LAN 実習マニュアル

第六版
2014 年 3 月 19 日（水）

目次

1 オペレーティングシステムは百花繚乱

1.1 この文章について

みなさん、LAN 実習室へようこそ。これからみなさんは、この実習室でさまざまな実習をしていくことになるわけですが、そのためには、まず最初に、この実習室にあるパソコンの使い方を習得していただく必要があります。この文章の目的は、LAN 実習室にあるパソコンの使い方をみなさんに説明することです。

ちなみに、この実習室が「LAN 実習室」と呼ばれる理由は、この実習室ができた 1993 年の時点では、コンピュータが LAN(local area network) に接続されていたのが、この学校の中ではこの実習室だけだったからです。

1.2 オペレーティングシステムって何？

コンピュータの基本的な動作をコントロールするプログラムは、「オペレーティングシステム」(operating system) と呼ばれます（オペレーティングシステムは、OS という略称で呼ばれたり、「基本ソフト」と呼ばれたりすることもあります)。オペレーティングシステムの具体的な例としては、Windows、OS X、iOS、Android、Firefox OS、Chrome OS などがあります。

1.3 Linuxって何？

オペレーティングシステムのもっとも核となる部分は、「カーネル」(kernel) と呼ばれます。

リーナス・トーバルズ (Linus Torvalds) という人が 1991 年に開発を開始したカーネルは、「Linux カーネル」(Linux kernel) と呼ばれます。

Linux カーネルにさまざまなプログラムを加えることによって、オペレーティングシステムとして使えるようにしたものは、「Linux ディストリビューション」(Linux distribution) と呼ばれます。

Linux ディストリビューションの具体的な例としては、Fedora、CentOS、openSUSE、Gentoo Linux、Vine Linux、Ubuntu などがあります。

「Linux」という言葉は、Linux カーネルの意味で使われる場合と、各種の Linux ディストリビューションの総称として使われる場合とがあります。

1.4 LAN 実習室のパソコンのオペレーティングシステムは何？

パソコンが使えるようになるためには、そのパソコンで使われているオペレーティングシステムの使い方を習得する必要があります。さて、それでは、LAN 実習室にあるパソコンは、オペレーティングシステムとして何を使っているのでしょうか。

LAN 実習室にあるパソコンがオペレーティングシステムとして使っているのは、Ubuntu という Linux ディストリビューションです。

したがって、LAN 実習室のパソコンが使えるようになるためには、Ubuntu の使い方を習得する必要がある、ということになります。

2 Ubuntu 事始め

2.1 ログインって何？

それでは、さっそくパソコンの電源を ON にしてみましょう。

パソコンの電源を ON にしてからしばらくすると、モニターに、「ログイン画面」と呼ばれる画面が表示されます。これは、「ログイン」(login) と呼ばれる作業をするための画面です。

ログインというのは、これからオペレーティングシステムを使うのが誰なのかということをオペレーティングシステムに通知する、という作業のことです。

ただし、オペレーティングシステムは、誰にでも自分の使用を許可するわけではありません。オペレーティングシステムを使うことができるのは、オペレーティングシステムにその名前があらかじめ登録されている人間だけです。

オペレーティングシステムに名前が登録されている人間のことを、そのオペレーティングシステムの「ユーザー」(user) と言います。そして、オペレーティングシステムに登録されているユーザーの名前のことを「ログイン名」(login name) または「ユーザー名」(user name) と言います。

2.2 あなたのログイン名は？

みなさんは、もうすでにユーザーとして Ubuntu に登録されています。ただし、みなさんのログイン名は、みなさんの本当の名前とは違っています。みなさんのログイン名は、英小文字の j の右側に入学年度（西暦）の下 2 桁を並べたものです。たとえば、入学年度が 2058 年度の人は、

```
j58
```

というログイン名になります。

Ubuntu のログイン画面には、Ubuntu に登録されているユーザーのログイン名が表示されています。ログインをするためには、その中から自分のログイン名を選択して、パスワードを入力する必要があります。

みなさんのパスワードは、みなさんのログイン名と同じです。たとえば、入学年度が 2058 年度の人は、`j58` というのがパスワードになります。

それでは、実際にログインしてみましょう。ログイン画面に表示されている自分のログイン名をキーボードの矢印キーかマウスで選択して、パスワードを入力してください（入力した文字は、画面には丸印で表示されます）。もしも間違った文字を入力してしまった場合は、Backspace というキーを押すことによって、その文字を消すことができます。正しく入力できたら、Enter というキーを押してください。

テンキー（キーボードの右端にある電卓のようなキー）を使って数字の入力をしたいという場合は、あらかじめ、テンキーの中にある Num Lock というキーを押しておく必要があります。

もしも、入力したログイン名が正しくなかった場合は、

> パスワードが違います。もう一度試してください。

と表示されますので、再度、パスワードを入力してください。

ログインすると、画面の上端と左端に、細長い長方形の領域が表示されます。画面の上端にある左右に細長い長方形の領域は、「メニューバー」(menu bar) と呼ばれます。そして、画面の左端にある上下に細長い長方形の領域は、「ローンチャー」(launcher) と呼ばれます（ローンチャーは、「ランチャー」と呼ばれることもあります）。

