1 はじめてのお使い

ログイン直後やGUI環境で「コマンド入力」をクリックした直後では, プロンプトが画面に出力され, 指示を待っている状態になっています。ここでコンピュータに対してキーボードをタイプして指示します。指示のことをコマンドといいます。

コンピュータの中には, 多数の文書やフォルダが格納されています。どのように格納されているのかを調べるコマンドの学習を始めましょう。Windowsの場合「フォルダ」といいますが, Linuxの世界では「ディレクトリ」といいます。これからディレクトリといわれたらWindowsの「フォルダ」のことと頭の中で読みかえてください。

実習 1.1　ここはどこ?(pwd)

1. 「コマンド入力」をクリックしなさい。
2. pwd を入力しなさい。出力をメモしなさい。
3. su - を入力し, 一旦管理者になりなさい。
   pwd を入力し, 出力をメモしなさい。
4. 管理者になるのは必要最小限にとどめるべきなので exit を 入力して, 一般ユーザに戻りなさい。一般ユーザに戻れたかどうかはプロンプトで確認できます。
5. もう一度, pwd を 入力して出力を確認しなさい。

pwd は現在位置を出力するコマンドです. Present Working Directory の略です。

一般ユーザでログインした直後と, 管理者(の名前は root ですが)でログインした直後では, 存在している位置が異なっていることがわかります。

実習 1.2　お出かけ(cd)

1. cd ..を入力しなさい。
2. 今, どこにいるのかを調べなさい。調べた結果をメモしなさい。
3. もう一度 cd .. を入力しなさい。
4. 今, どこにいるのかを調べなさい。調べた結果をメモしなさい。
5. 今度は cd /etc/rc.d を入力しなさい。今, どこにいるのかを調べなさい。調べた結果をメモしなさい。
6. もう一度 cd .. を入力しなさい。現在位置を調べるのですが, どこにいるのかを予測した上で現在位置を調べなさい。
7. 最後に cd だけ入力しなさい。現在位置を調べなさい。

実習を行ってみるとコンピュータ内部の住所は / で区切られた名前で指定されているようです。

cd コマンドは, 現在位置を変更する場合に用います。Change Directory の省略形です。

引数を与えた場合

与えた引数の場所へ移動します.

..は特殊な意味を持ち, 今いる位置より一つ上の場所を指示しています。

引数を省略した場合

お家に戻ります。

pwd にしろ cd にしろ, Directory の省略形が入っています。コンピュータ内部の住所は / で区切られたディレクトリの名前で指定されます。

特に, ログイン直後には, ユーザ毎に予め決められた, 位置にいます。ユーザ毎に決められた位置のことを「ホームディレクトリ」といいます。

ディレクトリはWindowsのフォルダと同じように, そこに複数の文書やフォルダ自身を格納することができます。

それでは、お家にどのようなものがあるか、調べてみましょう。(ホームディレクトリにどのような文書やデータがあるかを調べてみようということです)

実習 1.3　持ち物探し, 宝探し(ls)

1. `ls` を入力しなさい。出力に見覚えはありませんか?どのような出力になったのかを記録しなさい
2. `ls - a` を入力しなさい。新しい項目が出力されますが、新しく出力されたものに共通する点は何でしょう
3. `ls - l` を入力しなさい。
4. `ls -tl` を入力しなさい。前の出力との違いはどういう風に違うといえますか。
5. `ls -tlr` を入力しなさい。前 2 回との違いはどういう風にいえばよいですか?
6. `ls -tlra` を入力しなさい。どうしてそのような出力になったのか, 出力の様子を説明できますか?
7. `ls /etc/rc.d` を入力しなさい。何が表示されていますか?

`ls` は List の省略形です。 `cd` と似ている部分と, 似ていない部分があります。

似ていない部分は, `ls` の後ろにハイフン(マイナス)に続けて指定を行うことが可能です。ハイフンに続けて指定するものを「オプション」といいます。

オプションまで含めた部分をコマンドと解釈すれば, cd と似ている部分は以下のようになります。

引数を省略した場合

現在位置での持ち物をリストアップします。

引数を与えた場合

引数で指定した位置での持ち物をリストアップします。`cd` とは違って位置の移動は行われません。

`ls` は指定した位置での持ち物をリストアップします。ここで持ち物とは, 文書そのもののような「ファイル」とファイルをいれることを目的とした「ディレクトリ」です。ls の出力の直後にある, “/”, “*”, “@”などはファイルやディレクトリの名前ではありません。「ファイル」「ディレクトリ」をまとめてファイルといったりしますので, 頭を整理して聞いてください。

コマンドのオプションは “-” に続けて指定ます。複数のオプションを指定する場合

```
ls -tlra
ls -t -l -r -a
```

のどちらで書いても構いません(場合が多い。このあたりはコマンド毎に異なります)

`ls` コマンドでよく利用されるオプションを以下にまとめます。

オプション	オプションの意味
`l`	ファイルやディレクトリの名前だけではなく, 詳細な情報を表示する(long format の略)
`a`	ピリオドで始まるファイルやディレクトリは「隠しファイル」として利用される。すべてのファイルを表示する場合に用いる(all の略)
`t`	ファイルの出力順をアルファベット順ではなく, 時間順に表示する(time の略)
`r`	ファイルの出力順を逆順にする(reverse の略)
`R`	指定した位置より下のすべてのファイルを出力する(Recursive 再帰的の略)
`d`	ディレクトリの中を表示しない

実習 1.4　あちこち探検(ls の練習問題)

1. ホームディレクトリの一つ上にどのようなファイル(ディレクトリを含む)があるかを調べなさい。その際, 現在位置を移動しないで調べる方法を考えなさい.
2. ホームディレクトリの二つ上の場所にどのようなファイルがあるかを調べなさい。その際, 現在位置を移動しないで調べなさい。
   (ヒント: 二つ上の場所は, 一つ上の場所のもう一つ上の場所です。ディレクトリの区切りを示す記号は“`/`”です)
3. 位置を移動しないで`/usr/bin` の下にあるファイルを調べてみなさい。
4. 「隠しディレクトリ」`.config` の下にはどのようなファイルがあるかすべて書き出してみなさい。ディレクトリを別にすれば全部で 20 ヶのファイルがあるはずです。
   1. ディレクトリ`.config` へ移動しなさい。
   2. ディレクトリ`.config` 内にどのようなファイル/ディレクトリがあるのか調べなさい.
   3. もしディレクトリがあればその中も調べなさい。
5. すべて探し終えたなら, ホームディレクトリに戻りなさい.

「はじめてのお使い」はこれで終りです。

2 探し物はなんですか?

