

RAINBOW GUIDE

UNIX 操作入門編

立命館大学 情報システム部 情報基盤課 著

2010 年 4 月 1 日

序

現在，インターネットはその普及や発展に伴って，自宅のコンピュータや携帯電話から簡単に利用できるものとなりました．インターネットを通じてメールを送ったり，動画を見たり，通信販売にて商品を購入したりすることは，既に日常的に行われています．このようなインターネットサービスは情報科学の知識が多く使われており，その基盤として UNIX というオペレーティングシステム (OS) が活用されています．UNIX を学ぶことによって IT や情報科学についての知識を多く得ることができます．

立命館大学では，教育研究用ネットワークとして RAINBOW を提供しています．RAINBOW では UNIX 系 OS のひとつである Linux が利用可能です．Linux は，無料で配布されています．また，開発はインターネット上で活発に行われています．Linux の配布形態は様々であり，それらを Linux ディストリビューションと呼びます．RAINBOW では，Linux ディストリビューションのひとつである Vine Linux を採用しています．

本書では，RAINBOW 環境における UNIX の利用方法について説明します．基本的な UNIX の操作方法から，授業のレポート作成やプログラミング課題の作成など，普段の学生生活と密接に関わりのあるアプリケーションの使い方について説明します．本書の内容に触れることによって，コンピュータをより身近なものに感じ，IT や情報科学について興味，関心を持っていただくことが本書の目的です．

執筆者一同

目次

| | |
|---|----|
| 序 | i |
| 第 1 章 はじめに | 1 |
| 1.1 RAINBOW でできること | 1 |
| 1.2 Linux とは | 3 |
| 1.3 ユーザーアカウントとパスワード | 4 |
| 1.4 RAINBOW のコンピュータを利用する上での注意事項 | 6 |
| 1.5 本書の内容 | 7 |
| 第 2 章 目的別リファレンス | 9 |
| 2.1 レポートを作成したい | 9 |
| 2.2 プログラムを作成したい | 10 |
| 2.3 インターネットをしたい | 11 |
| 2.4 Linux 環境をカスタマイズしたい | 11 |
| 第 3 章 基本操作 | 13 |
| 3.1 ログインとログアウト | 13 |
| 3.2 GNOME メニューからのアプリケーションの起動 | 15 |
| 3.3 GNOME 端末からの起動 | 16 |
| 3.4 使ってみよう | 16 |
| 3.5 USB メモリへの保存 | 17 |
| 3.6 印刷 | 18 |
| 第 4 章 シェル | 23 |
| 4.1 シェルとは | 23 |
| 4.2 ファイルシステムとファイルの操作 | 24 |
| 4.3 プロセスの操作 | 31 |
| 4.4 ジョブの操作 | 33 |
| 4.5 コマンドの入力 | 36 |
| 4.6 ワイルドカード文字 (ファイル名の展開) | 37 |

| | | |
|---------------|--|------------|
| 4.7 | ヒストリの表示 (history) | 41 |
| 4.8 | 標準入出力とリダイレクション・パイプ | 43 |
| 第 5 章 | エディタ | 47 |
| 5.1 | Emacs | 47 |
| 5.2 | Emacs での日本語入力方法 | 74 |
| 5.3 | vi | 77 |
| 5.4 | gedit | 78 |
| 第 6 章 | L<small>A</small>T<small>E</small>X による文書作成環境 | 79 |
| 6.1 | L <small>A</small> T <small>E</small> X とは | 79 |
| 6.2 | L <small>A</small> T <small>E</small> X を使ってみる | 79 |
| 6.3 | L <small>A</small> T <small>E</small> X の使い方 | 82 |
| 6.4 | AUC T <small>E</small> X | 88 |
| 第 7 章 | 図・グラフの作成 | 93 |
| 7.1 | 図を作成する - Tgif | 93 |
| 7.2 | グラフを作成する - gnuplot | 97 |
| 7.3 | 画像を表示/変換する - ImageMagick | 100 |
| 7.4 | 高度な画像編集を行う - GIMP | 101 |
| 第 8 章 | プログラミング | 105 |
| 8.1 | 前準備 | 105 |
| 8.2 | C 言語でのプログラミング | 105 |
| 8.3 | FORTRAN でのプログラミング | 107 |
| 8.4 | 最後に | 108 |
| 第 9 章 | インターネット | 109 |
| 9.1 | インターネットとは | 109 |
| 9.2 | ブラウザ | 110 |
| 9.3 | Mozilla Firefox | 110 |
| 9.4 | メール | 116 |
| 9.5 | WebMAIL | 117 |
| 9.6 | Mew | 118 |
| 9.7 | Wanderlust | 139 |
| 第 10 章 | アプリケーション | 143 |
| 10.1 | 数値解析 Mathematica | 143 |
| 10.2 | マルチメディアプレイヤ XMMS | 146 |
| 10.3 | マルチメディアプレイヤ RealPlayer | 146 |

| | |
|---|-----|
| 10.4 日本語コード変換 nkf | 148 |
| 第 11 章 シェルの設定 | |
| 11.1 シェル変数 | 149 |
| 11.2 環境変数 | 152 |
| 11.3 エイリアス (別名) の設定 (alias , unalias) | 153 |
| 11.4 .cshrc と .login | 154 |
| 11.5 ログインシェルの変更 | 158 |
| 11.6 その他のシェル | 159 |
| 第 12 章 GNOME | |
| 12.1 GNOME とは | 161 |
| 12.2 GNOME の各部説明 | 162 |
| 12.3 GNOME 上での日本語の扱いについて | 166 |
| 12.4 GNOME で困ったときは (ヘルプシステム) | 166 |
| 第 13 章 困ったときには | |
| 13.1 トラブルシューティング | 167 |
| 13.2 その他 Q&A | 173 |
| 付録 | |
| 付録 A リモートアクセス | 177 |
| 付録 B バックアップ | 184 |
| 付録 C ソフトウェア一覧 | 192 |
| 付録 D その他 , 知つておくべきこと | 195 |

第1章

はじめに

1.1 RAINBOW でできること

1.1.1 RAINBOW とは

立命館大学のコンピュータネットワークや多種多様のコンピュータ設備を立命館統合情報システム RAINBOW(Ritsumeikan Academic Information Network Bridging Our World) と呼びます。RAINBOW は、インターネットに接続されています。RAINBOW では実際どのようなことができるのでしょうか。いくつか代表的なものを挙げて説明していきたいと思います。

1.1.2 World Wide Web へのアクセス

World Wide Web とは、インターネット上に構築された世界規模の情報資源で、WWW もしくは単に Web(ウェブ)と呼ばれています。Web 上の情報を閲覧するためのソフトウェアを Web ブラウザと呼びます。Web ブラウザを利用して情報の閲覧することを Web ブラウジングと言い、インターネットの代表的な利用方法として急速に普及しています。

インターネットに接続された RAINBOW の端末から、世界中の膨大な情報を閲覧したり、入手したりすることができます。例えば、新聞社のホームページにアクセスすれば、新聞記事の電子版が読めますし、研究のために他大学から論文を手に入れることもできます。

その他にも、自分と同じ趣味の人の書いた Web ページを探したり、パソコンの見積書を注文したり、資料請求や企業へのエントリなど就職活動にも利用できます。また、Web ページを作成することで、世界に向けて自分自身が情報の発信者になることもできます。是非チャレンジしてみてください。

1.1.3 メールの送受信

RAINBOW では、インターネットを利用したサービスの中で、Web ブラウジングと並び、基本的なサービスの 1 つである、E メールを利用することができます。E メールとは、インターネットを利用して配達される手紙のようなものだと思ってください。

RAINBOW では、皆さんができる独自のユーザーアカウントを持ち、それに対応した“E メールア

ドレス”を持ちます。友人のEメールアドレスに対してEメールを送信することで、簡単に連絡を取ることができます。

メールならではのメリットとして、同じ内容を一度に複数の宛先に出すことができます。そのため、ゼミやコンパの連絡、旅行の打ち合せなどが簡単に行えます。さらに、メールアドレスを持つ世界中の人にメールを出すことができるようになります。卒業して離ればなれになった同窓生と同窓会の打ち合せもできますし、海外の研究者に直接質問をすることもできるでしょう。

1.1.4 レポートの作成

RAINBOWには \LaTeX という文書組版システムが用意されています。また、 Tgif という作図ツールや gnuplot というグラフ描画ツールも用意されています。

これらのツールを使用することで、ワープロソフトを利用しなくても簡単に美しくレポートを作成することができます。最初はワープロより難しいと感じるかもしれません、使い慣れてくると、章構成や、目次、索引、文献リストの作成などを自動生成してくれるため、非常に便利です。また、ワープロソフトでは、記述が難しい複雑な数式も簡単に美しく表現できます。このように、レポートの作成に大変向いていることが分かると、手放せなくなることでしょう。

使いこなせるようになると市販の書籍のような美しくデザインされた文書が作成できるようになります。本書の執筆・レイアウトも全て \LaTeX を利用しています。

1.1.5 プログラムの作成

RAINBOWでは、C,C++,Java,Perl,Python,Rubyなど、様々なプログラムを開発する環境が整えられています。プログラミングの授業に利用できるのはもちろんですが、RAINBOW環境にインストールされていない最新のフリーソフトのソースファイルを手に入れて、自分でコンパイルして利用することもできます。趣味でプログラムを作成して公開し、有名ソフトの作者として世界に名を馳せるのもよいでしょう。

1.2 Linux とは

1.2.1 Linux の特徴

RAINBOW の UNIX 環境では OS として Linux を利用しています。Linux は、当時ヘルシンキ大学の大学生であった Linus Torvalds によって開発された UNIX 互換カーネル（OS の本体）です。以下に、Linux の特徴についてまとめます。

- オープンソース

オープンソースは、誰でも改良や再配布を行うことができるライセンス形態です。そのため、オープンソースソフトウェアは、インターネットなどを通して、ソースコードが公開されています。Linux は代表的なオープンソースソフトウェアのひとつです。ソースコードが公開されているため OS の内部構造の学習にも適しています。

- UNIX 互換 OS

Linux は UNIX 互換の命令を備えています。そのため、ほかの UNIX 上で作られたソフトウェアのほとんどが簡単に移植できます。

- ディストリビューション

Linux 自体は OS のカーネルでしかありません。実用的に Linux 上で作業を行うためには、多くのソフトウェアをインストールする必要があります。この作業は複雑で面倒なため、Linux カーネルと様々なソフトウェアを組み合わせた，“ディストリビューション”がよく利用されます。ディストリビューションを利用することによって、簡単に実用的な Linux 環境が構築できます。有名なディストリビューションとしては、Fedora, Ubuntu, Vine Linux などがあります。

1.2.2 Vine Linux について

RAINBOW では、数あるディストリビューションの中から、Vine Linux を採用しています。以下に Vine Linux の特徴をオフィシャルページ^{*1}から引用しました。

Vine Linux は、使いやすい日本語環境を提供する Linux 配布パッケージ（ディストリビューション）です。インストールの直後から快適な日本語環境で作業ができるように、さまざまな配慮を行っています。

このように、Vine Linux は特に日本語環境に配慮したディストリビューションです。メンテナンスも日本人によって行われており、安心して日本語を利用することができます。RAINBOWにおいても、日本語のサポートが充実していることを理由に、Vine Linux を採用しています。

^{*1} <http://www.vininux.org/>

1.3 ユーザーアカウントとパスワード

コンピュータの世界では、コンピュータを使う人のことをユーザーと呼びます。ユーザーがコンピュータを利用する権利のことをユーザーアカウントまたはアカウントと呼びます。立命館大学では、入学時に RAINBOW を利用するためのユーザーアカウントを発行しています。ユーザーアカウントは RAINBOW のサービスやコンピュータを利用するためのユーザー ID とパスワードから構成されます。ユーザー ID はコンピュータにログインするときに、自分が誰であるかを知らせるために用います。パスワードはコンピュータを利用する場合に(ログインするときに)本人であることを証明するためのものです。

パスワードはシステムと共有の秘密です。キャッシュカードにおける暗証番号にあたります。暗証番号を他人に教えないことと同じで、パスワードを他人に教えてはいけません。当然ですが、メールでパスワードを流すことも絶対にしないでください。パスワードが漏洩してしまうと、システムのクラッキングやプライバシー情報の漏洩といった問題につながってしまうおそれがあります。そのため、パスワードの管理は徹底して行い、定期的に変更するようにしてください。

1.3.1 パスワードの変更

変更方法

パスワードを変更するには `passwd` というコマンドを使います。

```
% passwd  
Changing password for user ***** 自身のユーザーIDが表示されます  
Enter login(LDAP) password: oldpasswd 現在のパスワードを入力します  
New password: newpasswd 新しいパスワードを入力します  
Retype new password: newpasswd もう一度新しいパスワードを入力します
```

このとき、

- 新しいパスワードは二度入力する
- 入力したパスワードは一切画面には表示されない

ことに注意してください。そして画面に

```
LDAP password information changed for *****  
passwd: all authentication tokens updated successfully.
```

と表示されればパスワードが変更されたことになります。エラーメッセージが表示されたときには、再度 `passwd` を試してみてください。

1.3.2 変更時の注意

強いパスワード・弱いパスワード

世の中には、パスワード解析しようとする悪意あるプログラムが存在します。そういうたったプログラムの対策をするために、パスワードを変更する際には、強いパスワードにする必要があります。

では、実際にどのようなパスワードが強い（破られにくい）のか、また弱い（破られやすい）のか、以下に例を挙げます。

- 強いパスワード（推測されにくいパスワード）
 - 大文字と小文字がランダムに混ざっている
 - 文字の他に数字や記号が含まれている
 - 長さが 8 文字以上
- 弱いパスワード（推測されやすいパスワード）
 - ユーザー名と同じ
 - 自分の名前
 - 親族、知人、有名人の名前
 - 自分の電話番号や誕生日
 - 入手しやすい自分に関する情報
 - 地名
 - 固有名詞
 - 辞書に載っている言葉
 - 全部同じ文字
 - 以上のパスワードの綴を逆にしたもの
 - 以上のパスワードの前後に数字をつけたもの

以上の全てを満たすパスワードを考えましょう。

定期的に変更する

強いパスワードにしたからといって安心しないで、定期的にパスワードを変更することも重要です。また、ログイン履歴を確認して、おかしいと思ったらすぐにパスワードを変更してください。

1.4 RAINBOW のコンピュータを利用する上での注意事項

RAINBOW のコンピュータは単独で動作しているわけではなく、ネットワークを通じて多くのコンピュータとつながって動作しています。したがって、あるコンピュータの障害が、他のコンピュータや他のユーザーに影響を与えることがあります。お互いに気持よく RAINBOW を利用するために、UNIX の使用において必ず守ってほしいことを以下に書きますので、協力をお願いします。

- いきなり電源を切らない

UNIX では、ファイルに対する変更をすぐにディスクに書き込むとは限りません。頻繁なディスクアクセスを避けるために、アクセス速度の速いメモリ上にだけ書き込み、一定の時間経過後などにディスクに書き込み、整合性を取ります。

いきなり電源を切ると、メモリ上にだけある最新の情報をディスクに書き込む暇がなく、ファイルの整合性が取れなくなります。最悪の場合はファイルが破損したり、ファイルがなくなったりするかもしれません。

- ログアウトを忘れずに

RAINBOW のコンピュータを利用したあと、退出の際には必ずログアウトを確認してください。ログアウトをしないまま放置しておくと、誰かがあなたの名前を偽って他の人にメールなどでいたずらしたり、場合によっては取り返しのつかない事件になるかも知れません。

- トラブルが発生したら

何かトラブルが発生した場合には、まず RAINBOW GUIDE の FAQ を読んでみてください。それでも解決しない場合は、必ず最寄りの RAINBOW STAFF か RAINBOW サービスカウンターまで連絡してください。

1.5 本書の内容

基本的な UNIX の操作については、基本編にあたる章に目を通すことで、理解できます。より発展した操作や、環境のカスタマイズを行いたい場合は、応用編に目を通してください。

準備編

第 1 章 はじめに RAINBOW 環境を利用する上で知っておくと良い事項と、利用する上での注意事項について説明しています。

第 2 章 目的別リファレンス RAINBOW 環境で行えることを、目的別に分けて説明しています。この章に目を通すことによって、自分の目的にあった RAINBOW GUIDE の読み方ができます。

基本編

第 3 章 基本操作 RAINBOW の UNIX 環境を操作する際の基本的な操作方法について、説明しています。この章の内容を習得することで、以降の章の内容がよりわかりやすくなります。

第 4 章 シェル RAINBOW GUIDE はシェルによる操作を中心に解説しています。本章では、シェルを用いたファイル操作方法について説明しています。

第 5 章 エディタ UNIX で広く利用されているテキストエディタである Emacs について説明しています。

第 6 章 L^AT_EX による文書作成環境 大学のレポートを美しく仕上げるために文書組版システム L^AT_EX について説明しています。

第 7 章 図・グラフの作成 L^AT_EX で作成したレポートに図やグラフを挿入したい場合に利用するアプリケーションの使い方と L^AT_EX 文書への図の挿入の方法について説明します。

応用編

第 8 章 プログラミング RAINBOW の UNIX 環境で利用できるプログラミング言語についてサンプルプログラムとともに簡単に説明します。

第 9 章 インターネット Web へのアクセスやメールの利用の仕方について説明します。

第 10 章 アプリケーション 上で説明したもの以外で便利なアプリケーションについて紹介します。

第 11 章 シェルの設定 4 章で説明した“シェル”をより利用しやすくするための設定の仕方について説明しています。

第 12 章 GNOME RAINBOW の UNIX 環境の標準 GUI 環境である GNOME について説明します。

第2章

目的別リファレンス

2.1 レポートを作成したい

UNIXを取り扱う授業などで, L^AT_EXでのレポート作成が必須とされています。L^AT_EXを利用した文書の作成では、まず L^AT_EX の記述規則に基づいた tex ファイルを作成します。この作業は、エディタを使用したテキストファイルの作成と同様の方法で実現できます。次に、完成した tex ファイルを plateX というコマンドライン上のツールを利用して、印刷用の原稿を作成します。その際にシェルの知識が求められます。各種操作についての詳細は、下記を参照してください。

シェルの操作

テキストエディタ Emacs の起動や、L^AT_EX 文書のコンパイル、文書のプレビュー、ファイル操作などでシェルの知識が必要なります。ほとんどの操作がコマンドラインで行われます。シェルの操作については、

第4章 シェル

をお読みください。

エディタの操作

レポートはエディタを使って記述します。一般的な UNIX では、Emacs が利用されています。Emacs の操作方法については、

第5章 エディタ

をお読みください。

L^AT_EX を使った文書の作成

L^AT_EX は独自の文法体系に基づいた記述をすることで、章構成、段落や目次などを自動で作成してくれます。習得すれば、出版物相当のものが作成できます。L^AT_EX については

第6章 L^AT_EX による文書作成環境

をお読みください。

図・グラフの作成

レポートを作成する際に、図やグラフなどをレポート中に記述することが必要な場合があります。RAINBOW 環境では、図作成ツールとして Tgif、グラフ描画ツールに gnuplot を提供しています。各ツールの使い方については

第7章 図・グラフの作成

をお読みください。

2.2 プログラムを作成したい

RAINBOW 環境では、様々なプログラミング言語がサポートされており、実際にプログラムを作成する環境も用意されています。プログラミングでは、まず、ソースコードと呼ばれるプログラミング言語が記述されたファイルを作成します。次に作成されたソースコードをコンパイラと呼ばれるソフトウェアを利用して、実行可能なプログラムに変換します。この作業過程の中で、コマンドを使用するシェルの操作や、エディタを使用したファイルの編集作業が行われます。各種ツールやコマンドの操作についての詳細は、下記を参照してください。

シェルの操作

プログラムのコンパイルや実行にシェルの操作が必要となります。シェルの操作については、

第4章 シェル

をお読みください。

エディタの操作

プログラムはエディタを使って記述します。エディタについては、

第 5 章 エディタ

をお読みください。

プログラミング

プログラムはプログラミング言語を利用して記述します。プログラミング言語については

第 8 章 プログラミング

をお読みください。

2.3 インターネットをしたい

RAINBOW の環境は、インターネットにつながっており、様々なサービスが利用できます。UNIX 環境でもメールや Web ブラウジングが行えます。インターネットについては、

第 9 章 インターネット

をお読みください。

2.4 Linux 環境をカスタマイズしたい

Linux は、自分にとって使いやすいうようにカスタマイズすることができます。

シェルの設定

シェルでは、自分自身でコマンドのエイリアスを作成したり、パラメータを設定することで柔軟な設定を行うことができます。シェルの設定については、

第 11 章 シェルの設定

をお読みください。

外観の設定

起動後の外観も使いやすい環境づくりにとっては、重要です。RAINBOW で採用している GNOME 環境では、壁紙の変更や外観の変更が可能です。GNOME の設定については

第 12 章 GNOME

をお読みください。

第3章

基本操作

3.1 ログインとログアウト

3.1.1 アカウント・パスワードとは

UNIX を基本とするシステムを利用する人は、必ずアカウントを持ちます。アカウントはユーザー名とパスワードの 2 つからなります。ユーザー名^{*1}は識別子であり、コンピュータにログインするときに、自分が誰であるかを知らせるために用います。パスワードはコンピュータを利用する場合に、本人であることを証明するためのものです。

パスワードはシステムと共有の秘密です。いわばキャッシュカードにおける暗証番号にあたります。暗証番号を他人に教えないのと同じで、パスワードを他人に教えてはいけません。当然ですが、メールでパスワードを流すことも絶対にしないでください。

3.1.2 ログイン

RAINBOW 環境の Vine Linux を起動すると図 3.1 のような画面が表示されます。それではログインしてみましょう。

まず、画面の指示に従いユーザー名を入力し Enter キーを押します。続いてパスワードを入力し Enter キーを押します。このとき、パスワードは画面には表示されません。画面を肩越しに盗み見られても、パスワードを他人に知られないようにするためです。ユーザー名とパスワードが正しければ、システムにログインできます。もし間違っていた場合は、再度ユーザー名とパスワードを入力します。

3.1.3 ログアウト

ログアウトを行わずにコンピュータを放置すると、他人に秘密のファイルを見られたり、大切なファイルを削除されたりする危険性があります。長時間席を立つときは、忘れずにログアウトをしましょう。

^{*1} ユーザー ID、アカウント名または、ログイン名とも呼ばれます。

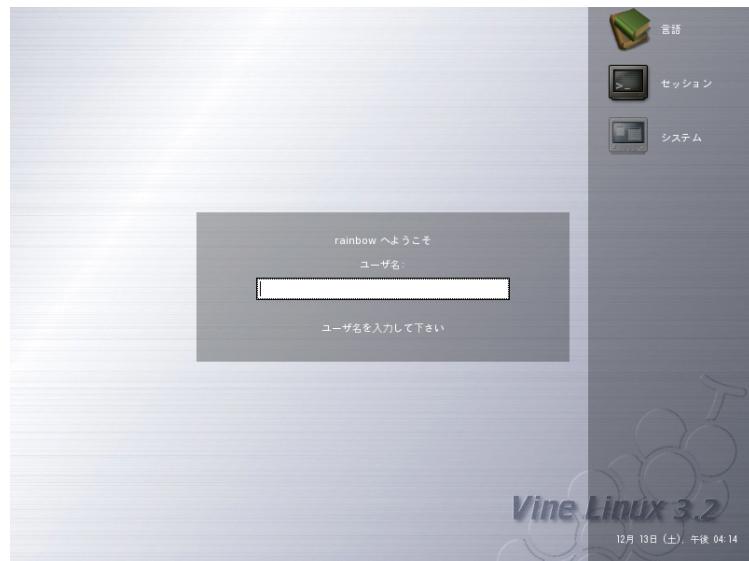


図 3.1 ログイン画面

GNOME でログアウトをするには、図 3.2 の左上にある“アクション”ボタンを押し、“ログアウト”を選択します。ログアウトするか、あるいは電源を切るかを確認するメッセージが表示されますので、ログアウトを選択し、“OK”をクリックしてください。



図 3.2 ログアウト

3.1.4 シャットダウン

GNOME でコンピュータをシャットダウンするには“アクション”ボタンを押し“ログアウト”を選択します。そしてアクションの中からシャットダウンを選択し、“OK”をクリックします。

ログイン画面からコンピュータをシャットダウンしたい場合は“システム”を選択し、アクションの中から“コンピュータを停止”を選択し“OK”をクリックします。

3.2 GNOME メニューからのアプリケーションの起動

3.2.1 GNOME 端末の起動

UNIX を使う上で重要なことは、シェル上でのコマンド操作です。ここでは最初にシェルを立ち上げるためにウインドウである GNOME 端末^{*2}を起動してみましょう。

まず、デスクトップの上部にある“メニュー・バー”の中から“アプリケーション”を左クリックします。するとサブメニューが現れるのでその中から“システム・ツール”を選び、“GNOME 端末”的項目を左クリックします。



図 3.3 GNOME 端末

“GNOME 端末”というタイトルのウインドウが現れ、その中にシェルのプロンプトが表示されます。これで GNOME 端末を使える状態になります。

3.2.2 他のアプリケーションの起動

RAINBOW 環境にはエディタ Emacs や Web ブラウザ Firefox など様々なアプリケーションがインストールされています。これらも GNOME 端末と同様に“メニュー・バー”から起動することができます。

^{*2} GNOME 用ターミナルエミュレータ

3.2.3 日本語の入力

GNOMEにおける標準日本語入力システムとしてSCIMが採用されています。SCIMでは全角/半角キーを押すことによって、日本語の入力が可能な状態になります。

3.3 GNOME 端末からの起動

3.3.1 コマンドの実行

まず3.2.1章のようにGNOME端末を開いてみましょう。

GNOME端末上でコマンド名を打ち、Enterキーを押すことによって各種コマンドを実行することができます。まずははじめにここではejectコマンドを実行してみましょう。下のようにGNOME端末上で“eject”とキーボードを打ち、Enterキーを押してください。

```
% eject
```

ejectコマンドが実行され、コンピュータのCDトレイが開かれたと思います。“コマンドが見つかりません。”と表示される場合はejectの綴りを確認してください。

3.4 使ってみよう

RAINBOW環境には様々なエディタがインストールされていますが、ここではその中でもEmacsを使ってみましょう。Emacsを起動するにはGNOME端末でemacsコマンドを実行します。

```
% emacs a.txt
```

コマンドのオプションには開くファイル名を指定します。ここでは“a.txt”と言うファイルを開いています。

Emacsが起動すると図3.4のようなウインドウが開かれます。

それでは文章を書いてみましょう。注意すべき点として、Emacsでは半角/全角キーによって日本語入力を切り替えることができません。日本語の入力を行いたいときは、Emacsが立ち上がった状態でCtrlキーを押しながら¥を入力します。ウインドウの下の方に<Anthy: あ>と表示されると、日本語の入力が可能になります。もう一度同じことを行うと英字入力に戻ります。そして、漢字変換はローマ字打ちで日本語に変換したい言葉を入力した後、希望の漢字になるまでSpaceキーを押して文字変換します。Enterキーを押すとその漢字に確定します。

