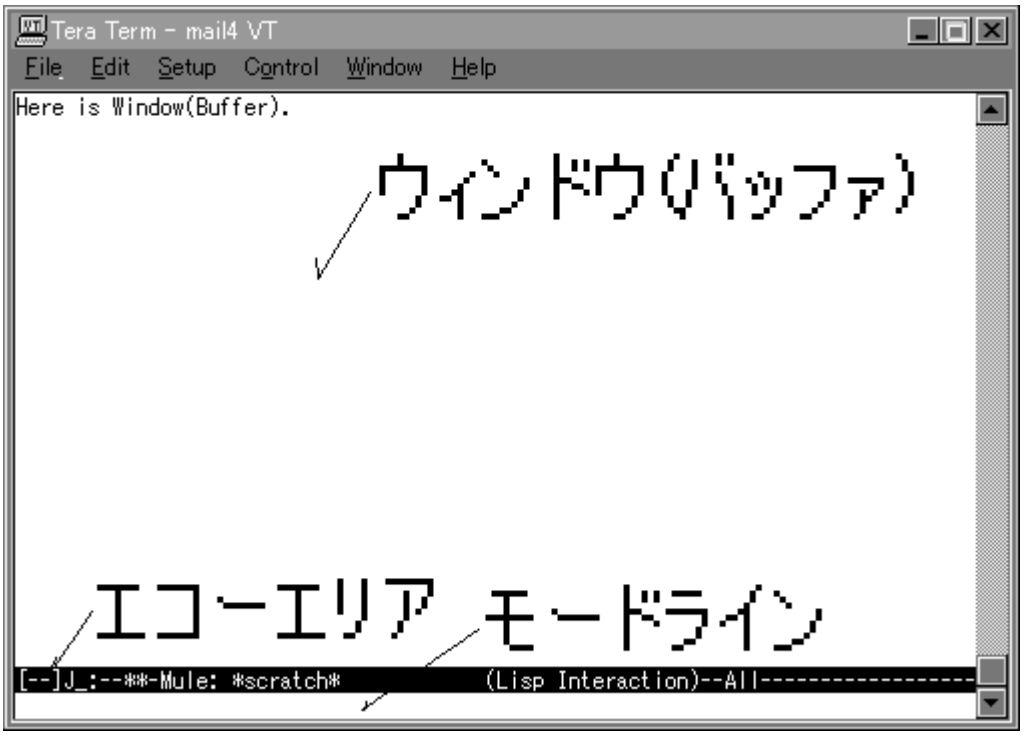

図 2.1: 起動時の画面

- **エコーエリア**

  Emacs からのメッセージが表示されたり，コマンドを入力するところです．
  本テキストではエコーエリアに表示されるメッセージは『』で囲みます．

2.5 終了

終了するには C-x C-c と入力します．場合によってはメニューから終了することも出来ますが，メニューがない場合もあるのでしっかりと覚えてください．

2.6 ファイルを開く

C-x C-f と入力するとエコーエリアに『Find file: ~/』と表示されますのでファイル名を入力すればそのファイルを新しいバッファで開くことが出来ます．

2.7 バッファを切り替える

C-x b と入力するとエコーエリアに『Switch to buffer: (default xxxxxx)』と表示されるので，xxxxxx が切り替えたいバッファの名前ならばリターンキーを押して切り替えます．
他のバッファに切り替えたい場合はバッファ名を入力してリターンを押します．バッファ名がわから

ない場合は TAB を押せば一覧が表示されます.
また，C-x C-b と入力すると直接一覧が表示されるので表示したいバッファ名のところにカーソルを移動して1 を入力するとそのバッファが開きます.

2.8 バッファを閉じる

C-x k を入力すると『Kill buffer: (default xxxxxx)』と表示され，xxxxxx で指定したバッファを閉じることが出来ます．デフォルトは現在開いているバッファです.
他のバッファを閉じたい場合にはバッファ名を入力します．バッファ名がわからない場合は TAB を押せば一覧が表示されます.

2.9 複数のファイルを編集する

Emacs はウィンドウを複数に分割して同じファイルの違う部分を同時に表示させたり，別のファイルを表示編集することが出来ます.
ウィンドウを上下二つに分けて同じファイルを編集するにはC-x 2 と入力し，元の一つのウィンドウに直すためにはC-x 1 と入力します．カーソルを他のウィンドウに移動させるにはC-x o を入力します.

2.10 テキストの入力

vi とは異なりキーボードからの入力がそのままバッファに入力されます.

2.11 テキストを削除する

カーソルの右の文字を削除するにはC-d[2]を入力し，カーソル位置から行末まで削除するにはC-k[3]を入力します．Delete やBackspace は使えないこともあるので覚えてください.