先に進む前に “ls -la”の出力の説明をしておきましょう。

```
合計 13026
drwxr-xr-x 26 otofuji users    1384 2005-05-13 17:50 ./
drwxr-xr-x  5 root    root       96 2005-03-15 16:49 ../
drwxr-xr-x  5 otofuji users     440 2005-05-13 17:46 .OOo/
-rw-r--r--  1 otofuji users     229 2005-02-25 13:04 .Xmodmap
-rw-------  1 otofuji users    5524 2005-05-13 17:49 .bash_history
drwx------  4 otofuji users      96 2005-03-18 17:40 .cache/
-rw-r--r--  1 otofuji users    4526 2005-02-23 19:17 .canna
       省略
-rw-r--r--  2 otofuji users    8014 2005-05-10 21:39 How2Use.sxw
-rw-r--r--  2 otofuji users 5691206 2005-04-06 19:55 LinuxPersonalWorkstation.pdf
-rw-r--r--  2 otofuji users  156548 2005-05-12 05:25 ja-JP.xpi
       省略
-------------------------------------------------------------------------------
      ①          ②       ③       ④             ⑤—⑥          ⑦
```

一行に一つのファイルあるいはディレクトリの情報が出力されています。それぞれの列の意味を表にまとめると次のようになります。

列	意味
①	1 文字目: 普通のファイル(-)かディレクトリ(d)かを示す。 2-4 文字目: 所有者に対する許可権, 左から順に読みだし可能(r). 書き込み可能(w), 実行可能(x)を表す。不可能な場合 - となる 5-7 文字目: グループに対する許可権 8-10 文字目: すべてのユーザに対する許可権
②	ファイル/ディレクトリの所有者
③	ファイル/ディレクトリの所有グループ
④	ファイル/ディレクトリの大きさ。英字換算での文字数を表す。
⑤⑥	ファイル/ディレクトリが作成された日時, あるいは最終的に変更された日時
⑦	ファイル/ディレクトリの名前。先頭がピリオドで始まるファイル/ディレクトリは「隠し属性」を持つので, 表示させる場合には-a オプションが必要。

..は一つ上のディレクトリを示すことを説明しましたが, “.”は現在位置を示します。

ls の動作をまとめておくと以下のようになります。

- ls のみ
  ls .の動作と同じ。現在位置にあるファイルやディレクトリの名前を表示する。「隠し属性」を持っている
  ファイルやディレクトリを表示させるには-a オプションが必要。
- ls ファイル名
  ファイル名がディレクトリでない場合, ファイル名を持つファイルを表示する
- ls ディレクトリ名
  ディレクトリ内部のファイル一覧を表示する。
- ls -d ファイル/ディレクトリ名
  ファイル/ディレクトリの名前を表示する。ディレクトリを指定しても, 名前だけを表示する。

実習 2.1 見付けにくい物ですか(ワイルドカード)

1. ディレクトリ.config へ移動しなさい。
2. 移動できたかどうかは「ここはどこ?」で調べることができます。
3. さらにその中のディレクトリ xfce4 へ移動しなさい。
4. さらにその中のディレクトリ mcs_settings へ移動しなさい。

5. ホームディレクトリへ移動しなさい。
6. cd .config/xfce4/mcs_settings で 1-3 の動作をまとめて行えることを確認しなさい.
7. 以下の操作を行って “*”,“?”, “[]”の意味することを考えなさい。
    1. ls のみ
    2. ls d*
    3. ls m*
    4. ls *a*
    5. ls [dm]*
    6. ls [a-m]*
    7. ls *[a-z].*
    8. ls ?o*
    9. ls *o*
    10. ls [^a-m]*
8. ホームディレクトリへ戻りなさい。

ファイル名やディレクトリ名を指定する場合に,特殊な文字を利用することができます。これらの文字をワイルドカードと呼びます。

文字	意味
*	任意の 0 個以上の文字を示す。
?	任意の 1 個の文字を示す
[]	かっこ内のすべての文字を示す ただし最初の文字が^の場合には否定形

実習 2.2　まだまだ探す気ですか(複数の引数, ディレクトリ名にもワイルドカード)

1. ホームディレクトリで「隠しファイル」でないものをすべて表示させなさい。
2. ホームディレクトリでファイル名として “x”を含んでいるものだけを表示させなさい。
3. ls j* *s* を入力してファイルを表示させなさい。予めどのような出力が出て来るのかを予想してから Enter キーを押しなさい。.
4. 「隠しディレクトリ」.config へ移動しなさい。さらにディレクトリ xfce4 へ移動しなさい。可能ならば一回だけの操作で移動しなさい。
5. ls * を実行しなさい。予めどのような出力が出て来るのかを予想してから Enter キーを押しなさい(おそらく予想と違ったものが出力されるのではないかと思います).
6. 前の実習で, なぜそのような出力になったのか説明できるようになったら, ls *a* を実行しなさい。予想通りの出力が得られましたか?
7. ls */*p* を実行しなさい。 予めどのような出力が出て来るのかを予想してから Enter キーを押しなさい
8. ホームディレクトリへ移動しなさい。
9. ディレクトリ/usr の二つ下にある名前の中に q を含むファイル名/ディレクトリ名を表示するにはどうしたらよいでしょうか?ディレクトリを移動しないで調べなさい.

普通のファイルにも, いろいろなソフトで利用されるファイルがあります。それらはどのように区別したらいいのでしょうか?

実習 2.3　一体何を探しているのか(普通のファイルの種類)

1. 準備
    1. 「ほぼメモ帳」を起動しなさい。
    2. test だけ入力しなさい。
       [ファイル]→[名前をつけて保存]の順に呼び出し, test0 というファイル名で保存しなさい。
    3. 内容が「test テスト」となるように「テスト」の部分を追加しなさい。
       [ファイル]→[名前をつけて保存]の順に呼び出し, test1 というファイル名で保存しなさい。.
    4. 「ほぼメモ帳」を終了しなさい。
2. file test0 を実行しなさい。出力を記録しなさい。
3. file test1 を実行しなさい。出力を記録しなさい。
4. file test* を実行しなさい。出力に納得してください。
5. ls *.pdf を実行した後 file *.pdf を実行しなさい。

6. ls H* を実行した後 file H* を実行しなさい。
7. ls .config を実行した後 file .config を実行しなさい。
8. ls -l /usr/X11R6/bin/xpdf を実行した後 file /usr/X11R6/bin/xpdf を実行しなさい。
   ファイル名の補完機能を利用すると便利に入力できるはずです。
9. ls -l /etc/rc.d/rc.inet1 を実行した後 file /etc/rc.d/rc.inet1 を実行しなさい。