編集したファイルを保存するにはEmacsのツールバーにある保存アイコンをクリックするか、Ctrlキーを押しながらx,sの順番でキーを押します。

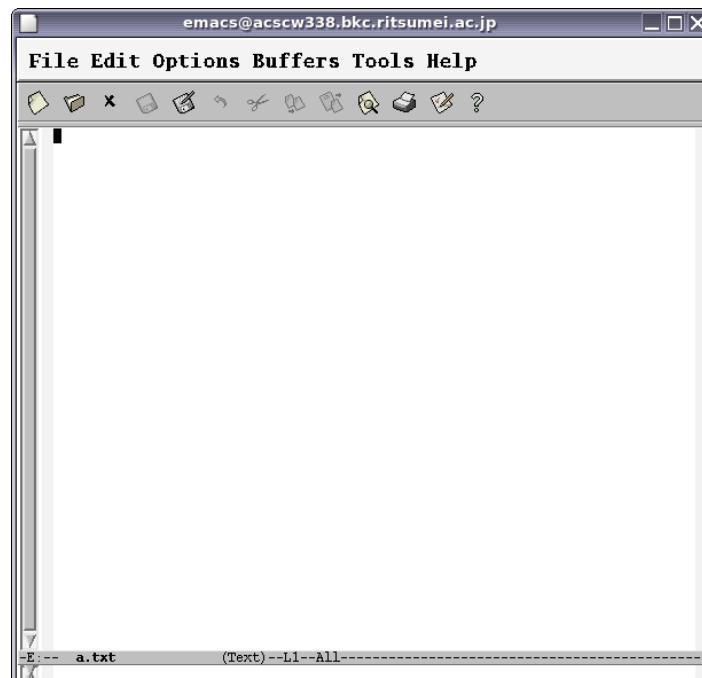


図 3.4 Emacs

次に作成した文章を印刷してみましょう。Emacs 上で編集中のファイルを印刷するにはツールバー上のプリンタアイコンをクリックします。最後に Emacs を終了するにはウインドウの右上の×ボタンをクリックします。Emacs についてのさらに詳しい使い方は第 5 章を参照してください。

3.5 USB メモリへの保存

最後に作成した文章を USB メモリに保存してみましょう。

3.5.1 マウント

Linux で USB メモリにアクセスする場合は通常、マウントと呼ばれる操作が必要です。RAINBOW 環境で USB メモリのマウントを行うには `usb-mount` コマンドを使用します。

```
% usb-mount
```

実行し、デスクトップ上にアイコンが表示されれば成功です。/mnt 以下の `usb1`, `usb2`, `usb3`, `usb4` のいずれかのディレクトリに USB メモリがマウントされています。

```
% ls /mnt/usb1  
% ls /mnt/usb2  
% ls /mnt/usb3  
% ls /mnt/usb4
```

3.5.2 ファイルのコピー

次に、先ほど作成したファイルを USB メモリにコピーします。

```
% cp a.txt /mnt/usb1
```

/mnt/usb1 の部分は usb[1-4] の中でファイルが表示されたものに変更してください。

3.5.3 アンマウント

最後に USB メモリを取り出すときは必ず、

```
% usb-umount
```

を実行しアンマウントした後に取り出してください。

3.6 印刷

3.6.1 プリントアウト

プリントアウトには `lp` コマンドを用います。例えば、`file_name` という名前のファイルを印刷するには、

```
% lp file_name
```

と入力します。また、ファイル名を省略した際には標準入力^{*3} の内容を印刷します。

テキストファイルや画像ファイルなどファイルにはさまざまな形式がありますが、多くのファイルは `lp` コマンドで印刷することができます。しかし、PDF ファイル・DVI ファイルについては `lp` コマンドで直接印刷できません。PDF ファイルの印刷方法については 3.6.4 節を、DVI ファイルの印刷方法については 6.2 節を参照してください。

また、`-d` オプションを用いることでプリンタを指定することができます。例えば、`printer_name` というプリンタで印刷がしたい場合、

^{*3} 標準入力については 4.8.1 節を参照してください。

```
% lp -d printer_name file_name
```

となります。

3.6.2 プリンタの状態とプリント要求の取消

プリント要求は、それぞれに対し番号が割り当てられ、キュー（待ち行列）に入れられます。lpstat コマンドでこのキューの状態を見ることができます。

```
% lpstat  
lp-91           rs012345      22528   2008年12月24日 10時23分40秒  
lp-92           rs012345      1540    2008年12月24日 10時24分45秒
```

一旦出したプリント要求を、取り消したい場合があるかもしれません。そのようなときには、lpstat コマンドで取り消したい要求のジョブ番号を調べ、cancel コマンドを実行します。ここで、上で表示されている 91 を取り消したいときは、

```
% cancel 91
```

と入力します。

なお、これによって取り消したいプリント要求がキューから削除されます。ただし取り消せるものは、自分で印刷を指示したものに限ります。ジョブがプリンタまで送られてしまうと、cancel では中断できません。

3.6.3 ツールを使った印刷

ここでは、PostScript ファイルを様々な形式で印刷するためのツール PSUtils について紹介します。

- psselect - 印刷のページの指定

psselect コマンドを使うことで、PostScript ファイルの指定したページだけを印刷することができます。例えば、2, 4, 5 ページだけを抜き出して印刷したい場合は、次のようにします。

```
% psselect -p2,4,5 file_name.ps | lp
```

また、次のようにして、2 ページ目から 4 ページ目までを印刷することもできます。

```
% psselect -p2-4 file_name.ps | lp
```

また、奇数ページ、偶数ページだけを印刷することもできます。奇数ページを印刷する場合は、-o

オプションを指定します。

```
% psselect -o file_name.ps | lp
```

偶数ページを印刷する場合は、-e オプションを指定します。

```
% psselect -e file_name.ps | lp
```

- **psnup** - 複数ページを一枚に印刷

psnup コマンドを使うと、2 ページ分を縮小して 1 枚の紙に印刷することができます。

```
% psnup -2 file_name.ps | lp
```

- **psbook** - ページの配置操作

psbook を使うと、印刷したものを綴じて冊子を作ることができます。まず、

```
% psbook file_name.ps | psnup -2 | lp
```

として印刷してください。次に、印刷した紙を二枚を組にして、コピー機で両面印刷します。そして真中を綴じれば、冊子になります。

これまでに紹介したコマンドは、PSUtils のほんの一部です。そのほかにも EPS ファイルの大きさを調整する **epsffit** コマンドや、複数の PostScript ファイルを結合させるための **psmerge** コマンドなどがあります。

3.6.4 PDF ファイルの閲覧と印刷

PDF (Portable Document Format) は、米 Adobe Systems 社が開発した電子文書形式です。PDF は特定のプラットフォームに依存せず、同じ見栄えで文書を表示できます。そのため、インターネットでの文書配布用フォーマットとして広く用いられており、また企業内での書類の電子化や、広告の配布に利用されるケースが増えてきています。

ここでは、Adobe Reader を利用して PDF ファイルを閲覧、印刷する方法について説明します^{*4}。
file_name.pdf という名前の PDF ファイルを閲覧するには次のように入力します。

```
% acroread file_name.pdf
```

^{*4} PDF ファイルを閲覧するためのコマンドとして、他に **xpdf**, **gv** などがあります。

閲覧している PDF ファイルを印刷したい場合は、メニューバーから [ファイル (F)]-[印刷 (P)] を選択します。図 3.5 のようなウインドウが開きます。

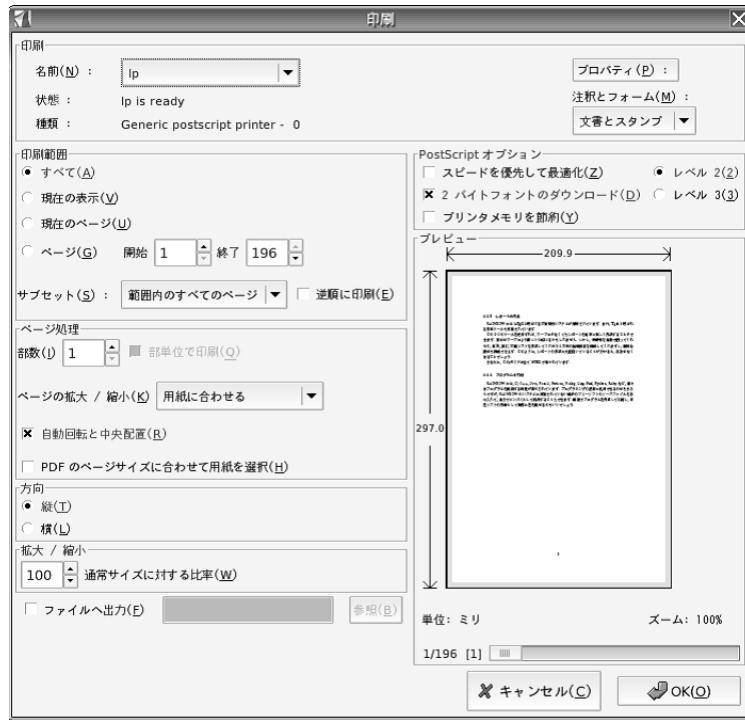


図 3.5 印刷画面

PostScript ファイルとして出力したい場合は [ファイルへ出力] にチェックを入れます。

あとは自分の好みにあわせて印刷範囲やページ処理を設定した後、OK ボタンを押すと印刷されます。

第4章

シェル

この章では、まずシェルとはどのようなものかについて説明し、そして具体的な例を紹介しながらシェルの便利な機能について説明していきます。シェルにはいくつかの種類がありますが^{*1}、RAINBOW 環境での初期設定では tcsh を使うようになっていますので^{*2}、この tcsh について説明します。また tcsh の設定ファイルの内容、およびログインシェルを他のシェルに変更する方法については、第 11 章シェルの設定で説明します。またこれ以降シェルのキー操作を説明するために、Ctrl キーを押しながら文字キー(x)を押すことを、“Ctrl-x”と表記します。

4.1 シェルとは

シェルとは、一体どのようなものなのでしょうか。実は、皆さんはコンピュータにログインしてから、今までずっとシェルを使ってきていたのです。これまでに基本的なコマンドについて説明してきましたが、これらのコマンドを実行する際には、“%”の文字（プロンプト^{*3}）が表示されている後にコマンドを入力して、Enter キーを押しました。すると、コマンドが実行され、また “%” の文字が表示され… という繰り返しだったと思います。このように、我々はコンピュータと対話的に作業を進めていきます。このとき、“%” の文字を表示してコマンドを入力するように促しているのは、実は ls や cat と同じく、シェルという 1 つのプログラムだったのです。私たちユーザーは、直接オペレーティングシステム (OS) とやりとりをしているのではなく、シェルがユーザーと OS の仲介をしているのです。

シェルは英語で書けば shell です。シェルというプログラムは貝殻のように OS を覆い隠して、我々ユーザーがコンピュータを使いやすいように、OS では提供できないユーザーに対するきめ細かなサービスを提供しています。それではこれから、シェルが提供している便利な機能を説明していきます。

^{*1} sh, csh, ksh, tcsh, bash, zsh など。それぞれ、B シェル、C シェル、K シェル、TC シェル、バッシュ、Z シェルなどと読まれることが多いようです。

^{*2} 初期設定のシェルは /bin/csh が登録されていますが、RAINBOW 環境における Linux 環境では、これは実際には csh ではなく tcsh となります。なお、Solaris など一部の環境では、/bin/csh が本当に csh である場合があります。echo \$tcsh として、tcsh: Undefined variable. とのエラーメッセージが表示された場合は、tcsh ではなく csh が動作しています。注意してください。

^{*3} 英語で prompt。コマンドの入力を促していることからこう言われます。表示する文字 “%” は、設定をカスタマイズすることによって自由に変更できます。

4.2 ファイルシステムとファイルの操作

4.2.1 ファイルとディレクトリ

UNIX システム上で扱われるプログラムやデータなどは、ファイル (file) という形態で管理されています。ファイルはユーザーには連續したバイト列に見えます。

| | | | | | | | | | | | | |
|---|---|---|---|---|---|--|---|---|---|---|---|---|
| H | e | l | l | o | , | | w | o | r | l | d | . |
|---|---|---|---|---|---|--|---|---|---|---|---|---|

ファイルには名前 (ファイル名) が付けられ、ファイルへのアクセスはファイル名を指定することで行います。

次の例は、hello.c というファイルの内容を表示したものです。ファイル内容を表示するには、cat というコマンドを使います。% はプロンプト (prompt) と呼ばれるもので、コマンドの入力が可能な状態であることを示しています。

```
% cat hello.c
#include <stdio.h>
main()
{
    printf("Hello, world.\n");
}
```

ファイルの特殊なものとして、ディレクトリファイル (directory file) があります。ディレクトリファイルは、他のファイルを参照するための情報を格納しているファイルです。UNIX のファイルシステムは、ディレクトリを用いることによって木構造を形成しています。木構造の根 (root) は特にルートディレクトリと呼ばれ、/ (スラッシュ) で表されます。

図 4.1 は、ファイルとディレクトリの概念図です。この図は、ルートディレクトリの下に 2 つのディレクトリ dir1, dir2 があり、dir1 の下にファイル file1, file2 が、dir2 の下にディレクトリ dir3 とファイル file3, file4 があることを示しています。この図における dir2 と dir3 のように、2 つのディレクトリ間に上下関係がある場合、dir2 は dir3 の親ディレクトリ であると呼びます。逆に、dir3 は dir2 のサブディレクトリであると呼びます。

4.2.2 ディレクトリ操作

この節ではディレクトリに対して操作を行う表 4.1 のコマンドを説明します。

カレントディレクトリの表示 (pwd)

ユーザーが現在作業をしているディレクトリをカレントワーキングディレクトリ (current working directory: cwd) またはカレントディレクトリ (current directory) と呼び、pwd というコマンドを用いることによって得ることができます。

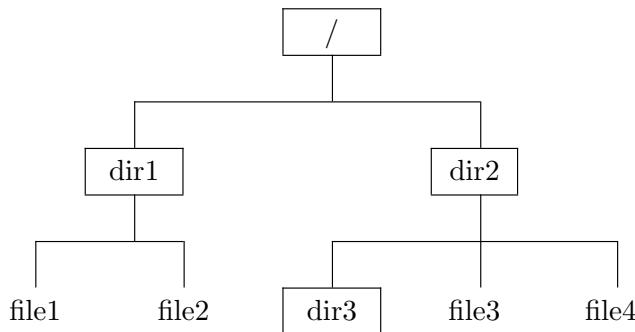


図 4.1 ファイルとディレクトリ

表 4.1 ディレクトリ操作コマンド

| コマンド名 | 内容 |
|-------|---------------|
| pwd | カレントディレクトリの表示 |
| cd | カレントディレクトリの変更 |
| mkdir | ディレクトリの作成 |
| rmdir | ディレクトリの削除 |
| ls | ファイルリストの出力 |

```
% pwd
/homer/stu0/ra012345
```

この例では、現在作業中のディレクトリは /homer/stu0/ra012345 です。これは、ルートディレクトリ / の下の homer というディレクトリの下の stu0 というディレクトリの下の ra012345 というディレクトリを表しています。このように、親子関係にあるディレクトリやファイルは / で区切って表示されます。境界を表すこの記号を、デリミタ (delimiter) と呼びます。

カレントディレクトリの変更 (cd)

カレントディレクトリの変更 (移動) には、cd というコマンドを使います。

```
% cd /
% pwd
/
% cd /homer/stu0/ra012345          ルートディレクトリに変更
% pwd                                絶対パスで指定
/homer/stu0/ra012345
```

最上位のルートディレクトリからの位置を表したディレクトリを絶対パス (absolute path) といい、カレントディレクトリからの相対的な位置を表したディレクトリを相対パス (relative path) と呼びます。

特殊なディレクトリとして，“.”，“..”があります。“.”はドット(dot)と呼ばれ，カレントディレクトリを表しています。“..”はドット-ドット(dot-dot)と呼ばれ，現在のディレクトリに対する親ディレクトリ^{*4}を表しています。

```
% pwd
/homer/stu0/ra012345
% cd ..
% pwd                               親ディレクトリに移動
/homer/stu0
% cd ./ra012345                   相対パスで指定
% pwd
/homer/stu0/ra012345
```

ディレクトリの作成 (mkdir)

ディレクトリを作成するには，mkdirというコマンドを使います。

```
% ls
Mail
% mkdir pascal                     pascalというディレクトリを作成
% ls
Mail  pascal
```

ディレクトリの削除 (rmdir)

ディレクトリを削除するには，rmdirというコマンドを使います。ただし，中身が空のディレクトリしか削除できません。^{*5}

```
% ls
Mail  pascal
% rmdir pascal
% ls
Mail
```

ファイルリストの出力 (ls)

ディレクトリ内にあるファイルのリストを出力するには，lsというコマンドを使います。lsは，カレントディレクトリだけでなく，ファイルやディレクトリを指定して，それらの情報を出力することができます。出力結果の一例を示します。

^{*4} カレントディレクトリがルートディレクトリの場合は，特別に自分自身になります。

^{*5} 空ではないディレクトリを削除したい場合は rm コマンドを使用してください。

```
% ls
Mail
% ls -a
. .Xauthority .cshrc .login .xsession
.. .Xdefaults .emacs .twmrc Mail
% ls -al
合計 80
drwxr-xr-x 35 ra012345 student 4096 Feb 1 01:17 .
drwxr-xr-x 20 root sys 4096 Jan 10 12:32 ..
-rw----- 1 ra012345 student 554 Feb 1 01:16 .Xauthority
-rw-r--r-- 1 ra012345 student 360 Jan 29 13:20 .Xdefaults
-rw-r--r-- 1 ra012345 student 1144 Feb 2 01:30 .cshrc
-rw-r--r-- 1 ra012345 student 3202 Jan 23 03:28 .emacs
-rw-r--r-- 1 ra012345 student 233 Feb 10 02:30 .login
-rw-r--r-- 1 ra012345 student 2474 Jan 5 17:10 .twmrc
-rwxr-xr-x 1 ra012345 student 229 Feb 15 04:01 .xsession
drwx----- 2 ra012345 student 4096 Feb 1 13:59 Mail
```

ls コマンドにはさまざまなオプションがあります。例えば、ファイル名が“.”から始まるファイル、ディレクトリは単に ls を実行しただけでは表示されませんが、-a オプションを指定すると、これらのファイルも表示することができます。また、-l オプションを指定すると、詳細な情報を得ることができます。-l オプション指定時の実行結果は、次の形式で出力されます。

```
-rw-r--r-- 1 ra012345 student 3202 Jul 7 2000 hello.c
```

これは 表 4.2 に示す情報を表しています。

表 4.2 ファイルの情報の説明

| | | |
|-------|---------|----------------|
| 許可モード | 所有者 | 読み書き可 (rw) |
| | グループメンバ | 読み込み可 (r) |
| | その他 | 読み込み可 (r) |
| リンク数 | | 1 |
| 所有者 | | ra012345 |
| グループ | | student |
| サイズ | | 3202 バイト |
| 最終更新日 | | 2000 年 7 月 7 日 |
| ファイル名 | | hello.c |

4.2.3 ファイル操作

この節ではファイルに対して操作を行う表 4.3 のコマンドを説明します。

表 4.3 ファイル操作コマンド

| コマンド名 | 内容 |
|-------|-----------------|
| chmod | ファイルへのアクセス許可の変更 |
| cp | ファイルのコピー |
| rm | ファイルの削除 |
| mv | ファイルの移動・ファイル名変更 |

ファイルへのアクセス許可の変更 (chmod)

ファイルやディレクトリには許可モードという属性があります。これは、そのファイルにアクセス（読み、書き、実行）できるユーザーを制限するのが目的です。例えば、メールの内容などは個人のプライバシーに関係するものであるため、所有者のみ読み取りと書き込みが可能というモードに設定するのが一般的です。

ls -l コマンドで得られた情報が、

```
-rwxr-xr-x 1 root      root      78460 Jun 25 2001 patch
```

であるとき、このファイルの許可モードは表 4.4 のようになっています。

表 4.4 許可モードの説明

| | 読み取り (r) | 書き込み (w) | 実行 (x) |
|---------|----------|----------|--------|
| 所有者 | 許可 | 許可 | 許可 |
| グループメンバ | 許可 | 不許可 | 許可 |
| 他のユーザー | 許可 | 不許可 | 許可 |

許可モードを変更するには、chmod というコマンドを使います。このコマンドの書式は次のようになっています。

chmod [オプション] モード ファイル

モードは、次の表 4.5 に示す 8 進数の論理和^{*6} で指定します。

表 4.5 8 進数での許可モードの指定

| | |
|-----|----------------|
| 400 | 所有者は読み取り可 |
| 200 | 所有者は書き込み可 |
| 100 | 所有者は実行可 |
| 040 | グループメンバは読み取り可 |
| 020 | グループメンバは書き込み可 |
| 010 | グループメンバは実行可 |
| 004 | その他のユーザーは読み取り可 |
| 002 | その他のユーザーは書き込み可 |
| 001 | その他のユーザーは実行可 |

^{*6} この他にも、現在の許可モードに読み取り許可を追加するのに、`chmod +r filename` という書式で、8 進数を使わずに指定する方法もあります。

例として、.cshrc というファイルの許可モードを変更してみます。

1. 許可モードを調べる
2. 所有者のみ読み取り可に変更する
3. 元に戻す

```
% ls -l .cshrc
-rw-r--r-- 1 ra012345 student      329 Feb 1 01:16 .cshrc
% chmod 400 .cshrc
% ls -l .cshrc
-r----- 1 ra012345 student      329 Jan 1 01:16 .cshrc
% chmod 644 .cshrc
% ls -l .cshrc
-rw-r--r-- 1 ra012345 student      329 Feb 1 01:16 .cshrc
```

ファイルのコピー (cp)

ファイルの複製を作成するには、cp というコマンドを使います。

```
% ls
test1.p
% cp test1.p test2.p
% ls
test1.p  test2.p
```

また、-r オプションをつけて実行することで、ディレクトリを中のファイルごとまとめて複製することもできます。

```
% mkdir pascal
% cp test1.p pascal
% ls pascal
test1.p
% cp -r pascal modula2
% ls modula2
test1.p
```

ファイルの削除 (rm)

ファイルを削除するには、rm というコマンドを使います。

```
% ls
modula2  pascal  test1.p  test2.p
% rm test1.p
% ls
modula2  pascal  test2.p
```

また，-r オプションをつけて実行することで，ディレクトリの中のファイルごとまとめて削除することができます。ただし，操作を誤ると多量のファイルを消してしまうことになるため，注意して使ってください。

```
% rm -r pascal  
% ls  
modula2 test2.p
```

ファイルの移動・ファイル名変更 (mv)

ファイルを移動するには，mv というコマンドを使います。また，移動先ファイルを別のファイル名にすることにより，ファイル名を変更することができます。

```
% ls  
modula2 test2.p  
% mv test2.p test3.p  
% ls  
modula2 test3.p
```

また，ディレクトリもファイルと同様に，mv で移動および名前変更ができます。

4.3 プロセスの操作

4.3.1 プロセスとは

プロセスとは，いったいどのようなものでしょうか。それを知るために，まず最初にプロセスの状態を画面に表示させてみましょう。プロセスの状態を表示させるには，ps コマンドを使います。

```
% ps  
 PID TTY      TIME CMD  
 591 pts/1    00:00:05 tcsh  
 691 pts/1    00:00:01 xclock  
 896 pts/1    00:00:00 ps  
 747 pts/1    00:00:01 xterm
```

ps コマンドを実行すると，CMD と書かれた欄に，現在実行しているプログラム名が表示されました。つまり，プロセスとは，プログラムを実行する単位のことだったのです。あるプログラムを実行すると，プロセスが作られます。みなさんは，自分で作ったプロセスを操作することができます。それでは，実際にプロセスを操作(生成，表示，終了)してみましょう。

4.3.2 プロセスを作る

それでは、プロセスを作成してみましょう。プロセスを作るのは、非常に簡単です。何かプログラムを実行すればいいのです。

```
% ls
```

例えば、この場合、プロセスが作られ、ls というプログラムが実行されます。プログラムの実行が終了すると、作られたプロセスも終了します。

4.3.3 プロセスを表示する (ps)

ps コマンドを使って、もう一度プロセスを表示させてみましょう。

```
% ps
 PID TTY      TIME CMD
 591 pts/1    00:0:05 tcsh
 691 pts/1    00:0:01 xclock
 896 pts/1    00:0:00 ps
 747 pts/1    00:0:01 xterm
```

このように、ps コマンドは、プロセス ID(PID) とプログラム名 (CMD) を表示します。プロセス ID とは、プロセスに付与された番号のことです。みなさんは、このプロセス ID を使ってプロセスを操作することができます。

4.3.4 プロセスを終了させる (kill)

プロセスを終了させるということは、実行中のプログラムを終了させるということです。通常の場合、プログラムは処理の最後に到達すると終了しますが、なんらかの理由で、処理の途中で終了させなければいけない場合が考えられます。この方法は二種類あります。Ctrl-c を押す方法と、kill コマンドを使う方法です。

それでは、まず Ctrl-c を押す方法でプロセスを終了させてみましょう。

```
% cat
abc
abc
123
123
^C
```

このように、プログラムを実行しているときに Ctrl-c を押すと、実行中のプロセスを途中で終了させることができます。Ctrl-c を押すことでは終了させられないプロセスを終了させるには、kill コマンド

を使います。使い方は、

```
% kill process_id
```

です。なお、指定するプロセス ID は、ps コマンドを使って調べます。

ときどき、kill コマンドを使用しても終了しないプロセスが存在します。このような場合は -KILL オプション、または -9 オプションを使ってみましょう⁷。

```
% kill -KILL process_id      または,  
% kill -9 process_id
```

これで、大抵のプロセスは終了させることができます。ただし、他のユーザーのプロセスは kill を使って終了させることはできません。

4.3.5 フォアグラウンドとバックグラウンド

通常、何も指定しないでプログラムを実行した場合、その処理が終了するまで次のコマンドを入力することはできません。しかし、実行結果がすぐに必要でない場合や、時間のかかる仕事は、

```
% lp file &  
[1] 1099
```

のように “&” 記号をつけて実行します。このようにすると、実行したプロセスの終了を待たずにプロンプトが表示され、すぐに次のコマンドを入力することができます。“&” 記号をつけて実行したプロセスは、裏で実行を続けます。このようにプロセスを裏で実行することを、バックグラウンドでプロセスを実行すると呼びます。逆に、“&” 記号をつけないで実行することを フォアグラウンドでプロセスを実行すると呼びます。

バックグラウンドでプロセスを実行すると、ジョブ番号 ([] 内の数字) とプロセス ID を表示します。ここでは、ジョブとは、一連の仕事をしているプロセスの集合だと理解してください。

4.4 ジョブの操作

4.4.1 ジョブとは

ジョブとは、一連の仕事をしているプロセスの集合です。ジョブを構成するプロセスは、1 つもしくはそれ以上となります。

例えば、

⁷ ただし、このオプションを指定した場合、プロセスをいきなり強制終了させます。場合によっては、これでも消えずにコンピュータのメモリ内に残ったままの状態にもなりうるので、扱いには注意してください。なお、同オプションを指定してもプロセスが終了しない場合は、そのまま放置せず、最寄りの RAINBOW STAFF に相談に行きましょう。

```
% ls | less
```

と入力すると、パイプでつながった2つのプロセスができます^{*8}。この2つのプロセスからなる集合が1つのジョブに相当します。

4.4.2 ジョブを作る

プロセスを作ると、自動的にジョブも作られます。パイプでつながったプロセスの集合は、1つのジョブになります。

4.4.3 ジョブを表示する (jobs)

ジョブを表示させるには、jobs コマンドを使います。

```
% jobs
[1] + 実行中です          xclock
[2] - 実行中です          xterm
[3] 実行中です            emacs
```

このように、jobs コマンドは、ジョブ番号（[] 内の番号）とその状態、そしてプログラム名を表示します。ジョブ番号とは、ジョブにつけられた番号のことです。このジョブ番号を使うことで、ジョブを操作します。

4.4.4 ジョブの指定方法

ジョブを操作するには、% を使ってジョブを指定します。指定の方法には、次のものがあります。

表 4.6 ジョブの指定方法

| 指定 | 意味 |
|-------|-----------------------------|
| %n | ジョブ番号 n のジョブ |
| %+ | 現在のジョブ (jobs で + がついているもの) |
| %- | 直前のジョブ (jobs で - がついているもの) |
| %str | 文字列 str から始まるプログラム名のジョブ |
| %?str | 文字列 str を含む文字列をプログラム名に持つジョブ |
| % | %+ と同じ |
| %% | %+ と同じ |