2.3 パソコンをシャットダウンしてみよう

パソコンの電源を ON にしたいときは電源スイッチを押せばいいわけですが、電源を OFF にしたいときは、電源スイッチを押すのではなくて、「シャットダウン」(shutdown) という処理をパソコンに実行させる必要があります。シャットダウンというのは、動作している状態にあるさまざまな機能を穏やかに終了させて、電源が OFF になっても大丈夫な状態にすることです。シャットダウンをパソコンに実行させることを、単に「パソコンをシャットダウンする」と言います。

パソコンをシャットダウンしたいときは、次のような操作をします。

(1) メニューバーの右端にある歯車のアイコンをクリックする。
(2) そうすると、アイコンの下にメニューが表示されるので、その中の「シャットダウン」をクリックする。
(3) そうすると、

> すべてのプログラムを終了して、コンピューターをシャットダウンしますか？

というメッセージが表示されるので、そのメッセージの下にある「シャットダウン」というボタンをクリックする。

この操作をすると、シャットダウンが実行されて、それが完了すると、自動的に電源が OFF になります。

それでは、実際にパソコンをシャットダウンしてみてください。

2.4 ローンチャーでプログラムを起動してみよう

Ubuntu では、画面の左端にある「ローンチャー」(launcher) と呼ばれる領域を使うことによって、さまざまなプログラムを起動することができます（ただし、ローンチャーを使うことによって起動することができるのは、ウィンドウを持っているプログラムだけです。ウィンドウを持っていないプログラムを起動する方法については、第 3 節で説明します)。

ローンチャーの上に表示されているアイコンをクリックすると、そのアイコンで示されるプログラムがまだ起動していない場合は、そのプログラムが起動します。たとえば、地球とキツネが描かれているアイコンをクリックすると、Firefox というプログラムが起動します。これは、ウェブを閲覧するためのブラウザーです。

ウィンドウを持っているプログラムは、ローンチャーの上に自分のアイコンを表示します。最小化されたウィンドウを復元したいときや、別のウィンドウに隠されているウィンドウを前面に表示したいときは、ローンチャーの上に表示されている、そのプログラムのアイコンをクリックします。

2.5 ローンチャーでプログラムを検索してみよう

ローンチャーの上にアイコンが表示されていないプログラムは、それを検索することによって起動することができます。

プログラムを検索したいときは、まず、ローンチャーの一番上にあるアイコン（「検索アイコン」と呼ぶことにします）をクリックします。すると、その右側に、検索用の入力欄が表示されます。そこに検索したいプログラムの名前を入力すると、そのプログラムのアイコンが入力欄の下に表示されます。

それでは、試しに、「数独」というパズルを検索してみましょう。検索アイコンをクリックして、入力欄に、

```
sudoku
```

という文字列を入力してください。すると、数独のアイコンが入力欄の下に表示されますので、そのアイコンをクリックしてください。そうすると、数独が起動するはずです。

2.6 ローンチャーにプログラムを登録してみよう

ローンチャーには、プログラムを登録することができます。ローンチャーにプログラムを登録するというのは、そのプログラムが動作していないときでも、そのプログラムのアイコンがローンチャーの上に表示されるようにする、ということです。ローンチャーにプログラムを登録しておくと、検索しなくてもプログラムを起動することができますので、とても便利です。

ローンチャーにプログラムを登録したいときは、次のような操作をします。

(1) 登録したいプログラムを起動する。
(2) そうすると、ローンチャーの上にそのプログラムのアイコンが表示されるので、そのアイコンをマウスで右クリックする。
(3) そうすると、アイコンの右側にメニューが表示されるので、そのメニューの中の「Launcher に登録」をクリックする。

それでは、試しに、ローンチャーに数独を登録してみてください。

2.7 ローンチャーへのプログラムの登録を解除してみよう

ローンチャーへのプログラムの登録は、いつでも解除することができます。

ローンチャーへのプログラムの登録を解除したいときは、ローンチャーの上に表示されている、解除したいプログラムのアイコンを右クリックして、表示されたメニューの中の「Launcher への登録を解除」をクリックします。

それでは、試しに、ローンチャーへの数独の登録を解除してみてください。

3　魔法の呪文を唱えると

3.1　コマンドって何？

Windows や OS X と同じように、Ubuntu も、マウスなどのポインティングデバイスを使うことによってさまざまな操作をすることができます。

しかし、ポインティングデバイスというのは、オペレーティングシステムを操作するための唯一の手段ではありません。オペレーティングシステムの多くは、「コマンド」(command) と呼ばれるものを使って操作することも可能です。

コマンドというのは、オペレーティングシステムに実行してほしい動作を、文字列として記述したもののことです。それをパソコンに入力すると、そのとおりにパソコンが動くわけです。つまり、魔法の呪文のようなものだと考えることができます。

3.2　端末を起動してみよう

コマンドを使ってオペレーティングシステムを操作するためには、コマンドを入力するためのプログラムを起動する必要があります。そのようなプログラムは、Windows では「コマンドプロンプト」(Command Prompt) と呼ばれ、OS X では「ターミナル」(Terminal) と呼ばれ、Ubuntu では「端末」(Terminal) と呼ばれます。

それでは、ローンチャーを使って端末を起動してみましょう。ローンチャーの検索アイコンをクリックして、入力欄に、

```
terminal
```

という文字列を入力してください。そうすると、「端末」と書かれたアイコンが表示されますので、それをクリックしてください。そうすると、端末が起動します。

端末は、非常によく使うプログラムですので、ローンチャーに登録しておくといいでしょう。

3.3　シェルって何？

Windows のコマンドプロンプトや OS X のターミナルや Ubuntu の端末は、その中で、「シェル」(shell) と呼ばれるプログラムを動かしています。