ファイル LinuxPersonalWorkstation.pdf を見るためには, 特別なソフトを必要としました。また How2Use.odt は「ワープロ」で作成したものでした(と予想されます)。

ls の出力だけでは, すべて「普通のファイル」としか判断できなかったものも file コマンドを使って, ファイルの内容について予想がつくようになります。

3 ファイルの内容を確認する

普通のファイルで, 内容が単純な「テキスト」である場合には「エディタ」のようなプログラムを起動しないでも内容を確認することができます。

利用するコマンドは cat, more, less です. more, less は一画面だけ出力すると一旦停止次の操作を続けることができます。

実習 3.1 確認してみよう

1. ls -l test* を実行し,
   file test* をさらに実行しなさい
2. cat test0 を実行しなさい。ファイル test0 の内容が表示されているはずです
3. cat test1 を実行しなさい。
4. cat test* を実行しなさい
5. file .canna を実行し
   cat .canna を続けて実行しなさい.
6. more .canna を実行しなさい。
   1. 「スペース」キーを押しなさい. Enter キーは不要です。
   2. もう一度「スペース」キーを押しなさい. Enter キーは不要です。
   3. キー b を押しなさい. Enter キーは不要です
   4. /forward Enter を押しなさい.
   5. キー q を押しなさい. Enter キーは不要です.
7. less .canna を実行し, more と同じ操作を行いなさい. More とどこが違うのか捜し出しなさい.
8. file /usr/man/man1/ls.1.gz を実行しなさい。出力をよく見ておきなさい。
   1. cat /usr/man/man1/ls.1.gz を実行しなさい。
      画面がおかしくなったら何度か Enter キーを押してプロンプトが出る状態までにしなさい。
   2. more /usr/man/man1/ls.1.gz を実行しなさい。
      終了させなさい。
   3. less /usr/man/man1/ls.1.gz を実行しなさい。
      今度は出力が英語ですがきちんと, more, less の動作ができることを確認しなさい。
      最後に終了しなさい。
9. file /bin/ls を実行しなさい。出力をよく見ておきなさい。
   1. cat /bin/ls を実行しなさい。
      画面がおかしくなったら何度か Enter キーを押してプロンプトが出る状態まにしなさい。
   2. more /bin/ls を実行しなさい。
      終了させなさい。
   3. less /bin/ls を実行しなさい。
      今度はうーん

file コマンドの出力に “text”と書かれているものは, そのまま画面に表示することが可能なようです。ものによっては less では人間が読める形式になるものもあります。
(最後の/bin/ls はコンピュータが理解できる形式なんです)

cat を使えば複数のファイルをまとめて画面に表示させることもできます。
(cat は concatenate: 連結するの略です)

4 今日も作る、運ぶ、コピー、そして捨てられる

ここで学習する内容は多いのでひとつの実習が終了したら頭を整理してください。また、実習の間の関連も実際の利用状況を考えて, どのような利用方法が可能なのかを考えるクセをつけてください。

ここで学習するコマンドは以下の通りです.

- ディレクトリを作成する。(`mkdir`: Make Directory の略)
- ディレクトリを削除する。(`rmdir`: Remove Directory の略)
- ファイルを移動する。この中にはファイル名の変更も含まれます(`mv`: Move の略)
- ファイルをコピーする。(`cp`: Copy の略)
- ファイル/ディレクトリを削除する(`rm`: Remove の略)

実習 4.1　作る

0. ホームディレクトリにいることを確認しなさい. もし, ホームディレクトリにいない場合にはホームディレクトリへ移動しなさい.
1. ホームディレクトリにあるファイル/ディレクトリを表示させなさい。
   オプションをうまく使って, 最新のファイル/ディレクトリが最も下になるように表示させなさい。
2. `mkdir may.ex` を実行しなさい。
3. ホームディレクトリにあるファイル/ディレクトリを最新のものが最も下に表示されるように表示させなさい。
   ディレクトリ `may.ex` が今作られたことを確認しなさい。
4. ディレクトリ `may.ex` へ移動しなさい.
5. 現在ディレクトリを確認しなさい.
6. ディレクトリの中のファイルを表示させなさい。
7. `cd -` を実行しなさい。
   現在位置を確認しなさい。
8. `cd -` を実行しなさい。
   現在位置を確認しなさい。
9. 一つ上のディレクトリへ移動しなさい.
10. 現ディレクトリのファイル/ディレクトリを表示させなさい. 最新のものが最も下に表示されるように表示させなさい。
11. `mkdir may.ex` を実行しなさい. 出力に注意すること!!
12. `mkdir` を実行しなさい. 出力に注意すること!!
    メッセージが気がかりな人はメッセージにしたがって調べてみること.
13. `rmdir` を実行しなさい. 出力に注意すること!!
    メッセージが気がかりな人はメッセージにしたがって調べてみること.
14. `rmdir .config` を実行しなさい
15. `rmdir may.ex` を実行しなさい.
16. 現ディレクトリのファイル/ディレクトリを表示させなさい. 最新のものが最も下に表示されるように表示させなさい。
17. `mkdir may.ex` を実行しなさい。
18. 最新のディレクトリに移動しなさい。
    現在位置を確認しておきなさい。

実習からディレクトリの作成/削除がわかったと思いますが, 注意点を三つ.

- Linux(UNIX)のコマンドは, 一般にうまくいったときには余分なメッセージを出力しません。エラーや警告があるときのみメッセージが出力されます。メッセージが出力され, 自分が意図していないメッセージが出たならば十分注意してください.
- `cd` の引数 “`-`”は “`-`”という名前のディレクトリを示しているわけではありません。「直前にいたディレクトリ」を示します. したがって“`cd -`”は直前の位置へ戻ることを示します。
- “`mkdir --help`”の出力を注意深く眺めると, 複数のディレクトリを一回の操作で作成する方法がわかります。

実習 4.2　コピー, ファイル名の変更

この実習では, 一つの操作を行うごとに現ディレクトリのファイルたちがどのように変化しているのかを必ず確認してください。ファイルの数は多くないので“ls -l”で確認することをお勧めします(もっと頭を使ってもらっても構いません).

0. 現在位置を確認しなさい.(may.ex にいるはず. そうなっていない人は移動すること)
   cp ../test* . を実行しなさい. (最後のピリオドを忘れないように!!)
1. cp test0 test2 を実行しなさい。
2. cp test1 test3 を実行しなさい。
3. mv test3 test4 を実行しなさい。
4. mv test4 test1 を実行しなさい。
5. cp -p ../test0 . を実行しなさい。(“.”, “..”が何を意味していたのかを探しだしてください。,また, 確認の出力を注意深く見てください)
6. ディレクトリ/etc/rc.d の中のファイル名をディレクトリを移動しないで表示させなさい。その際, ファイルの所有者, 他の人に対する許可権を確認しなさい.(現ディレクトリの確認不要)
7. ディレクトリ/etc/rc.d の中にあるファイルで, その名前に“net”を含んでいるファイル名だけを表示させなさい。その際, ファイルの所有者, 他の人に対する許可権を確認しなさい.(現ディレクトリの確認不要)
8. cp /etc/rc.d/*net* . を実行しなさい.
9. /etc/passwd- の情報を表示させなさい.(/etc/passwd ではありません!! 現ディレクトリの確認不要)
10. cp /etc/passwd- . を実行しなさい.
11. cp * を実行しなさい.
12. mv * を実行しなさい.
13. cp --help を実行しなさい。mv --help を実行しなさい
    それぞれ画面が流れていくので
    cp --help | more, mv --help | more を再度実行し, 出力を眺めなさい.
    (現ディレクトリの確認不要. more ってどこかで見ましたよね? 使い方は一緒です)

