UNIX コマンドとエディタ

目 次

1 UNIX 機の利用開始と終了

この授業では，Windows が動作しているコンピュータから，UNIX 機に遠隔ログイン (remote login) して実習を行います。

UNIX では，コンピュータの利用開始の手続きを，ログイン (login) と呼びます。遠隔ログイン (remote login) とは，目の前 (**local**) のコンピュータから，ネットワークで接続されている遠隔 (**remote**) のコンピュータにログインすることです。

本校の教育用コンピュータから UNIX 機に遠隔ログインする方法については,「演習用コンピュータへのログイン手順」を参照してください。ログアウトするときには，`exit` コマンドを実行してください。

2 コマンド

2.1 コマンドの実行

UNIX が搭載されているコンピュータの操作の基本は，キーボードからコマンドを入力して実行することです。本章では，コマンド実行に関わる基本事項を学びます。

- **コマンド (command)** はコンピュータへの命令です。
- UNIX にログインして表示される

  [*user*@*host* ~]$

  を**プロンプト (prompt)** と呼びます[1]。プロンプトが存在する行を**コマンド行 (command line)** と呼びます。
- プロンプトが現れれば，キーボードからコマンドを入力できます。入力の誤りは `<BS>` (Backspace キー) で修正できます。続いて `<ENTER>`（エンター・キーまたはリターン・キー）を押せば，入力したコマンドが実行されます。

  例) date コマンドで現在の日付と時刻を表示する

  ```
  date<ENTER>
  ```
- UNIX のコマンドにおける大文字と小文字は，異なるものとして区別されます[2]。
- ほとんどのコマンドには，**引数 (ひきすう; argument)** を使って，コマンドの動作に関わる指示を与えることができます。引数が必須のコマンドもあります。

  引数は，コマンド行に入力したコマンドの名前に続いて，空白で区切って記述します。

  引数として記述すべき事柄はコマンド毎に異なります。

  例) 引数 `+%D` を付けて date コマンドを実行する

  ```
  date +%D<ENTER>
  ```

[1] プロンプトの表示形式は設定や状況によって変わります。利用者が設定を変更することも可能です。
[2] Windows のコマンドには，大文字・小文字の区別がありません。

練習

1. `cal` コマンドを実行してください。今月のカレンダーが表示されましたか。
2. 大文字で `CAL` と入力して `<ENTER>` を押してください。Shift キーを押しながら英字を入力すると大文字になります。

   「コマンドが見つかりません (Command not found.)」としかられますね。UNIX に cal コマンドはありますが，CAL コマンドはありません。
3. `cal 10 1966` を実行してください。10 や 1966 は，cal コマンドの引数です。

   cal コマンドは，引数が無ければ今月の，引数があれば指定の月あるいは年のカレンダーを表示します。
4. 途中に空白を入れずに `cal101966` を実行してください。

   cal101966 というコマンドはありません。コマンドと引数，および複数の引数を区切るには空白が必要です。
5. `man date` を実行してください。man はコマンド等のマニュアルを表示するコマンドです。

   表示が一画面におさまりきらないため，最終行に何やら表示されて止まっています。`<SPACE>` (スペース・キー) を押すと次の画面に進みます。`q` を打つと終了します。
6. `man` を実行してください。

   man コマンドには，調べたいコマンドの名前（コマンド名）等を，引数として与える必要があります。引数が必要かどうかは，コマンドによって異なります。
7. `clear` コマンドを実行してみましょう。

   きれいになりましたか。

2.2 困ったときには

- キーボードから入力できない。

  CTRL-q を押してみてください。CTRL-q は Ctrl キーを押しながら q キーを押すことを意味します。
- プロンプトが出ない。

  CTRL-d を押してみてください。それでもだめなら CTRL-c を押してみてください。CTRL-c は現在実行しているコマンドを強制終了します。
- 英字のキーを押すと大文字で入力される。Shift キーと一緒に押せば小文字になる。