シェルというのは、人間が入力したコマンドを解釈して実行するプログラムのことです。シェルにはさまざまなものがあるのですが、これからみなさんが実習で使うことになるのは、bash という名前のシェルです。

シェルは、「私は今、人間がコマンドを入力するのを待っています」ということを人間に伝えるために、「プロンプト」(prompt) と呼ばれる文字列を出力します。端末のウィンドウの中に表示されている、

ログイン名@ホスト名:~$

という形のものが、シェルが出力したプロンプトです。ちなみに、アットマークの右側に表示される「ホスト名」というのは、ネットワークの上でパソコンを識別するための名前のことです。

プロンプトの右側には、文字と同じ大きさの白い長方形が表示されています。この長方形は、「カーソル」(cursor) と呼ばれるものです。カーソルは、キーボードから入力した文字が表示される場所を示しています。

3.4　カレンダーを出力させてみよう

第 2 節で説明したように、Ubuntu では、ローンチャーを使うことによってさまざまなプログラムを起動することができます。しかし、ローンチャーを使うことによって起動することができるのは、ウィンドウを持っているプログラムだけです。では、ウィンドウを持っていないプログラムを起動するためにはどうすればいいのでしょうか。

ウィンドウを持っていないプログラムを起動する方法は、Windows でも OS X でも Ubuntu でも同じです。それは、プログラムを起動するコマンドを入力するという方法です。

Windows でも OS X でも Ubuntu でも、プログラムを起動するコマンドは、そのプログラムの名前です。つまり、プログラムの名前をコマンドとしてシェルに入力すると、そのプログラムが起動する、ということです。

オペレーティングシステムとして Ubuntu を使っているパソコンには、カレンダーを出力する、cal という名前のプログラムが入っています。このプログラムはウィンドウを持っていませんので、起動するためにはコマンドをシェルに入力する必要があります。

プログラムを起動するコマンドはそのプログラムの名前ですから、cal は、

```
cal
```

というコマンドをシェルに入力することによって起動することができます。

それでは、このコマンドをシェルに入力してみましょう。もしも間違った文字を入力してしまったら、Backspace を押してカーソルを戻してください。正しく入力できたら、Enter キーを押してください。そうすると、端末のウィンドウの中に今月のカレンダーが表示されるはずです。

3.5 あなたが生まれたのは何曜日？

cal というプログラムは、今月のカレンダーだけしか出力できないというわけではありません。西暦 1 年から西暦 9999 年までの範囲内ならば、いつのカレンダーでも出力することができます。それでは、今月以外のカレンダーを出力させたいときは、いったいどうすればいいのでしょうか。

シェルを使ってプログラムを起動するときに、そのプログラムの動作についての細かい指示を与えたいという場合は、プログラムの名前の右側に、その指示を意味する単語を付け加えたものをコマンドとして入力します。プログラムの名前の右側に付け加える単語のことを、コマンドの「引数」(argument) と言います。

カレンダーを出力してほしい月と年を cal に指示したいときは、

```
cal 月 年
```

というコマンドを入力します。つまり、出力してほしい月と年を cal というプログラム名の右側に付け加えればいいわけです。

なお、コマンドに含まれているそれぞれの単語の間には、1 個以上の空白を入れる必要があります。

それでは、実際に、月と年を指定して cal を起動してみましょう。たとえば、

```
cal 2 1936
```

というコマンドを入力してみてください。そうすると、西暦 1936 年 2 月のカレンダーが出力されるはずです（二・二六事件が何曜日だったかが分かりますね）。

それでは次に、同じようにして、自分が生まれた月のカレンダーを出力させてみてください。

3.6 ユリウス暦からグレゴリオ暦へ

cal コマンドを入力するときに、月を省略して年だけを指定すると、指定された年のすべての月のカレンダーが出力されます。たとえば、

```
cal 1752
```

というコマンドを入力すると、1752 年のすべての月のカレンダーが出力されるはずです。

ところで、1752 年のカレンダーには、ものすごく変なところがあるのですが、気が付きましたか。これは、機械の故障でも目の錯覚でもありません。実は、イギリスとその植民地では、1752 年というのは本当にこんなカレンダーだったのです。

3.7 その者、真の名前を知りたれば

コマンドで起動することができるのは、ウィンドウを持たないプログラムだけではありません。ウィンドウを持っているプログラムをコマンドで起動するということも可能です。

しかし、プログラムをコマンドで起動するためには、そのプログラムの本当の名前を知っている必要があります。通称やニックネームを入力しても、

```
コマンドが見つかりません
```

とシェルに言われるのがオチです。

たとえば、数独の本当の名前は、「数独」ではありません。本当の名前は、

```
gnome-sudoku
```

と言います。この名前をシェルに入力して、数独が起動するかどうかを確かめてみてください。

ところで、シェルは、プログラムを起動するコマンドが入力された場合、原則として、そのプログラムが終了するまで次のコマンドを受け付けません。でも、ウィンドウを持っているプログラムをコマンドで起動した場合は、そのプログラムがまだ動いているときでも、次のコマンドを受け付けてくれると便利ですよね。