^{*8} パイプについての詳細は 4.8.3 節で説明します。

4.4.5 ジョブを終了させる

フォアグラウンドのジョブを終了させるには、Ctrl-c を入力します。また、停止しているジョブやバックグラウンドで実行しているジョブを終了させるには kill コマンドを使います。

```
% kill %1
```

このように、% を使ってジョブを指定します。

4.4.6 ジョブを停止(サスPEND)させる(stop)

フォアグラウンドのジョブを停止(サスPEND)させるには、Ctrl-z を入力します。また、バックグラウンドのジョブを停止させるには、stop コマンドを使います。

```
% stop %1
```

また、kill コマンドを使って、

```
% kill -STOP %1
```

とすることもできます。

4.4.7 ジョブをフォアグラウンドで実行する(fg)

停止しているジョブ、またはバックグラウンドで動いているジョブをフォアグラウンドで実行するには、fg コマンドを使います。

```
% fg %1
```

のように使います。fg を省略して、

```
% %1
```

としても同じです。

4.4.8 ジョブをバックグラウンドで実行する(bg)

停止しているジョブをバックグラウンドで実行するには、bg コマンドを使います。

```
% bg %1
```

のように使います。

4.5 コマンドの入力

シェルにコマンドを実行させたいときは、プロンプトのあとにコマンド名を入力し、Enterキーを押します。プロンプトに入力をしているときに打ち間違いをした場合には、エディタと同じようにコマンド入力を編集することができます。操作方法を表4.7に示します。

表4.7 tcshの操作方法

| 操作 | 機能 |
|--------|---------------------|
| Ctrl-f | 入力位置を1つ進める |
| Ctrl-b | 入力位置を1つ戻す |
| Ctrl-a | 入力位置を先頭に移動 |
| Ctrl-e | 入力位置を文字列の最後に移動 |
| Ctrl-h | 一つ前の文字を削除 |
| Ctrl-d | 入力位置の文字を削除 |
| Ctrl-k | 入力位置より後ろに続く文字列を全て削除 |
| Ctrl-l | 画面を再描画 |

tcsh は、ファイル名の一部を入力するだけで、途中までの入力に一致するファイルの一覧を表示したり、残りの入力を補完してくれる機能を持ちます^{*9}。例えば test... といったファイル名を入力しようとし、うっかり入力するファイル名を忘れた場合、次の操作で一覧を表示できます。

```
% ls
test1-1.txt test1-2.txt test.c test.1
marina.gif misako.jpg
% vi test          ...この後のファイル名がわからない
(Ctrl-d を入力)
test1-1.txt test1-2.txt test.c test.1
% vi test1-1      ...ファイル名の補完を行いたい
(Tab or Ctrl-i を入力)
% vi test1-1.txt   ...ファイル名の入力が完了する
```

またファイル名と同様に、コマンド名に対しても補完や一覧の機能を使うことができます。この場合、シェル変数 path に設定されているコマンドサーチパスからコマンド名の検索が行われます。

```
% v          ...v で始まるコマンドを知りたい
(Ctrl-d を入力)
vcam
vcstime
(略)
vimtutor           vine-update-gnome-font-install
                     virmf
```

この例のように補完機能を使用することによって、ファイル名がうろ覚えの場合でも、ls コマンドでファイル名を確認をせずに済みます。またタイプミスも抑制することができます。

4.6 ワイルドカード文字 (ファイル名の展開)

シェルは、ユーザーからのコマンドの実行要求にメタキャラクタといわれる特別な文字が含まれていると、まずこれらの文字に対する処理を行ってからコマンドを実行します。本節では、その中のワイルドカード文字といわれる

* ? ~ [] { }

について説明します。

ワイルドカード文字がコマンド行に含まれていると、シェルはこれらをファイル名に展開します。例を示しながら、順に説明していきます。ここでは、カレントディレクトリに以下のファイルがあるとします。

^{*9} csh も同様の機能を持ちますが、補完には ESC キーを用います。

```
% ls
abocado      apple.f    carrot.txt  pomato      tomato.c
abocado.f    apple.txt   orange      pomato.f    tomato.o
abocado.txt   carrot     orange.f   pomato.txt  tomato.txt
apple        carrot.c   orange.txt  tomato
```

4.6.1 任意文字数の文字列を表す “*”(アスタリスク)

例えば、カレントディレクトリに存在するファイル名が apple で始まるファイルをリストアップしたい場合にはどうすればよいのでしょうか。あらかじめ apple , apple.f , apple.txt というファイルがあるとわかっていれば、ls apple apple.f apple.txt とタイプすればよいのですが、どのようなファイルがあるかわからない場合には、すべてのファイルをリストアップするしか方法はないのでしょうか。そういうときには、シェルに働いてもらいましょう。

シェルは、“*”を任意文字数(0個を含む)の文字列にマッチするファイル名に展開し(置き換え)ます。これを用いて、カレントディレクトリにある apple で始まるファイルをリストアップしてみます。

```
% ls apple*
apple  apple.f  apple.txt
```

この場合、シェルは、ls apple* を ls apple apple.f apple.txt に展開してから ls コマンドを実行します。ここでは apple も含まれていることに注意してください。

“*”はいくつあっても構いません。カレントディレクトリにある ファイル名に “c” を含むものをリストアップしてみます。

```
% ls *c*
abocado      abocado.txt   carrot.c    tomato.c
abocado.f    carrot       carrot.txt
```

4.6.2 任意の1文字を表す “?”

“?”は任意の1文字にマッチするファイル名に展開されます。最初の1文字は何でもよいですが、次の文字からが omato と続くファイルをリストアップしてみます。

```
% ls ?omato
pomato  tomato
```

“?”は、“*”と違い、そこには何か必ず1文字がなければなりません。これを用いて、ファイル名が7文字のファイルをリストアップしてみます。

```
% ls ????????
abocado apple.f
```

4.6.3 ホームディレクトリをあらわす “~”(チルダ)

“~”は自分のホームディレクトリのパス名に展開されます。自分のホームディレクトリにあるディレクトリ test にあるファイルをリストアップしてみます。

```
% ls ~/test
abocado      apple.f      carrot.txt  pomato      tomato.c
abocado.f    apple.txt    orange       pomato.f    tomato.o
abocado.txt   carrot      orange.f    pomato.txt  tomato.txt
apple        carrot.c    orange.txt  tomato
```

“~ユーザー名”は、ユーザー名で指定したユーザーのホームディレクトリのパス名に展開されます。ユーザー名 ra012345 のユーザーのホームディレクトリにあるディレクトリ test の中身をリストアップしてみます。

```
% ls ~ra012345/test
abocado      apple.f      carrot.txt  pomato      tomato.c
abocado.f    apple.txt    orange       pomato.f    tomato.o
abocado.txt   carrot      orange.f    pomato.txt  tomato.txt
apple        carrot.c    orange.txt  tomato
```

4.6.4 文字を指定する “[]”

[文字列] は、 “[]” 内にマッチさせる文字を列挙した書き方です。文字列内のいずれか 1 文字にマッチした場合に、その文字に展開されます。ファイル名の最後が .c または .f であるファイルをリストアップしてみます。

```
% ls *.[cf]
abocado.f  apple.f  carrot.c  orange.f  pomato.f  tomato.c
```

ここで、* はさきほど説明したように任意の文字列を表しています。

[文字₁–文字₂] は、文字₁ と文字₂ の範囲にある 1 文字とマッチするファイル名に展開します。数字 1 文字であれば、[0-9] ですし、大文字 1 文字であれば、[A-Z] になります。では、例としてファイル名が a, b, c, d のいずれかで始まるファイルをリストアップしてみます。

```
% ls [a-d]*
abocado      abocado.txt    apple.f      carrot      carrot.txt
abocado.f    apple          apple.txt    carrot.c
```

[]の中にはいくつも - をつかって範囲指定することもできます。最初の文字が a から d まで、または、s から z までのファイルをリストアップしてみます。

```
% ls [a-ds-z]*
abocado      apple        carrot       tomato      tomato.txt
abocado.f    apple.f     carrot.c     tomato.c
abocado.txt  apple.txt   carrot.txt  tomato.o
```

4.6.5 共通しない部分を表す “{ }”

共通文字列 { 文字列₁ , 文字列₂ , ... } という表現は、よく似たファイル名の列挙の際、タイプする手間と誤りを少なくするために使います。ファイル名のうち異なる部分を列挙して、{ } で囲みます。余分な空白を入れてはいけません。

tomato.c と tomato.txt のファイルの内容を表示してみます。ここでは tomato.o は表示したくありませんので、cat tomato.* とはしません。

```
% cat tomato.{c,txt}
これは ファイル tomato.c の内容です。
これは ファイル tomato.txt の内容です。
```

4.6.6 ワイルドカードの展開の確認 (echo)

複雑なワイルドカードの組み合わせを用いて、ファイル一括消去などの処理する場合は、ファイル名の展開に注意する必要があります。.c で終るファイル全部を消去する場合に、rm * .c と、間違って * と .c の間に空白を 1 つ入れてしまうと、カレントディレクトリのファイル全部と .c (つまり全てのファイル) を消してしまうことになります。

そのため、複雑なワイルドカードを用いる際には、本当に自分の思った通りに展開されるのか、確認する必要があります。そのための方法のひとつとして、与えられた引数を表示するだけのコマンドである echo を使って確認することができます。echo を実行する際にシェルが引数を展開するため、実際には展開後の文字列が表示されます。

```
% echo /usr/{,s,local}/}bin
/usr/bin /usr/sbin /usr/local/bin
```

4.6.7 ワイルドカード一覧

これまでに説明したワイルドカードを表 4.8 にまとめます。

表 4.8 ワイルドカード文字

| メタキャラクタ | 説明 |
|---|--------------------------------|
| * | 任意の文字列 (0 個を含む) に一致 |
| ? | 任意の 1 文字に一致 |
| [] | 括弧内のいずれかの 1 文字に一致。- を使った範囲指定も可 |
| { str ₁ , str ₂ , ... } | 括弧内の文字列それぞれについて別に展開 |
| ~ | 自分のホームディレクトリ |
| ~username | ユーザー名が username のホームディレクトリ |

4.7 ヒストリの表示 (history)

シェルは、最近実行したコマンドをヒストリ (履歴) としてある程度記憶します。ヒストリを表示することで、一連の操作でどこでタイプミスをしたか、あるいは席を離れている間にどんな悪戯をされたのかが分かります。また、一度実行したコマンドを簡単に再実行することもできます。

いくつ記憶するように設定されているのかを調べるには、echo \$history とします。実際に実行してみると、

```
% echo $history
100
```

100 個記憶するように設定されているのがわかりました。これは、history という名前のシェル変数の値を表示しています（詳しくは 11.1 節を参照してください）。

記憶している内容（履歴リスト）を表示するには、history コマンドを使います。

```
% history
 83 14:03  kterm &
 84 14:07  ls
(略)
178 17:02  emacs -nw abocado.txt
179 17:08  cat abocado.txt
180 17:10  firefox &
```

以前に cat abocado.txt を実行し、もう一度このコマンドを実行するには、メタキャラクタ “!” と履

歴リストの cat abocado.txt の左側に表示されている番号 179 を使って、

```
% !179  
cat abocado.txt  
これは ファイル abocado.txt の内容です.
```

とします。シェルはメタキャラクタである “!” があるので、履歴リストの 179 番に対応するコマンドを実行したわけです。

直前のコマンドは!!で実行できます。

```
% !!  
cat abocado.txt  
これは ファイル abocado.txt の内容です.
```

さらに、“!”+ 文字列 で、その文字列で始まる、一番最近実行したコマンドを実行します。

```
% !cat  
cat abocado.txt  
これは ファイル abocado.txt の内容です.
```

また、上記のメタキャラクタ ! を使ったヒストリの指定方法の他に、tcsh ではより直観的に、Ctrl-p , Ctrl-n によってヒストリリストから文字列を取り出し、編集することもできます。

```
% cc -c test.c  
% ls -al  
(Ctrl-p を 2 回入力する)  
% cc -c test.c          ...前前回の入力  
(Ctrl-n を 1 回入力する)  
% ls -al              ...前回の入力
```

ヒストリに関する機能については、詳しくは tcsh のオンラインマニュアルを参照してください。

4.8 標準入出力とリダイレクション・パイプ

4.8.1 標準入出力とは

UNIX には、標準入力、標準出力、標準エラー出力といった概念があります。これらは、通常それぞれ、

標準入力: キーボードからの入力

標準出力: 端末の画面あるいは仮想端末 (GNOME 端末 等)

標準エラー出力: 端末の画面あるいは仮想端末 (GNOME 端末 等)

に割り当てられています。例えば、cat コマンドは、ファイル名を指定しなかった場合には、標準入力からデータを読み込み標準出力に出力します。

```
% cat  
RAINBOW  
RAINBOW  
^D
```

(キーボードから文字を入力)
(そのまま出力される)
(Ctrl-d を入力)

この場合、キーボード（標準入力）から文字をタイプして Enter キーを押すと、そのタイプした文字が画面（標準出力に）に表示されます。終了するときは、Ctrl-d (Control キーをしながら d) を押すと（標準入力というファイルからの）入力が終わります。

4.8.2 標準入出力のリダイレクション

リダイレクションは、標準入出力を指定したファイルに切り替える機能です。cat コマンド等は、ファイルを指定しなければ入力として標準入力を使用します。ここでは < という文字を使って標準入力をキーボードからファイル tomato.c に切り替えてみます^{*10}。

```
% cat < tomato.c  
これは ファイル tomato.c の内容です。
```

次に、cat コマンドとリダイレクトを用いてファイルの連結をしてみます。> を使って実行結果の出力を標準出力から text というファイルにリダイレクトします^{*11}。本来なら連続して画面に表示されますが、リダイレクトにより出力先がファイルになっており、画面には表示されません。しかし、画面に出力されるはずだった結果は、そのまま 1 つのファイルとなり、結果的に複数のファイルを連結して 1 つのファイル (text) とすることができます。

*10 cat tomato.c と同じ結果になりますが、ここではリダイレクションの例として < を使っています

*11 cat コマンドは、引数に複数のファイルを指定することができます

```
% cat abocado.txt apple.txt carrot.txt > text
% cat text
これは ファイル abocado.txt の内容です。
これは ファイル apple.txt の内容です。
これは ファイル carrot.txt の内容です。
```

それでは、次の例は何をしているのでしょうか。

```
% cat < tomato.txt > tomato.bak
% cat tomato.bak
これは ファイル tomato.txt の内容です。
```

これは `cat` コマンドを使ってファイルのコピーを実現しています^{*12}。<で入力を `tomato.txt` に>で出力を `tomato.bak` に指定しています。このように < と > を同時に使うこともできます。

さらに `>>` を使って、さきほど作成したファイル `text` のおわりに `orange.txt` の内容を追加してみます。

```
% cat orange.txt >> text
% cat text
これは ファイル abocado.txt の内容です。
これは ファイル apple.txt の内容です。
これは ファイル carrot.txt の内容です。
これは ファイル orange.txt の内容です。
```

`>`, `>>` を用いた場合は、標準エラー出力はリダイレクトされません。もし、標準エラー出力も一緒にリダイレクトしたいときには、それぞれ `> &`, `>> &` を用います^{*13}。

`>` および `> &` を使ったリダイレクションを使うときに注意することがあります。それは、出力先に既存のファイルを指定した場合には、そのファイルは上書きされ、もとの内容は失われてしまうということです。この動作を行わないようにする、すなわち、既存のファイルにリダイレクションを行おうとしたらエラーになるようにするには、`set noclobber` というおまじない^{*14}をします。

```
% set noclobber
% cat apple.txt > apple
apple と言うファイルはすでに存在します。
```

もし、`set noclobber` を実行していた場合で、既存ファイルに上書きするリダイレクションを行いたいとき、すなわち `noclobber` を無視してリダイレクションを行いたいときには、`!>` をつけて `>!` や `> &!` のようにします。

*12 もちろん、`cat tomato.txt > tomato.bak` としてもできます

*13 これは csh 系の場合です。sh 系であれば、標準エラー出力のリダイレクトは、もう少し細かく制御できます。

*14 シェル変数 `noclobber` をセットするということです。1回設定すれば、シェルを終了するか、`unset noclobber` を実行するまで有効です。

また、`>> (>> &)` と `noclobber` の関係は次の通りです。`set noclobber` を実行していた場合、存在しないファイルを指定するとエラーになります。さらにこのとき、`>>! (>> &!)` を使えば `noclobber` を無視するのでエラーにはなりません。

4.8.3 パイプ

パイプとは、コマンドの標準出力（実行結果）を次のコマンドの標準入力にする機構です。このパイプ機構により、既存の（シンプルな）コマンド群を組み合わせて、複雑な処理を行うことが可能となります。このように、標準入力から入力を受け取り、加工してから標準出力に結果を出力しするコマンドを フィルタ と呼びます。UNIX には、組み合わせて使うのに便利な、多くの基本的なフィルタが用意されています。

ここでは、パイプ機構を使った簡単な例を紹介します。`ls` コマンドと `wc` コマンドを使って、カレントディレクトリにあるファイルの数を表示してみます。`wc` コマンドは、入力の文字数、単語数、行数を数えるコマンドです（詳しくはオンラインマニュアルをみてください）。

```
% ls | wc -l  
21
```

この場合 `ls` コマンドは、1 ファイルあたり 1 行の形式でファイルのリストを出力します。ここでは、メタキャラクタ `|` を使って `ls` コマンドの出力を `wc` コマンドの標準入力としています。`wc` がこの入力が何行あるか数えることにより、結局ファイルの数を標準出力（画面）に出力します。

表 4.9 にリダイレクションおよびパイプの使い方をまとめます。

表 4.9 リダイレクト・パイプ

| 書き方 | 説明 |
|---|---|
| <code>> file</code> | 標準出力を <code>file</code> に切り替える * ¹ |
| <code>> & file</code> | 標準出力およびエラー出力を <code>file</code> に切り替える * ¹ |
| <code>>> file</code> | 標準出力を <code>file</code> につけたす * ² |
| <code>>> & file</code> | 標準出力およびエラー出力を <code>file</code> につけたす * ² |
| <code>< file</code> | 標準入力を <code>file</code> に切り替える |
| <code><< str</code> | 次に文字列 <code>str</code> が現れる前までをコマンドの入力にする |
| <code>cmd₁ cmd₂</code> | <code>cmd₁</code> の標準出力を <code>cmd₂</code> の標準入力にする |
| <code>cmd₁ & cmd₂</code> | <code>cmd₁</code> の標準出力および標準エラー出力を <code>cmd₂</code> の標準入力にする |

*¹ *file* に既存ファイルを指定した場合、シェル変数 noclobber により次のように動作が異なります。noclobber がセットされていないときは *file* が上書きされます。セットされているときはエラーとなり *file* が上書きされることはありませんが、`>!`、`>` &`!` のように`!`をつけると強制的に上書きされます。

*² noclobber がセットされている場合は、*file* がなければエラーになります。

第5章

エディタ

エディタとは、ディスク上のファイルを編集するツールです。エディタにはさまざまな種類がありますが、この章ではテキストを編集するテキストエディタについて説明をしていきます。一般的なユーザーがエディタを用いて扱うファイルとしては、以下のものが挙げられます。

- 個人用の環境設定ファイル
- 授業で出されたレポートが納められたファイル
- 友達からのメールが納められたファイル

また、プログラムなどを作ったり、コンピュータやワープロでレポートなどを作ったことのある人なら分かると思いますが、一度だけのファイル編集でプログラムが動いたり、完璧なレポートができたりすることはあまりないと思います。

以下に UNIX 上での一般的なファイルを扱う流れを示します。

1. エディタで扱いたいファイルを編集する。
2. 満足いくかどうか UNIX のコマンドを使って確かめてみる。（例えば、コンパイルやプリントアウト）
3. 満足がいかなかつたり失敗した場合、1 に戻ってファイルを再編集する。

上の流れを見れば、UNIX 上でのエディタの役割がわかってもらえると思います。

ここでは、RAINBOW 環境で扱うことのできる Emacs, vi, gedit について紹介していきます。

5.1 Emacs

Emacs では、一時的な作業領域であるバッファへファイルを読み込み編集します。メモリが許す限り一度に複数のバッファ、つまり複数のファイルを扱うことができます。バッファの内容を変更しただけでは、ディスク上のファイルの内容が変わることではなく、それをセーブすることによりファイルの内容が変更されます。

Emacs の特徴を以下に示します。

- 編集した結果が画面上で確認できる。
- ユーザーが、個人の好みに合わせて操作を変更したり、新たに機能を拡張することができる。
- 多言語の文字が表示でき、それを編集できる。
- 簡単に電子メールや NetNews の読み書きができる。
- UNIX コマンドを簡単に操作できる。
- Emacs のウインドウを複数開けて操作できる。
- 一定周期のオートセーブでなく、一定の時間内に次の入力がないときオートセーブされ、遅く感じない。
- 文字に色つけることができるので、ファイルが見やすい。

5.1.1 Emacs を使う前に

Emacs を操作するときには、Ctrl キーや ESC キーを使うことが多くなります。そこで、以下 Ctrl キーを押しながら何か文字キーを押す操作を、

C-文字

という表現で表します。Ctrl キーを押しながら z キーを入力する場合は、“C-z”となります。また ESC キーを押して、離してから x キーを押す操作を、M-x と表します^{*1}。“M-x recover-file”とある場合、ESC キーを押してから x キーを、続けて recover-file と入力してください。

Emacs を使用していて異常が発生した場合には、以下のコマンドを試してみてください。

C-g

C-g は、実行しているコマンドを中止するコマンドです。間違ったコマンドを入力してしまったときや、何をやっているか分からなくなったときは、このコマンドを数回入力してみてください。

5.1.2 Emacs の起動

マウスを動かして適当な端末上 (GNOME 端末や kterm など) に持っていき、カーソルが白くなっていることを確認してください。シェルプロンプトに対して次のコマンドを入力すれば、Emacs が起動します。

% emacs

また、Emacs の起動時にファイル名を引数としてあたえると、そのファイルをバッファに読み込んだ状態で起動できます。指定したファイル名のファイルが存在しない場合は、新規作成になります。例として foo.txt というファイルを編集、あるいは新規作成するために Emacs を起動するには以下のコマンド

^{*1} あるいは端末によっては、ALT キーを押しながら z キーを押しても同じキー操作として解釈されます。

を端末に入力します。

```
% emacs foo.txt
```

5.1.3 Emacs の画面の説明

ここでは Emacs 画面の構成要素を説明します。Emacs では、編集を効率良く行うためにバッファにはいろいろな情報が付加されています。

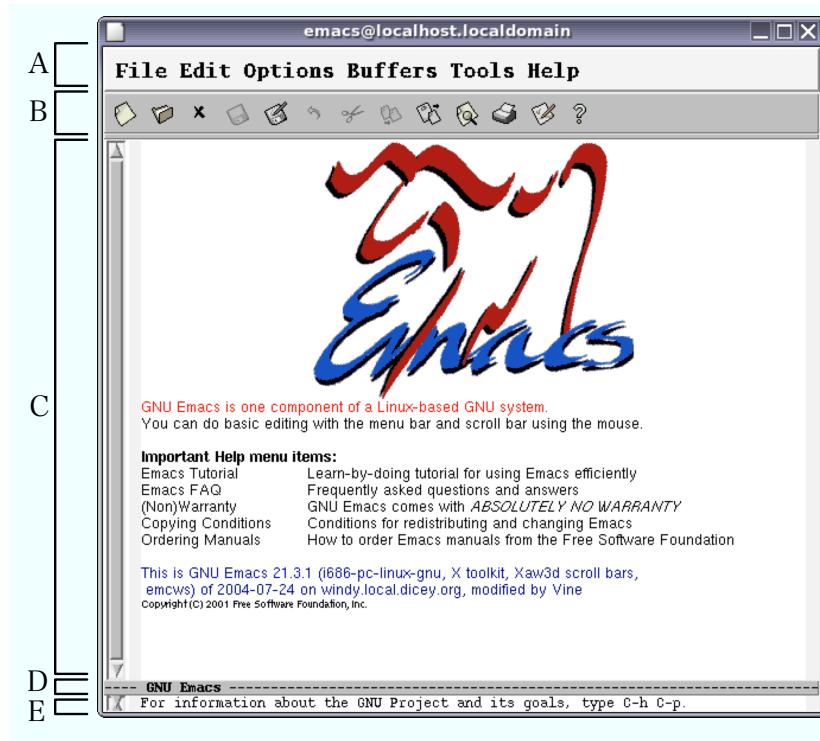


図 5.1 画面の構成要素

図 5.1 は Emacs の起動状態です。その画面は A から E で表され、それぞれ以下の内容を表示しています。

- A: 起動状態でのメニューバーは File , Edit , Options , Buffers , Tools , Help の 6 つからなっています。メニューバーの機能は以下のようになっています (扱うファイルの種類や Emacs のモードによって、メニューバーは多少異なり、扱えるコマンドが増えますので試してください)。
- File : Emacs の基本的操作のファイルオープン , セーブ , バッファの切替え , Emacs の終了 , さらに新しい Emacs のウインドウを開けたり , 閉じたりすることができる。
 - Edit : ファイルの編集に使われる Cut や Paste , Undo ができる。
 - Options: Emacs のいろいろな設定を変えたり , 言語やフォント , 日本語入力方法などを設定できる。

- Buffers：過去に扱ったバッファ状況を知ることができ、再度呼び出すことができる。
- Tools: ファイルの内容を検索したり、バッファの内容を印刷したり、デバッガを使ったり、メールを見ることができる。
- Help: チュートリアルやキーバインド情報を知ることができる。

また、それぞれのメニュー・ウインドウの右側にそれぞれの操作に対するキーボードからの操作法が対応して書かれていますので、マウス操作が嫌いな方はそれらを参照して操作してください。

- B: ここは、様々な機能をボタンをクリックすることで利用できるツールバーです。ここでは詳しく説明しませんが、いろいろな機能が使えるので試してみてください。
- C: ここは、バッファ内容を表示する領域で、基本的にこの領域に表示されているものを編集していくことになります。
- D: バッファ状態などを表示するモードラインです。モードラインの内容は図 5.2 の通りです（起動する場所により多少異なる場合があります）。

```

-E:-- *scratch*      (Lisp Interaction Encoded-kbd)--L1--All-----
  12  3   4           5           6   7

```

図 5.2 モードライン

1. 入力モード^{*2}

- <Anthy: あ> : 日本語入力モード (かな)
- <Anthy: ア> : 日本語入力モード (カナ)
- <Anthy: A> : 日本語入力モード (半角英数)
- <Anthy: A > : 日本語入力モード (全角英数)

2. 現在の文字コード^{*3}

- J : JIS コード
- E : 日本語 EUC
- S : シフト JIS コード

3. 現在扱っているバッファの変更状態

- : バッファ内容未変更
- ** : バッファ内容変更
- %% : バッファ内容変更不可 (書き込み許可がない)

4. バッファの名前

- *scratch* : Emacs立ち上げ時、またはファイルを特定していない場合
- バッファ名 : バッファを扱っているとき

^{*2} 入力モードについては 5.2 でくわしく説明しています。

^{*3} 日本語の文字コードについては 148 ページを参照してください。

ディレクトリ名 : Dired を扱っているとき

5. 編集モード

(説明は省略します)