2.12 ファイルの保存

現在編集しているバッファを保存するにはC-x C-s と入力します．別の名前を付けて保存するためにはC-x C-w と入力します．エコーラインに『Write file ~/』と表示されるので保存したいファイル名を入力します.

2.13 カーソルの移動

カーソルを現在の位置から上下左右に一字分だけ移動させるには，カーソルキーを使うことが出来る場合がほとんどですが，環境によってはそれができないことがあります．そのときには，表 2.1 に示すキーを入力してカーソル移動できます．[4]．一単語だけ前後に移動させたりすることもできます.

[2] delete の略.
[3] kill の略.
[4] それぞれ，Backward，Forward， Previous，Next の頭文字になっています.

表 2.1: カーソルキーの移動

コマンド	動作
C-p	前の行
C-b	後の文字
C-f	先の文字
C-n	次の行

C-の代わりにM-を使います．また行の先頭と最後に移動するキーもあり，C-aa が先頭，C-e が行末 (end) に移動します．
さらにバッファの先頭と最後にカーソルを移動するコマンドもあり，M-<がバッファの先頭，M->がバッファの最後に移動します．

2.14 ヤンク機能 (コピー＆カット＆ペースト)

コピーするにはまずコピーしたい文字列の先頭までカーソルを移動してC-Spaceを入力します．次に文字列の最後に移動してM-w と入力すると文字列が Emacs に記憶されます．
次にペーストしたい位置までカーソルを移動してC-y と入力すれば Emacs に記憶されている文字列がペーストされます．カットしたい場合はM-w の代わりにC-w を使用します．

2.15 文字列の検索

C-s と入力すると『I-Search:』と表示されます．この検索はインクリメンタルサーチといい，検索したい単語を 1 文字入力するたびにそれに該当する文字列を検索します．目的の単語が見つかるまで文字を入力すればよいので，unitedstatesofamerica といった長い単語を検索したい場合に全てを入力する手間を省けます．また，続けてその単語を検索したい場合には，そのままC-s を押します．また，上方向検索を行う場合には，C-r を使います．

2.16 文字列の置換

ESC %と入力するとエコーエリアに『Query replace:』と表示されるので置換前の文字列を入力してリターンを押します．続いて『with:』と表示されるので置換する文字列を入力しリターンを押せば置換前の文字列にカーソルが移動し『—Query replacing xxxx with yyyy: (? for help)—』と表示されるので置換していいのならy を，置換せずに次に進むのならn を入力します．全ての置換が完了するとエコーエリアに『Done』と表示されます．

2.17 他のファイルを読み込む

現在編集しているバッファのカーソル位置に他のファイルを読み込むためにはC-x i と入力します．すると『Insert file: ~/』と表示されるので読み込みたいファイル名を入力します．

2.18 C モード

Emacs では C 言語を編集しやすくするモードがありM-x c-mode(c-mode の c-はコントロールキーではなく文字列です) で移行できます.

2.19 チュートリアルの起動方法

C-x T とタイプすると Emacs の操作方法を実践で覚えることの出来るチュートリアルファイルが開きます．mule ではC-x T のあとにJapanese と入力すれば日本語によるチュートリアルファイルが開きます.

2.20 その他

Emacs は統合環境と呼ばれるほどに非常に機能の多いエディターで Emacs Lisp アプリケーションと呼ばれるメーラー，ニュースグループリーダーなどを利用することが出来ます.

第3章　viエディタの使い方

3.1　概要

vi[1]は Unix の標準的なエディタで Windows で使われるメモ帳などのエディタとはコマンドモード，テキストモード（挿入モードとも呼ぶ）と呼ばれる二つのモード（状態）が有ることが最も異なります．コマンドモード時にはその名の通り編集に関連するコマンドを入力し，テキストモード時には実際の文章などを入力します．vi はこの二つのモードを使いこなすことによって必要最低限のキーストロークで編集を行えるように作られているので，慣れれば編集の効率を上げることが出来ます．

3.2　起動方法

vi を起動するためには次のように入力します．

vi [ファイル名]Enter [2]

ファイルが存在する場合はそのファイルを開き，存在しない場合はファイルを作成します．
例えば sample.txt というファイルを開くには

vi sample.txtEnter

と入力します．図 3.1 は何も指定せずに起動した場合の初期画面です．

3.3　コマンド モード とテキスト モード

3.3.1　コマンド モード

vi は起動時にコマンドモードになっており，このモードのときにキーボードに入力すると，入力は vi に対してのコマンド（命令）として解釈され，入力した文字や記号によって様々な操作を行なうことが出来ます．大文字小文字は区別され，似た動作をします．