プログラムを起動するコマンドの末尾にアンパサンド (&) を書いておくと、シェルは、起動したプログラムがまだ動いていても、次のコマンドを受け付けてくれます。たとえば、

```
gnome-sudoku &
```

というコマンドで数独を起動すると、数独がまだ動いているうちに、次のプロンプトが出力されます。試してみてください。

4 ファイルをどんどん作ろう

4.1 リダイレクトで流し込む

パソコンには、ハードディスクや SSD(solid state drive) などの、データを保存するという機能を持つデバイスが搭載されています。ハードディスクや SSD の中では、データは、「ファイル」(file) と呼ばれる箱に入れて管理されています。

個々のファイルは、それに与えられた名前によって識別されます。ファイルに与えられた名前は、「ファイル名」(file name) と呼ばれます。ファイルを操作するプログラムは、コマンドの引数としてファイル名を受け取って、それによって指定されたファイルに対してさまざまな操作を実行します。

ところで、プログラムが出力したデータは、普通、端末のウィンドウに表示されます。でも、実は、プログラムが出力したものは、ファイルの中に流し込むことも可能です。それをしたいときは、プログラムを起動するコマンドの右側に、「大なり」という文字 (>) と、データが格納されるファイルに付けたい名前とを追加します。つまり、

コマンド > ファイル名

というコマンドを入力すればいいわけです。

それでは、

```
cal 1970 > expo70.txt
```

というコマンドを入力してみてください。端末のウィンドウには何も表示されませんね。でも、このコマンドによって 1970 年のカレンダーが出力されて、それが expo70.txt というファイルの中に格納されたはずです。このように、「大なり」という文字を使うことによってプログラムの出力先をファイルに切り換えることを、出力を「リダイレクトする」(redirect) と言います。

ファイルに付ける名前には、普通、ドット (.) で始まる文字列が末尾にくっついています。ドットとその右側の部分は、「拡張子」(extension) と呼ばれます。拡張子は、ファイルに格納されているデータの形式をあらわしています。ちなみに、 .txt というのは、ごく普通のテキストという属性をあらわす拡張子です。

さて、それでは、 expo70.txt の内容を見てみましょう。ファイルの内容を見たいときは、cat という名前のプログラムを使います。

cat ファイル名

というコマンドを入力すると、引数で指定された名前を持つファイルの内容が出力されます。ですから、

```
cat expo70.txt
```

というコマンドを入力することによって、 expo70.txt の内容を出力させることができるはずです。

4.2 みんな、自分専用のディレクトリを持っている

ハードディスクや SSD の中には、ファイルを整理するために、ファイルを何個でも入れることのできる箱を作ることができるようになっています。そのような、ファイルを入れることのできる箱は、「ディレクトリ」(directory) と呼ばれます。

オペレーティングシステムのユーザーには、一人にひとつずつ、「ホームディレクトリ」(home directory) と呼ばれるディレクトリが与えられていて、その中には、ファイルを自由に作ることができるようになっています。先ほどみなさんが作った、 expo70.txt というファイルも、みなさんのホームディレクトリの中にあります。

ディレクトリの中にどんなファイルがはいっているのかということを知りたいときは、ls というプログラムを使います。

```
ls
```

というコマンドで ls を起動すると、ディレクトリの中にあるすべてのファイルの名前が出力されます（青色で出力されているのは、ファイルではなくてディレクトリの名前です）。

ただし、このコマンドだと、ファイルの名前だけしか表示されません。それぞれのファイルの属性（更新日時や大きさなど）について知りたいときは、

```
ls -l
```

というように、`-l` という単語をコマンドに追加します。ちなみに、この `-l` のような、先頭にマイナスの付いた単語は、「引数」ではなく「オプション」(option) と呼ばれます。

`-l` というオプションを付けて ls を起動すると、ls は、ファイルの名前だけではなくて、ファイルを最後に更新した日付と時刻や、ファイルの大きさなども出力します。

4.3 分身の術をマスターする

次に、ファイルのコピーを作ってみましょう。

ファイルのコピーを作りたいときは、cp という名前のプログラムを使います。cp を起動するコマンドには、二つの引数を付ける必要があります。1 個目の引数でコピー元のファイルを指定して、2 個目の引数でコピー先のファイルに名前を付けます。つまり、

```
cp [ファイル名 1] [ファイル名 2]
```

というコマンドを入力すると、[ファイル名 1] で指定されたファイルをコピーすることによって、[ファイル名 2] という名前を持つ新しいファイルが作られる、ということです（[ファイル名 2] を名前として持つファイルが既に存在している場合は、そのファイルの元の内容が消えてしまいますので、くれぐれも注意してください）。

それでは、以前に作った `expo70.txt` というファイルをコピーすることによって、`expo70b.txt` という名前のファイルを作ってみましょう。つまり、

```
cp expo70.txt expo70b.txt
```

というコマンドを入力すればいいわけです。コピーができたら、`expo70b.txt` の内容を cat に出力させて、`expo70.txt` と同じ内容になっているかどうかを確かめてください。

4.4 気に入らない名前とはさようなら

次に、ファイルの名前を変更してみましょう。

ファイルの名前を変更したいときは、mv というプログラムを使います。cp と同じように、mv を起動するコマンドにも 2 個の引数を書く必要があります。1 個目は変更する前の名前で、2 個目は変更後の名前です。つまり、

```
mv [ファイル名 1] [ファイル名 2]
```