6. 行数

現在バッファの何行目を編集中かを表します

7. バッファのうち上から何 % の位置が表示されているか

All : バッファ内容すべてが表示されている

Top : バッファの先頭部が表示されている n

Bot : バッファの末尾部が表示されている

E: ここは、ユーザーへのメッセージを表示するエコー領域で、検索、保存などのコマンドを Emacs 上で行うときに使います。

バッファの構成とモードラインの情報は、Emacs を使っていくうちに覚えますが、Emacs の使いはじめには、これらの情報にも注意して操作を行ってください。

5.1.4 Emacs の終了

Emacs を終了したいときは、まず、メニューバーの “File” をクリックし、そこで “Exit Emacs” をクリックします。また、カーソルを Emacs 上に置いて以下の 2 通りある操作のどちらかを行っても Emacs を終了させることができます。

- Ctrl キーを押しながら x キーを入力し、その後 Ctrl キーを押しながら c キーを入力する。
C-x C-c

- ESC キー入力の後 x キーを入力すると、Emacs の一番下のエコー領域に M-x という文字が出力されるので、その後 save-buffers-kill-emacs と入力する。

M-x save-buffers-kill-emacs

Emacs を終了するとき、バッファの内容が変更されていれば、そのバッファを保存するかどうかを聞いてきます。保存する場合は y キーを、しない場合は n キーを入力して終了します。バッファの内容が変更されていなければそのまま終了します。バッファの保存をせずに Emacs を終了しようとすると、Emacs がそれらのバッファについてどうすべきかをユーザーに問い合わせるので安心ですが、バッファを編集したら隨時保存を心がけておく方が無難です。また、Emacs が正常に終了しなかった場合^{*4} でも #ファイ

^{*4} 異常終了することを“落ちる”と呼ぶことがあります。

ル名# というファイルに保存してくれるので、あわてず Emacs を立ち上げ直して編集し直したり、ファイル名を変更するなどして対処してください。

5.1.5 Emacs を動かしてみよう

Emacs を使用して図 5.3 の文章を書いてみます。

```
鳴かぬなら殺てしまえホトトギス  
鳴かぬなら鳴かしてみようホトトギス  
鳴かぬなら鳴くまで待とうホトトギス
```

図 5.3 練習用文章

Emacs を起動するため端末に以下のコマンドを入力してください。

```
% emacs
```

Emacs を起動した状態では英字入力であるので、日本語の文章を入力するために日本語入力へ切替えます。Emacs で英字入力から日本語入力へ切替えるためには Ctrl を押しながら ¥ を入力すると覚えておいてください。(5.2 で Emacs 上での日本語入力の詳細を説明します。)

```
C-¥
```

これで、日本語入力が可能です。もう一度同じことを行うと英字入力に戻ります。

それでは 1 行目の文をローマ字打ちで入力してみてください。入力された文章は、ひらがなで表示されます。この状態で、Space キーを押すと漢字変換を行います。適切な変換対象がない場合は Ctrl キーを押しながら i キーを入力して文節を縮めるか、Ctrl キーを押しながら o キーを入力して文節を伸ばし、適切な文節へ変更した後、再度変換をおこなってみてください。Enter キーを押せば、その漢字に決定します。

続いて 2 行目、3 行目を入力します。ここで注目してほしいのですが、鳴かぬなら と ホトトギス は繰り返し文章に登場しています。そこでコピー & ペーストを使用してみましょう。

まず、2 行目に文章を入力したいので、Enter キーを入力してください。これで改行されるので、続いて 鳴かぬなら のコピーに取り掛かります。コピーする部分の先頭にカーソルを持っていき、Ctrl キーを押しながら Space キーを入力します。この場合は 鳴 になります。

```
C-Space
```

次にコピーする部分の一番後ろの次の文字(右の文字)にカーソルを持っていきます。今回は、殺 という文字になります。続いて Ctrl キーを押しながら w キーをと入力してください。

```
C-w
```

C-w を入力した際にコピーする範囲が消えてしまいますが、Ctrl キーを押しながら y キーを入力すれば元に戻るので安心してください。

C-y

コピーした文を次はペーストします。ペーストしたい場所へカーソルを持っていき、C-y を入力してください。

C-y

これで、鳴かぬなら という文がコピー & ペーストできました。今までの説明をもとに、他の部分も同様に書いてみましょう。

文章が書けたのでファイルへ保存しましょう。保存操作を行うと保存したいファイル名を聞いてきますので、hototogisu.txt というファイル名で保存してみましょう。まず、Ctrl キーを押しながら x キーを入力し、その後 Ctrl キーを押しながら s キーを入力します。

C-x C-s

すると、Emacs のエコー領域に、

File to save in: ~ /

と表示されるので、

hototogisu.txt

と入力し、Enter を入力すると、ファイルに保存することができます。

作業が完了したので Emacs を終了させます。終了方法は、Ctrl キーを押しながら x キーを入力し、その後 Ctrl キーを押しながら c キーを入力します。

C-x C-c

これで Emacs が終了します。端末から ls を入力すると hototogisu.txt というファイルが存在することが確認できます。ここで、先ほど書いたファイルを見てみます。端末で次のコマンドを入力してください。

more hototogisu.txt

先ほど書いた文章が表示されます。

この要領で Emacs でファイルを編集し、友達に送るメールやレポート、を作成し、また、環境設定を行なうことができます。次に Emacs の基本操作を紹介します。

5.1.6 ファイル編集に伴う Emacs の基本操作

- Emacs を使用する上での一連の動作

1. ファイルの読み込み (作成)

新しくファイルを作成したり、既存のファイルをオープンするためには、起動時の引数としてファイル名を指定する以外に、

C-x C-f

と入力する方法があります。入力後、エコー領域に

Find file: ~/

と表示されるので、オープンしたいファイル名または作成したいファイル名を入力します。既存のファイルの場合そのファイルの中身が表示され、新しいファイルの場合はエコー領域に“(New file)”と表示されファイルが作成されます。

2. ファイルの編集 (エディット)

基本的にキーを入力すれば、バッファ内容に文字入力が反映されます。

3. ファイルの保存 (セーブ)

作成されたり更新されたファイルを保存するためには、ファイルをセーブする動作が必要になります。ファイルをセーブするためには、

C-x C-s

と入力してください。入力後エコー領域に

Wrote filename

と表示されファイルの中身がセーブされます。もしファイルの中身が更新されないまま C-x C-s を入力するとエコー領域に

(No changes need to be saved)

と表示されます。つまり、内容が全く更新されていないため、改めてセーブする必要がないということです。編集されたファイルがセーブされているか否かはモードラインを見れば分かります。モードライン上のバッファ名の左側を見ると、ファイルの中身を更新、または、新たに作成してあるにもかかわらずセーブしていない場合 “**” と表示されます。もしセーブされていれば “--” と表示されます。

図 5.4 ファイルがセーブされた場合のモードライン

セーブするためのコマンドとして C-x C-s 以外に C-x C-w があります。C-x C-s はファイル名を変更せずにセーブしますが、ファイル名を変更してセーブするのであれば

C-x C-w

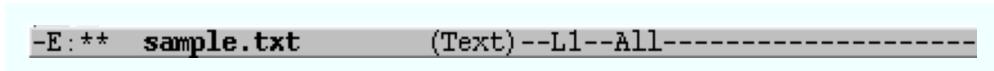


図 5.5 ファイルがセーブされていない場合のモードライン

と入力してください。このコマンドを入力するとエコー領域に、

`Write file: ~/`

と表示されるので、別のファイルにセーブする場合、そのファイル名を入力すればセーブできます。このとき、現在読み込んでいるファイルも残っています。ファイル名を変えたくないのであれば、そのまま Enter キーを押します。上書きするかどうかを聞かれるので y キーを押すことで上書きして保存ができます。

4. Emacs の終了

Emacs を終了するには、

`C-x C-c`

と入力します。このときもしセーブされていないファイルが存在すると、エコー領域に

`Save file filename? (y, n, !, ., q, C-r or C-h)`

と表示されるので、セーブしたいのであれば “y” を、そうでなければ “n” を入力します。

- Emacs が作成するファイルについて

Emacs は、編集中のファイルをセーブせずに間違って消してしまったり、ファイルを更新した際に、バックアップのファイルを自動的に作成してくれます。Emacs を終了した後にコマンドライン上で ls を入力してください。例えば編集中にファイルの名前が sample であったとすると、前者の場合には #sample# というファイル（オートセーブファイル）が、後者の場合には sample~ というファイル（バックアップファイル）が作成されます。

- オートセーブファイル

オートセーブファイルは、ある決まった回数だけ文字入力を行うか、もしくはファイルをセーブせずに強制的に終了してしまった場合（意図的でない場合も含む）に Emacs が自動的に作成してくれるファイルです。このファイル（例えば #sample#）があれば、sample というファイルを復元することができます。

#sample# が存在しており、かつ sample が存在しない、もしくは sample の中身が #sample# よりも古い場合、sample を読み込もうとすると、エコー領域に

`sample has auto save data; consider M-x recover-file`

と表示されるので、

`M-x recover-file`

と入力すると

`Recover file: ~/`

と入力を求めてくるので復元したいファイル名（ここでは “sample”）のパス名を入力します。すると今度は

`Recover auto save file /home/graduate01/mcp40081/#sample#? (yes or no)`

と尋ねてくるので “yes” と入力すると元のファイルが復元されます。

- バックアップファイル

Emacs はファイルを更新してからセーブすると、更新前（デフォルトでは 1 つ前）のファイルが保存されるようになっています。更新されたファイル名が sample ならば、バックアップファイルは sample~ になります。また個人的に設定すれば、もっと多くのバックアップファイルを残しておくことができます。そのためには .emacs.el^{*5} ファイルに

```
(setq version-control t)
```

の 1 行を加えてください。これによりバックアップファイルは “sample.~1~” のように数字がつき、バックアップの履歴が分かるようになります。数字が若いほど、古いファイルになります。またバックアップファイルの数が増えてくるとファイルをセーブするときに、

```
Delete excess backup versions of /home/graduate01/mcp40081/sample? (y or n)
```

と聞いてくることがあります。これは、バックアップが多すぎるので削除しますか？ ということを意味しています。もし “y” を入力すると、バックアップファイルの中で最も古いファイル 2 つと最も新しいファイル 2 つを残して削除されます。

5.1.7 ファイル編集時の Emacs のコマンド

Emacs では、ファイルの編集や保存、カーソルを 1 つ移動させるコマンドなどが用意されています。Emacs 上でこれらのコマンドを実行するためには、

```
M-x commandname
```

で実行できます。しかし、カーソル移動を含め頻繁に実行するコマンドに対し、コマンド名を逐一入力していくは面倒です。そこで Emacs では、これらのコマンドを簡単に実行するために、多くのコマンドがキーにバインドされています。コマンドの中から通常の使用で使うものを紹介していきます。

- ファイル編集に関する操作

- カーソルの移動方法

テキストの中で、カーソルを移動させる方法は表 5.1 の通りです。

- テキストの削除

文字を削除するための方法は、3 種類あります。削除させる方法は表 5.2 の通りです。

- アンドウ機能

アンドウ機能とは、編集中に必要なテキストを間違って消してしまったり、ミスに気づかず編集を続けてしまった場合に、元に戻したいというときに便利な機能です。C-/ , C-_ または C-x u で 1 つ手前の状態に戻すことができます。当然、これらのキーを繰り返し入力すれば、押した回数の分だけ手前の状態に戻ることができます。ただし、戻ることのできる回数には限

^{*5} emacs.el の詳細については、73 ページを参照してください。

表 5.1 カーソル移動

| | |
|-------------------|-----|
| 右(前)へ移動する | C-f |
| 左(後)へ移動する | C-b |
| 上へ移動する | C-p |
| 下へ移動する | C-n |
| 1ワード(単語)右(前)へ移動する | M-f |
| 1ワード(単語)左(後)へ移動する | M-b |
| 行頭に移動する | C-a |
| 行末に移動する | C-e |

表 5.2 テキストの削除

| | |
|--------------------------|-----|
| カーソルの1つ手前(左側)の文字を削除する | DEL |
| カーソルの下にある(重なっている)文字を削除する | C-d |
| カーソルの位置から行末までを削除する | C-k |

度があります。

– 画面移動に関する操作

これまでに述べたカーソル操作でファイルの中を移動することも可能ですが、画面単位で移動することができます。また1度にファイルの先頭や末尾に移動することもできます。画面移動させる方法は表5.3の通りです。

表 5.3 画面移動

| | |
|--------------|-----|
| 1画面下に移動する | C-v |
| 1画面上に移動する | M-v |
| ファイルの先頭に移動する | M-< |
| ファイルの末尾に移動する | M-> |

– 繰り返し操作について

これまでに述べたコマンドを繰り返し入力する場合、その繰り返しの回数を指定して実行する方法があります。まずC-uと入力し、続けて繰り返しの回数を入力し、目的のコマンドを入力します。例えばC-fを10回繰り返す、つまり10文字分右にカーソルを移動させたい場合、C-u, 10, C-fの順に入力すると、エコー領域にC-u 1 0 C-fと表示されコマンドが実行されます。

- カット & ペースト(キル&ヤンク)

Emacsには便利なカット&ペーストの機能が備わっています。カット&ペーストとは、ファイルの一部を切り取るか或いは写し取って、他の場所へ移動させる又はコピーするということです。エディタを利用したことのある人ならカット&ペーストという言葉には慣れているかもしれません

が、Emacs では“カット”を“キル”，“ペースト”を“ヤンク”と表現しています。よってコマンド名にも“kill”，“yank”という言葉が使用されています。

- 指定されたリージョンの削除

先に、文字や行を削除するコマンドとして DEL , C-d , C-k を紹介しましたが、ここではまず切り取りたい領域を指定して削除する方法を述べます。Emacs ではこの領域のことをリージョンと呼びます。

1. 指定したいリージョンの先頭にカーソルを持って行き、C-Space または C-@ と入力しマークを設定します。このときエコー領域に Mark set というメッセージが表示されます。
2. 指定したいリージョンの末尾の 1 つ後(右)までカーソルを持って行きます。この時点でリージョンを指定したことになっています。指定したリージョンを確認したければ、C-x C-x と入力してください。指定したリージョンの先頭にカーソルが移動します。もう一度 C-x C-x と入力すれば元の位置に戻ります。
3. リージョンを指定した後、C-w と入力すると、指定したリージョンが削除されます。

上記の手順 1 では、指定したいリージョンの先頭でマークを設定していますが、指定したいリージョンの末尾の 1 つ前(右)でマークを設定した後、カーソルをリージョンの先頭に移動することによってもリージョンを指定することができます。

- リージョンを利用したテキストの移動またはコピー

先に C-w で削除されたリージョンのテキストは キルリング という特別なバッファに保存されています。このバッファの内容を呼び戻すことで保存された内容を挿入することができます。

1. リージョンを指定して、移動したいテキストを削除(上の方法と同じ)
2. 挿入したい場所にカーソルを移動して C-y と入力すると、削除されたテキストが挿入されます。

- リージョンを利用したテキストのコピー

上に挙げた 2 つの方法は、指定されたリージョンのテキストを削除してから、別の場所に挿入する方法ですが、ここでは元のテキストを削除せずにリージョンを指定して挿入する、つまりコピーする方法を述べます。

1. リージョンを指定した後、M-w と入力します。画面上では何も変化しませんが、キルリングに保存されています。
2. 移動の場合同様、挿入したい場所にカーソルを移動して C-y を入力すると、保存されたテキストが挿入されます。

C-k による削除についてですが、このコマンドによって削除された内容もキルリングに保存されています。よって C-y を入力すると、その内容を戻すことができます。また C-o と入力すると、カーソルの位置している行に空行を挿入することができます。編集コマンドのまとめを表 5.4 に示します。

表 5.4 ファイル編集時のコマンド

| キー入力 | コマンド名 | 動作 |
|---------|-------------------------|-------------------------------|
| C-x C-f | find-file | ファイルを読み込む、又は新規作成する |
| C-x C-s | save-buffer | ファイルをセーブする |
| C-x C-w | write-file | バッファの内容をファイルに書き出す |
| C-x C-c | save-buffers-kill-emacs | Emacs を終了する |
| C-f | forward-char | 1 文字右(前)へ移動する |
| C-b | backward-char | 1 文字左(後)へ移動する |
| C-n | next-line | 1 文字下へ移動する |
| C-p | previous-line | 1 文字上へ移動する |
| M-f | forward-word | 1 ワード右(前)へ移動する |
| M-b | backward-word | 1 ワード左(後)へ移動する |
| C-a | beginning-of-line | 行頭へ移動する |
| C-e | end-of-line | 行末へ移動する |
| DEL | delete-backward-char | カーソルの手前の文字を削除 |
| C-d | delete-char | カーソルと重なっている文字を削除 |
| C-v | scroll-up | 1 画面下へ移動する |
| M-v | scroll-down | 1 画面上へ移動する |
| M-< | beginning-of-buffer | ファイルの先頭に移動する |
| M-> | end-of-buffer | ファイルの末尾に移動する |
| C-/ | undo | アンドゥを行う |
| C-_ | | |
| C-x u | advertised-undo | |
| C-u n | universal-argument | 続けて入力するコマンドを n 回繰り返す。 |
| C-Space | set-mark-command | リージョンの先頭(末尾)をマークする |
| C-@ | | C-Space と同じ。 |
| C-x C-x | exchange-point-and-mark | カーソルをマークしたリージョンの先頭または末尾に移動させる |
| C-w | kill-region | マークしたリージョンを削除してキルリングに移す |
| M-w | copy-region-as-kill | マークしたリージョンを削除せずにキルリングに移す |
| C-y | yank | キルリングに保存されたテキストを復元する |
| C-o | open-line | 空行を挿入する |

5.1.8 ウィンドウについて

複数のファイルの内容を同時に表示しながら編集したい場合、複数のウィンドウを開くか、あるいはウィンドウを分割することで行えます。ここではウィンドウの操作について説明します。

複数のウィンドウを開く

Emacs で複数のウィンドウを開く方法は、マウスでメニューバーの “File” をクリックし、そこで、“New Frame” を選ぶか、

C-x 5 2

を実行します。そうすると図 5.6 のように、Emacs のウィンドウが 2 枚開いた状態になります。こうすれば、複数のファイルを編集するときに見やすくなると思います。さらに Emacs のプロセスは 1 つのままでありますので、重たく感じることはないと思います。また、Emacs のウィンドウを複数開いているときに、ウィンドウを 1 つだけ閉じたい場合は、閉じたいウィンドウをフォーカスして、

C-x 5 0

を実行します。

ウィンドウの操作

Emacs ではウィンドウを分割して使用できます。

C-x 2

と入力するとウィンドウが上下に分割されます（図 5.7 参照）。互いのウィンドウ間においてカーソルを移動させるためには

C-x o

と入力してください。また片方のウィンドウを消去するためには、消去したい方のウィンドウにカーソルを移動させてから

C-x 0

を入力してください。

5.1.9 バッファの操作

Emacs でファイルを読み込み編集する場合、ファイルの中身を一度バッファにコピーし、実際にはこのバッファの内容を編集します。ファイルにセーブするという動作は、バッファの内容を、ある名前のつ

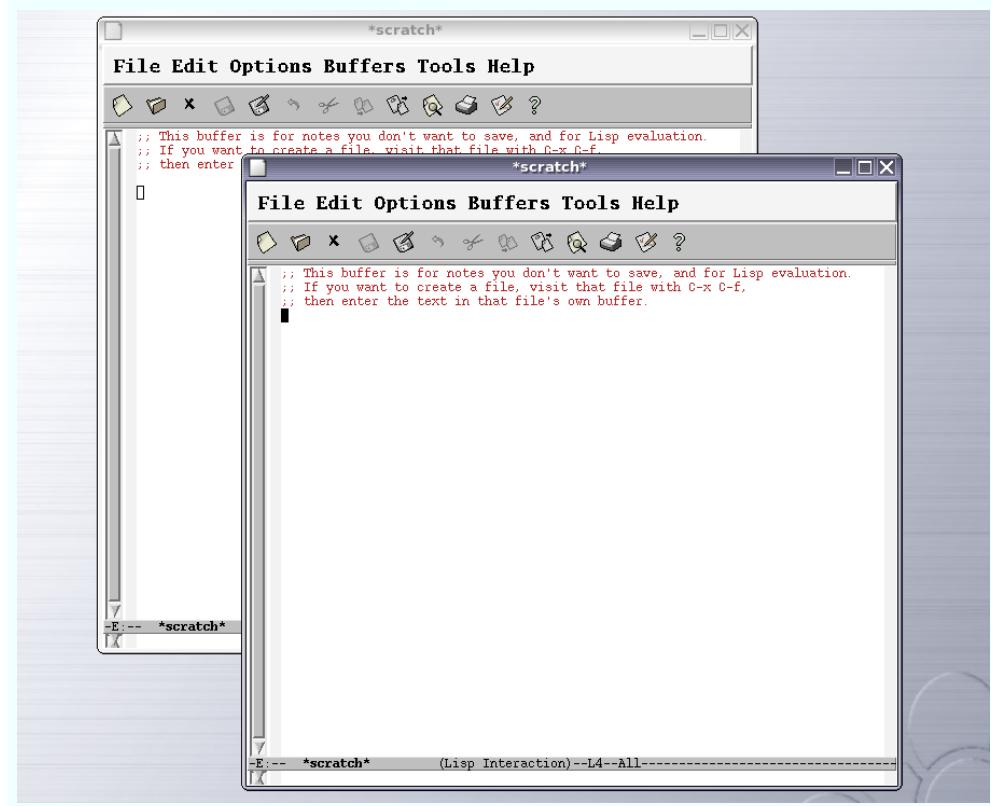


図 5.6 New Frame で開いたウインドウ

表 5.5 ウインドウのコマンド

| キー入力 | コマンド名 | 動作 |
|-------|-------------------------|---|
| C-x 2 | split-window-vertically | 現在使用しているウインドウを上下に 2 分割する |
| C-x 1 | delete-other-windows | 現在使用している以外のウインドウを削除する |
| C-x 0 | delete-window | カレントウインドウ (現在カーソルが存在しているウインドウ) を削除する (但し 2 つ以上ウインドウをオープンしているときのみ) |
| C-x o | other-window | 他のウインドウに移動する |

いたファイルにコピーすることです。

Emacs で複数のファイルを並行して編集している場合で、他のファイルを編集するために現在のファイルのバッファをクローズせずに、他のファイルのバッファに移るためには、

C-x b

と入力します。するとエコー領域に

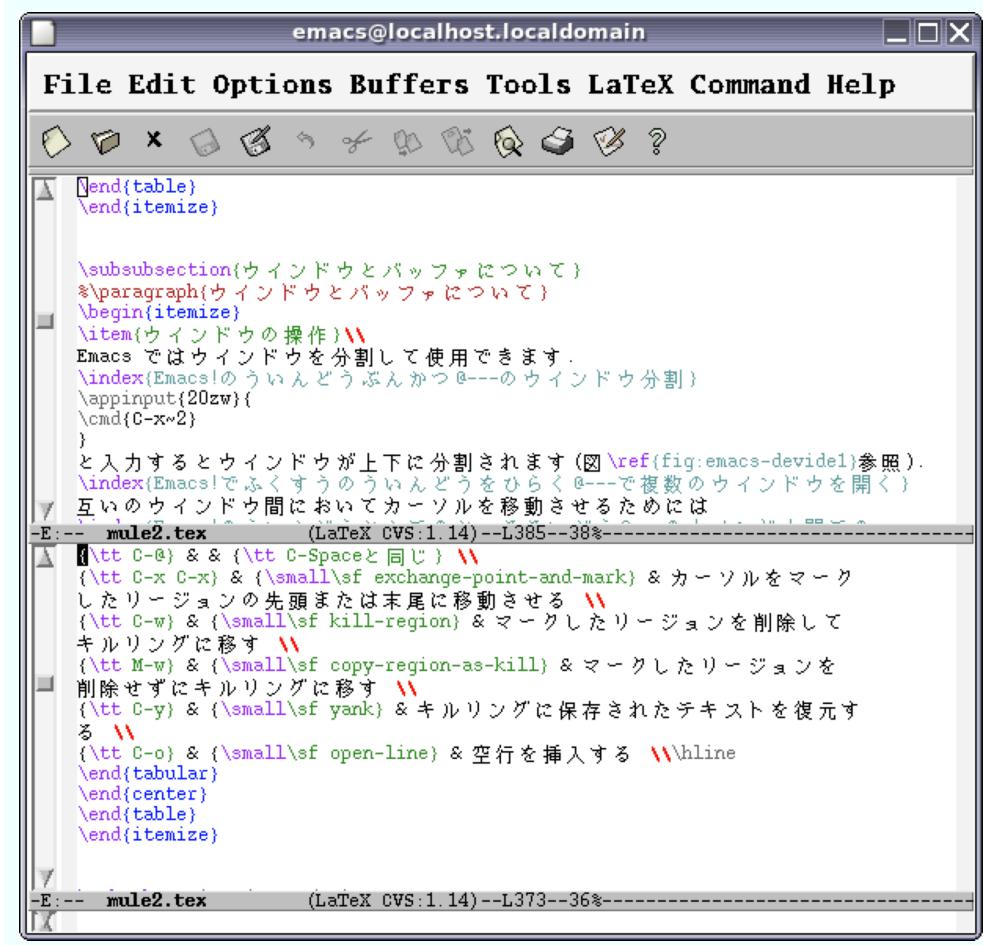


図 5.7 ウィンドウを上下に分割した場合の画面

Switch to buffer: (default
test.tex)

というメッセージが表示されます。ここで Enter を入力すると “default” に続いて表示されているバッファ（ここでは “test.tex”）に移動し、別のバッファの名前を入力すれば、指定されたバッファに移動します。

また、現在オーブンしているバッファのリストとその属性を見る場合は、

C-x C-b

と入力します。すると、図 5.8 のような画面が表示されます。

バッファリストを用いると、バッファの名前と属性を見るだけでなくバッファのセーブや削除の操作もできます。*Buffer List* ウィンドウ内でのフィールドに関する説明を表 5.6 に示します。

また、バッファリスト内での MR フィールドにおけるマークの説明を表 5.7 に示します。

バッファリスト上でバッファを操作するためには、まず *BufferList* ウィンドウにカーソルを移動

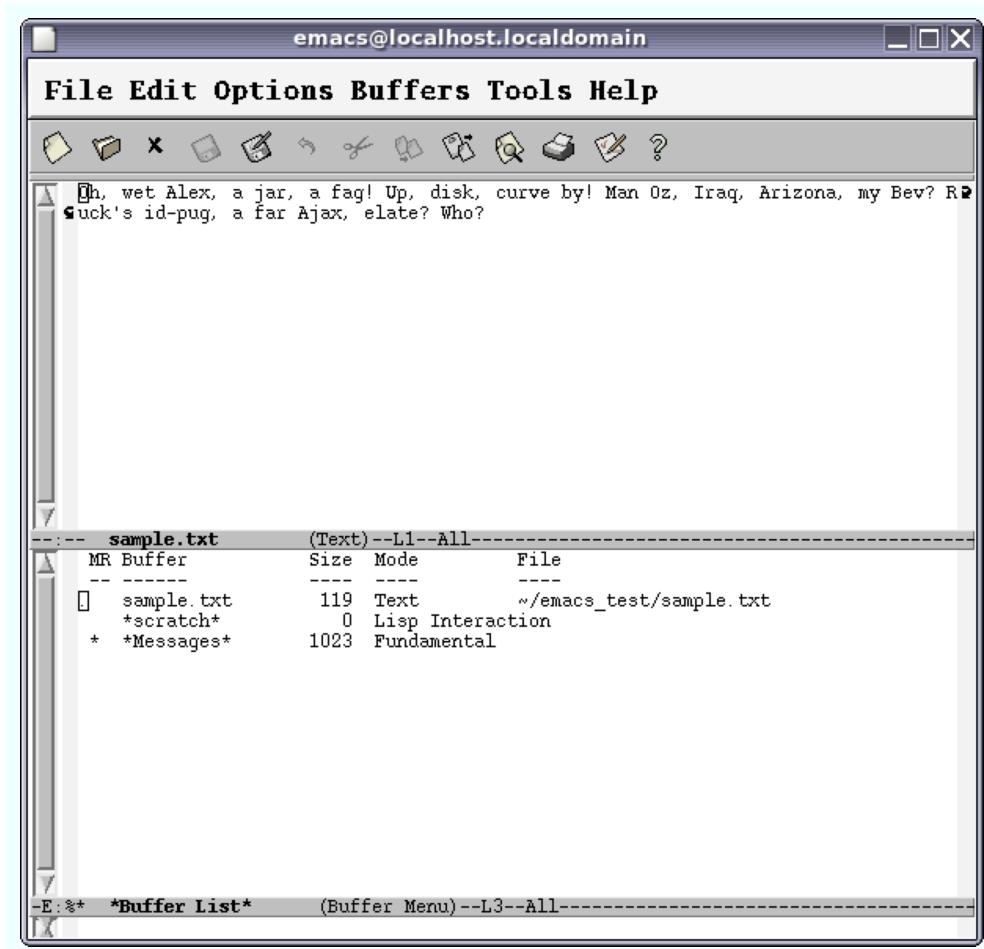


図 5.8 バッファリスト

表 5.6 バッファリスト内におけるフィールドの説明

| フィールド名 | 意味 |
|--------|-----------------|
| MR | バッファの属性に関するマーク |
| Buffer | バッファの名前 |
| Size | バッファのサイズ |
| Mode | メジャー モード |
| File | オープンしているファイルのパス |