  キーボードの Caps Lock ランプが点灯していれば，大文字入力になります。Caps Lock キー，または Shift キーと同時に Caps Lock キーを押してください。

- exit コマンドでログアウトしようとすると「中断した job が残っています」と表示されてログアウトできない。

  fg とタイプしてエンターを押すと画面にアプリケーションが現れます。それを正しい方法で終了させてからログアウトを試してください。

2.3 ファイル

コンピュータで作業したことを残しておきたいときには**ファイル (file)** を利用します。ファイルを作るには，その中身に加えて，ファイルの名前（**ファイル名**）が必要です。ファイルはファイル名で区別され，ファイル名を使ってファイルの中身を見たり，中身を変更できます。

ファイルの作成や利用の方法はいろいろとありますが，この章では，コマンド行の操作だけでファイルを扱います。

2.3.1 ファイルを作って見てみよう — ls, cat, >

- ls コマンドは所持するファイルの名前 (ファイル名) を表示するコマンドです。
- 「`cat` ファイル名」を実行すれば，ファイルの内容が表示されます[3]。
- コマンド [と引数] の後ろに「`>` ファイル名」を指定すると，コマンドの実行結果は画面に表示されません。その代わりに，指定したファイル名の新しいファイルができ上がり，そこに結果が書き込まれます。

では，実際に試してみましょう。

1. `ls` コマンドを実行してみましょう。

   まだファイルを作っていないので，何も表示されません。

2. `date` コマンドを実行してみましょう。

   現在の時刻が表示されましたね。

3. `date > now` を実行しましょう。

   date コマンドの実行結果は画面に表示されません。その代わりに，now という名前のファイルに書き込まれました。

4. 再度，`ls` を実行してみましょう。

   now という名前のファイルがありますね。

5. `cat now` を実行して，now の中身を見てみましょう。

[3] cat コマンドで内容を表示できるのは，テキストファイルと呼ばれるファイルのみです。

練習

1. `now cat` を試してみましょう。

   しかられましたね。一般に，プロンプトに続いては，ファイル名 (now) ではなく，コマンド名 (cat) を打つ必要があります[4]。

2. `cat dog` を実行してみましょう。

   cat コマンドの引数には，既存のファイルの名前を与えなければいけません。

3. 引数を与えずに `cat` を実行してみましょう。

   プロンプトが現れませんね。第 2.2 章を読んで，正しい対処をしてください。

2.3.2 ファイル名の付け方

- UNIX では / をファイル名に含めることはできません。
- 小文字と大文字は区別されます[5]。特に理由がなければ，小文字のファイル名を使うのが慣例です。
- UNIX では，.（ドット; ピリオド）で始まる名前のファイルは，多くのコマンドで特別に扱われます。

当面，英数字と . (ドット)，_ (アンダーバー) のみを組み合わせたファイル名の利用を推奨します。ただし，ドットで始まるファイル名は避けるのが無難です。

ファイルはファイル名で識別して操作するものですから，ファイル名はファイルの内容を連想できるものにしましょう。ドットやアンダーバーはファイル名を構成する単語を区切るために使われます。

練習

1. `date > a/b` を実行してみましょう。

   エラーメッセージが表示されましたね。エラーの意味は，ディレクトリの学習をすればわかります。

2. `ls` を実行しましょう。続いて `date > Now` を実行して，`ls` を実行しましょう。

   now と Now は区別されていますか。

3. now と Now の内容を見比べてください。

4. `cal` を実行してください。次に `cal > this_month` を実行してから，ファイルができたか，確認してください。

[4] 例外はありますが，細かなことは無視しましょう。

[5] Windows のファイルシステムでは，ファイル名の大文字・小文字は区別されません。そのため，大文字・小文字のみが異なる名前のファイルを，同じフォルダに作ることはできません。