というコマンドを入力すると、[ファイル名 1] というファイルの名前が [ファイル名 2] に変更される、ということです。

それでは、まず、自分が生まれた年のカレンダーを、`jibun.txt` という名前のファイルに入れてください。それができたら、そのファイルの名前を `birth.txt` に変更してみましょう。この場合、mv を起動するコマンドは、

```
mv jibun.txt birth.txt
```

ということになります。さて、本当に名前が変更されたでしょうか。ls と cat を使って確かめてみてください。

4.5 地球にやさしい？

次に、ファイルを削除してみましょう。

ファイルを削除したいときは、rm というプログラムを使います。rm を起動するコマンドには、削除したいファイルの名前を引数として指定します。つまり、

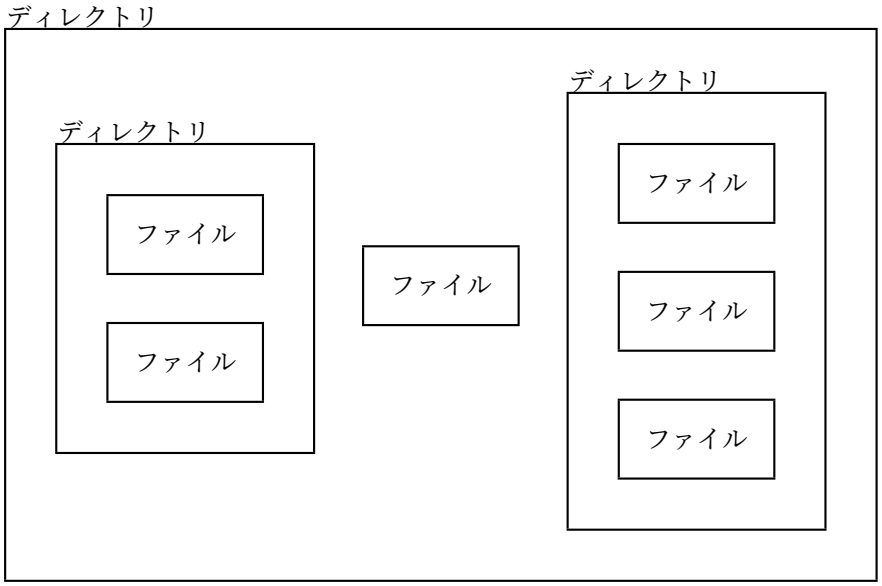

図 1: ディレクトリの入れ子

```
rm ファイル名
```

というコマンドを入力すると、ファイル名という名前のファイルが削除される、ということです。

それでは、まず、昨年のカレンダーを sakunen.txt という名前のファイルに入れてください。それができたら、そのファイルを削除してください。つまり、

```
rm sakunen.txt
```

というコマンドを入力すればいいわけです。それでは、本当にファイルが削除されたかどうか、ls で確かめてみましょう。

5 パス名を制する者はコマンドを制す

5.1 パス名って何？

ファイルやディレクトリを指定するための記述のことを、「パス名」(path name) と言います。ファイルやディレクトリの名前もパス名の一種ですが、ファイルやディレクトリの名前だけがパス名ではありません。それらの名前を組み合わせたパス名というのもあります。

この節では、パス名について説明したいと思います。

5.2 親ディレクトリの下に子ディレクトリ

第 4.2 項で説明したように、ディレクトリというのはファイルを入れることのできる箱のことです。しかし、ディレクトリの中に入れることができるものは、ファイルだけではありません。

図 1 を見てください。この図のように、ディレクトリには、ファイルだけではなく、ディレクトリを入れることもできるのです。つまり、ディレクトリは何重にでも入れ子にすることができるということです。このようにディレクトリを何重にも入れ子にしておくと、ファイルを系統的に整理することができるので、とっても便利です。

ディレクトリの中にディレクトリまたはファイルがあるという構造は、図 2 のような親子関係の図によってあらわすこともできます。この図では、外側のディレクトリの下に、その内側にあるディレクトリまたはファイルが描かれています。ファイルやディレクトリがどこにあるかということについて誰かに説明するときは、このような図を頭の中に描いて、「このディレクトリの下にあるディレクトリ」とか、「ここから二つ上がったところ」というような言い方をするのが普通です。また、ひとつ上のディレクトリのことを「親ディレクトリ」(parent directory) と呼ぶこともあります。