させます。

- カーソルの操作

BufferList 内でのカーソル操作は編集の場合と同じですが “Space” で下方へ移動させることができます。

- 保存と削除

表 5.7 MR フィールドにおけるマークの説明

| バッファの属性を示すマーク | |
|---------------|---------------|
| マーク | 意味 |
| . | 現在表示中 |
| * | 変更点が存在 |
| % | 書込み不可 |
| バッファの処理を示すマーク | |
| マーク | 意味 |
| D | 削除 |
| S | セーブ |
| > | ウインドウを割り当てて表示 |

バッファに “S”(セーブ) マークをつける場合は “s” を入力します。バッファに “D”(削除) マークをつける場合は “d” , または “k” を入力します。しかしマークをつけただけでは実行されませんので、実行させるためには “x” を入力します。

- ファイルの表示

“>”(表示) マークをつける場合は “m” を入力します。これも実行させるためには “q” を入力します。

- 操作のキャンセル

またマークをキャンセルするときは “u” を入力します。

以上バッファリストに関する基本的な操作について述べましたが、他の操作についても、表 5.8 に記しておきますので試してみてください。

表 5.8 バッファリスト内の操作

| キー入力 | 動作 |
|-------|-------------------------------|
| Space | 1 行下に移動する |
| s | バッファにセーブマークをつける |
| d | バッファに削除マークをつける |
| k | “d” と同じ |
| u | マークを消去する |
| DEL | 1 つ上のバッファのマークを消去する |
| x | “s” , “d” , “k” によるマークを有効にする |
| m | バッファにウインドウ表示のためのマークをつける |
| f | バッファリストのウインドウに指定したバッファを表示する |
| o | バッファリスト以外のウインドウに指定したバッファを表示する |
| q | “m” によるマークを有効にする |

5.1.10 文字列の検索

文字列の検索（サーチ）

ファイルサイズが大きくなると、そのファイルの中からある単語や文字列を目で見つけ出すのは大変です。Emacs では検索（サーチ）を行うための機能がいくつか用意されています。基本的には次の 3 種類です。

1. 単純文字列検索

この検索方法では、検索対象となる文字列を入力すると、ファイル中で該当する文字列の先頭にカーソルが移動します。

例えば、“main”という文字列を検索したい場合は、まず

M-x search-forward

と入力します。すると、エコー領域に

Search:

と表示されるので、続けて“main”と入力すれば、文字列“main”的直後にカーソルが移動します。search-forward コマンドは、コマンドを入力した時点におけるカーソルの位置から前向きに（ファイルの末尾に向かって）検索を行います。もし該当する文字列が見つからなかった場合には、

Search failed: main

と表示されます。現時点におけるカーソルの位置から後向きに（ファイルの先頭に向かって）検索したいときは、

M-x search-backward

と入力してください。前向き検索と同様に検索したい文字列を入力すれば検索できます。search-forward コマンドは C-s RET で、search-backward コマンドは C-r RET で実行することもできます。

2. インクリメンタルな検索

C-s または M-x isearch-forward と入力してください。エコー領域に

I-search:

と表示されます。次に検索したい文字列を入力してください。単純文字列検索と違って、1 文字入力ごとに、マッチする文字列のところにカーソルが移動しているのがわかると思います。もし文字を入力している途中で、

Failing I-search: main

というメッセージが出力されたら、最後の文字を入力した時点で該当する文字列がないことを意味しています。つまり上の場合、“mai”までマッチする文字列はあっても“main”までマッチする文字列はないということです。対象となる文字列を検索し、そのまま検索を終了する場合は、C-g あるいは ESC キーを 3 回入力します。

インクリメンタル検索の特徴は、検索の文字列を途中で変更できるところです。検索文字列を入力しているときに DEL を使って文字を変更することもできます。

次に一度対象となる文字列を検索した後、終了せずに C-s を入力してください。もし該当する文字

列がファイルの中に複数ある場合、次の該当個所にカーソルが移動します。さらに C-s の入力を続けると、同じメッセージが表示され、続けて入力すると

Wrapped I-search: main

というメッセージが表示され、ファイルの先頭から最初に合致する文字列の先頭まで移動し、検索を再開します。以上、前向きのインクリメンタル検索について述べましたが、後向きのインクリメンタル検索もできます。後向きに検索する場合は、C-r を入力してください。検索中の操作は C-s の場合と同じです。

もう 1 つ便利な機能があります。それは、C-s で検索している途中で C-r を入力することにより、検索の方向を変えて同じ文字列を検索できる機能です。逆方向の検索についても同様です。

3. 正規表現を用いた検索

正規表現を用いると、検索対象となっている文字列を細かく指定することができます。まず正規表現に使用する文字とその性質を表 5.9 に挙げます。

表 5.9 正規表現に使用する文字とその性質

| | |
|----|----------------------------|
| . | “CR”(改行) 以外の任意の 1 文字にマッチする |
| * | 直前の文字や記号の任意個の繰り返しにマッチする |
| ^ | 行の先頭にマッチする |
| \$ | 行の末尾にマッチする |
| \< | 単語の先頭にマッチする |
| \> | 単語の末尾にマッチする |

この検索を行うコマンドは前向き検索の場合

M-x isearch-forward-regexp

後向きの場合

M-x isearch-backward-regexp

と入力します。

例を幾つか挙げます。今、“co”で始まり “t”で終わるような単語を検索したいとします。そこで仮に co[a-z]*t という正規表現を用意したとします。“[a-z]”は“a”から“z”までの文字のいずれかという意味です。もちろん “[abc’]”とすれば“a”，“b”，“c”，“'”の文字のいずれかという意味になります。

この正規表現では確かに，“count”，“combat”という単語を検索できます。続けて検索するときは C-s (後向きは C-r) を入力してください。しかし上の正規表現では“coordinate”，“locomotion”や“recognition”といった単語にも引っ掛かります。そこでもっと明確に文字列を指定するために、今度は \<co[a-z]*t\> という正規表現を用意します。意味は単語の先頭文字が“c”，2 文字目が“o”であり，3 文字目より“a”から“z”までの任意個の繰り返しから成る文字列が存在し、単語の末尾文字が“t”であるような文字列ということです。これにより目的の単語を

検索することができます。

表 5.10 文字列の検索のコマンド

| キー入力 | コマンド名 | 動作 |
|---------|-------------------------|------------------------------|
| C-s | isearch-forward | インクリメンタルサーチを前向きに実行する |
| C-r | isearch-backward | インクリメンタルサーチを後向きに実行する |
| C-s RET | search-forward | 単純文字列検索を前向きに実行する |
| C-r RET | search-backward | 単純文字列検索を後向きに実行する |
| ESC C-s | isearch-forward-regexp | 正規表現を用いたインクリメンタルサーチを前向きに実行する |
| ESC C-r | isearch-backward-regexp | 正規表現を用いたインクリメンタルサーチを後向きに実行する |

| インクリメンタルサーチの実行に伴うキー操作 | |
|-----------------------|----------|
| キー入力 | 動作 |
| DEL | 検索文字列の修正 |
| C-g または ESC ESC ESC | 処理の中止 |

5.1.11 文字列の置換

文字列の検索と同じく、ファイルサイズが大きくなると置換する文字を探すのは大変です。検索と同じように、置換を行うための機能が Emacs には用意されています。文字列の置換には次の 2 種類があります。

1. 単純な文字列置換

この置換方法は、変更前の文字列と変更後の文字列を指定すると、カーソルの位置より下にある変更前の文字列が全て変更後の文字列に置換されるものです。

M-x replace-string

と入力すると、エコー領域に

Replace string:

とメッセージが出力されるので、変更前の文字列を入力すると続けて

Replace string: 変更前の文字列 with:

と出力されるので、これにしたがって変更後の文字列を入力すると、文字列が置換されます。置換を行う状況にもありますが、この方法では

- 一部の文字列だけを置換する

- どの文字列が置換されたかを見る
ということができません。
2. 対話的な文字列置換

一部の文字列だけを置換したり、どの文字列が置換されたかを見るなどの要望に対応した置換に、対話的な文字列置換があります。

対話的な置換は検索の場合と同様、正規表現を使用することができます。正規表現の構成は検索の場合と全く同じです。（正規表現の構成については、正規表現を用いた検索の項を参照してください。）

対話的な置換を実行するためには、

M-x query-replace-regexp

と入力します。対話的な文字列置換とほとんど動作は同じであり、置換対象となる文字列の指定に正規表現を用いる点が異なっているだけです。置換に関するキー操作は先に述べた操作と同じです。

表 5.11 文字列の置換に関する操作

| キー入力 | コマンド名 | 動作 |
|---------|----------------------|------------------------|
| (なし) | replace-string | 単純な文字列置換を実行する |
| M-% | query-replace | 対話的な文字列置換を実行する |
| ESC M-% | query-replace-regexp | 正規表現を用いた対話的な文字列置換を実行する |

対話的な置換の実行中におけるキー操作

| キー入力 | 動作 |
|---------------------|---------------------------------|
| Space | 文字列を置換し、次の変更前の文字列の存在する場所に移動する |
| y | Space の場合と同じ |
| DEL | 文字列を置換せずに、次の変更前の文字列の存在する場所に移動する |
| n | DEL の場合と同じ |
| . | 文字列を置換した後、処理を中止する |
| C-g または ESC ESC ESC | 文字列を置換せずに、処理を中止する |

5.1.12 ファイル操作に関する便利なモード

Emacs には、ファイルを簡単に操作する環境が用意されています。

- Dired モード

Dired (DIRectory EDitor) モードは、ディレクトリの中にあるファイルを簡単に操作するモードです。Dired モードを起動する方法を以下に挙げます。Dired モード上では簡単なキー入力でファイルの読み込み、コピー、圧縮などができます。

1. C-x d または M-x direc と入力し ,

Dired (directory): ~ /

と表示された後 , 続けてディレクトリのパスを入力する.

2. C-x C-f と入力し ,

Find file: ~ /

と表示された後 , ファイル名ではなくディレクトリ名を入力する.

Dired モードが起動されたときの画面を図 5.9 に示します.

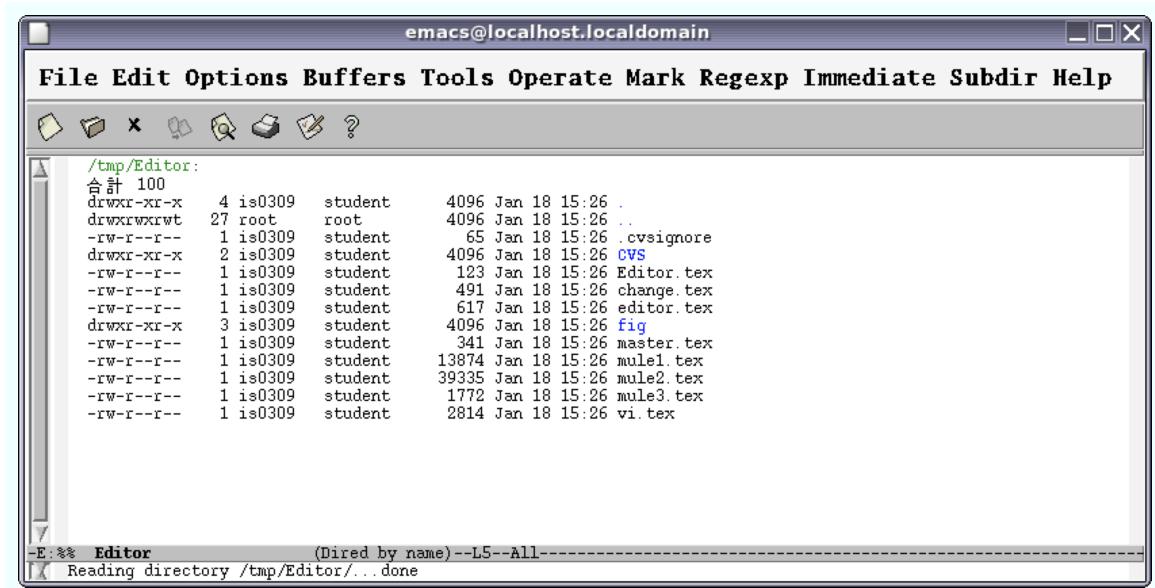


図 5.9 Dired が起動されたときの画面

Dired モードにおける簡単な操作法をいくつか挙げます. カーソルの操作に関しては , 編集の場合とほとんど同じですが , “n” , “p” でカーソルを上下に移動させファイルを指定することができます.

ファイルをオープンするときは , “e”(または “f”) を入力します. “v” でファイルをオープンすることもできますが , 書き込みは不可です.

ファイルを削除するときは , まず “d” を入力しファイルに削除マーク “D” をつけます. 削除マークをつけた後に “x” を入力すると削除が実行されます. また , “#” を入力すると Emacs のオートセーブファイルに , “~” を入力すると Emacs のバックアップファイルに削除マークをつけることができます.

ファイルをコピーするときは , “C” を入力するとエコー領域に

Copy to: ~ /

と表示されますのでコピー先のファイル名もしくはディレクトリ名を入力します. またファイル名を変更するときは “R” を入力すると , エコー領域に

Rename guide.tex to: ~ /

と表示されるので , 変更後のファイル名を入力してください. 表 5.12 に操作の一覧を挙げます.

表 5.12 Dired モードでの操作

| キー入力 | コマンド名 | 動作 |
|-------|---------------------------|--------------------------------|
| n | dired-next-line | 次のファイルにカーソルを移動する |
| Space | | nと同じ |
| p | dired-previous-line | 1つ前のファイルにカーソルを移動する |
| DEL | dired-unmark-backward | 削除マークを消しながら、1つ前のファイルにカーソルを移動する |
| e | dired-find-file | ファイルを読み込む(書き込み可) |
| f | | eと同じ |
| v | dired-view-file | ファイルの内容を見る(書き込み不可) |
| C | dired-do-copy | ファイルをコピーする |
| R | dired-do-rename | ファイル名を変更する |
| d | dired-flag-file-deletion | ファイルに削除マークをつける |
| ~ | dired-flag-backup-files | Emacs のバックアップファイルに削除マークをつける |
| # | dired-flag-auto-save-file | Emacs のオートセーブファイルに削除マークをつける |
| u | dired-unmark | 削除マークを消す |
| x | dired-do-flagged-delete | 削除マークのついているファイルを削除する |
| G | dired-do-chgrp | ファイルのグループ属性を変更する |
| O | dired-do-chown | ファイルの所有者(オーナ)属性を変更する |
| M | dired-do-chmod | ファイルのパーミッションを変更する |
| Z | dired-do-compress | ファイルを圧縮および展開する |

- shell モードの説明

Emacs を使用中、シェルコマンドを実行する必要が生じたときに、シェルを利用する方法がいくつかあります。

- 一度 Emacs を終了させてから、コマンドを実行し再び Emacs を起動する。
- Emacs を起動する際に、“emacs &”と入力しバックグラウンドで動作させる。ただし、リモートログインしているときなど、この機能が使えない場合があります。
- Emacs をサスペンド(一時停止)させて、UNIX シェルに移動してコマンドを実行し、コマンドライン上で、“fg”と入力して Emacs に戻る。ただし、この機能は端末上で起動した (“emacs -nw”) 場合に限ります。
- Emacs 上でシェルを起動し、コマンド実行後 C-x b または C-x k を入力してシェルバッファを消す。

1 の場合、Emacs を終了させごとに逐一編集中のファイルを保存し、Emacs を起動し直さなければならぬので面倒です。

2 の場合，カーソルを Emacs のウインドウから UNIX シェルのウインドウに移動させるだけで，シェルを使用することができます。Emacs に戻るときも Emacs のウインドウにカーソルを移動させるだけです。

3 の場合，Emacs を終了させるのではなくて，ファイルを保存する必要はありません。UNIX シェルの画面にもすぐ移動できますし，“fg” コマンドで，サスPENDする前の画面に戻ることができます。しかし，サスPENDしていることを忘れて，Emacs を幾つも起動すると，マシンの負荷が高くなることに注意してください。

4 の場合，Emacs 上でシェルを起動するためには，

M-x shell

と入力してください。プロンプトが表示されれば，通常の UNIX シェルと同様に使用することができます。起動直後の画面を図 5.10 に示します。

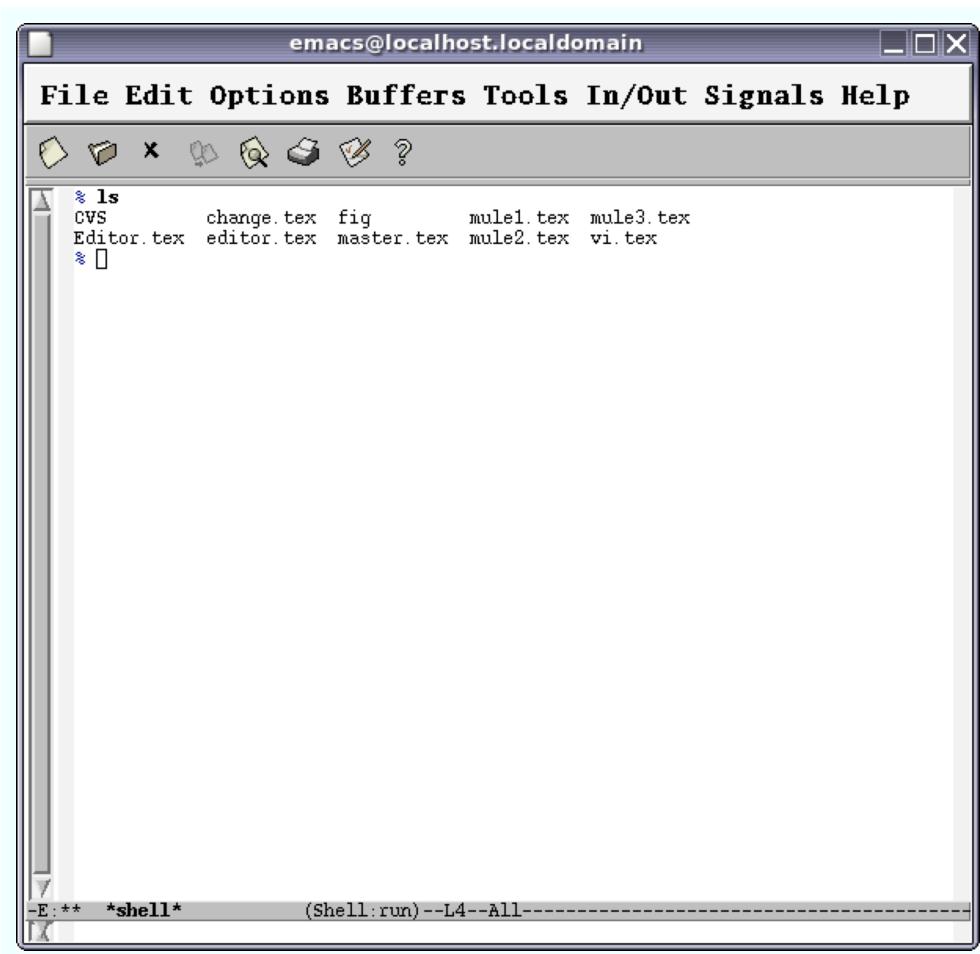


図 5.10 M-x shell 起動直後の画面

シェルモードを起動しているバッファも通常のバッファ操作と同様，C-x b でひとつ前のバッファに戻る等の操作が可能です。

Emacs シェルモードを利用する上で注意することがあります。最も注意しなければならないのは、telnet, ftp, ssh 等のネットワークコマンドを利用する場合です。通常 UNIX シェル上でこれらのコマンドを利用するとログイン名やパスワードをたずねてきます。当然ながら入力したパスワードは画面には表示されません。しかし、Emacs のシェルモードでパスワードを入力すると画面に表示されてしまいます。

- ヘルプ機能

Emacs には簡単なチュートリアル（説明書のようなもの）がついています。このチュートリアルには簡単なキー操作が書いてあります。日本語版のチュートリアルを起動したいときは、

M-x help

すると、エコー領域に

Type one of the options listed, or SPACE or DEL to scroll:

と表示されるので、“t”と入力してください。

- info システムについて

info システムは、Emacs に関するマニュアルや説明はもちろんのこと、他のツール等のドキュメントを読むためのシステムです。info システムを起動するためには

M-x info

とします。

info 上で使用する主なキー操作を表 5.13 に挙げます。

表 5.13 info での操作

| | |
|----------|-------------|
| m | メニュー項目を選択する |
| u | 1 つ前の画面に戻る |
| q | info を終了する |

“*”の横にある名前がメニュー項目になっています。info システムではドキュメントが階層的に構成されていますので、あるメニュー項目を選択すると、その下にまたメニュー項目がいくつか現われるという構成になっています。実際に項目を選択するときは、画面上で “m” を入力し

Menu item:

と出力されてから項目を入力するか、もしくは目的の項目までカーソルを移動させてから “m” を入力し

Menu item (default Emacs):

と出力されてから Enter を入力するかのどちらかで選択することができます。また “u” を入力すれば 1 つ上のメニューに戻ることができます。

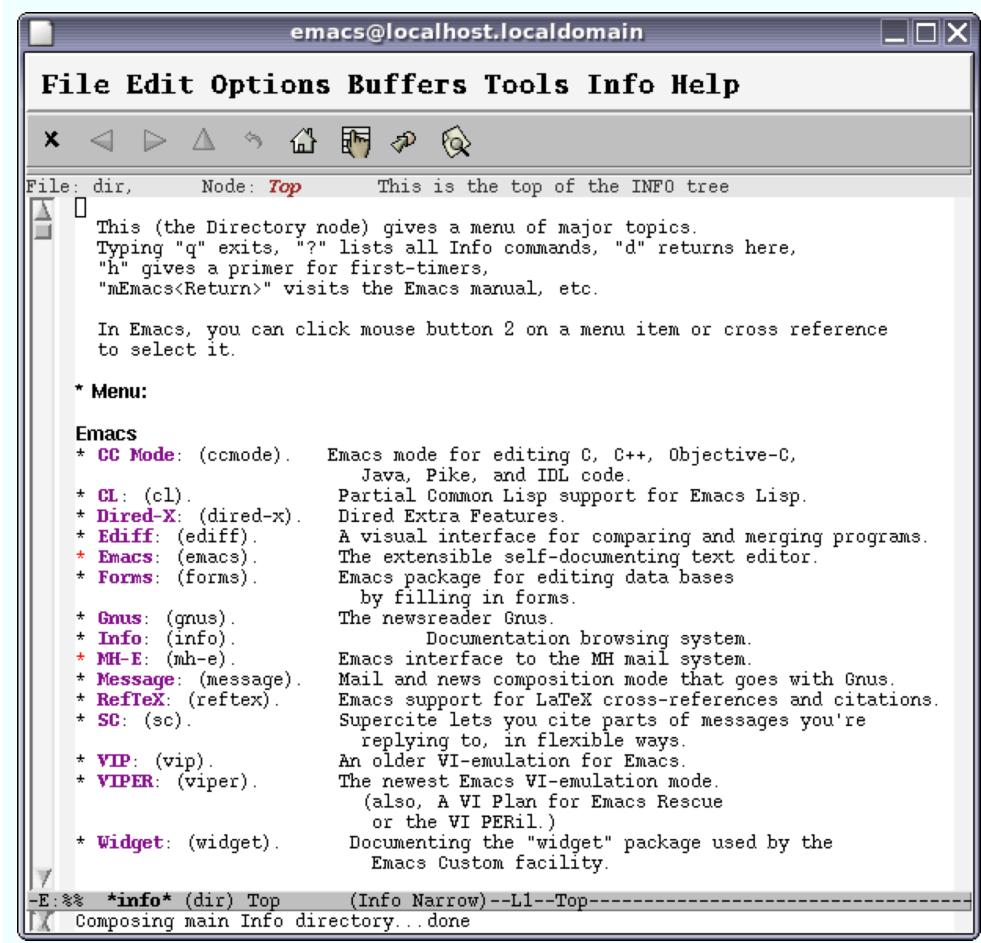


図 5.11 M-x info 起動直後の画面

5.1.13 応用例

ここでは、Emacs を操作する上で便利な設定を紹介します。Emacs の個人用の環境設定は `.emacs.el` ファイルで行います。なお環境設定ファイルとして `.emacs` ファイルも利用できますが、`.emacs.el` ファイルと `.emacs` ファイルが両方ともある場合、`.emacs` ファイルは無視されることに注意してください。

以下に設定の例を紹介します。枠内のテキストを `.emacs.el` ファイルに追加してください⁶。

- ファイルセーブしたとき、ファイル名~を作らなくする

```
(setq auto-save-default t)
(setq make-backup-files nil)
```

⁶ Emacs Lisp という Lisp プログラミング言語の方言で記述されています。

- Emacs 立ち上げ時に自動的に Dired モードに移行

```
(switch-to-buffer (find-file-noselect default-directory))
```

- メニューバーを消す

```
(menu-bar-mode nil)
```

- スクロールバーを消す

```
(scroll-bar-mode nil)
```

- メニューバーの “Read Mail” で Mew を起動する

```
(define-key menu-bar-tools-menu [rmail] '("Read Mail" . mew))
```

- 文字に色をつける

```
(require 'hilit19)
```

5.2 Emacs での日本語入力方法

Emacs 上で日本語を入力するには以下の方法があります。

- anthy.el
- SKK

ここでは主に anthy.el の使用方法を説明したいと思います。

5.2.1 anthy.el

anthy.el は、Emacs に日本語入力環境を提供するシステムです。Anthy を利用して、かな漢字変換を実現します。

1. anthy.el の起動と終了

Emacs 上で、C-¥と入力します。すると、Emacs のモード行に、<Anthy: あ> と表示され（図 5.12 参照）、ローマ字かな変換モードに入ります。これで anthy.el が起動したことになります。この状態でもう一度 C-¥ と入力すると anthy.el は終了し、Emacs のモード行から <Anthy: あ> がなくなります。

2. anthy.el を利用した漢字入力

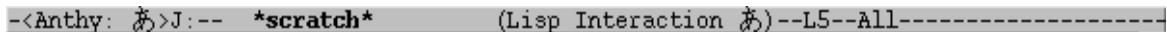
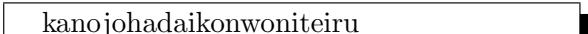