cp はファイルのコピーをします。 mv はファイル/ディレクトリ名の変更に用いることができます。

- mv ファイル 1 ファイル 2
  “ファイル 1”の名前を“ファイル 2”に変更します。もし “ファイル 2”が既に存在していた場合, “ファイル 1”の内容をもつ”ファイル 2”になってしまいます。
- cp ファイル 1 ファイル 2
  “ファイル 1”を“ファイル 2”にコピーします。もし “ファイル 2”が既に存在していた場合, “ファイル 2”の内容は失われ“ファイル 1”の内容をもつ “ファイル 2”になります。
- cp ファイル 1 … ディレクトリ
  “ファイル 1”からのファイルをディレクトリ内にコピーします. ディレクトリ内に同じ名前のファイルがあった場合, ファイルの内容は上書きされます.
  ファイルを複数指定し, 最後がディレクトリでない場合, エラーとなります。

ディレクトリの指定として “.”を利用することがよくあります(“.”は現在のディレクトリを示します)

実習 4.3　ファイルの移動, ディレクトリ名の変更

この実習では, 一つの操作を行うごとに現ディレクトリのファイルたちがどのように変化しているのかを必ず確認してください。ファイルの数は多くないので“ls -l”で確認することをお勧めします(もっと頭を使ってもらっても構いません).

1. 現在位置は may.ex のはずです。確認しなさい。
2. 現在位置で新しいディレクトリ may.ex2 を作成しなさい。
3. 現在位置のファイル test0, test2 をディレクトリ may.ex2 にコピーしなさい.
   コピーされたことを確認しなさい(新しく作成されたディレクトリ内部のファイルも確認しなさい)
   ここまでは復習.
4. mv may.ex2/test0 . を実行しなさい.
5. mv may.ex2 may.ex3 を実行しなさい.
6. mv may.ex2/test2 . を実行しなさい.

7. mv may.ex3/test2 . を実行しなさい.
8. ディレクトリ may.ex3 を削除しなさい.

コマンド mv はファイル名の変更だけではなく,

ファイルのディレクトリ間での移動

ディレクトリ名の変更

にも利用できることがわかります.

実習 4.4　総合演習(1)

ちょっと難しいかもしれません.

1. 新しいディレクトリ may.ex4 を作成しなさい.
2. 作成したディレクトリ may.ex4 へ移動しなさい.
3. 上のディレクトリにある test で始まるファイルたちを, 現在位置に移動させなさい.
   (ヒント: mv も cp と同じ操作を指示できます)
4. 一つ上のディレクトリへ移動しなさい.
5. ディレクトリ may.ex4 にある, すべてのファイルを現在位置に移動させなさい.
6. ディレクトリ may.ex4 を削除しなさい。
   (うまく操作ができていないと削除できません).

実習 4.5　ディレクトリ本体の移動

この実習では, 一つの操作を行うごとに現ディレクトリのファイルたちがどのように変化しているのかを必ず確認してください。ファイルの数は多くないので“ls -l”で確認することをお勧めします(もっと頭を使ってもらっても構いません).

0. cp -r /etc/rc.d . を実行しなさい.
1. mv rc.d test5 を実行しなさい.
2. mv test5 test0 を実行しなさい.
3. mv test5 . . を実行しなさい.
4. ホームディレクトリへ移動しなさい. そしてファイルを確認しなさい.

ディレクトリ自体の移動も mv を用いて行うことができます。

実習 4.6　ファイルの削除

この実習では, 一つの操作を行うごとに現ディレクトリのファイルたちがどのように変化しているのかを必ず確認してください。

1. ホームディレクトリにいることを確認しなさい.
2. ディレクトリ test5 が存在していることを確認しなさい。
3. ディレクトリ test5 へ移動しなさい.
4. ファイルを確認しなさい.
5. rm rc.4 を実行しなさい.
6. ファイル名に net を含む物だけを表示させなさい.
7. rm *net* を実行しなさい.
8. rm * を実行しなさい.
9. 一つ上のディレクトリへ移動しなさい.
10. ディレクトリ test5 を削除しなさい.

rm でファイルを削除することができます.

実習 4.7　ファイル/ディレクトリの削除(ちょっと危険な)

この実習では, 一つの操作を行うごとに現ディレクトリのファイルたちがどのように変化しているのかを必ず確認してください。

1. cp -r /etc/rc.d test5 を実行しなさい.
2. ls test5 を実行しなさい.
3. rm test5/* を実行しなさい.
4. ディレクトリ test5 を削除しなさい.
5. もう一度 cp -r /etc/rc.d test5 を実行しなさい.

6. rm -r test5 を実行しなさい.

rm の-r オプションで指定したディレクトリ(中に存在するディレクトリも含め)全部を一挙に削除することが出来ます

ここまでで学習してきたコマンドは mkdir, rmdir, cp, mv, rm です. コマンドの詳細については

```
man コマンド名
```

で調べることができます(man コマンドってのもあるということです).

実習 4.8　総合演習

1. ファイル D.sh を指示にしたがって転送しなさい。
   ディレクトリ may.ex へ移動しなさい.
   sh ../D.sh を実行しなさい.
   ファイルを確認しなさい. aaa0, aaa1, ..., ddd9 までの合計 640 のファイルが新規に作られているはずです.
2. 新しく作られたファイルの中身をいくつか調べてみて, ファイルの内容とファイル名の関係を推測しなさい。
3. ディレクトリをいくつか作成し, 以下のルールになるようにファイルを整理しなさい.
   ファイル名が *XYZn* のファイルはディレクトリ dir.*X*/dir.*Y* に入っているようにする。
   ただし *X*, *Y*, *Z* は a から d, *n* は 0 から 9 までとする.
   (ディレクトリを全部で 4+16 作成する必要があります)

5 ファイル操作の復習

ここまでのまとめです.

cd	ディレクトリを移動する.
ls	ファイル名を表示する
cp	ファイルをコピーする. オプション-r を使えば, ディレクトリ全体のコピーも行える.
mv	ファイル/ディレクトリ名の変更 ファイルのディレクトリをまたがる移動 ディレクトリの移動
rm	ファイルの削除. オプション-r を使えば, ディレクトリ全体の削除も可能.
mkdir	ディレクトリを作成する.
rmdir	ディレクトリを削除する. ディレクトリの中が空になっていないと削除できない. rm -r を利用する場面の方が多いか?
file	ファイルがどうのような内容のものであるか表示する
cat	ファイルの内容を表示する.
more	ファイルの内容を一画面ごとに表示する.
less	ファイルの内容を一画面ごとに表示する. More の高級版 (歴史的に more が先に作られていたので, 名前が less になってしまった)
pwd	現在位置の表示
man	コマンドの詳細な使用方法を表示する. オプションを忘れてしまったとき位に利用する方が無難か?