2.3.3 ファイル名の変更とファイルの複写・削除 — mv, cp, rm

コマンドを使って，作成したファイルの名前を変更したり，ファイルの内容を別のファイルに複写したり，ファイルを削除することができます。そのためのコマンドは mv, cp, rm です。ファイル名を引数として次の形式で使います。

mv *file1 file2* — *file1* の名前 (ファイル名) を *file2* に変更

cp *file1 file2* — *file1* の内容を *file2* に複写 (*file1* と同じ内容のファイル *file2* を作る)

rm *file* — *file* を削除

注意

- 削除したファイルは二度と戻りません。rm コマンドを実行するときには心を落ち着かせてから`<ENTER>` を押すことを習慣づけましょう[6]。
- cp や mv の最後の引数 (上で *file2* と表記したもの) に既存のファイルの名前を指定すると，元のファイルは上書きされて無くなります[7]。このことは「コマンド `>` ファイル名」でファイルにコマンドの実行結果を書き込むときも同様です[8]。誤って必要なファイルを消さないように，既存のファイルをよく確認してから実行しましょう。

練習

1. 再び `ls` を実行してから `cat now` を実行しましょう。
2. ファイル now の名前を old に変更します。`mv now old` を実行してください。名前が変わったことを ls コマンドで確認してください。cat コマンドでファイルの中身も見てみましょう。
3. old と同じ内容のファイル old.copy を作ります。`cp old old.copy` を実行してください。ファイルができたか ls で確認してから，old と old.copy の内容を見比べてみましょう。
4. old.copy を削除します。`rm old.copy` を実行してください。うまくいったか確認してください。
5. 続いて old と this_month を削除してください。結果を確認してください。

2.3.4 コマンドのオプション

多くのコマンドには，**オプション (option)** と呼ばれる，コマンド実行に必須ではない引数を指定できます。オプションは，コマンド動作の詳細を指定するもので，通常，`-` か `--` に続いて指定します。各コマンドのオプションは man コマンドを使って調べることができます。

例) `ls -a`

ls のオプション -a は . (ドット) で始まるものを含めて，すべてのファイルの名前を表示するためのものです。

[6] rm 実行時，本当にファイルを削除してよいか，確認のメッセージを出すように設定しているシステムもあります。
[7] cp や mv の実行時に，既存のファイルを上書きしてよいかどうか，確認のメッセージを出すように設定されているシステムもあります。
[8] 設定によっては上書きされないこともあります。

注意) ここで表示される . (ドット) で始まる名前のファイルは，コンピュータを正常に使うために必要なファイルです。削除したり，名前を変えたりしないでください。

練習

1. `cal > month` を実行してから，`cat month` を実行してください。

   ファイル month の内容が表示されましたね。

2. `cat -n month` を実行してみましょう。

   cat は -n というオプションを受け付けます。-n は行番号を付して表示するために使います。

3. `ls` を実行してから，`ls -l`（エル）を実行してみましょう。

   ls コマンドのオプション -l は，ファイルのいろいろな情報を表示する指定です。ここでは，ファイルの最終変更日時がわかることにだけ，注目することにしましょう。

4. `ls -a -l` と `ls -al` の実行結果を比較してください。

3 エディタ GNU Emacs を使おう

UNIX 系の OS で利用可能なエディタは vi を始めとして，いろいろとあります。その中でも，vi と並んで代表的なエディタとして GNU Emacs があります。本章では，この GNU Emacs の使い方の基本を紹介します[9]。

3.1 キー表記法

GNU Emacs の説明文書では，次のキー表記を使うのが一般的です。

`C-文字` コントロールキー (`<CTRL>`) を押したまま，文字を押します。例えば、`C-f` はコントロールキーを押したままで f のキーを押すことです。