3.3.2　テキスト モード

テキストモードではキーボードからの入力はそのまま画面に表示され，実際に文章など入力するのはこのモードで行います．

[1] VIsual の略です．
[2] ここで使用されている表記法の [ファイル名] というのは BNF 記法と呼ばれ簡単に説明すると [...] は省略可能で<...>は省略不可ということです．

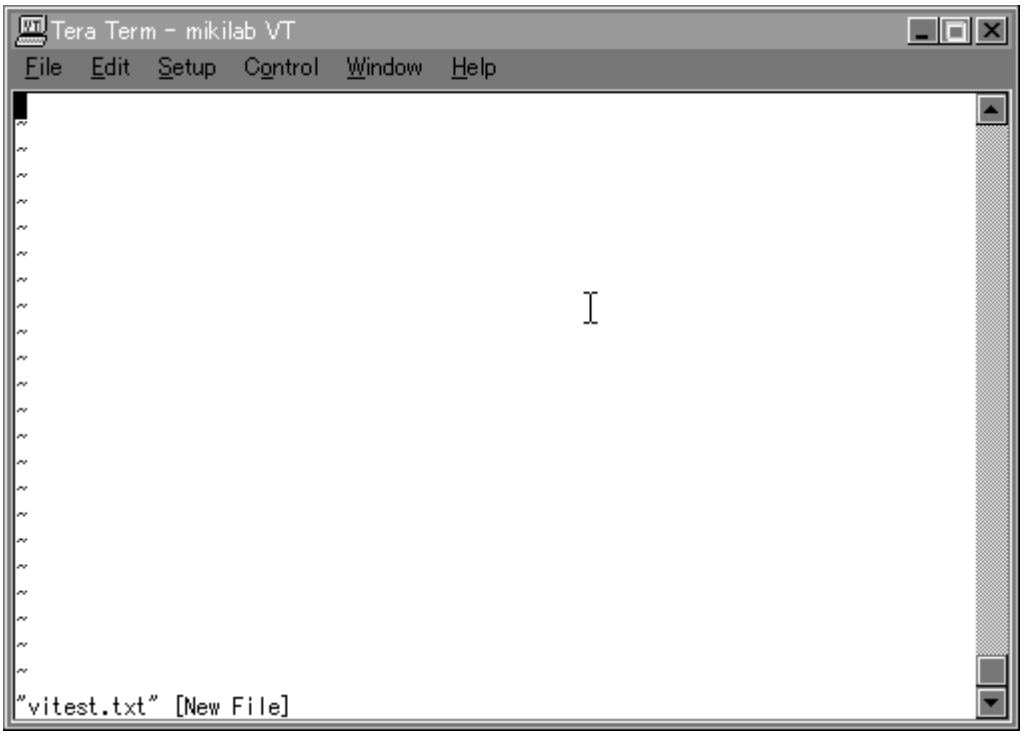

図 3.1: vi 起動時の画面

図 3.2: モードの表示

3.3.3 モードの切り替え

- **コマンドモードからテキストモードに切り替える**

  ESC を押します．既にコマンドモードのときはビープ音が鳴ります．

- **テキストモードからコマンドモードに切り替える**

  多くある中で表 3.1 に基本的なものを挙げます．

図 3.2 の様に画面下の方に-- INSERT --と出ているかどうかでモードを見分けることが出来る vi も有りますが基本的には画面上にはモードが表示されません．その場合 ESC を押してビープ音がなればコマンドモードです．
いずれにしても，ESC を押すとコマンドモードになるので必要に応じてテキストモードに切り替えてください．

表 3.1: モード切り替えコマンド

コマンド	動作
ESC	コマンドモードに切り替えます
a	カーソルの右からテキストを挿入します
A	行末からテキストを追加します
i	カーソルの左からテキストを挿入します
I	行頭からテキストを挿入します
o	下の行からテキストを追加します
O	上の行からテキストを追加します

表 3.2: 保存と終了コマンド

コマンド	動作
:w	上書き保存します
:w <ファイル名>	ファイル名をつけて保存します
:w! <ファイル名>[3]	既に存在するファイル名で保存します
:q	vi を終了します
:q!	編集したファイルを保存せずに終了します
:wq, ZZ	ファイルを保存して終了します

3.4 文字を入力する

テキストモードなら直接入力を開始し，コマンドモードならa，i などを入力してテキストモードに移行してから入力します.

abc
↓ **a**
abc　カーソル位置が右へ移動してテキストモードに移行
↓ **xyz**
abxyzc　xyz が挿入される

3.5 ファイルの保存と終了方法

編集したファイルを保存するためにはコマンドモードで表 3.2 のようなコマンドを入力します．起動時にファイル名を指定しなかった場合に保存するには名前を付けて保存してください．また，ファイルはカレントディレクトリに保存されます.
例えば，新規作成したファイルに test.txt という名前を付けて保存する場合,

```
:w test.txt
```

とします.