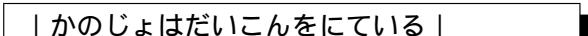
図 5.12 anthy.el の起動

上記で述べたように C-¥ でローマ字かな変換モードに入りましょう。

ローマ字仮名変換モードで文字列（ローマ字打ち）を入力すると、これから変換される部分が、かな文字で以下のように縦棒で囲まれて表示されます。

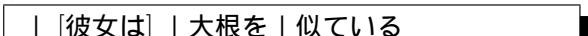


と入力すると、



と表示されます。このような状態を フェンスモード と呼びます。

この状態で、Space を入力すると適当な漢字に変換されます。この状態をかな漢字変換モードと呼びます。



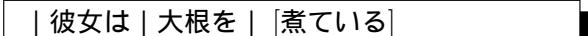
この場合、3 つの文節に分かれて、それぞれ漢字に変換されています。このとき、それぞれの文節において希望する漢字に変換されていないときは Space でさらに変換していくことになります。

最初，“彼女”の部分が反転しています。この反転している部分が、いま漢字変換を対象としている文節であることを示しています。これらの文節では、文節伸ばしや、文節縮め、文節の移動などができます。文節の操作を表 5.14 にまとめます。

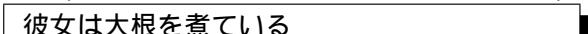
表 5.14 文節の操作

| | |
|---------|-----|
| 文節伸ばし | C-o |
| 文節縮め | C-i |
| 右の文節に移動 | C-f |
| 左の文節に移動 | C-b |

ここでは，“似ている”を“煮ている”に変換したいので、C-f を二回入力します。すると、カーソルが“似ている”的文節に移動し反転します。この状態で Space キーで次候補に変換していき、目的の状態にします。最後に望みの状態になれば、Enter キーを入力し確定します。



の状態で、Enter を入力してください。確定され、



と表示されます。

3. モードの切り替え

フェンスモードでないときに変換モードを切り替えることができます。例えば、<Anthy: あ> のときに q を入力すると入力モードが、<Anthy: ア> となって自動的にカタカナに変換される

モードになります。

4. anthy.el における基本操作

ここでは、上記で述べた部分も含めて、anthy.el における基本操作を表 5.15 としてまとめます。

表 5.15 かな漢字変換

| | | |
|-----------------|-------------|---------------------|
| anthy.el の起動と終了 | | C-¥ |
| モードの切り替え | あ から アへの切替え | q |
| | あ から Aへの切替え | l |
| | あ から Aへの切替え | L |
| | ア から あへの切替え | q |
| | ア から Aへの切替え | l |
| | ア から Aへの切替え | L |
| | A から あへの切替え | C-j |
| | A から あへの切替え | C-j |
| 入力した文字列の修正 | カーソルを左に戻す | C-b |
| | カーソルを右に進める | C-f |
| | カーソル位置の文字削除 | C-d |
| | カーソル以降を全消去 | C-k |
| | カーソルを左端へ | C-a |
| | カーソルを右端へ | C-e |
| 漢字変換 | | Space または C-n |
| 変換と確定間の操作 | 次候補 | Space または C-n |
| | 前候補 | C-p |
| | 文節伸ばし | C-o |
| | 文節縮め | C-i |
| | 右の文節に移動 | C-f |
| | 左の文節に移動 | C-b |
| | 変換中止 | C-g または Backspace |
| 確定 | | Enter , C-m または C-j |

5.2.2 SKK

SKK は、Emacs 上に高速で効率的な日本語入力環境を提供するシステムで、GPL にしたがって配布されているオープンソースソフトウェアです。SKK には、以下のような特徴があります。

- ユーザー辞書と共有辞書によるかな漢字変換
- 文体に依存しないかな漢字変換

- Emacs との高い親和性
- 再帰的辞書登録

SKK にはチュートリアルプログラムが用意されています。このプログラムを用いて操作方法をマスターすることができます。このチュートリアルプログラムは、Emacs 上で、

C-x t

または、

M-x skk-tutorial

と入力すれば起動します。詳しくはそちらを参照してください。

5.3 vi

vi は、Emacs と並び UNIX で広く使われているエディタです。ここでは、vi について簡単に説明します。豊富な機能を備えた Emacs とは異なり、vi は比較的シンプルなエディタです。オンラインマニュアルにほぼ全ての機能が記述されています。詳細についてはそちらを参照してください。

なお、vi にはいくつかの種類がありますが、以下ではオリジナルの vi の機能を強化した vim について説明します。

5.3.1 vim の基本的な操作方法

vim でファイルを編集するには、GNOME 端末や kterm の上で、次のように入力して vim を起動します。

% vim *filename*

ここで、表 5.16 のキーバインドを駆使してファイルを編集します。vim には、“カーソルを移動させるモード”と“文字を入力するモード”的 2つのモードがあります。vim の起動時は“カーソルを移動させるモード”になっていて、文字を入力したい所までカーソルを移動させます。次に“文字を入力するモード”に切り替えるコマンドを入力し、文字を入力します。カーソル位置での文字の入力が終了すると、ESC で“カーソルを移動させるモード”に戻ります。この作業を繰り返して、ファイルの編集を行います。

5.3.2 キーバインド

表 5.16 に、vim で使う基本的なキーバインドを載せておきます。このほかにも多くのキーバインドがありますが、使っていくうちに覚えていくと思います。

表 5.16 vi の基本的なキーバインド

| | |
|---------------|--|
| h | 左へ一文字分移動 |
| j | 下へ一文字分移動 |
| k | 上へ一文字分移動 |
| l | 右へ一文字分移動 |
| x | カーソル位置の文字を削除 |
| dd | カーソルのある行を削除 |
| yy | カーソルのある行をコピー |
| P | カーソルの前に削除やコピーしたテキストをペースト |
| p | カーソルの後に削除やコピーしたテキストをペースト |
| u | アンドゥ |
| i | カーソルの前にテキストを挿入 , ESC でカーソルを移動させるモードになる |
| a | カーソルの後にテキストを追加 , ESC でカーソルを移動させるモードになる |
| O | カーソルのある行の前に新しい行を追加 , ESC でカーソルを移動させるモードになる |
| o | カーソルのある行の後に新しい行を追加 , ESC でカーソルを移動させるモードになる |
| ZZ,:wq | ファイルを保存して終了 |
| :w | ファイルのセーブ |
| :q | vim の終了 |
| :q! | vim の強制終了 |

5.4 gedit

gedit は , Windows に付属しているメモ帳のようなエディタです. メモ帳に似ているため , gedit は扱い易く見えますが , 拡張性がなく , Emacs や vi と違い , 便利な機能を使用することができません. また , サーバでは gedit を使用することができない環境であることがあります. 普段から gedit は使用せず , Emacs や vi でプログラミングやレポートを書くように心掛けることで , 操作に慣れるとよいでしょう. ここでは起動方法の紹介にとどめます. 起動方法は , GNOME 端末などで次のように入力します.

```
% gedit
```

第6章

L^AT_EXによる文書作成環境

6.1 L^AT_EXとは

T_EX¹はスタンフォード大学の Donald E. Knuth 教授が制作した組版処理ソフトウェアです。T_EXの特徴として組版処理の美しさ、柔軟性・移植性の高さがあります。また、無料で配布されていることから、世界的に普及しています。特に、数式の記述能力は非常に高く、次の(6.1)(6.2)式のような数式を簡単に記述することができます。

$$\sum_{n=1}^{\infty} \iint_{|z|<1} \left[\frac{\sqrt{n+1}}{\pi} z^n - f_n(z) \right]^2 dx dy < 1 \quad (6.1)$$

$$J = \frac{\partial(f_1, \dots, f_n)}{\partial(x_1, \dots, x_n)} = \det \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix} \quad (6.2)$$

L^AT_EX²は、DEC 社の Leslie Lamport が制作したマクロパッケージです。マクロパッケージとは、マクロと呼ばれる、文書の組版に関わる新しい命令を独自に作成するための仕組みを、ひとつにまとめたものです。この L^AT_EXを使用することによって、T_EXの細かな知識を必要とすることなしに、T_EXのすばらしい文書整形を行うことができます。以下、pL^AT_EX 2_εのことを単に L^AT_EXと呼びます³。

6.2 L^AT_EXを使ってみる

本節では、L^AT_EXの使い方について、簡単な例を用いて説明します。以下の文章を L^AT_EXで処理する場合について考えます。

¹ T_EXはAMS(American Mathematical Society)の商標です。T_EXは“テック”または“テフ”と読むことが多いようです。

² “ラテック”または“ラテフ”と発音されることが多いようです。

³ pL^AT_EX 2_εとは、日本語が使えるよう改良された L^AT_EX 2_εのことです。

これは RAINBOW GUIDE に掲載されているサンプルです。

この例を用いて, LATEX のソースファイルの作成方法から, 実際に印刷するまでの過程について述べます。

まず, はじめにエディタを用いて, test.tex を作成します。LATEX のソースファイル名は, “.tex” で終わります。このファイルに対して, 図 6.1 のように入力します。

```
\documentclass[a4paper]{jarticle}
\begin{document}
これは RAINBOW GUIDE に掲載されているサンプルです。
\end{document}
```

図 6.1 test.tex のリスト

次に test.tex を DVI ファイルと呼ばれるファイルに変換します。また, この処理をコンパイルと呼びます。DVI とは device independent の略で, デバイス(出力機器)に依存しないことを表します。DVI 命令を解釈できるプレビューやプリンタドライバを用意すれば, それがどの機種で動作していようと, 同一の結果となることが保証されます。DVI ファイルを作成することによって, 印刷イメージをディスプレイで確認したり, プリンタを用いて印刷したりできます。ソースファイルの作成から印刷までの流れを図 6.2 に示します。

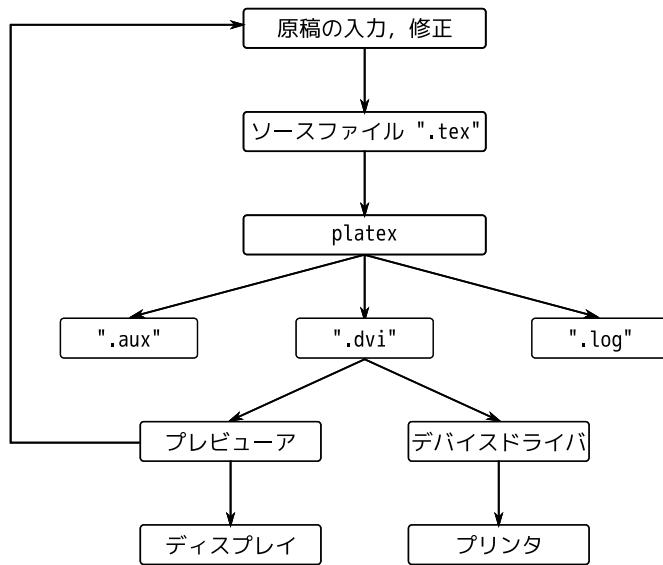


図 6.2 ソースファイルの作成から印刷まで

ソースファイルを DVI ファイルに変換するコマンドは pdflatex です。test.tex を DVI ファイルに変換する場合, 次のように入力します。

```
% plateX test.tex
```

正常に終了すると、ソースファイルと同じディレクトリに `test.dvi`, `test.aux`, `test.log` が作成されます。`test.dvi` は変換された DVI ファイルです。`test.aux` ファイルは、表や図の名前や、参考文献などをつけるときに、相互参照をするためのデータを集めたものです。また `test.log` は `plateX` のコマンドを実行したときの記録が集められています。

それでは、印刷イメージをディスプレイで確認しましょう。印刷イメージの確認には、DVI ファイルビューアの `xdvi` を使用します。`test.dvi` をディスプレイに表示するには、以下のようにコマンドを入力します。

```
% xdvi test.dvi
```

すると、図 6.3 のような印刷イメージがディスプレイに表示されます。なお、`xdvi` では、DVI ファイルの拡張子 `.dvi` を省略できます。

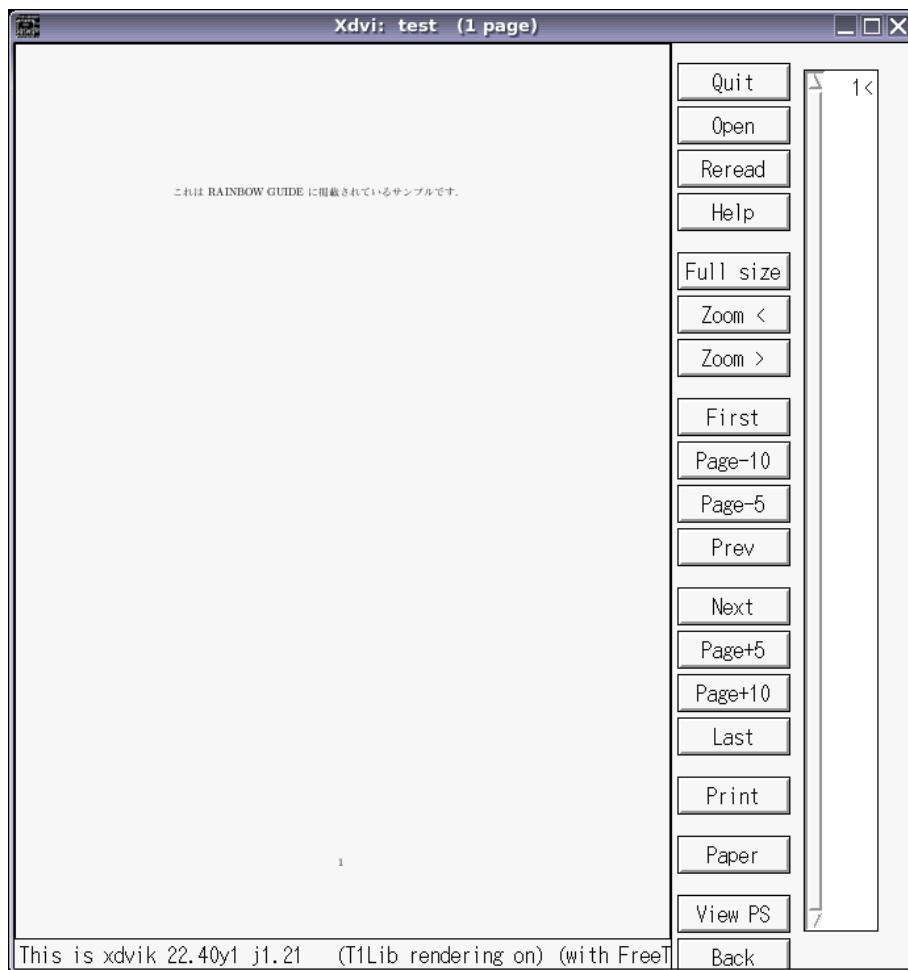


図 6.3 プレビューによる印刷イメージの表示

プレビューアでの表示結果を見て、望みどおりの表示であれば、印刷してみましょう。プリンタで印刷するためには、まず PostScript 言語で書かれたファイルへ変換しなければなりません。そして、その変換されたファイルをプリンタへ出力します。DVI ファイルを PostScript ファイルへ変換するためのコマンドは dvips です。また、ファイルをプリンタへ出力するためのコマンドは lp です。test.dvi をプリンタに出力するためには、この 2 つのコマンドとパイプを使って、以下のように入力します。

```
% dvips test.dvi | lp
```

正しく印刷できたでしょうか。また、以下のようにコマンドを入力することによって、デフォルトプリンタ以外でも印刷できます。

```
% dvips test.dvi | lp -d プリンタ名
```

-d とプリンタ名の間に空白をいれないことに注意してください。

また、dvipdfmx というコマンドを用いることによって DVI ファイルを PDF ファイルに変換できます。以下のようにコマンドを入力します。

```
% dvipdfmx test.dvi
```

6.3 L^AT_EX の使い方

L^AT_EX のソースファイルは、キーボードから入力できる文字だけを用いて作成します。しかし、書体を変更する場合に難しいキー操作をしたり、罫線を—で指定したりといったことではありません。書体を変更するには“ という書体に変更”，罫線を引くには“罫線を引く”といった意味の命令を利用します。これらのような命令をソースファイルに対して記述することによって作成します。

一般に L^AT_EX のソースファイルは、文書のスタイルを設定するコマンド \documentclass で始まり、また文書の本体は \begin{document} から \end{document} の間に記述します。また、\documentclass と \begin{document} の間に、文書スタイルの一部を変更するための場所であるプリアンブルと言われる領域があります。

```
\documentclass [< クラスオプション >] {< クラス名 >}
\usepackage{< パッケージ名 >}
    < プリアンブル >
\begin{document}
    < 文書の本体 >
\end{document}
```

図 6.4 一般的な L^AT_EX のソースファイルの形式

ここでは、文書のスタイルを指定する `\documentclass` コマンドの部分の基本的な説明だけにとどめます。

文書のスタイルを指定しておけば、文書の内容を記述するだけで L^AT_EX が自動的に文書の体裁を整えます。文書のスタイルを決定するファイルを、**クラスファイル** と呼びます。

日本語および欧文の文書のスタイルを定義したクラスファイルには以下のものがあります。

表 6.1 標準クラス

| スタイル | 機能 |
|-----------------------|--------------|
| <code>article</code> | 一般の文書形式(欧文) |
| <code>report</code> | レポートの形式(欧文) |
| <code>book</code> | 書籍の形式(欧文) |
| <code>jarticle</code> | 一般の文書形式(日本語) |
| <code>jreport</code> | レポートの形式(日本語) |
| <code>jbook</code> | 書籍の形式(日本語) |

そして、このクラスファイルは、図 6.4 のリストの一番上の行にあるように、

```
\documentclass [< クラスオプション >] {< クラス名 >}
\usepackage{< パッケージ名 >}
```

と記述します。なお `< クラスオプション >` はクラスファイルを部分的に修正することができるもので、省略しても構いません。表 6.2 に基本的なクラスオプションを示します^{*4}。なお、クラスオプションは “`11pt, titlepage`” のように、 “,” でつなげることで複数指定できます。

表 6.2 標準クラスのオプション

| オプション | 機能 |
|-------------------------------|---|
| <code>11pt,12pt</code> | 本文の文字を 11 ポイント、12 ポイントの文字で印字する |
| <code>twocolumn</code> | 文書を 2 段組にする |
| <code>titlepage</code> | <code>article, jarticle</code> スタイルで表紙を作る |
| <code>a4paper, b5paper</code> | 日本語 A4、日本語 B5 のレイアウトに設定する |

ここで空白について説明します。L^AT_EX の印刷イメージでは、ソースファイル中の空白や改行コードは空行を除いて無視されます^{*5}。しかし、実際に L^AT_EX のソースファイルを書く場合、プログラミング言

^{*4} この他にも、フリーで色々と便利なものが存在します。また、最近では市販されているクラスファイルもあります。

^{*5} まったく無視されるわけではありません。きれいな出力を得たいなら、空白 1 つにも気をつかう必要があります。

語におけるソースファイルと同じように見やすく書くとよいでしょう。他の人が見てもわかるようにするため改行やスペースを利用します。

以降、レポート作成に際し、よく使用されるコマンドについて簡単に説明します。L^AT_EX について詳しい解説をしている書籍やウェブサイトが多くありますので、より詳しい説明や、本書に載っていないコマンドについては、それらを参考にするとよいでしょう。

6.3.1 基本的なコマンド

メタデータの作成 L^AT_EX では、作成するドキュメントに、著者が誰であるかなどといった、メタデータと呼ばれる情報を付与できます。

- タイトル

タイトルの設定には、`\title{< タイトル >}` を使用します。

- 著者

著者の設定には、`\author{< 著者名 >}` を使用します。なお、タイトルおよび著者欄では任意の場所で改行できます。改行には後に説明するコマンドを使用します。

- 日付

日付の設定には、`\date{< 日付 >}` を使用します。日付の部分を空白にするか、`\today` と書くことで、L^AT_EX のソースファイルをコンパイルする日付になります。

文書作成 L^AT_EX には、文章を美しく作成するためのいくつかのコマンドがあります。以下にその代表的なコマンドを示します。

- 改行

文章の途中で改行するには、`\backslash` を使用します。

- 段落分け

文章の段落を区切るには、一行以上の空白で区切るか、`\par` を使用します。

- 改ページ

任意の部分で改ページするには、`\newpage` を使用します。

- 表紙の作成

表紙の作成には`\maketitle` を使用します。このコマンドにより、上で設定するメタデータを用いた表紙が整形されます。整形される形式は、文書のスタイルに従います。

- 目次の作成

目次を作成するには、`\tableofcontents` を使用します。

- コメント

L^AT_EX では、文書中にコメントを挿入できます。L^AT_EX ファイル中のコメントは、実際に作成されるドキュメントには表示されません。`%` を書くことで、`%` 以降の文書をコメントとみなします。

以上のことと踏まえた文書の作成例を図 6.5 に示します。

```
\documentclass[a4paper]{jarticle}
\title{\LaTeXによる文書作成}
\author{立命太郎}
\date{\today}
\begin{document}

\maketitle % 表紙の作成

\newpage
\tableofcontents % 目次の作成

\newpage
これは RAINBOW GUIDE に掲載されているサンプルです.

\par
みなさん、\LaTeXでレポートを書きましょう。
\end{document}
```

図 6.5 L^AT_EX による文書作成

箇条書き L^AT_EX で箇条書きをしたい場合は、itemize 環境あるいは enumerate 環境を使用します。普通の・(中点)などを用いる箇条書きの場合は itemize 環境を、番号が振られる箇条書きの場合は enumerate 環境を使用します。箇条書きの要素を示すために、\item を用います。

- Linux
 1. RedHat
 2. Debian
 3. Gentoo
- Windows
- Mac OS

このような箇条書きの場合、図 6.6 のように記述します。

数式の書き方 L^AT_EX には、この章のはじめに紹介したとおり、複雑な数式ですらコマンドすべて記述する仕組みがあります。細かい数式のコマンドに関しては、専門の書籍を参照するか、インターネットで検索してください。ここでは文書中に数式を挿入する方法を紹介します。

まず文の途中に数式を挿入する方法について説明します。文の途中に数式を挿入する場合、\$ を使用します。図 6.7 のように書くことで、 $E = mc^2$ といった数式を簡単に挿入できます。

また、別立ての数式の場合、数式を \[, \] で囲むか、equation 環境あるいは eqnarray を用います。eqnarray 環境のみ、数式が複数行にまたがる場合に使用します。equation 環境および eqnarray を用いた場合、自動的に数式の番号が振られるようになりますが、equation* のようにアスタリスクを付ける

```
\begin{itemize}
\item Linux
\begin{enumerate}
\item RedHat
\item Debian
\item Gentoo
\end{enumerate}
\item Windows
\item Mac OS
\end{itemize}
```

図 6.6 箇条書きの書き方

$E=mc^2$ といった数式を簡単に挿入できます。

図 6.7 文の途中への数式の挿入

ことで番号が振られないようになります。

$$\sum_{k=1}^n k = \frac{1}{2}n(n+1)$$

$$\int_0^{10} x dx = 50 \quad (6.3)$$

このような数式の場合、図 6.8 のように記述します。

```
\[
\sum_{k=1}^n k = \frac{1}{2}n(n+1)
\]
\begin{equation}
\int_0^{10} x dx = 50
\end{equation}
```

図 6.8 別行立ての数式の挿入

表の組み方 LATEX ではコマンドによって表を組むことができます。コマンドの組み合わせによっては複雑な表を組むことも可能ですが、ここでは簡単な表の作成方法について示します。

このような表 6.3 を考えます。一般的に表の作成には `table` 環境、`center` 環境、`tabular` 環境と、`caption` コマンド、`label` コマンドを用います。

表 6.3 表組みの例

| 左詰め | 中央 | 右詰め |
|-----|----------|---------|
| 幅は | 自動調整されます | 空白でも大丈夫 |

table 環境は表を配置し , center 環境で表をドキュメントの中央に配置し , tabular 環境で表を作成します . また caption コマンドで表の名前を付け , label コマンドで表への参照となるラベルを作成します . 表の作成の例を , 図 6.9 に示します .

```
\begin{table}[htbp]
\begin{center}
\caption{表組みの例}
\label{table:example_table}
\begin{tabular}{l|c|r}
左詰め & 中央 & 右詰め \\
\hline
幅は & 自動調整されます & \\
\hline
& & 空白でも大丈夫
\end{tabular}
\end{center}
\end{table}
```

図 6.9 表の組み方

tabular の横の `{l||c|r}` は , それぞれの行を左詰めに書くか (l) , 右詰めに書くか (r) , あるいは中央に書くか (c) を示します . また縦線 (|) は行間の罫線を示します .

それぞれの列では , 行の区切りの指定に & を使用します . また , 行の終端では , 改行するための \\ を忘れないでください . そして , 横の罫線を引くために , hline コマンドを使用します .

図表番号の参照 表を作成するときに , label コマンドを使用し , 表のラベルを作成しました . このラベルを用いることで , 図や表の番号の参照を容易にします . 図表番号の参照には , ref コマンドを使用します . 図 6.10 のようにして図表番号を参照します .