`M-文字` メタキー (普通は `<ALT>` がメタキーです) を押したまま，文字を押します。または，`<ESC>` を押して離してから文字を押します。

3.2 起動と終了

3.2.1 起動方法

GNU Emacs (以下，単に Emacs という) を起動するためのコマンドは `emacs` です。X Window System が動作している環境では，コマンド行オプションの有無によって，二通りの動作形態があります。

[9] GNU Emacs は UNIX の標準コマンドではありませんので，必ずしもすべての UNIX で利用できるとは限りません。

1. `emacs` (オプション無しで起動) : 起動時に新規ウィンドウを生成して動作する。

   Windows のアプリケーションを操作するのと同様の感覚で，マウスを使って Emacs を操作できます。

2. `emacs -nw` または `emacs --no-windows` : コマンドを入力したウィンドウ内で動作する。

   X Window System を用いずに Emacs の操作をすべてキーボードで行うことになります。

X が動作していなければ，いずれの起動法を使っても，Emacs はコマンドを入力したウィンドウ内で動作します。ただし，この資料は X が動作していることを前提として記述しています。

これから，Emacs の使い方を学びますが，できるだけマウスを使わないで操作する方法までを習得することが望ましいです。それは，マウス操作よりもキーボード操作の方が作業効率が良いことが多いし，マウスはいつも使えるとは限らないからです。

3.2.2 終了方法

マウスを使って Emacs を終了するには，メニューから

```
File -> Quit
```

を選択します。

キーボード操作による場合は

```
C-x C-c
```

です。

練習

1. コマンド行に `emacs` と打って Emacs を起動しましょう。起動元のウィンドウに新しいプロンプトが現れないことと，Emacs の起動画面を観察したら終了しましょう。
2. コマンド行に `emacs -nw` と打って Emacs を起動しましょう。起動画面を観察したら，キーボード操作で終了しましょう。

3.3 画面構成

コマンド行に `emacs &` と打って，Emacs を起動してください。新しいウィンドウに図 1 のような起動画面が表示されましたね。& をつけて Emacs を起動したので，起動元のウィンドウに新しいプロンプトが現れていることも確認してください。

- 最上行の File Edit Options ... 等と表示されている部分をメニューバー (menu bar) と呼びます。ここをマウスでクリックして，メニューから Emacs の操作を選択することができます[10]。すぐ下の行には，マウス操作のためのボタンが並んでいます。

[10] すべての操作がメニューでできる訳ではありません。

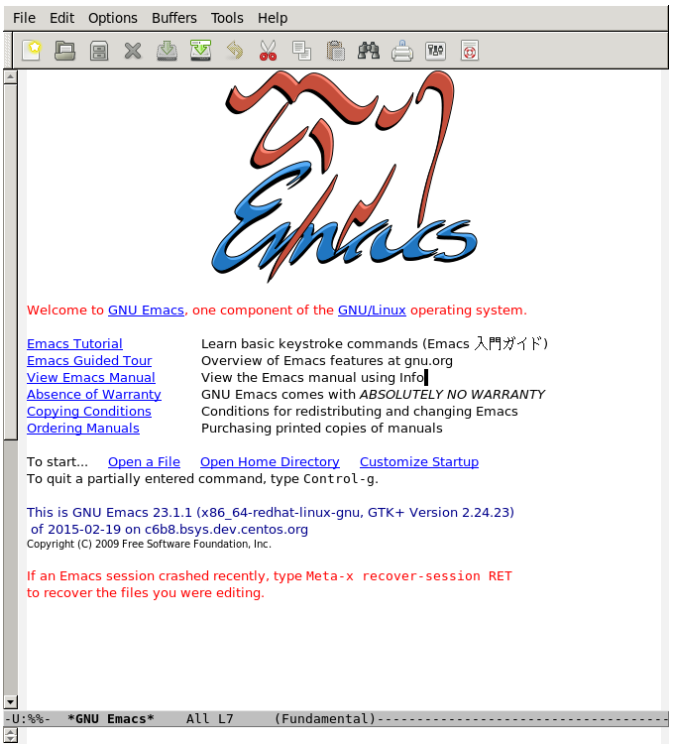

図 1: Emacs の起動画面例

- 最下行の For information about ... と表示されている行をエコー領域 (echo area) と呼びます。ここに Emacs からの様々なメッセージが表示されます。