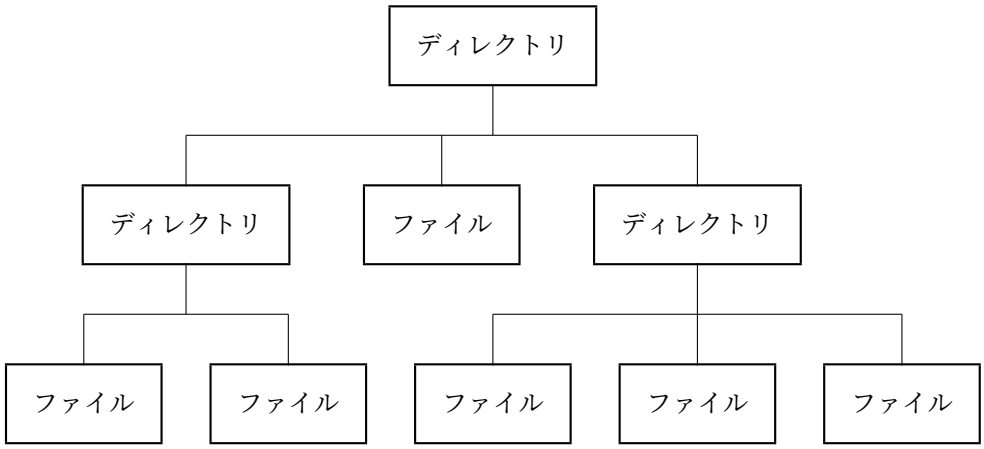

図 2: ディレクトリの親子関係

5.3 ルートディレクトリって何？

ディレクトリの親子関係は、親の方向へ上っていくと、かならず、親を持たないディレクトリに到達します。そのような、親を持たないディレクトリは、「ルートディレクトリ」(root directory) と呼ばれます（「ルート」というのは「根」という意味です）。ルートディレクトリは、スラッシュ(/) というパス名で指定することができます。

ls を起動するとき、

```
ls パス名
```

というように、パス名を引数として書くと、ls は、そのパス名がディレクトリを指定するものならば、そのディレクトリの下にあるものの一覧を出力します。

それでは、ls を使って、みなさんのパソコンのルートディレクトリの下にどんなものがあるか、調べてみましょう。

```
ls /
```

というコマンドを入力してみてください。そうすると、ls は、ルートディレクトリの下にあるものの一覧を出力します。

5.4 絶対パス名はルートディレクトリが出発点

パス名には、絶対パス名と相対パス名という 2 種類のものがあります。

「絶対パス名」(absolute path name) というのは、ルートディレクトリを出発点にして、親から子供へディレクトリを順番にたどっていくことによってファイルやディレクトリを指定するパス名のことです。

絶対パス名は、

```
/名前/名前/･･･/名前
```

というように、まず先頭にスラッシュを書いて（これはルートディレクトリの名前）、その右側に、親から子へという順番でディレクトリの名前をスラッシュ(/) で区切って並べていって（このスラッシュは単なる区切り文字）、最後に、指定したいファイルまたはディレクトリの名前を書きます。たとえば、それぞれのユーザーのホームディレクトリは、ルートディレクトリの下にある home というディレクトリの下にある、そのユーザーのログイン名と同じ名前のディレクトリですので、

```
/home/ログイン名
```

という絶対パス名で指定することができます。それでは、

```
ls /home/自分のログイン名
```

というコマンドをシェルに入力してみてください。そうすると、自分のホームディレクトリにあ

るものの一覧が出力されるはずです。

5.5 相対パス名はカレントディレクトリが出発点

シェルは、常に、どこかのディレクトリを「現在位置」として認識しています。シェルが現在位置として認識しているディレクトリは、「カレントディレクトリ」(current directory) と呼ばれます。カレントディレクトリは、ユーザーが自由に変更することができます。

シェルが起動した直後の状態では、カレントディレクトリは、ユーザーのホームディレクトリになっています。

ルートディレクトリを出発点とするパス名が「絶対パス名」と呼ばれるのに対して、カレントディレクトリを出発点とするパス名は、「相対パス名」(relative path name) と呼ばれます。シェルは、先頭がスラッシュではないパス名が入力された場合、それを相対パス名だと判断します。

ファイル名を単独で書くと、それは、カレントディレクトリの下にあるそのファイルを指定する相対パス名になります。たとえば、第 4.1 項で登場した、

```
cat expo70.txt
```

というコマンドの中にある `expo70.txt` というファイル名は、「カレントディレクトリの下にある `expo70.txt` というファイル」を指定する相対パス名です。

5.6 トキメキドットドットスラッシュ

ところで、カレントディレクトリのひとつ上にあるディレクトリ、つまりカレントディレクトリの親ディレクトリを相対パス名で指定したい場合は、どう書けばいいのでしょうか。

カレントディレクトリの親ディレクトリを指定するパス名は、ドットドット (`..`) です。ということは、

```
ls -l ..
```

というコマンドを入力すると、カレントディレクトリのひとつ上にあるディレクトリの内容についての情報が出力されるはずです。また、

```
ls -l ../ディレクトリ名
```

というコマンドを入力すると、カレントディレクトリと同じ階層にある別のディレクトリの内容についての情報が出力されるはずです。

ちなみに、カレントディレクトリ自身を指定する相対パス名は 1 個のドット (`.`) で、カレントディレクトリの二つ上にあるディレクトリ（親ディレクトリの親ディレクトリ）を指定するパス名はドットドットスラッシュドットドット (`../..`) です。

6 ファイルはディレクトリで整理整頓

6.1 ディレクトリを作ってみよう

ディレクトリは、自由に作ったり削除したりすることができます。

新しいディレクトリを作りたいときは、mkdir という名前のプログラムを使います。mkdir は、

```
mkdir パス名
```

というコマンドを入力することによって起動します。パス名のところには、新しく作られるディレクトリを指定するパス名を書きます。

それでは、新しいディレクトリを作ってみましょう。

```
mkdir renshuu
```

というコマンドを入力してください。そうすると、カレントディレクトリの下に、`renshuu` という名前のディレクトリができるはずです。ls を使って確かめてください。ls は、ディレクトリの名前を青色で表示しますので、 `renshuu` も青色で表示されるはずです。