6 あっち向いてホイ

新しいファイルを作成する方法として現在知っているのは, エディタ(ほぼメモ帳や Emacs)を使うことだけです. 画面への出力をファイルへ出力して新しいファイルを作成する. あるいは画面への出力内容を, 別のコマンドの入力にすることなどができます。キーワードは “**標準出力**”, “**標準入力**”, “**標準エラー出力**” です.

実習 6.1　猫は連結?

1. ホームディレクトリに test0, test1 があることを確認しなさい.
2. cat test0 test1 を実行しなさい.
3. cat test0 test1 > test1.all を実行しなさい.
   新しく作られたファイルが何であるか確認しなさい.
4. 新しく作られたファイルの内容を確認しなさい.
5. cat -n test0 test1 を実行しなさい
6. cat -n test0 test1 > test2.all を実行しなさい.
   新しく作られたファイルが何であるか確認しなさい
7. 新しく作られたファイルの内容を確認しなさい.

画面への出力のことを「**標準出力**」と呼びます. また, キーボードから入力している部分を「**標準入力**」と呼びます

コマンド > ファイル名

でコマンドの画面への出力(標準出力)をファイルに書き込むことができます. このような入力や出力の方向転換のことをリダイレクションと呼びます.

実習 6.2　練習問題

1. ホームディレクトリにいることを確認しなさい.
2. ホームディレクトリ直下にあるファイル/ディレクトリ一覧(隠しファイルを除く)を詳細な形式でファイル files.all へ書き出しなさい。
   内容も確認しなさい。
3. ホームディレクトリ直下にあるファイル/ディレクトリ一覧(隠しファイルを除く)を名前だけ出力した files-short.all を作りなさい.
   内容も確認しなさい.
4. ディレクトリ emacs-mule へ移動しなさい.
5. セクション毎に作成した 6 個のファイルを繋げ合わせた all.txt を作りなさい.
6. ホームディレクトリへ戻りなさい.

画面に出力したものをファイルへ書き出すといったのに, 動作が違うじゃないかと気が付いた人は勘が鋭くなってきています. コマンドによっては, 画面への出力と, リダイレクションの出力が違っている場合があります.

コマンドの画面への出力を別の入力にすることも可能です.

実習 6.3　more

1. more .canna を実行して, more の利用方法を思い出しなさい.
2. cat .canna | more を実行しなさい.
   どのようなことが起きているのか予想がつきますか?
3. cat emacs-mule/7*.txt | more を実行しなさい.
4. cat -b emacs-mule/7*.txt | more を実行しなさい.
   cat のオプションについては cat --help あるいは man cat で調べてみなさい.

“|”で二つのコマンドを繋合わせ, 最初のコマンドの出力を 2 番目のコマンドの入力に利用しています. “|”のことをパイプといったりします。

コマンドの中にはそれだけでは利用方法の見当がつかないものがかなりあります。典型的なものとして wc(ファイルの行数/単語数/バイト数を表示する)があります.

実習 6.4

1. cat test0 test1 を実行しなさい
2. cat test0 test1 | wc を実行しなさい
   勘が良い人は cat test[0-9] | wc でもよいことは分かっていると思いますが…
3. それでは ls | wc を実行しなさい
   出力の意味する所を考えなさい
   (files-short.all の内容がヒントになるかもしれません.wc –help も利用できるかも?).

Linux のコマンドは, パイプを用いて利用する事が多いので, 必要な情報のみを出力する事が多くなっ

ています.
　>があったのだから<があってもおかしくないと思った人は勘の良い人です. さらに, 多分「標準出力を…」ではなく「標準入力を…」と思った人はもっと勘の良い人です.
あまり良い例ではありませんが

実習 6.5　面白くない例

1. cat < .canna | more を実行しなさい

　>とは少し違って>>というのもあります。man bash を実行して>>を探してみてください. man bash は

```
gunzip -c /usr/man/ja/man1/bash.1.gz | nroff -man | less
```

を簡単にできるようにしたものです. 最後が less で終っているので, less の利用方法を思い出せば >>を探すことができるはずです.

7 ファイル内容の検索, ファイル名の検索(上級者向け)

　Windows で遊んでいると

- フォルダのアイコンを右クリック
- 検索をクリック
- ファイル名のすべてまたは一部
- ファイルに含まれている単語または句

という項目に出会うことができます。
　その中の「詳細設定オプション」を開いてみると

- システムフォルダの検索
- 隠しファイルとフォルダの検索
- サブフォルダの検索
- 大文字と小文字の区別

を指定して検索を行うことが可能となっていることがわかります.
　Linux でもこのようなことができないか? というのがこのセクションの主題です. 結論から先にいうと, 同様なことが可能です。しかし, Linux の場合, 機能の少ないものをパイプなどを用いて利用することが多いので, 同様のことを実現するために色々な方法が存在します.

実習 7.1　文字列を含むファイルを探す(grep)

1. E.sh がまだある人は実行しなさい.
   ディレクトリ games.files に移動しなさい.
2. grep you file1 を実行しなさい
3. grep you file4 を実行しなさい
4. grep you file* を実行しなさい
5. grep good file* を実行しなさい.
6. grep -n '[Yy]ou' file* を実行しなさい

　grep は指定したファイルたちの中から, 指定した文字列を探しだし, 見付けた行を表示します。指定したファイルが複数ある場合には, ファイル名も出力されます.
(オプション-v を指定すると, 指定した文字列を含まない行だけを出力することができます)
ot22

実習 7.2　こしゃくな使い方(正規表現)

1. ディレクトリ/usr/include には沢山のファイルがあり, ファイル名としては .h で終っているものが多数あります。そのようなファイルが何個あるか調べなさい.
2. grep '[0-9]\{1,\}BIT' /usr/include/*.h を入力しなさい。
   '数字 BIT'という文字が入っているものを検索します.
3. grep '^ *A' /usr/include/*.h を入力しなさい(^と*の間にはスペースが 1 個入っている

ことに注意)

行の先頭が A で始まっているものを検索します. ただし, 行の先頭にある空白は無視します.

grep で指定する検索文字列としては, 凝った指定ができるようになっています。指定方法は「正規表現」と呼ばれます.

指定したディレクトリより下にあるファイル/ディレクトリについて検索を行うには find コマンドを用います.

find コマンドの一般的な形式は

find ディレクトリ … 条件 …

です.