  この行に，Emacs で編集したいファイル名等を入力することもあります。この行は，キーボードからの入力を受け付ける状態になったとき，ミニバッファ (minibuffer) と呼ばれます。

- 残る部分を (Emacs における) ウィンドウと呼びます。図 1 では，様々な文字が表示されている一番広い場所がウィンドウです。ここで文書の編集等を行います。

  なお，X Window System における “ウィンドウ” (これまでウィンドウと呼んできたもの) のことを Emacs ではフレーム (frame) と呼びます。

- ウィンドウの最下行 (エコー領域を含めて数えると下から 2 行目) をモード行 (mode line) といいます。ここにはウィンドウに関する様々な情報が表示されます。

ここで少し Emacs を操作してみましょう。

1. キーボードから何か文字を打ってください。

   Emacs のウィンドウに文字を入力できましたか。

2. マウスを使い，メニューバーの File の項目から Split Window を選択してください。

   ウィンドウが分割 (split) されて二つになります。Emacs では複数のウィンドウを使ってファイルを編集できます。

3. 同じく File メニューの Remove Splits を選択してください。

   予想どおりになりましたか。

次に，ウィンドウ分割をキーボード操作で行ってみます。

1. まず，File メニューをクリックしてプルダウンメニューを表示し，メニュー項目をよく観察してみましょう。

   項目の右側に，括弧で囲まれた記号があります。この記号は，当該操作をキーボードで行うためのキーを表しています。

2. Split Window の項目の右には (C-x 2) とあるのを確認したら，この表示に従って，キーボードから `C-x 2` をタイプしてください。`C-x 2` は，コントロールキーを押しながら x を押し，続いて (空白は打たずに) 2 だけを押すことを意味します。

3. キーボード操作でウィンドウを一つにしましょう。File メニューをマウスで開き，キー操作による方法を確認したら，実行してください。

3.4 困ったときには

3.4.1 操作の中断

Emacs を使っていて良く分からない状況に陥ったときのために，次の Emacs のコマンド (キー操作) を是非覚えておいてください。

Emacs におけるコマンドの取り消し: `C-g`

1 回押しただけでうまくいかなければ 2 回押します。

たとえば，キー操作でウィンドウを二つに分割しようと思ったけど，途中で止めたくなったとします。ウィンドウ分割のためにまず `C-x` を押しますが[11]，続いて `2` を押す代りに `C-g` を押すと，その操作を取り消すことができます。

3.4.2 分割された Window を一つにする

Emacs では，操作の途中で自動的にウィンドウが二つになることがあります。その場合には，ウィンドウ内に表示された指示に従って対処するか，よく分からなければウィンドウを一つにする操作 (`C-x 1` やメニューの Remove Split など) をしてください。

3.5 Emacs におけるファイル編集の基本操作

3.5.1 例題 1 — ファイルの新規作成

ファイルの中身が

```
ls - list directory contents
mv - move files
```

[11] この状態で少し待って，エコー領域に入力したキー (C-x-) が表示されることを確認しておきましょう。

である commands_file という名前のファイルを Emacs を使って作成します。以下の操作を行ってください。

1. Emacs が起動していなければ `emacs &` で起動してください。
2. ファイルを新しく作成するには，`C-x C-f` を押します。このコマンドが第 B 章の Emacs コマンド一覧のどこに載っているかを確認しておきましょう。