次に、 `renshuu` の下にファイルを作ってみましょう。

```
cal > renshuu/kongetsu.txt
```

というコマンドを入力してみてください。この場合、cal が出力したものは、renshuu というディレクトリの下に作られた kongetsu.txt というファイルの中に格納されるはずです。それでは、本当にそうなったかどうか、cat を使って確かめてください。

6.2 上へも下へも自由自在

カレントディレクトリは、自由に変更することができます。カレントディレクトリを変更したいときは、

```
cd パス名
```

というコマンドをシェルに入力します。そうすると、シェルは、パス名で指定されたディレクトリをカレントディレクトリにします。たとえば、

```
cd renshuu
```

というコマンドを入力することによって、カレントディレクトリを renshuu に変更することができます。同じように、

```
cd ..
```

というコマンドを入力することによって、カレントディレクトリの親ディレクトリをカレントディレクトリにすることができます。

ここでちょっと余談ですが、cd というのは、プログラムの名前ではなくて、シェルの内部にある機能を実行させるためのコマンドです。このようなコマンドは cd のほかにもいくつかあって、それらは「内部コマンド」(internal command) と呼ばれます。

ところで、cd でカレントディレクトリを変更すると、シェルのプロンプトが変化する、ということにお気付きでしょうか。実は、プロンプトの中には、カレントディレクトリをあらわすパス名が含まれているのです。

プロンプトの中に含まれているカレントディレクトリをあらわすパス名は、ホームディレクトリまたはそれよりも下にあるディレクトリがカレントディレクトリになっている場合、チルダ (~) という文字を含んでいます。このチルダという文字は、みなさんのホームディレクトリを意味しています。

ホームディレクトリ以外のディレクトリがカレントディレクトリになっているときに、引数を何も付けないで、ただ単に、

```
cd
```

とシェルに入力すると、カレントディレクトリはホームディレクトリに戻ります。

6.3 過去を精算するのは楽じゃない

ディレクトリを削除したいときは、rmdir というプログラムを使います。rmdir を起動するコマンドは、

```
rmdir パス名
```

と書きます。この中のパス名というところには、削除したいディレクトリのパス名を指定します。

それでは、実際にディレクトリを削除してみましょう。まず、utakata という名前のディレクトリを自分のホームディレクトリの下に mkdir で作って、それが本当にできているかどうかを ls で確かめてください。utakata がちゃんとできていたら、今度はそれを削除してみましょう。つまり、

```
rmdir utakata
```

というコマンドを入力すればいいわけです。それでは、本当に削除されたかどうか、ls で確かめてみてください。

このように、rmdir を使うことによってディレクトリを削除することができるわけですが、ただし、rmdir で削除することができるのは、中に何も入っていない空のディレクトリだけです。試しに、少し前に作った renshuu というディレクトリを引数で指定して rmdir を起動するコマンドを入力してみてください（renshuu は、空ではなくて、そこには kongetsu.txt というファイルが入っているはずです)。この場合、rmdir は、

```
rmdir: ‘renshuu’ を削除できません: ディレクトリは空ではありません
```

というメッセージを出力するだけで、指定されたディレクトリを削除しません。つまり、何かが中に入っているディレクトリを削除したいときは、あらかじめその中のものを rm または rmdir で削除しておく必要がある、ということです[1]。

6.4 USB メモリーもひとつのディレクトリ

Ubuntu が動いているパソコンに USB メモリーを挿入すると、Ubuntu は、それをひとつのディレクトリとして認識します。

USB メモリーは、

/media/ ログイン名 / USB メモリー名

という絶対パス名で指定することができます。

Ubuntu は、USB メモリーが挿入されると、ローンチャーの上に USB メモリーのアイコンを表示します。USB メモリーをパソコンから抜きたいときは、その前にかならず、USB メモリーのアイコンを右クリックして、表示されたメニューの中にある「取り出し」をクリックする必要があります。

7 テキストエディターって、教科書編集者？

7.1 いえいえ、ちょっと違います

文章とかプログラムとかのような、文字が並んでできているデータは、「テキスト」(text) と呼ばれます。テキストを編集したいとき、つまり、テキストを新しく作ったり、すでにあるテキストを修正したりしたいときは、「テキストエディター」(text editor) と呼ばれるプログラムを使います（テキストエディターは、ただ単に「エディター」と呼ばれることもあります)。

これから先の実習で、みなさんは、gedit という名前のテキストエディターを使うことになります。そこで、この節では、gedit の使い方について説明したいと思います。

gedit を起動したいときは、

gedit パス名 &

というコマンドをシェルに入力します。パス名 のところには、編集の対象となるファイルを指定するパス名を書きます。存在しないファイルのパス名を指定した場合は、そのパス名で指定される新しいファイルが作られて、そこにテキストが保存されることになります。

ちなみに、コマンドの末尾にあるアンパサンド (&) は、第 3.7 項で説明したように、起動したプログラムが終了するのを待たずに次のコマンドを受け付けてほしい、とシェルに対してお願いするためのものです。

それでは、実際に gedit を起動してみましょう。まず、みなさんのホームディレクトリの下にあるはずの、renshuu というディレクトリをカレントディレクトリにしてください。それができたら、次に、

```
gedit ningen.txt &
```

というコマンドを入力してください。そうすると、gedit が起動します。

それでは次に、ウィンドウの中にある白い領域の左上を見てください。そこで縦棒が点滅しているはずです。この縦棒は、カーソルです。つまり、この縦棒は、キーボードで文字を入力した文字が挿入される位置を示しているわけです。