図 `\ref{fig:ref_label}` のようにして図表番号を参照します

図 6.10 ラベルの参照

なお，“図”や“表”は自分で記述する必要があります。

ラベルは一意な名前でなければならず，同じ名前のラベルが文書内で複数回出現すると，コンパイルでエラーが発生することがあります。また，ラベルは図表だけでなく，章や節，数式に加え箇条書きの要素と，あらゆるものに対して作成できます。

ラベルの便利な点として，図表の追加による図表番号の変更に自動で対応できることが上げられます。L^AT_EX では図表番号が自動で振られますが，図表の追加により図表番号が変更されようとも，その参照の番号は自動的に変更されます。

続きを読む Web で 以上で，レポート作成においてよく利用されるコマンドの説明を終わります。繰り返しになりますが，より詳しいコマンドの使用方法などは，専門の書籍を参照するか，インターネットで検索してください。

6.4 AUC T_EX

AUC T_EX は Per Abrahamsen 氏が開発した，Emacs において，L^AT_EX のソースファイルを書くときに，とても便利な環境を与えてくれる Emacs Lisp パッケージです^{*6}。AUC T_EX を使用すると，Emacs の内側から L^AT_EX を動かしたり，L^AT_EX に関連のあるツール群を動かしたりすることができます。

AUC T_EX は非常に頻繁にバージョンアップが行われ，デフォルトのキーバインドも大幅に変更されたりするので注意が必要です。まず，AUC T_EX を利用するために，下の1行をホームディレクトリにある.emacs.el ファイルに追加してください。

```
(require 'tex-site)
```

日本語を使う場合，以下の設定も書いておくと便利です。

```
(require 'tex-jp)
(setq TeX-default-mode 'japanese-latex-mode)
(setq TeX-force-default-mode t)
(setq-default japanese-LaTeX-default-style "jarticle")
(setq-default LaTeX-default-options '("a4paper"))
(setq-default japanese-LaTeX-command-default "pLaTeX")
```

以降の説明は，これらの設定がなされているものとします。

AUC T_EX の基本的なコマンドの使い方 では，基本的なコマンドを使って簡単な L^AT_EX のソースファイル(図 6.11 のソースファイル)を書いてみましょう。まず，C-x C-f で新しいファイル名を入力し，ファイルを開きます。ここでは，ファイル名の拡張子を“.tex”にします。モード行に“LaTeX”と表示されます。

^{*6} 同様な機能を果たす Emacs Lisp パッケージに cmutex, 野鳥(yatex)があります。

```
\documentclass[a4paper]{jarticle}
\begin{document}

\section{はじめに}
これは、サンプルです。
\subsection{使い方}
AUC \TeX{} は便利です。
\end{document}
```

図 6.11 サンプル・ソースファイル

文書スタイルの入力 6.3 章においても説明したように、 \LaTeX のソースファイルは一般に、文書のスタイルを決定するコマンド `\documentclass{jarticle}` で始まります。これを入力するために、以下のように入力します。

C-c C-e

また次のように入力することによっても、文書スタイルを決定することができます。

M-x LaTeX-environment

すると、ミニバッファにおいて、以下のように表示されて入力待ちの状態になります。

Environment type: (default document)

C-c C-e は環境を入力するコマンドで、ここでは入力する環境の種類、具体的には `\begin{ }` のかっこの中身に対して入力待ちをしています。この場合においては、`document` と入力すればよいのですから、そのまま Enter キーを入力します。

Enter キーを入力すると、ミニバッファに、次のように表示され、入力待ちとなります。

Document style: (default jarticle)

これは、文書のスタイル (`\documentclass[]{} の { } の部分`) に対して入力待ちをしています。この場合においては、`jarticle` スタイルとしますので、そのまま Enter キーを入力します。

最後に、ミニバッファに次のように表示され、入力待ちの状態になります。

Options: a4paper

これは、クラスオプション (`\documentclass[]{} の [] の部分`) に対して入力待ちをしています。通常、A4 の紙を使用するので、そのまま Enter キーを入力します。すると、文書スタイルが入力され、カーソルもちょうど `\begin{document}` の下に表示されます。

セクション・コマンドの入力 次に，本文を書いていきます。まずははじめに章立てを行います。`\section{ }` の部分です。これを入力するには，

C-c C-s

または

M-x LaTeX-section

と入力します。

6.3章では，詳しく説明しませんでしたが，章立てをするコマンドには，`section` や `subsection` などのようなレベルが存在します。さて，C-c C-s を入力すると，ミニバッファに以下のように表示され，入力待ちの状態になります。

Select level: (default section)

現在，デフォルトが `section` なので，そのまま Enter キー を入力します。すると，ミニバッファに以下のように表示され，入力待ちの状態になります。

What title:

ここでは，適当なタイトル名を考えて入力します。図 6.11 の通り，“はじめに”と入力して，Enter キー を入力します。すると，ミニバッファに以下のように表示され，入力待ちの状態になります。

What label: sec:

`label` についての説明は省略することにします。`label` の詳細については本書では省略します。ここでは，そのまま Enter キー を入力することによって，`label` をつけることにします。

環境の入力 文書スタイルの項で使用した C-c C-e を利用することによって，様々な環境の入力をすることができます。例えば，`itemize` 環境を入力したい場合，`Environment type:` のところで，`itemize` と入力することによって実現できます。また，`table` 環境を入力したければ，同様に `table` と入力することで実現できます。

外部コマンドの実行 図 6.11 で示されたソースファイルを L^AT_EX で処理する方法について説明します。処理のために，以下のように入力します。

C-c C-c

また，以下のような入力も可能です。

M-x TeX-command-master

すると、ミニバッファにおいて次のように表示され、入力待ちの状態となります。

Command: (default pLaTeX)

このとき、default で出力されているコマンドは pLaTeX となっています。ここで plateX と入力し、Enter キーを入力すると、pLaTeX が起動され、バッファの内容が処理されます。

すると、エコー領域にはまず次のように出力されます。

Type 'C-c C-l' to display results of compilation.

その後、もしソースファイルが正しく入力されていれば、次のように出力されます。

pLaTeX: successfully formatted {1} page.

次に以下のように入力します。

C-c C-l

また次のような入力も可能です。

M-x TeX-recenter-output-buffer

このとき、ソースファイルにエラーがあると、ミニバッファに以下のように表示されます。

pLaTeX errors in '* /work/latex/rainbow/sample output*. Use C-c ' to display

このような表示が出力された場合、表示通り、

C-c '

または

M-x TeX-next-error

と入力します。すると、Emacs のバッファが上下に分割され、上のバッファに対してソースファイル、下のバッファに対してエラーメッセージが出力されます。この場合、カーソルはソースファイルのエラーの場所を指しています。このエラーメッセージを参考にしながら、ソースファイルを修正します。

修正が終わったら、C-x C-s でファイルを保存して、もう一度 C-c C-c と入力します。

なお、label などを使用している場合には、エコー領域に、

You should run LaTeX again to get references right, {1} page.

と表示されることがあります。この場合はもう一度 LaTeX を実行することによって、正常終了することができます。

さて、ソースファイルを正しく処理できたら、もう一度 C-c C-c と入力してください。今度は、ミニバッファには

Command: (default View)

と表示され、入力待ちの状態になります。デフォルトで表示されている View はプレビューアを起動するコマンドです。そのまま Enter キー を押してみましょう。ミニバッファに以下のように表示され、入力待ちとなります。

View command: xdvi sample.dvi

これは、プレビューアを起動するために、xdvi sample.dvi というコマンドを実行するということを意味しています。ファイル名に間違いがなければ、Enter キー を入力します。すると、プレビューアが起動されます。

ここまで、外部コマンドの実行について述べてきました。実際に AUC-T_EX 利用する場合において、ソースファイルに間違いがなければ、C-c C-c と Enter キー の入力だけで、L^AT_EX による処理・プレビューアによる印刷イメージ表示ができます。

表 6.4 で表されるような数個のコマンドを覚えることによって、L^AT_EX のソースファイル作成のすばらしい環境を手に入れることができます。ぜひ、AUC-T_EX を使用されることをお薦めします。

表 6.4 よく使用される AUC-T_EX のコマンド

| キー入力 | 動作 |
|---------|--|
| C-c C-e | 環境の入力 |
| C-c C-s | セクションの入力 |
| C-c C-c | hoge 外部コマンドの実行 |
| C-c C-l | L ^A T _E X の処理の表示 |
| C-c ‘ | エラー箇所の表示 |

第7章

図・グラフの作成

本章では、RAINBOW の Linux 環境を用いて、図やグラフを扱う方法について説明します。UNIX 環境における、図やグラフを扱うソフトウェアは様々です。本章では、レポート作成をはじめとした、普段の学生生活において使われるソフトウェアについて説明します。

- 図表の作成 - Tgif
- グラフの作成 - gnuplot
- 画像の表示/変換 - ImageMagick
- 画像の編集 - GIMP

それぞれのソフトウェアの使い方の説明は必要最小限にとどめています。よりくわしい使い方を知りたい場合は、マニュアルページを参照したり、インターネットで検索したりして調べると良いでしょう。

7.1 図を作成する - Tgif

ここでは、図を作成する方法について説明します。図を作成するために、Tgif と呼ばれるプログラムを利用します。Tgif は William Chia-Wei Cheng 氏が開発した作図ツールです。

7.1.1 図を作成する

ここでは、Tgif の簡単な使い方を知るために、簡単な図の作成し、保存することを例にして説明します。

Tgif の起動

まず、Tgif を起動します。Tgif を起動するには、コマンドラインで次のように入力します。

```
% tgif
```

すると、次のように Tgif のウインドウが開きます。(図 7.1) このウインドウの上側と左側あるツールバーを操作して、中央のキャンバスに図を描いていきます。

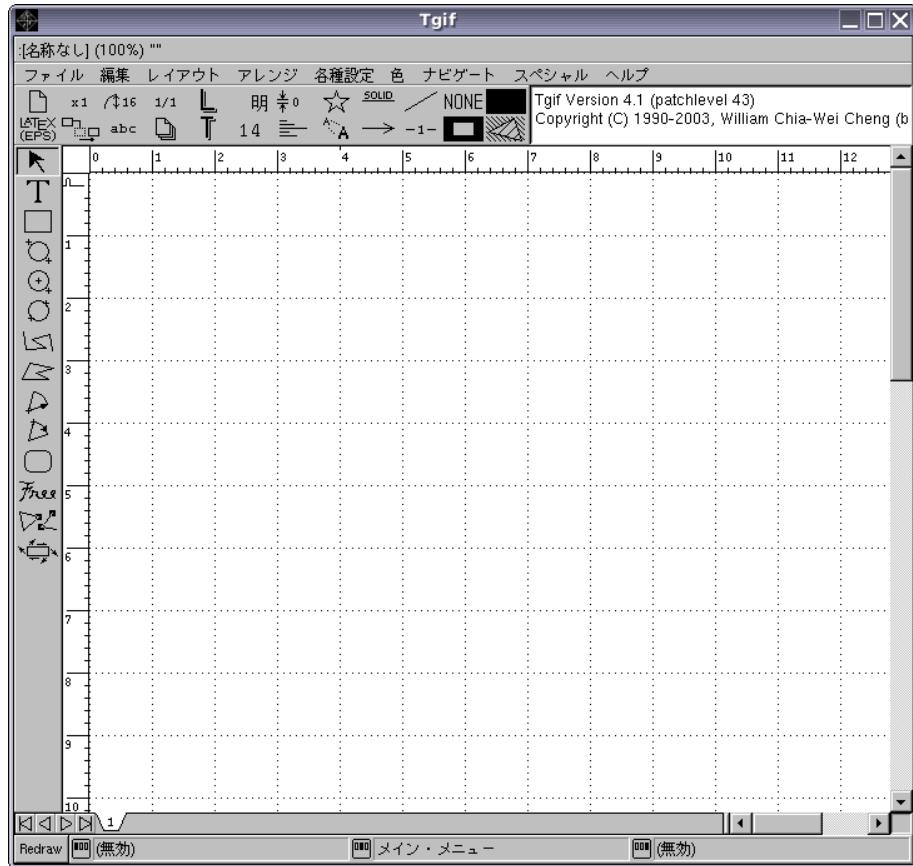


図 7.1 Tgif の起動画面

図形を描く

四角形を描くことを例として図形の描き方について説明します。まず、ウインドウ左側のツールバーにある“四角形描画モード”アイコンを左クリックします。この状態で、キャンバス上をマウスでドラッグすることによって四角形を描く事ができます。(図 7.2)

四角形と同様に円や多角形を描くことができます。ウインドウ左側のアイコンをクリックして、ためしてみてください。

図形を選択/移動/拡大する

一度描いた図形を選択することで、選択した図形に対して移動や拡大をすることができます。まず、ウインドウ左側のツールバーにある“オブジェクトの選択/移動/拡大縮小モード”をクリックします。この状態で、すでに描いた図形をクリックすると図形が選択されます。(図形のまわりにが表示されます。) この をドラッグすることで、図を拡大縮小することができます。また、以外の線の部分をドラッグすることで図を移動することができます。

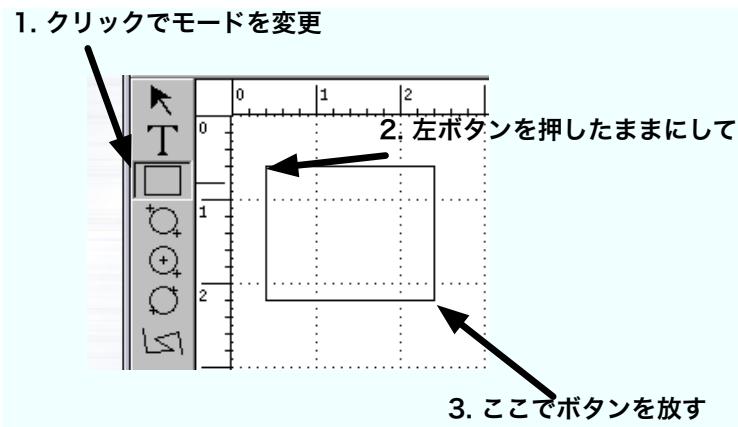


図 7.2 図形を描く

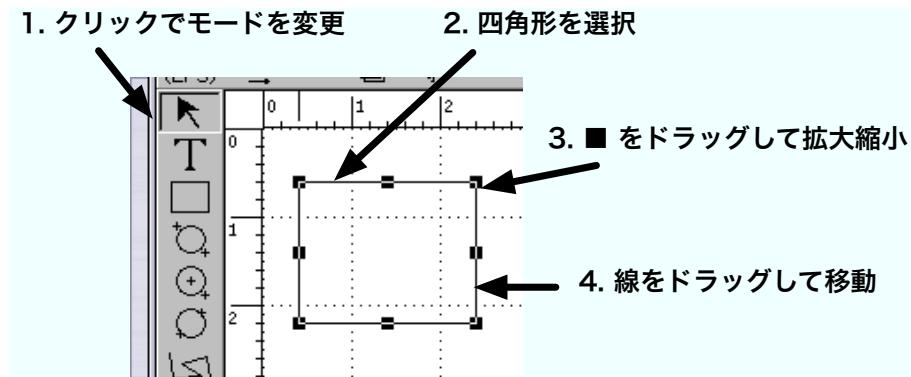


図 7.3 図形の選択/移動/拡大

文字を入力する

図形と同様に文字を入力することができます。まず、ウインドウ左側のツールバーにある“テキスト描画モード”をクリックします。この状態でキャンバスをクリックするとカーソルが現れ、キーボードから文字を入力することができます。入力が終了する場合、ウインドウ左側のツールバーにある“オブジェクトの選択/移動/拡大縮小モード”をクリックします。(図 7.4)
日本語を入力したいときには、Shift-Space を押します。アルファベット入力にもどりたいときは、同様に Shift-Space を押します。

7.1.2 作成した図の保存

図を保存するには、ウインドウ上部の“ファイル”メニューから“保存”をクリックします。表示されたダイアログに対してファイル名を入力した後、Enter キーを入力することによって、作成した図を保存することができます。

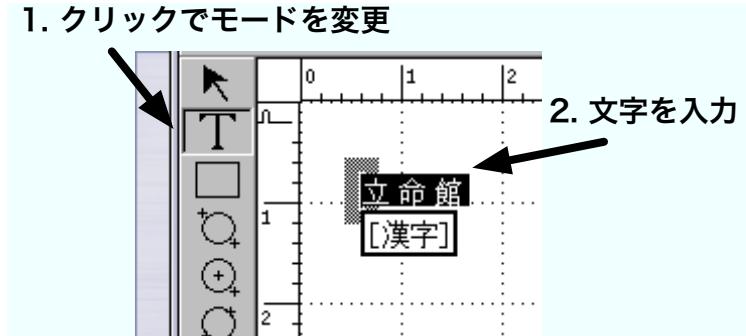


図 7.4 文字を入力する

ダイアログの“ワーキングディレクトリ”部分に表示されている場所に保存されるので注意してください。

7.1.3 L^AT_EX での図の利用

Tgif で描いた図は L^AT_EX のソースファイルの中に取り込むことができます。

ウインドウの上側図 7.5 で示される領域があります。この部分を選択ウインドウと呼びます。まず、選択ウインドウの下段左端のボタンにおいて LATEX(EPS) と表示されているかどうかを確認します。LATEX(EPS) 以外が表示されている場合、このボタンを複数回クリックすることによって LATEX(EPS) に変更します。

この状態でウインドウ上部の“ファイル”メニューから“印刷”をクリックします。すると, ****.eps として、ファイルが出力されます。このファイルは EPS (Encapsulated PostScript) と呼ばれる形式のファイルで、L^AT_EX の本文中に取り込むことができます。

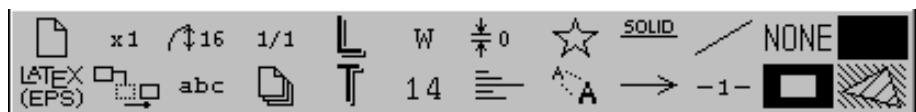


図 7.5 選択ウインドウ

EPS 形式のファイルを L^AT_EX の本文中に取り込むためには、graphics パッケージを使用します。このパッケージを使用するためには、以下のように記述します。

```
\documentclass{jarticle}
\usepackage{graphics}
```

このとき、以下のように文中で記述することによって、EPS 形式のファイルを図として取り込むことができます。

```
\includegraphics{filename.eps}
```

通常，EPS 形式のファイルを本文中に取り込む場合，`figure` 環境の中で使用します．例えば，`figure` 環境中で，センタリングをして，`sample.eps` を 50% の大きさで取り込む場合は以下のように記述します．

```
\begin{figure}[h]
\begin{center}
\scalebox{0.5} {\includegraphics{sample.eps}}
\end{center}
\end{figure}
```

7.2 グラフを作成する - gnuplot

ここでは，グラフの作成について説明します．グラフを作成するために，gnuplot と呼ばれるプログラムを利用します．gnuplot は Thomas Williams 氏と Colin Kelley 氏が開発した対話的にグラフを描くプログラムです．関数 ($x * x + x$ や $\cos x$ など) や離散的な数値の軌跡を描くことができます．

7.2.1 グラフを描く

gnuplot の起動

まず，gnuplot を起動します．gnuplot を起動するには，コマンドラインで次のように入力します．

```
% gnuplot
```

すると gnuplot のプロンプトが現れます．

```
% gnuplot
G N U P L O T
~ 中略 ~
Terminal type set to 'x11'
gnuplot>
```

以上で，gnuplot が立ち上りました．次に，実際にグラフを描くことについて説明します．

通常の関数のグラフを描く

グラフを描く場合，`plot` というコマンドを使用します．2 次元のグラフを描くときには，関数 $y = f(x)$ の $f(x)$ の部分だけを `plot` の引数として与えます．例えば， $y = x$ のグラフを描くためには，以下のコマンドを入力します．

```
gnuplot> plot x
```

$f(x)$ が演算子の入った数式であるとき、引数としては C や BASIC, FORTRAN で使用できる形式の式を入力することができます。例えば、 $y = x^2 + \cos x$ のグラフが描きたいときは、以下のコマンドを入力します。

```
gnuplot> plot x**2+cos(x)
```

3 次元のグラフを描くときは、splot というコマンドを使用します。例えば、 $z = x^2 + y$ のグラフを描きたいときは、以下のコマンドを入力します。

```
gnuplot> splot x**2+y
```

グラフの結果はウインドウに表示されます。(図 7.6)

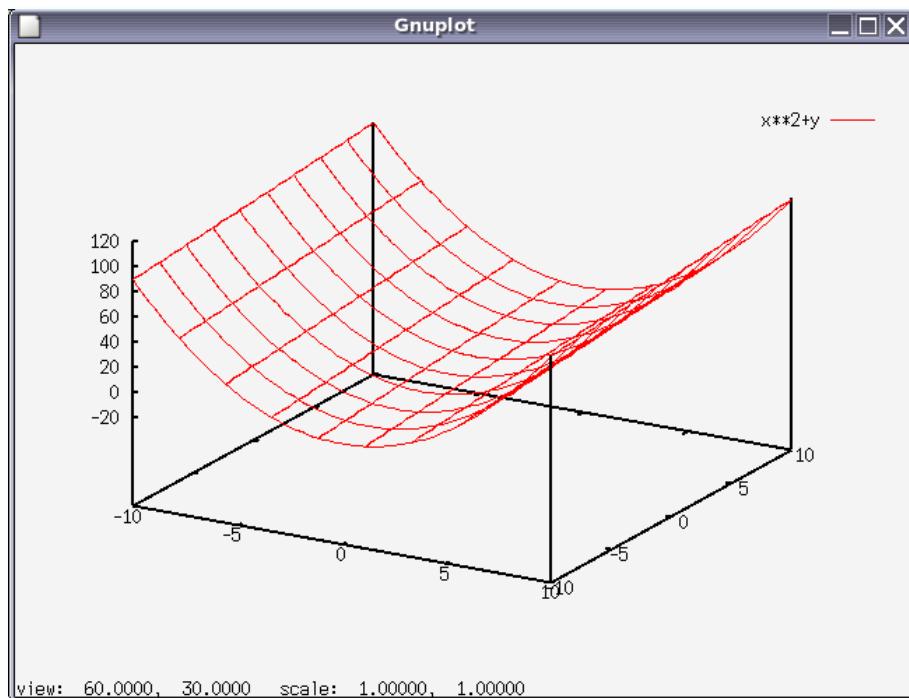


図 7.6 グラフの表示

媒介変数表示の関数のグラフを描く

媒介変数表示の関数をグラフに描く場合、媒介変数表示モードを利用します。まず、以下のコマンドを入力することによって、媒介変数表示モードに移行します。

```
gnuplot> set parametric
```

ここで plot の引数として x 座標、 y 座標に対応する関数をコンマ区切りで与えてやれば媒介変数 t の関数のグラフを描くことができます。例えば、半径 1 の円 ($x = \cos t, y = \sin t$) のグラフを描きたいときは、以下のコマンドを入力します。

```
gnuplot> plot cos(t),sin(t)
```

媒介変数表示モードを終了する場合は、以下のコマンドを入力します。

```
gnuplot> set noparametric
```

データのグラフを描く

データのグラフを描く場合、グラフとして描くデータのファイルをあらかじめ用意します。このファイルには、データを以下のように、1列目を x 座標の値、2列目を y 座標の値として、2列で記録します。ここでは、仮にファイル名を *data1.dat* とします。

```
10 20  
15 25  
25 30
```

これを、グラフとして描くためには、以下のコマンドを入力します。

```
gnuplot> plot "data1.dat"
```

グラフ上に点が3点打たれたウインドウが現れます。点の順番に線を引くようにしたければ、ファイル名の後に *with line* を付与します。すなわち、実際の入力は以下のコマンドになります。

```
gnuplot> plot "data1.dat" with line
```

7.2.2 グラフのファイルへの出力

グラフをプリンタに出力したり、*LATEX* の本文中に取り込むようにするために、グラフをファイルに出力する方法があります。単純にグラフをプリンタに出力したいときは、グラフを PostScript 形式でファイルに出力します。出力形式を PostScript にするためには以下のコマンドを入力します。

```
gnuplot> set terminal postscript
```

また、出力先を指定するコマンドによって、ウインドウの出力の代わりにファイルを出力することができます。*tmp.ps* というファイルを出力する場合、以下のコマンドを入力します。

```
gnuplot> set output "tmp.ps"
```

LATEX の本文中に取り込むようにしたいときは、前でも述べましたが EPS 形式でファイルに出力します。出力形式を EPS にするためのコマンドは、

```
gnuplot> set terminal postscript eps
```

です。あとは同様に、出力先のファイルを指定し、グラフを描いてください。(出力ファイル名の拡張子は *.eps* にしておきましょう。)

7.2.3 Help 機能

本章では、基本的な使い方についてしか述べませんでしたが、gnuplot には親切な Help 機能が存在します。Help 機能を使用する場合、以下のコマンドを入力します。

```
gnuplot> help
```

7.2.4 終了の方法

gnuplot を終了する場合、以下のコマンドを入力します。

```
gnuplot> quit
```

7.3 画像を表示/変換する - ImageMagick

本章では、画像を表示したり、変換したりする方法について説明します。本書ではこれらのために ImageMagick と呼ばれるプログラムを利用します。ImageMagick は画像ファイル表示ツール（イメージビューア）です。画像を表示する以外にも、画像形式の変換などを行うことができます。

7.3.1 画像を表示する

画像を表示するには、ImageMagick によって用意されている display コマンドを利用します。display コマンドで画像を表示する場合、コマンドラインに次のように入力します。ここでは開きたい画像のファイル名を仮に *filename.jpg* とします。

```
% display filename.jpg
```

画像は図 7.7 のように表示されます。

display コマンドは多くの画像形式に対応しています。Tgif で作成した eps ファイルなども表示できるので便利です。

表示されている画像を左クリックしたり、右クリックすることで、ImageMagick の様々な機能を呼び出すことができます。

7.3.2 画像を変換する

ImageMagick に用意されている convert コマンドを利用することによって簡単に画像の形式を変更することができます。例えば JPEG 形式の画像を EPS 形式の画像に変換する場合、コマンドラインに次のように入力します。このとき、仮に入力を *filename.jpg*、また出力を *filename.eps* とします。

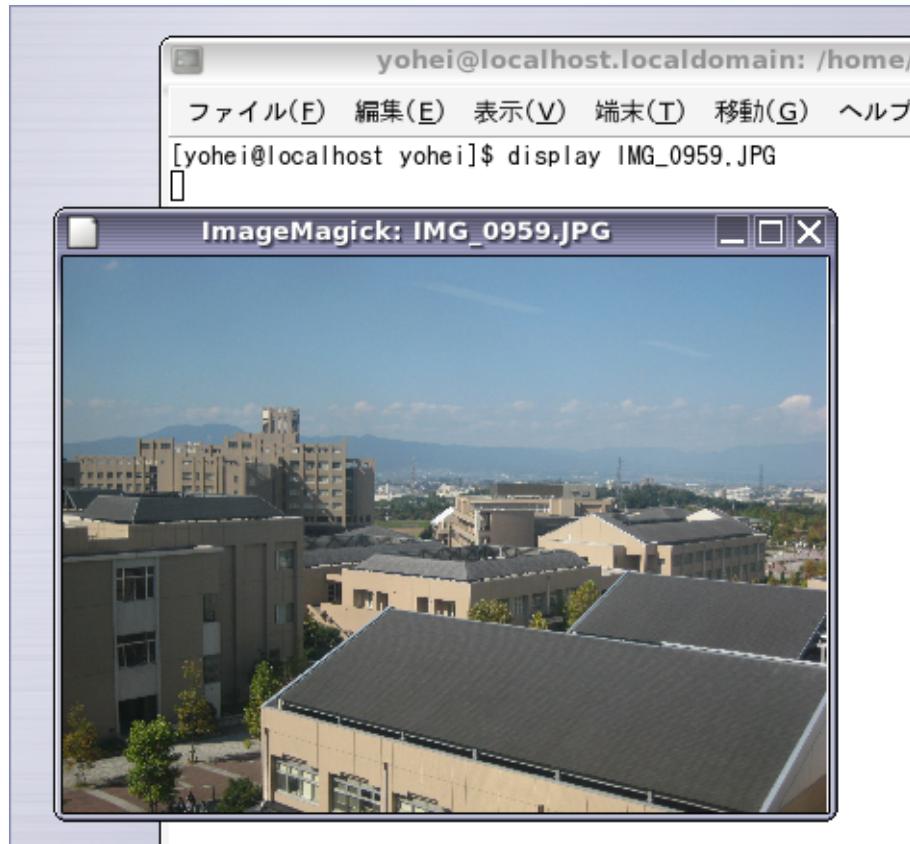


図 7.7 画像の表示

```
% convert filename.jpg filename.eps
```

このような方法によって、レポートにおいて JPEG 画像を使用できるようになります。

convert コマンドは他にも、画像のサイズを変更したり、色調を変えたりといった、多くの機能を持っています。

7.4 高度な画像編集を行う - GIMP

ここでは、高度な画像編集を行う方法について説明します。そのために、GIMP (GNU Image Manipulation Program) と呼ばれるプログラムを利用します。GIMP は Spencer Kimball 氏と Peter Mattis 氏が開発した、グラフィックや写真を処理することを目的としたプログラムです。Adobe Photoshop の代替として使用できることを目的として開発されています。

7.4.1 画像編集を行う

ここでは GIMP を使って簡単な画像編集を行うことについて説明します。GIMP を用いて指定の画像の編集を行う場合、コマンドラインに次のように入力します。ここでは開く画像を仮に *filename.jpg* とします。

```
% gimp filename.jpg
```

GIMP が *filename.jpg* を開いた状態で起動します。(図 7.8)