   最下行のエコー領域に Find file: `~/` というメッセージが現れます。カーソルもエコー領域に移動しましたので，ここにキーボードから文字を入力できます。エコー領域は，文字入力できる状態になったとき，ミニバッファと呼ばれるのでしたね。。
3. 作成するファイルの名前 `commands_file` をミニバッファに入力して `<ENTER>` を押しましょう。入力を誤ったら `<ENTER>` を押す前に `<BS>` を使って修正するか，`C-g` を押して 2. からやり直してください。

   ウィンドウ下部のモード行には，これから作成するファイルの名前 commands_file が表示されます。カーソルは上部のウィンドウに戻ります。
4. 作成するファイル commands_file の中身

   ```
   ls - list directory contents
   mv - move files
   ```

   を Emacs のウィンドウに入力してください。

   改行するには `<ENTER>` を押します。文字の修正には `<BS>` が使えます。カーソルの移動には，ここでは矢印のキーを使うことにします。
5. Emacs に入力したテキスト (文書) をファイルに保存するには `C-x C-s` を押します。

   エコー領域 (最下行) に Wrote ... というメッセージが表示され，保存に成功したことがわかります。これでファイル commands_file ができあがりました。
6. 一度 Emacs を終了し，ファイル commands_file の内容を cat コマンドで確認してください。

3.5.2 例題 2 — 既存ファイルの編集

第 3.5.1 節の例題 1 で作成したファイル commands_file の内容を

```
ls - list directory contents
mv - move files
cat - print files
```

に変更します。また，Emacs でのファイル編集作業と併行して UNIX のコマンドも使ってみます。

1. `emacs &` と打って Emacs を起動してください。
2. ファイル commands_file が存在することを， emacs コマンドを入力したウィンドウで ls コマンドを実行して確認してください。ファイル名が違っていたら，正しいものに変更してください。

3. 編集したいファイル commands_file を開きます。方法は第 3.5.1 節 例題 1 でファイルを新規作成したときと同じです。`C-x C-f` を押してください。最下行のミニバッファにはファイル名 commands_file を入力して `<ENTER>` を押してください。

   既存ファイル commands_file の内容が Emacs のウィンドウに表示され (Emacs に読み込まれ)，モード行 (下から 2 行目) には commands_file と表示されます。

4. ファイルに追加すべき内容 (`cat - print files`) を Emacs のウィンドウに入力しましょう。

5. 例題 1 と同じ方法で，Emacs のウィンドウに表示されている内容をファイルに保存してください。Emacs はまだ終了しないでください。

6. ファイル commands_file の内容を cat コマンドで確認しておきましょう。

3.6 ファイルとバッファ

図 2 は，コンピュータの構造を，これまでに行ったことを理解するために必要な部分に限って，ごく簡単に表現したものです。

CPU (central processing unit) はコンピュータの頭脳にあたる役割をし，コンピュータの動作を制御したり，様々な計算を行います。

主記憶装置 (main memory; 以下，単にメモリと呼ぶ) は，実行中のプログラムやデータを格納する場所です。CPU は，処理に必要なデータをメモリから読み取り，処理結果をメモリに書き込みます。

Emacs が動いているときには，Emacs のプログラムや編集中のテキスト (文書) はメモリにあります。Emacs がテキストを保持するために使うメモリ内の領域を，Emacs の用語で**バッファ** (buffer) といいます。

メモリに対するデータの読み書きは高速に行えますが，メモリの容量は比較的小さく，メモリ内のデータはコンピュータを停止すると消えてしまいます。そのため，メモリはデータの永続的な保存には使えません。

補助記憶装置 はメモリに比べて動作が遅い反面，記憶容量が大きく，中身はコンピュータを停止しても消えません。**ファイル**は補助記憶装置にあります。

さて，Emacs のウィンドウに表示されるテキストは，ファイルではなく，バッファの内容です。したがって，Emacs でファイルを作成・編集するには，まず，そのためのバッファを用意する必要があります。これを行うのが `C-x C-f` です。この操作ではファイル名を入力しますが，その名のファイルが存在しなければ，同名の空のバッファが用意されます (第 3.5.1 節 ファイルの新規作成)。ファイルが存在すれば，ファイルは同名のバッファに読み込まれます (第 3.5.2 節 既存ファイルの編集)。なお，バッファに読み込まれるのはファイルの内容のコピーですから，ファイルを読み込んでも元のファイルが無くなる訳ではありません。