実習 7.3 find を用いた検索の例

1. ホームディレクトリにいることを確認しなさい
2. find . -name 'test*' を入力しなさい.
   何が出力されましたか?
3. find . -type d を入力しなさい.
   何が出力されましたか?
4. find . -ctime -10 -exec ls \{\} \; を入力しなさい.
   何が出力されましたか?
5. find . -type f -exec grep excellent \{\} \; を入力しなさい.
   何が出力されましたか?

詳しいことは “man find”で調べてみましょう.

8 読める? 書ける?

実習 8.1 まずは実習

1. ホームディレクトリにいることを確認しなさい.
2. /etc/X11/app-defaults/XOrgCfg をホームディレクトリにコピーしなさい.
3. コピーしたファイルを mule を使って編集しなさい.
   1. ファイルの最後へ移動しなさい.
   2. label を Label に変更しようとしなさい.
      変更ができないはずです.
   3. 仕方がないので終了しなさい.
4. ファイル XOrgCfg の情報をなるべく詳しく調べなさい.
   1. コピーしたファイルは, 自分の持ち物になっていること
   2. 自分に対しても書き込み権がないことを確認しなさい

自分の持ち物であるにもかかわらず, 許可権が変になっているときには, 当然ながら許可権を変更しなければいけません。許可権を変更するコマンドは chmod です.

chmod *nnn* ファイル …

が一般的な形式です(もっと楽な方法もありますが). ここで *nnn* は三桁の 7 以下の数字です. 最初の桁が, 自分への, 次の桁がグループへの, 最後の桁がすべての人への許可権を示します. それぞれの桁は

4(r)	読み込み権あり
2(w)	書き込み権あり
1(x)	実行権あり

の**総和**となっています. ファイル XorgCfg に対する許可権はコピーした直後では “-r--r--r--”となっているために書き込み権がなかったことになります.

実習 8.2 許可権の変更

1. chmod 644 XorgCfg を実行しなさい.
2. 許可権がどのように変更されたか調べなさい.

3. 自分に対する書き込み権があることを確認した後, mule を用いて XOrgcfg が変更できることを確認しなさい.
4. a.out の許可権を確認しなさい.
5. 実行権があることを確認したのち, ./a.out を実行しなさい.
6. chmod 644 a.out を実行しなさい.
7. ./a.out を実行しなさい.
8. E.sh あるいは D.sh に実行権を与えなさい.
9. ./E.sh あるいは./D.sh を実行しなさい.

今まで sh xxx と実行していたファイル xxx に実行権を与えてやれば, そのままコマンドのように利用できるというわけです. このような例は/usr/local/bin の中のファイルたちです. どのようなファイルがあって, その内容はどうなっているのか調べてみなさい.

ディレクトリに対する許可権は少し分かりにくくなっています.

許可権	意味
読み込み(r)	ディレクトリ内のファイル名だけを見ることができる
書き込み(w)	ディレクトリにファイルを作成できる
実行(x)	ディレクトリ内のファイルの読み書き, 削除ができる

9 リンク

Windows には「ショートカット」という, 遠くのファイルを近くで操作できるような機能があります. 当然 UNIX にもあります.

実習 9.1 ハードリンク

1. /usr/local/bin へ移動しなさい.
2. ディレクトリ内のファイルを調べなさい。偶然(?)同じ大きさのファイルがないか調べなさい。
3. 同じ大きさのファイルの中身を調べなさい。

mule と emacs は大きさも同じ, 内容も同じです. これは偶然でしょうか? 実は同じファイルに対して別名を付けているのです. ls -l の出力が他のファイルと少し異なっています(2 番目の列). ls -l の出力から名前が二つあることがわかります。

野良猫は同じ一匹の猫なのにあちこちで別な名前を付けてもらっているのに似ています. 一つのファイルに複数の名前をつけることを「リンク」を張るといったりします. リンクを張るには

```
ln [-s] 存在するファイル/ディレクトリ 別名のファイル/ディレクトリ
```

を用います. -s は以下に述べるシンボリックリンクで用います.

実習 9.2 ソフトリンク(シンボリックリンク)

ハードリンクでは、ディレクトリに別名をつけることはできません. せっかく Windows の D ドライブが見えるのにわざわざ出かけていくのもシャクなので, ホームディレクトリから別名をつけてみましょう.

1. ホームディレクトリに移動しなさい
2. ln -s /mnt/d: D-drive を実行しなさい(名前は自分で利用し易い名前にしても構いません)
3. D-Drive へ移動しなさい.
4. 一つ上へ戻りなさい.

これで使いやすいなったはずです.

10 10 shell 変数, 環境変数

コンピュータを使っていろいろなことをやりはじめると, 同時に複数のことを覚えておかなきゃいけないことが出て来ます。そのような時には, コンピュータに覚えさせておくと便利です. 覚えさせる機能としてシェル変数があります.

実習 10.1　Shell 変数の紹介

1. ホームディレクトリへ移動しなさい.
2. echo HOME を実行しなさい.
3. echo $HOME を実行しなさい.
4. /etc/rc.d へ移動しなさい.
5. E=/etc/rc.d を実行しなさい.(=の両端にスペースをいれてはいけない)
6. cd $HOME を実行しなさい.
7. echo $E を実行しなさい.
8. cd $E を実行しなさい.
9. ls -a $HOME を実行しなさい
10. printenv を実行しなさい

記憶機能があることがわかります(高級メモリー電卓?) 代入する時には=を用いて, 利用する時には$をつけて用いればよいことがわかります. このような変数のことを「シェル変数」といいます.

実習 10.2　お遊び

環境を壊してしまうので, 新たに「コマンド入力」で窓を作ってから実習をしましょう.

1. PS1==== : を実行しなさい.
2. /usr/bin へ移動しなさい.
3. 直前の位置へ戻りなさい.
4. OLDPWD=/etc を実行しなさい.
5. 直前の位置に戻りなさい(あれ?)
6. files='ls -l' を実行しなさい.
7. $files を実行しなさい

シェル変数を使う時に引用符をうまく使うと便利な使い方ができます, 引用符には', “, `の 3 種類があり, それぞれ利用方法が異なります。

実習 10.3　引用符の利用

利用するシェル変数として D, E の二つを用います. 引用符の違いに注意してしなさい. それぞれの, 実行が終ったら echo $D, echo $E でシェル変数の中身を確認しながら実習を行いなさい.

1. E=abc を実行しなさい
2. E=abc def を実行しなさい
3. E='abc def' を実行しなさい
4. E=”abc def” を実行しなさい
5. D=”abc” を実行しなさい
6. D='abc $E' を実行しなさい
7. D=”abc $E” を実行しなさい
8. 今どこにいるのか確認しなさい
9. E=`pwd` を実行しなさい
10. D=`ls` を実行しなさい
11. この項は実習する必要はありません
    E=`date +%Y%m%d.%H%M` を実行し,
    mule tst.$E を実行するとどのようなファイルができることになりますか?
    (本当は mule tst.`date +%Y%m%d.%H%M`でも大丈夫なんですが)

引用符の意味の違いは以下のようになります.