7.2 キーボードでどんどん入力

さて、それではいよいよテキストの入力です。

```
Mine Fujiko
Kusanagi Motoko
Akagi Ritsuko
Saotome Kazuko
```

[1] 中身のあるディレクトリを一発で削除するコマンドを書くことも可能ですので、興味のある人は自分で調べてみてください。

```
#include <stdio.h>

int main(void)
{
    int n;
    printf("整数を入力してください。: ");
    scanf("%d",&n);
    printf("%d の 2 乗は%d です。\n", n, n*n);
    return 0;
}
```

図 3: プログラムのサンプル

というように、自分が知っている人（友人、有名人など、誰でもかまいません）の名前を入力してみましょう（漢字を入力することも可能ですので、漢字で入力してもかまいません）。

7.3 保存されてると安心

ここで、タブに表示されている ningen.txt というファイル名の左側に注目してください。そこに、1 個のアスタリスクが表示されているはずです。このアスタリスクは、「テキストが変更されているにもかかわらず、現在のテキストがファイルに保存されていない状態になっている」ということを示しています。

テキストをファイルに保存したいときは、ツールバーの中にある、「保存」と書かれたアイコンをクリックするか、または、Ctrl と書かれたキー（「コントロールキー」と呼ばれます）を押しながら S のキーを押します。そうすると、編集中のテキストがファイルに保存されて、タブに表示されていたアスタリスクが消滅します。

それでは、入力されたテキストが本当にファイルに保存されているかどうか、cat を使って確かめてみてください。

8 ワクワクドキドキ、はじめての C

8.1 プログラムを入力してみよう

これからみなさんは、LAN 実習室で、C 言語によるプログラミングの実習をすることになります。そこで、この節では、C 言語によるプログラミングの実習はどのように進めていけばいいのか、ということについて説明したいと思います。

まず最初に、C 言語によるプログラミングのためのディレクトリを作りましょう。自分のホームディレクトリの下に、c という名前のディレクトリを作ってください。それができたら、そのさらに下に kadai00 というディレクトリを作ってください。そして、kadai00 をカレントディレクトリにしてください。つまり、プロンプトに表示されているカレントディレクトリが、

```
~/c/kadai00
```

になっていればいいわけです。

次に、テキストエディターでプログラムを入力しましょう。

```
gedit nijou.c &
```

というコマンドで gedit を起動して（このように、C 言語のプログラムを保存するファイルには、.c という拡張子を付けるのが普通です）、図 3 のプログラムを入力してください。プログラムの意味は、まだ理解できなくてもかまいません。

プログラムの入力が終わったら、それをファイルに保存してください（入力が完全に終わったときだけではなくて、その途中で頻繁に保存することを強くお勧めします。なぜなら、そうすることによって、何らかのアクシデントによって保存ができなくなった場合に、被害を最小限に留めることができるからです）。

8.2 プログラムをコンパイルしてみよう

C 言語で書かれたプログラムは、コンピュータに理解できる言語（そのような言語のことを「機械語」(machine language) と言います）に翻訳しなければ、コンピュータに実行してもらうことができません。プログラムを機械語に翻訳することを、プログラムを「コンパイルする」(compile) と言います。

プログラムをコンパイルしたいときは、「コンパイラ」(compiler) と呼ばれるプログラムを使います。Ubuntu には、gcc という名前の C 言語のコンパイラが含まれていますので、それを使うことによって、C 言語のプログラムをコンパイルすることができます。

gcc を起動するコマンドは、

```
gcc パス名
```

と書きます。この中の パス名 というところには、C 言語のプログラムが保存されているファイルを指定するパス名を書きます。そうすると、gcc は、指定されたファイルの中のプログラムをコンパイルして、a.out という名前のファイルに機械語のプログラムを出力します。

それでは、nijou.c をコンパイルしてみましょう。そのためのコマンドは、

```
gcc nijou.c
```

ということになります。プログラムの中にエラー（誤り）が含まれていた場合は、それについてのメッセージが画面の上に出力されます。エラーがなかった場合は、画面には何も出力されません。

gcc が出力するエラーメッセージには、エラーが発見された行の番号とエラーの内容が含まれています。もしもエラーがあった場合は、そのエラーメッセージを参考にしてプログラムを修正してください。修正が終わったら、プログラムを保存して、もう一度それをコンパイルしてください。

なお、エラーメッセージが指摘する行の番号は、あくまでエラーが発見された場所です。修正しなければならない場所は、かならずしもエラーが発見された場所とは限りません。

8.3 プログラムを実行してみよう

プログラムにエラーがなかった場合は、a.out という名前のファイルが作られて、そこに機械語のプログラムが格納されます。それでは、a.out ができているかどうか、ls を使って確かめてみてください。

a.out ができていたら、それを実行させてみましょう。カレントディレクトリの下にある a.out をコンピュータに実行させたいときは、

```
./a.out
```

というコマンドを入力します。ちなみに、このコマンドの先頭にあるドット (.) は、第 5.6 項で紹介した、カレントディレクトリを指定するパス名です。

先ほど入力していただいたプログラムは、1 個の整数を読み込んで、その 2 乗を出力する、という動作をします。プログラムを起動すると、

```
整数を入力してください。:
```

というメッセージが出力されますので、整数をひとつ入力して、Enter を押してみてください。そうすると、入力した整数の 2 乗が出力されるはずです。

索引