[3] !は強制的に何かをするコマンドに使用されます.

表 3.3: 移動系コマンド

コマンド	動作
h	左に移動します
j	下に移動します
k	上に移動します
l, space	右に移動します
0	行頭に移動します (Windows の Home)
$	行末に移動します (Windows の End)
G	ファイルの最後に移動します
:行番号	指定行に移動します

3.6 カーソルを移動する

コマンドモードでカーソルを移動するためには通常カーソルを使いますが，カーソルの使えない環境も存在します．そのような環境では，表 3.3 のコマンドを使用します[4]．

3.7 文字を削除する

テキストモードで文字を削除するためには Backspace，もしくは Delete が使用できます[5]．
コマンドモードで文字を削除するためには表 3.4 のコマンドを使用してください．

次の三行があるとします
```
first line
second line
third line
```
↓ **dd**
```
first line
third line
```
二行目が削除されます
↓ **x**
```
first line
hird line
```
一文字削除されます
↓ **dw**
```
first line
line
```
一単語削除されます

3.8 テキストを置換する

置換には 1 文字を対象とするもの，文字列を対象とするものなどいくつかのタイプの置換があり，r 以外はは入力後テキストモードになります．詳細は
表 3.5 を参照してください．削除系コマンドと同様に3s によって 3 文字列を，5cc によって 5 行を置換することができます．

[4] コマンドの前に数字を付けることにより複数回コマンドを実行したのと同じだけ移動することが出来ます．また，一般的に使用できます．
[5] 設定によって使えない場合もあります．

表 3.4: 削除系コマンド

コマンド	動作
x	カーソル位置の文字を削除します
X	カーソルの左の文字を削除します
dd	現在の行を削除します
dw	カーソル位置の単語を削除します
d<移動系コマンド>	カーソル位置から移動先までを削除します

表 3.5: 置換系コマンド

コマンド	動作
r	1 文字を置換します
R	カーソル位置から ESC を押すまで置換します
s	カーソル位置の文字列を置換します
cc	行を置換します
C	カーソル位置から行末までを置換します

```
abc
  ↓ r1   b が 1 に置換されます
a1c
```

3.9 文字列の検索

文字列を検索するには表 3.6 のコマンドを使用します．例えばカーソル位置より下の行にある section という文字列を検索するには/section と入力し，続けて次の候補を検索するにはｎを入力します．また，ファイルの終端まで検索するとファイルの先頭から検索を続けます．

3.10 コピー＆カット＆ペースト

Windows のクリップボードに相当するものとして vi には一時バッファというものが有り，ここにコピーコマンド (yy) や削除系のコマンド (dd など) で削除された文字 (列) が記憶されています．ペー

表 3.6: 検索系コマンド

コマンド	動作
/<文字列>	指定した文字列をファイルの終わりに向かって検索します (下方向検索)
?<文字列>	指定した文字列をファイルの先頭に向かって検索します (上方向検索)
n	次の文字列を検索します
Shift を押しながら n	検索方向と逆方向に文字列を検索します

表 3.7: ヤンク系コマンド

コマンド	動作
yy	現在の行をコピーします
yw	カーソル位置の単語をコピーします
dd	現在の行をカットします
dw	カーソル位置の単語をカットします
p	現在のカーソル位置に一時バッファの内容を挿入します.

表 3.8: その他便利なコマンド

コマンド	動作
e: <ファイル名>	続けて他のファイルを編集します
:r <ファイル名>	他のファイルをカーソルの下の行から追加します
:set number	行番号を表示します
Ctrl+g	現在の行番号を表示します
:set autoindent	オートインデントを有効にします. 無効にするには:set noautoindent

ストするためにはp を使います.
これらのコマンドも数字を使って3yy で 3 行を，4yw で 4 つの単語をコピーすることが出来ます. 表 3.7 に一覧をまとめます[6].

次の一行があるとします

```
left right
```

↓ **yw**　カーソル位置の単語 left をコピーします

```
left right
```

↓ **$**　行末に移動します

```
left right
```

↓ **p**　一次バッファ内の left を貼り付けます

```
left rightleft
```

↓ **yy → p**　行をコピーし，次の行に貼り付けます

```
left rightleft
left rightleft
```

3.11 その他のコマンド

以上に出ていないコマンドで覚えていると便利なものを表 3.8 に挙げます.

[6] また，これらの操作をヤンクと呼びます.