引用符	意味
'	その中の文字をそのまま代入する
“	シェル変数についてはその値に置き換える
`	コマンドの実行結果そのものになる. シェル変数を利用した場合, その値に置き換わる

こう使わなきゃいけないというルールはありませんが, 次のような場合には便利かも知れません. 二つのディレクトリ/usr/share/emacs/22.0.50/lisp/international,

/usr/X11R6/lib/X11/locale/ja の間でファイル操作(コピーなど)を行う必要が生じた.

- /usr/share/....に移動し S=`pwd`を
- /usr/X11R6/...へ移動し X=`pwd`を実行

すれば, 長ったらしいファイル指定をせずに$S, $X で二つのディレクトリを利用することができます.

シェル変数 D に対して

```
export D
```

のように実行したものを「**環境変数**」と呼び, 他のプログラムからでも利用できるようになります.

11 PATH(重要な環境変数)

コマンドを実行するたびに, どうして ls だけで実行できるの? 素数を求めるときにどうして./a.out と指定しなきゃいけなかったの? と思ったことはありませんか? ls って一体なにものなのでしょうか? 実体は./a.out と何ら違いはありません. ただ, すべての人が利用できるようになっているだけです. また, 格納場所も決っています.

実習 11.1

1. echo $PATH を実行しなさい.
   出力は:(コロン)で区切られたいくつかの部分からなりたっています. それぞれの部分をメモしなさい.
2. 今まで利用してきたコマンドがどこに格納されているのか”which コマンド”で調べなさい.
   コマンドの中には win, mule などがあります. また, xeyes も調べてみなさい.

コマンドは, 環境変数 PATH で指定されたディレクトリ内の実行可能なファイルたちであることがわかります.

ログイン, あるいは新しいコマンド入力の窓を開く際には, /etc/profile が実行されます. その後, ホームディレクトリの.profile が実行されます.

実習 11.2

1. /etc/profile の内容から PATH=となっている行を見付けなさい.
   echo $PATH の出力と一致していることを確認しなさい.

12 落ち穂拾い

Mule の起動時に最後に“&”を最後につけましたが, その意味は.... コマンド入力は原則的に一行で一つのコマンドですが, 以下のようにすると, 複数のコマンドを指定できます.

- コマンド<Return>
  コマンドを実行します. コマンドが終了しないと戻ってきません.
- コマンド 1; コマンド 2<Return>
  コマンド 1 が終了すると, コマンド 2 を実行します. コマンド 2 が終了するまで戻ってきません.
- コマンド 1 & コマンド 2<Return>
  コマンド 1 とコマンド 2 を同時に開始します. コマンド 2 が終了するまで戻ってきません.

mule &と入力していたものは, これの特殊系です.

調子にのって“xeyes &”なんて実行させたらどうなるでしょうか? mule の場合には, 自分で終了させることができましたが...

実習 12.1 プロセス

1. ps ax を実行しなさい.
   現在コンピュータで実行されているプログラムの全てが表示されます.
2. xeyes & を実行しなさい.
3. ps ax を実行し, 出力の中に xeyes があることを確かめなさい.
4. pkill xeyes を実行しなさい. xeyes が終了することを確認しなさい.

実習 12.2　最後の遊び

1. oneko-1.1b.tar.gz を転送しなさい.
2. gunzip -c oneko-1.1b.tar.gz | tar xf – を実行しなさい.
3. 作成されたディレクトリへ移動しなさい.
4. ファイル README に書いてある順序に, コマンドを実行しなさい
5. 作成された実行形式ファイルを探しだし, 実行してみなさい.

ちょっと難しいかもしれない課題

誰でも, oneko を実行できるようにするにはどのようにしたらよいでしょう?

13 Emacs(Mule)自習書

`LinuxPersonalWorkstation.pdf` の 115~118 ページを読んで来ているとしてすすめます。
使用にあたっては TGU での環境が少し違っている点があるので注意として列挙しておきます.

1. パネルのエディタから直接起動されるもの, およびパネルのエディタから「万能エディタ」を選択して起動されるものが Emacs/Mule です. この場合, 起動される位置がホームディレクトリ決め打ちににになっています。
2. 開発中のバージョンになっています(最新バージョンより新しい)。
3. 起動方法で
    1. `mule` で起動: 日本語が利用できます
    2. `emacs` で起動: 日本語が利用できません

と能書きは以上です.

実習 13.1　自習書

1. 「コマンド入力」をクリックしなさい.
2. `mule -q&` を入力しなさい.
3. `C-h T` を押しなさい.
4. 「Emacs 入門ガイド」が表示されるので, 自習しなさい.
   その際, カーソルの移動は出来る限り矢印キーを利用しないで実習しなさい.

- LinuxPersonalWorkstation.pdf にミスプリントが一箇所あります.ミスプリントを探し出しなさい.

実習 13.2　復習

以下の設問に自習書を参考に**しないで**答えなさい

1. カーソル移動
    1. 一文字右へ
    2. 一文字左へ
    3. 一行上へ
    4. 一行下へ
    5. ファイルの先頭へ
    6. ファイルの最後へ
    7. 行の先頭へ
    8. 行の最後尾へ
    9. 指定した行へ
2. 削除
    1. カーソル位置の文字を削除
    2. カーソル位置の直前の文字を削除
    3. カーソル位置より行の最後まで削除
3. Cut & Paste
    1. 範囲指定のやり方
       でマーク指定。その後カーソルを移動
    2. 切り取り(指定した範囲の)
    3. コピー(指定した範囲の)
    4. 貼り付け
4. 検索
    1. カーソル位置からファイルの後ろに向かっての検索
    2. カーソル位置からファイルの先頭に向かっての検索
5. その他
    1. 強制終了
    2. ファイルの上書き保存
    3. ファイルに名前を付けて保存

実習 13.3　コマンドラインでの編集

カーソルキーを利用しないで実習をしないでといったのには理由があります。Emacs/Mule でのカーソル移動方法は Linux の他の場所で利用されています.

0. 「コマンド入力」が起動されていることを確認しなさい
1. Emacs/Mule のカーソル移動を思い出して, 以下の操作を行いなさい.
    1. 一行上へ
    2. さらに一行上へ
    3. 行の先頭へ
    4. 一文字削除
    5. C-c で終了.
2. Emacs/Mule のカーソル移動を思い出して, 以下の操作を行いなさい.
    1. 上へ向かって rm を検索
       C-c で終了
    2. 上へ向かって mule を検索
       C-c で終了
    3. その他 mule のキー操作を行ってみなさい(例: 一文字削除, 一単語削除など)
       最後は C-c で終了

「コマンド入力」では一行だけ表示されている「mule」と同じ動作をしていることが分かります. 通常は「ファイル」の最後尾にいると考えればいいわけです。

この機能と「ファイル名/コマンドの補完」機能を利用すると, キータイプの時間が大幅に節約することが可能になります.