Emacs を終了するとバッファは消えるので，保存したいバッファの内容はファイルに入れる必要があります。それをするのが `C-x C-s` です。

編集中のバッファが保存済かどうかは，モード行 (下から 2 行目) におけるバッファ名の左側の表示でわかります。ここが `--` から `**` に変わったら，バッファとファイルの内容に違いが生じたこと，すなわち，バッファの内容を変更したのに未保存であることを表しています。

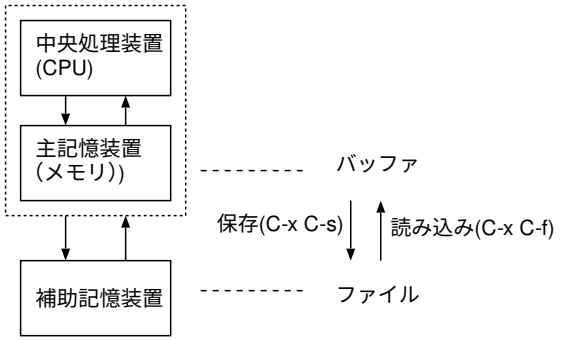

図 2: ファイルとバッファ

では，上記のことを確認するために，以下の操作を行いましょう。

1. Emacs が起動していなければ，`emacs &` と打って起動してください。commands_file をバッファに読み込んでいなければ，読み込む操作をしてください。

2. Emacs のウィンドウに表示されている，commands_file という名のバッファを消してみます。メニューから

   ```
   File -> Close (current buffer)
   ```

   を選んでください。

3. ls コマンドを使って，消したバッファと同じ名前のファイルが存在することを確認してください。

   バッファを消してもファイルは消えませんね。

4. Emacs を終了せずに，引き続き commands_file を編集するにはどうしたらいいでしょう?

   ファイル commands_file を再度バッファに読み込んでください。

5. Emacs のモード行 (下から 2 行目) に表示されているバッファ名 commands_file のすぐ左側の表示が -- であることを確認してから，バッファ (Emacs のウィンドウ) 3 行目の cat の説明を

   ```
   cat - concatenate and print files
   ```

   に変更してください。

   モード行の表示が ** に変わりましたね。

6. UNIX のコマンドでファイル commands_file の内容を確認してください。

   バッファの内容を変更しただけでは，ファイルは変わりません。

7. バッファの内容をファイルに保存してください。

   モード行の表示が -- になりましたね。バッファとファイルの内容が同じになりました。

8. ファイル commands_file の中身を cat コマンドで確認してください。

3.7 まとめと練習

3.7.1 まとめ—Emacs によるファイル編集の流れ

Emacs を用いてファイルを編集するときの流れは次のとおりです。

1. Emacs を起動する。— `emacs &` (X ウィンドウシステムが使えないときには & を付けない)
2. 新規ファイル用のバッファを用意する。または，編集したいファイルをバッファに読み込む。— `C-x C-f`
3. バッファへの文字の追加・削除・変更などの編集作業を行う。
4. バッファの内容をファイルに保存する。— `C-x C-s`
5. 必要に応じて 2 から 4 の作業を繰り返す。複数の文書を編集するときでも Emacs を終了して再度起動する必要はない。
6. すべての作業が終わったら Emacs を終了する。 — `C-x C-c`

3.7.2 練習

1. 次の内容を持つファイル emacs_motion を Emacs で作成してください。

```
           backward forward
character  C-b      C-f
line       C-p      C-n
```