14 総合演習

実習 14.1 ちょっとした練習問題です.

1~1000 までの素数を求めようとしてプログラムを書いて見ましたが, 何箇所かにタイプミスがあります。タイプミスを修正して, 1000 までの素数を求めなさい(プログラムの作成でよく見られる典型的な手順です).

0. mule-ex.c を転送しなさい.
1. エラー出力がなくなるまで以下の操作をしなさい.
    1. gcc mule-ex.c をタイプする
       MAXodd, prme がおかしいといってくる.
        1. MAXodd については MAXODD にすべて変更
        2. prme については prime にすべて変更
2. ./a.out をタイプ. 出力は 1000 までの素数を求めた結果となっています
3. mule-ex.c が読みにくいので以下の操作を行って読みやすいものにしなさい.
    1. mule-ex.c を mule を利用して編集する.
    2. main を検索する.
    3. 下の行の{にカーソルをあわせる
    4. M-C-q を押す
       (プログラミング言語 C の文法を知っているので, きれいになったはず)
    5. 保存して終了
4. 興味のある人は#define MAXTMP の行の 1000 を 10000 に変更して手順 1,2 をやってみてください.

実習 14.2 必要な人は…(辞書登録)

自分の姓名が一発で出ない人は辞書に登録しておくと便利になります。

1. mule を起動する。
2. 無理矢理にでも登録したい単語を入力する
3. 登録したい単語を範囲指定する
   (範囲指定を忘れた人は, 無駄なあがきかもしれません. もう一度復習しましょう)
4. M-x canna-touroku-region を入力する.
   M-x の意味は分かっていますよね.
5. 後は設問に答えていくだけ.

実習 14.3　タイプの練習もかねて

LinuxPersonalWorkstation.pdf の 115 ページから 118 までを以下のようにファイルに作成しなさい.

1. すべてのファイルはディレクトリ emacs-mule に作成する。
2. 段落は一行の空白行を用いる
3. セクション毎にファイルを作成する. ファイルの名前は “セクション番号.txt”とする
   ファイルとしては 7.2.txt, 7.2.1.txt, 7.2.2.txt, 7.2.3.txt, 7.2.4.txt, 7.2.5.txt ができることになります。
4. 脚注を入力する必要はありません。
5. 普通の入力が難しい太字はそのままで構いません.
6. 表はそれらしく見えるように入力してください.

15 ファイルの日本語コード

mule を起動させると画面下部に

```
-かんな-J:-- tutorial.txt 3% L37 (Text Fill)---------------
```

のように表示されています.

表示	意味	
日本語入力	「かんな」,「逐次」のように表示される	
日本語コード	日本語で書かれた文書は, ファイルのコード体系が異なる場合がある. ファイルのコード体系を自動認識して表示する	
	J	JIS コード(iso-2022-jp-)
	E	EUC コード. Linux/Unix でよく利用される(euc-jp-)
	S	シフト JIS コード. Windows/Macintosh で利用される(sjis-)
	u	UTF コード. 国際化に最もよく対応したもの. Windows の内部コードとしても利用されている.(utf-8-)
改行モード	Windows と Linux では, 行の最後の改行の扱いが異なっている.	
	:	Linux 流の改行(unix)
	(DOS)	Windows 流の改行(dos)
	(Mac)	Macintosh 流の改行(mac)
変更有り	ファイルに変更があったかどうか	
	**	変更有り
	%%	読み込み専用(書き込み不可)
ファイル名		
位置		
編集モード	ファイル名によっていろいろなモードになる. また, サブモードもある.	

実習 15.1 Windows と Linux でのファイルの相互利用

この実習は Linux と Windows とを切替えて使用するので時間がかかります. また USB メモリを利用します.

1. mule を用いて日本語のファイルをホームディレクトリに作成しなさい.
   2 行以上のファイルである事
   ファイル名は win-linux.txt とする
2. 作成したファイルを/mnt/m にコピーしなさい
3. Windows を起動し,「メモ帳」を使って USB メモリ上にある(はず)win-linux.txt を編集しなさい。
   どのようになっているかを確認しなさい.
4. 「メモ帳」を使って新規ファイル USB メモリに win-linux2.txt を作成しなさい.
   2 行以上のファイルであること.
   win-linux.txt の内容と異なる事
5. Linux を起動し/mnt/m/win-linux2.txt を編集しなさい.
   どのようになっているかをよく見なさい.

日本語の入ったファイルを操作する場合, 特に異なるコンピュータ間で利用する場合には, ここで見たようにいろいろとトラブルことがあります.

mule を使って, ファイルの日本語コード体系, 改行コードの変更を行うことができます.

```
C-x Enter F
```

に続けてファイルの属性を指定します. 指定方法は表中のタイプライタ体で示したものを利用します.
例: Windows で利用するファイルに変換する場合, C-x Enter F に続けて

```
sjis-dos
```

を指定する

16 いろいろなモード

編集しているファイルの名前などから判断して, いろいろなモードになります. 基本的なキーの機能は変化しませんが, モード毎に拡張機能が追加されます. どのような機能がキーに割り当てられているのかを調べるには

```
C-h b
```

を用います.

dired モード

ファイル名としてディレクトリを指定して起動した場合, あるいは M-x dired で明示的に起動した場合に起動されます. ディレクトリ中のファイル一覧が表示され, ファイル操作を行うことができます. M-x dired で起動した場合, ファイル名に対してワイルドカード指定を行って, 指定したファイルのみを操作対象にすることも可能です.

よく使いそうな機能だけを抜きだしてみました.

キー	機能	キー	機能
C	ファイルのコピー	^	上のディレクトリへ
D	ファイルの削除(マークのみ)	e または f	ファイルの編集
R	ファイル名の変更	g	表示の更新
U	すべてのマークをはずす	u	マークをはずす
Z	ファイルの圧縮	v	書き込み禁止でファイルを開く

アウトラインモード

長い文章などを書く時に利用するとよいかもしれません.

行の先頭が'*'で始まる行は, 一個の場合セクションを, 2 個の場合サブセクションを示します. セクションの本体を隠したり, 表示させたりできます.

言語ごとのモード

ここでいう言語とは, 日本語, 英語のような人間が使う言語ではなく, コンピュータ用の言語をさします. 主要な言語に対応しています.

C(.c, .h, .cxx), Java(.java), HTML(Web 用の言語)(.html), シェルスクリプト(.sh), Perl(.pl), TeX(.tex)などがよく利用するものでしょうか. 括弧内のタイプライタ体で書かれたファイル名で終了するファイルを編集(しようと)すると, 自動的にそのモードに変化します.

必要になったら, どのような機能がキーに割り当てられているか調べてから利用ください.