   ファイル作成後も Emacs を終了しないでください。

2. 起動中の Emacs を使って，既存のファイル commands_file に，これまでと同じ形式で，cp と rm の説明

```
cp - copy files and directories
rm - remove files or directories
```

   を追加してください。作業を終えたら Emacs を終了してください。

3.8 その他の操作

3.8.1 ファイル名を指定してファイルに保存

編集中のバッファの内容を，ファイル名を指定して保存するには `C-x C-w` を使います。既存のファイルをバッファに読み込んでからこのコマンドを使うと，元のファイルを異なるファイル名で保存することができますので，UNIX の cp コマンドでファイルを複写するのと同様のこともできます。

以下では，ファイル commands_file と同じ内容を持つファイル commands_copy を作ります。

1. ファイル commands_file をバッファに読み込んでください。

2. C-x C-w を押しましょう。

   エコー領域 (ミニバッファ) にメッセージ Write file: ~/ が現われます。

3. commands_copy と打って，<ENTER>を押しましょう。

この操作で新しいファイル commands_copy が作成されましたが，モード行に表示されているバッファ名も commands_copy に変ったことに注意してください。C-x C-w を実行すると，編集中のバッファの内容が別のファイルに書き込まれるだけではなく，バッファ自体の名前も変わります。引き続きこのバッファで編集作業をすれば，それは元の commands_file ではなく commands_copy に対してなされます。

3.8.2 Emacs 終了時のファイル保存

バッファの内容を変更したにもかかわらず，それをファイルに保存しないまま Emacs を終了しようとすると，バッファの内容をファイルに保存するかどうかを尋ねるメッセージが，ウィンドウ下部のエコー領域に表示されます[12]。その場合には，保存の必要性を判断して，y, n, yes, no 等で答えてください。

[12]Emacs には，ファイル編集以外の用途に用いられるバッファ (例えば *scratch*) が存在します。 それらの内容を変更しても，終了時に保存のメッセージは現れません。

A UNIX コマンド

分類	コマンド	機能
オンラインマニュアル	`man` *command*	*command* のマニュアルを表示
ディレクトリ操作	`ls`	カレント・ディレクトリに存在するファイルの名前を表示
	`cd` *directory*	カレント・ディレクトリを *directory* に変更
	`pwd`	カレント・ディレクトリ名の表示
	`mkdir` *directory*	*directory* の作成
	`rmdir` *directory*	*directory* の削除
	`mv` *directory1 directory2*	*directory1* の名前を *directory2* に変更
ファイル操作	`cp` *file1 file2*	*file1* を *file2* に複写
	`cp` *files directory*	*files* を *directory* に複写
	`mv` *files directory*	*files* を *directory* に移動
	`mv` *file1 file2*	*file1* の名前を *file2* に変更
	`rm` *files*	*files* を削除
その他	`cat` *files*	*files* の内容を表示
	`more` *files*	*files* の内容を一画面毎に表示
	`file` *files*	*files* の種類を表示
	`diff` *file1 file2*	*file1* と *file2* の違いを表示

B GNU Emacs の主要コマンド

`C-x C-c`	終了
`C-g`	コマンドの取り消し
`C-x u`	変更の取り消し (undo)
`C-_`	変更の取り消し (undo)
`M-x help <ENTER>`	ヘルプ
`M-x help <ENTER> t`	チュートリアル
`C-x C-f`	ファイルを開く (バッファへのファイル読込み)
`C-x C-s`	現在のバッファをファイルに保存
`C-x C-w`	現在のバッファに名前をつけて保存
`C-x s`	編集中のバッファをすべてファイルに保存
`C-d`	カーソル位置の 1 文字を削除
`C-k`	行末まで消去 (kill)
`C-y`	最後に保存した kill-ring の内容取りだし (yank)
`C-s`	検索
`M-%`	置換
`C-a`	カーソルを行頭に移動
`C-e`	カーソルを行末に移動
`C-v`	次の画面を見る
`M-v`	前の画面を見る