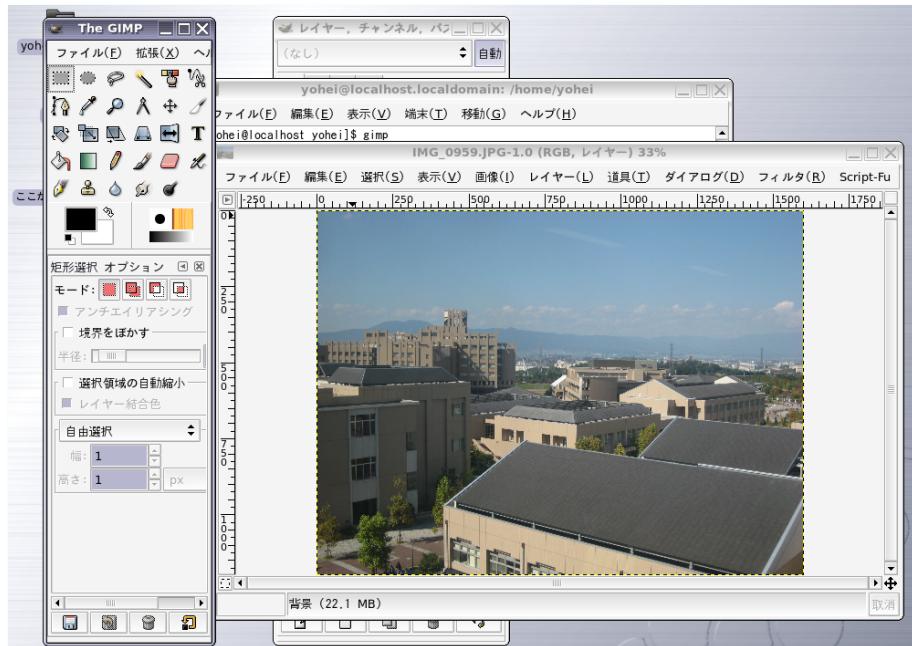


図 7.8 GIMP

はじめて GIMP を起動する場合、GIMP の初期設定を行うためのウィザードが起動します。その場合、「次へ」や「OK」などのボタンを押すことによってウィザードを最後まで進めます。ウィザードが終了すると、GIMP が起動します。

GIMP は非常に高機能なソフトウェアです。写真の修正やロゴ画像の作成といった、様々なグラフィック処理が可能です。

例えば、画像をモザイク調にするといった処理を行うことができます。このとき、画像が表示されているウィンドウのメニューから、[filtration]-[歪み]-[Mosaic] を選択します。すると、ダイアログが表示されるので、「OK」のボタンをクリックします。しばらくすると、画像がモザイク調に変換されます。

このように、GIMP を使うことによって、高度な画像処理を簡単にを行うことができます。しかし、あまりに多くの機能が搭載されているため、すべてを説明することはできません。GIMP の使い方をもっと知りたければ、インターネットで検索することによって調べることができます。

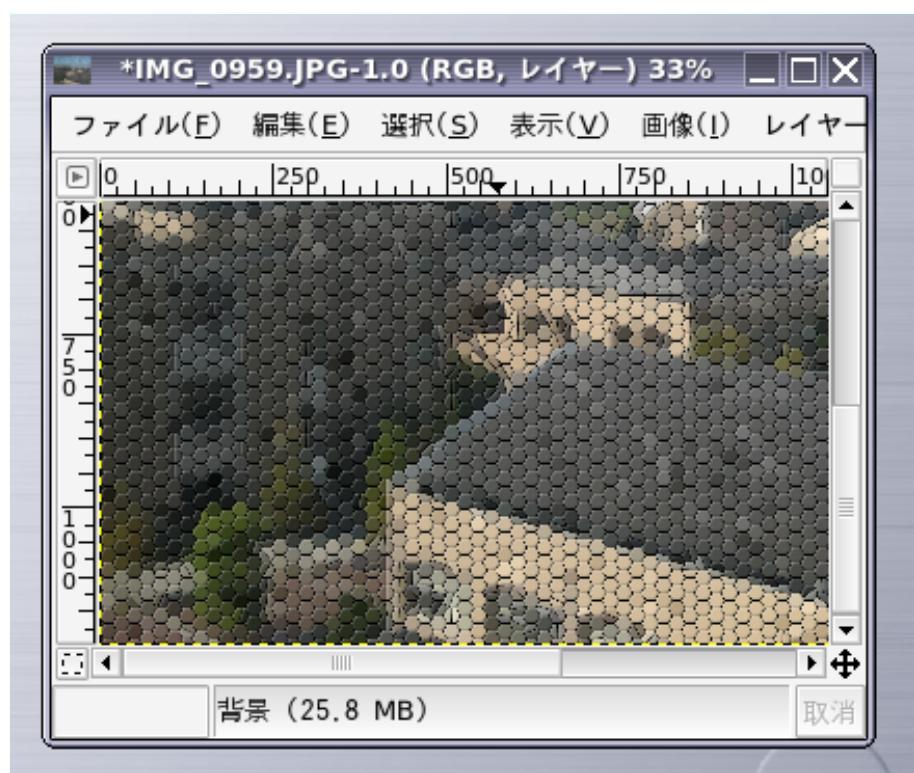


図 7.9 Mosaic 化された画像

第8章

プログラミング

本章では C 言語と FORTRAN における簡単なプログラミングの流れを説明します。それぞれの言語仕様に関しては触れませんので、言語仕様に関して詳しく知りたい場合は専門書をご利用下さい。

8.1 前準備

プログラミングをする場合、プログラムソースを保存するディレクトリを用意します。ターミナルを立ち上げ、次のコマンドを入力することによって、ディレクトリを作成します。

```
% mkdir program
```

作成したディレクトリに移動し、そのディレクトリの中に更に C 言語用と FORTRAN 用のディレクトリを以下のコマンドで作成します。（ディレクトリの移動は cd コマンドで行えます）

```
% mkdir {cProg,fProg}
```

これで準備は終わりです。プログラミングを始めましょう。

8.2 C 言語でのプログラミング

ここからは C 言語でのプログラミングを行います。まずははじめに、先ほど作成した cProg ディレクトリに移動します。

8.2.1 コーディング

本節では実際のコーディング作業の流れについて説明します。コーディングとはプログラムを書くことを指します。本書では、コーディングするためのエディタとして Emacs を用います。Emacs を起動するために以下のコマンドをターミナルに入力します。この例では作成するファイル名を hello.c とします。

```
% emacs hello.c
```

Emacs が起動したことを確認しましたら図 8.1 に示されているプログラムをそのまま Emacs に入力してみましょう。全角空白などを入力しないように注意して下さい。

```
#include <stdio.h>
int main(void){
    printf("Hello World!\n");
    return 0;
}
```

図 8.1 C のプログラムソース

プログラムソースの入力後、保存します。以上の手順で C 言語のソースファイルができます。

8.2.2 コンパイル

次に、入力したソースファイルをプログラムとして実行できるようにするためにコンパイルをします。コンパイルはターミナルで行います。このため、ターミナルを立ち上げてプログラムを作ったディレクトリに移動します。すると、hello.c というファイルがあるはずです。

ここで、以下のコマンドを入力することによって、ソースファイルをコンパイルすることができます。コンパイルのコマンドは gcc を用います。以下の例を参考にしてください。

```
% gcc -o 実行ファイル名 ソースコード名.c
```

hello.c の実行ファイル名を hello としてコンパイルする場合、以下のようになります。

```
% gcc -o hello hello.c
```

コンパイルエラーが表示されなければコンパイル成功です。エラーが表示される場合はプログラムソースに書き間違いがあります。この場合、プログラムソースを適宜修正する必要があります。

コンパイルが成功するとカレントディレクトリに、実行ファイルが生成されます。(今回の場合だと hello です。)

8.2.3 プログラムの実行

実行ファイルのあるディレクトリで以下のコマンドを入力して実行してみて下さい。

```
% ./hello
```

ターミナルに以下のように出力されれば成功です。

```
% ./hello  
Hello World!
```

8.3 FORTRAN でのプログラミング

次に、FORTRAN でのプログラミングを行います。準備として、はじめに作成した fProg ディレクトリに移動します。

8.3.1 コーディング

FORTRAN のプログラミングにおいても、Emacs を使用します。以下のコマンドをターミナルに入力します。この時作成するファイル名は hello.f とします。

```
% emacs hello.f
```

C 言語の場合と同様に、図 8.2 に示されているプログラムをそのまま Emacs に対して入力します。各行の行頭に 6 個の半角空白を入力する必要があるので注意してください。

```
PRINT*, "Hello World!"  
END
```

図 8.2 FORTRAN のプログラムソース

プログラムソースの入力が完了したら保存します。以上で FORTRAN のソースファイルが完成します。

8.3.2 コンパイル

次に C 言語の場合と同様にコンパイルを行います。コンパイルのためのコマンドとして g77 を用います、以下の例を参考にしてください。

```
% g77 -o 実行ファイル名 ソースコード名.f
```

hello.f の実行ファイル名を hello としてコンパイルする場合は以下のようになります。

```
% g77 -o hello hello.f
```

コンパイルエラーが表示されなければコンパイル成功です。エラーが表示された場合はプログラムソースに間違いがあるので適宜修正する必要があります。

コンパイルが成功すると、カレントディレクトリに実行ファイルが生成されます。(今回の場合だと hello です。)

8.3.3 プログラムの実行

実行ファイルのあるディレクトリで以下のコマンドを入力することによって、プログラムを実行することができます。

```
% ./hello
```

ターミナルに以下のように出力されれば成功です。

```
% ./hello  
Hello World!
```

8.4 最後に

今回は C と FORTRAN の簡単なプログラムを作成して実行することについてを説明しました。このように RAINBOW 環境ではプログラミングを行うための環境が整っています。実力とアイディアがあれば、どのようなプログラムも組み上げることができます。その可能性は無限大です！さあ、世界に名だたるプログラマーを目指して頑張りましょう！

第9章

インターネット

9.1 インターネットとは

インターネットとは、世界中のコンピュータが接続されているネットワークです。インターネットの起源は分散型コンピュータネットワークの研究プロジェクトである ARPANET であるといわれています。インターネットは全体を統括するコンピュータは存在せず、世界中にあるサーバによって構成されています。

9.1.1 WWW

インターネットにおける代表的な利用形態として Web ページを閲覧する行為があります。その行為は、WWW (World Wide Web) と呼ばれるシステムを利用します。WWW とは 1989 年に CERN の Tim Berners-Lee によって提唱されたインターネット上のリソースをハイパーテキスト^{*1}と呼ばれる形態で検索するための広域情報システムです。

その特徴としては

- ハイパーテキストによる Internet 上に分散された情報の結合

WWW は HTML (Hyper Text Markup Language) と呼ばれる言語で記されたハイパーテキストによってなりたちます。この言語は文書のフォーマットとアンカーと呼ばれる他の資源に対するリンク情報を制御します。HTML が用いられる事によって、インターネット上の様々な情報を予備知識がなくても利用することが可能です。

- URL によるアクセス方法の統一

HTML で示される情報の所在は URL (Uniform Resource Locator) と呼ばれる書式で書かれています。URL を用いることによって、Internet 上のさまざまな情報に対して、同じ方法でアクセスすることができます。

^{*1} ハイパーテキストとは任意の順序で読むことができる画像や音声データ等メディアを越えた情報を含む新しいタイプの文書です。

9.2 ブラウザ

Web ページを閲覧する場合に Web ブラウザを使用します。本章では RAINBOW で利用できるブラウザとして “Mozilla Firefox” について紹介します。

9.3 Mozilla Firefox

9.3.1 はじめに

Mozilla Firefox とは、Mozilla Foundation が開発・公開しているオープンソースの Web ブラウザです。特徴として

- マルチプラットフォームに対応
- HTML を独自に拡張し、高機能化
- アドオン^{*2}によるブラウザの機能拡張

などが挙げられます。

9.3.2 起動/終了

Mozilla Firefox は、GNOME 端末などで `firefox &` と入力することによって、起動することができます。起動すると図 9.1 のウインドウが開きます。

終了する場合は、メニューの [ファイル]–[終了] を用います。

Mozilla Firefox を起動したまま、ログアウトをするとトラブルの原因となりますので避けてください。

Firefox には図 9.2 のようなインターフェースによってスムーズな操作性を実現しています。

- A: メニューバーにより FireFox のすべての操作を行うことができます。各コマンドの説明は表 9.1 に書かれています。
- B: ナビゲーションツールバーはアイコンによって表示されたボタンでワンクリックによって手軽にコマンド実行を行うことができます。
- C: ブックマークツールバーは気に入ったサイトを登録したものがこのバーに表示されて手軽にサイトを閲覧することができます。

9.3.3 メニューバー

Mozilla Firefox のウインドウの上部にはメニューバーがあります。メニューバーの主なメニューについて表 9.1 で説明します。

^{*2} RAINBOW 環境において、アドオンのインストールはできません。



図 9.1 Mozilla Firefox の起動画面

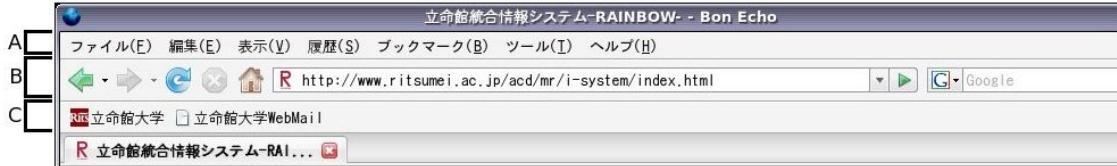


図 9.2 コマンドツールバー

*3 設定方法については 9.3.6 節を参照してください。

*4 スタイルシートとは、Web ページなどのフォントの種類や大きさ、色など、文書の見栄えに関する情報をひとまとめにしたものです。

*5 初期設定は立命館大学在学生のトップページです。

表 9.1 メニュー説明

| ファイル | |
|------------------|---|
| 新しいウインドウ | サブメニューを選択すると、Mozilla Firefox の新規画面を開きます。 |
| 新しいタブ | サブメニューを選択すると、現在開いている Mozilla Firefox の画面に新規タブを開きます。 |
| URL を開く | URL を直接指定して Web ページを開きます。 |
| ファイルを開く | ファイルを開きます。 |
| タブを閉じる | 現在表示しているタブを閉じます。 |
| 名前を付けてページを保存 | カレントドキュメントをファイルに保存します。 |
| ページの URL をメールで送信 | カレントドキュメントの URL をメールで送信します。 |
| ページ設定 | カレントドキュメントのページを設定します。 |
| 印刷プレビュー | カレントドキュメントの印刷プレビューを表示します。 |
| 印刷 | カレントドキュメントを印刷します。 |
| 設定とデータのインポート | Mozilla Firefox にデータをインポートします。 |
| オフライン作業 | オンライン状態からオフライン状態に変更します。 |
| 終了 | すべてのウインドウを閉じ、Mozilla Firefox を終了します。 |
| 編集 | |
| 元に戻す | 直前に行なった操作を取り消します。 |
| やり直し | 元に戻す操作を取り消します。 |
| このページを検索 | カレントドキュメントからキーワード検索をします。 |
| 設定 | フォントやキャッシュなどのいろいろな設定を変更します*3。 |

| | |
|---------------|--|
| 表示 | |
| ツールバー | 各ツールバーの表示、非表示を変更します。 |
| ステータスバー | ステータスバー（ウインドウ下部のバー）の表示、非表示を変更します。 |
| サイドバー | ウインドウ左端に履歴などのバーの表示、非表示を変更します。 |
| 中止 | 読み込みを中止します。 |
| 更新 | カレントドキュメントを読み込み直し、再描画します。 |
| 文字サイズ | 画面全体のフォントの文字サイズを変更します。 |
| スタイルシート | スタイルシート ^{*4} の使用、不使用を変更します。 |
| 文字エンコーディング | 表示に使用する言語を選択します。文字化けを起こしたときは変更してみてください（日本語の3項目とUnicode、どれかを選んでみましょう）。 |
| ページのソース | カレントドキュメントのソース（HTML）を表示します。 |
| 全画面表示 | カレントドキュメントを全画面に表示します。 |
| 履歴 | |
| 戻る | ドキュメント履歴を1つさかのぼります。 |
| 進む | ドキュメント履歴を1つ進みます。 |
| ホーム | 指定してあるホームページ ^{*5} を開きます。 |
| サイドバーに表示 | 今までアクセスしたページの履歴が表示されます。ここを選択すると選択したページが表示されます。 |
| ブックマーク | |
| このページをブックマーク | カレントドキュメントのURLをブックマークに登録します。 |
| このページを購読 | 現在表示されているページで提供されているフィードのプレビューを表示できます。フィードが複数ある場合は、タイトル一覧が表示され、プレビューするフィードを選択できます。 |
| すべてのタブをブックマーク | タブで表示しているドキュメントをまとめてブックマークに登録します。 |
| ブックマークの管理 | 登録されているブックマークの編集をおこないます。 |
| ツール | さまざまな検索エンジンの利用やクッキー、パスワードなどの管理をおこないます。 |
| Web検索 | 単語をWebで検索します。 |
| ページの情報 | カレントドキュメントの情報を表示します。 |
| ヘルプ | 操作の説明を表示します。分からなくなったときなどに使います。 |

9.3.4 ナビゲーションツールバー

ナビゲーションツールバーの各コマンドについて説明します。

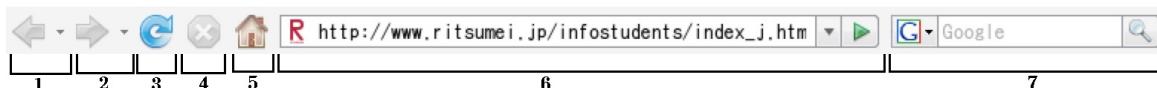


図 9.3 コマンドツールバー

- 1: 表示されたページより前のページに戻ることができます。
- 2: 進むボタンは一度戻ったページから再び進むことができます。
- 3: ページを再度読み込みます。
- 4: 読み込みを中止します。
- 5: 設定されているホームのページに戻ります。
- 6: ロケーションバーは現在表示しているページの URL を表示しています。直接 URL を入力することでページを表示することもできます。
- 7: 検索バーは検索エンジンを利用して Web を検索できます。検索したい語句を入力すると選択された検索エンジンの検索結果が表示されます。

9.3.5 印刷

本節では、Mozilla Firefox における Web ページの印刷方法について述べます。

まず最初に、メニューバーの [ファイル] から [印刷] を選択します。そこで、開いたウインドウ上の [プロパティ] ボタンをクリックすると、図 9.4 のようなウインドウが開きます。



図 9.4 印刷画面

次に、“プリンタ名:”に“CUPS/***/***”⁶を選択します。必要ならば、印刷の順序・紙の縦横・大きさなどを好みに合わせて設定した後、最後に印刷ボタンをクリックすることによって印刷を実行するこ

⁶ ***の部分には”lp0”のようにプリンタ名が記述されています。

とが可能です。RAINBOW 環境におけるプリンタは、基本的にカラー印刷をすることができませんから“色:”の設定項目は“グレースケール”を選択するとよいでしょう。

9.3.6 設定

本節では Mozilla Firefox における環境設定について述べます。RAINBOW 環境において、Mozilla Firefox を快適に利用するためには、利用者によって設定しなければならない項目がいくつか存在します。本節では、それらの必要な設定について、例を挙げながら紹介します。

“キャッシュ”的設定 RAINBOW 環境において、利用者が個人で保存可能なファイル量は制限されています^{*7}。そのため、キャッシュ容量を大きく設定すると、保存可能なファイル量が少なくなる問題が発生します。そのため、RAINBOW 環境において Mozilla Firefox を利用する場合、キャッシュ容量を少なめに設定することを推奨しています。なお、本学における推奨環境は 0MB としています。

キャッシュの設定を行う場合は、まず最初に、メニューバーの [編集] から [設定] を選択します。そこで、開いた設定 ウィンドウ上の [詳細] タブの [ネットワーク] を選択すると、図 9.5 のようなウィンドウが表示されます。このウィンドウを用いてキャッシュの容量を設定します。

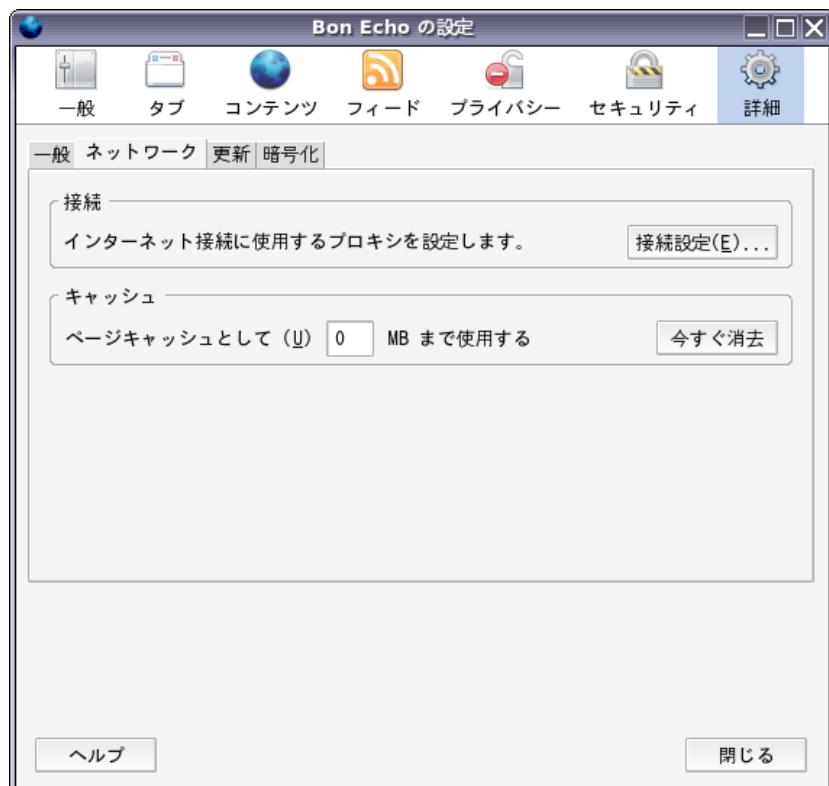


図 9.5 設定 – キャッシュ

^{*7} 詳しくは Appendix を参照してください

“プロキシ”の設定 RAINBOW 環境から、外部の WWW に接続する場合、プロキシサーバを経由する必要があります。そのため、Mozilla Firefox でプロキシサーバを利用するための設定する必要があります。

プロキシサーバの設定をするためには、まず最初にメニューバーの [編集] から [設定] を選択します。そこで、開いた設定 ウィンドウの [詳細] タブの [ネットワーク] の [接続設定] を選択します。

ここでは、“自動プロキシ設定スクリプト URL”を選択し、

`http://www.ritsumei.ac.jp/proxy.pac`

を入力します。以上の手順によってプロキシサーバの設定が完了します。

9.4 メール

9.4.1 メールとは

メールとは、日本語に直すと“手紙”的ことです。ただし、ここで扱う“メール”は E メールとも呼ばれ、コンピュータネットワークを使用して配達されるものです。この章では、メールを書く方法について述べていきます。

UNIX は標準でメールコマンドを備えています。しかし、UNIX の種類が異なると使用法が異なり、また、メールがたまつくると使用しづらいという問題点があります。ここでは、RAINBOW のコンピュータで利用できる

- ブラウザから利用する WebMAIL
- Emacs から利用する Mew
- Emacs から利用する Wanderlust

の 3 つを紹介します^{*8}。

9.4.2 メールを書くときの注意事項

ここで紹介する WebMAIL およびメールソフトは MIME に対応しているので、メールの件名 (Subject) に日本語を使用できますが、メールを出すときはメールを受け取る側が MIME 対応のメールリーダを使っているかどうかを、よく考えて日本語を使うようにしましょう。

また、メールのタイトル (Subject) や本文などには半角片仮名や機種依存文字を使用しないでください。

^{*8} 注意: WebMAIL や Wanderlust は IMAP というプロトコルを用いてメールをサーバ上で管理し、Mew は POP というプロトコルを用いてメールをローカルの各自ホームディレクトリに取り込んで管理します。IMAP と POP のメールリーダを併用すると、一旦 POP でローカルのホームディレクトリに取り込んだメールは再び IMAP でそのメールを閲覧することができなくなりますのでご注意ください。

9.5 WebMAIL

WebMAIL は Mozilla Firefox 等のブラウザから使用するメールリーダです。ブラウザ上で使用するため、WWW が利用できる場所であればどこでも使うことができます。

WebMAIL の URL は、

http://webmail.ritsumei.ac.jp

です。表示に成功すると図 9.6 のようなログイン画面が表示されます。

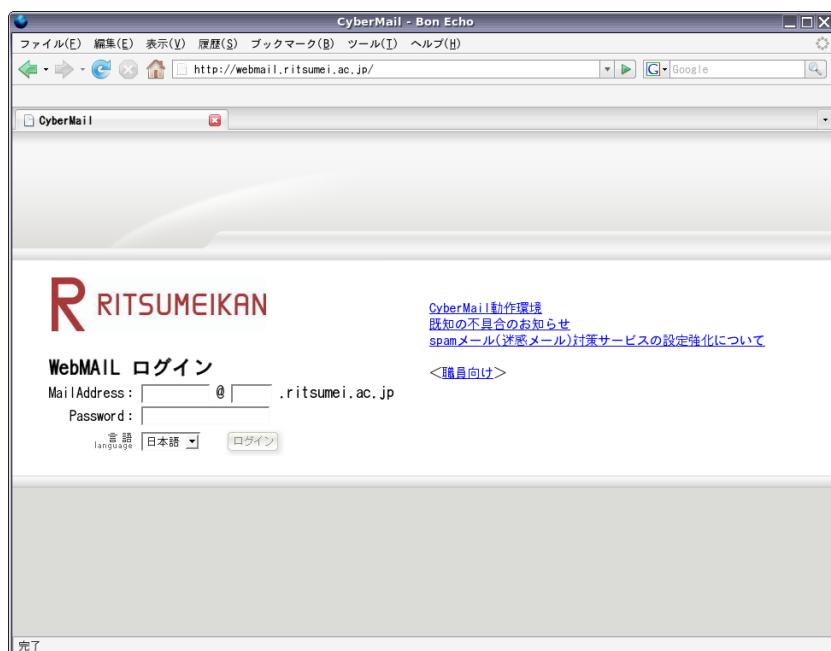


図 9.6 WebMAIL のログイン画面

WebMAIL を使用するためには、RAINBOW のユーザー ID とパスワードが必要です。ユーザー ID@ドメインには“ユーザー ID@ドメイン”*9、パスワードには UNIX で利用しているパスワードを入力してログインを押すと WebMAIL を使うことができます。

詳しくは、RAINBOW GUIDE (カラー刷) の“WebMAIL を使っての電子メールの利用”を参照してください。

*9 ドメインとはメールアドレスが “xx000000@xx.ritsumei.ac.jp” の場合’@’の後ろ 2 文字のことです

9.6 Mew

Mew は Emacs 上で動く MIME 対応メールリーダです。MIME に対応しているので、画像などのバイナリファイルを添付したメールを送ったり、それを受け取ったりすることもできます。また Mew はメールを読み書きをする機能の他、ソートや分類など、メールを扱う上で必要不可欠な、数多くの機能を提供しています。

9.6.1 Mew の初期設定

Mew を初めて使用する場合は、最初に初期設定をする必要があります。コマンドラインから、以下のコマンドを入力してください。

```
% imsetup
```

いくつか質問されますので、それに対して答えていきます。基本的にはデフォルトのまま変更せずに Enter を入力するだけですが、変更すべきところが 5箇所ほどあります。まずメールアドレスの設定ですが、

```
What is your E-mail address(es)? [xx000000@xx.ritsumei.ac.jp]
```

というように聞かれます。“ユーザー ID@ed.ritsumei.ac.jp” のように入力してください。

```
What kind of mail spool do you use? (local/POP/IMAP) [local]
```

と聞かれるところでは、“POP” と入力してください。次に

```
What kind of POP authentication mechanism? (POP/APOP/RPOP) [POP]
```

と聞かれますので、“POP” と入力してください。

```
What is your POP account name? [xx000000]
```

と聞かれるところでは、“ユーザー ID@ed.ritsumei.ac.jp” のように入力してください。@以降の部分も含めるように注意してください。POP サーバの設定では、

```
What is your POP server name or IP address? [mail.xx.ritsumei.ac.jp]
```

と聞かれますので、“pop.ritsumei.ac.jp” と入力してください。SMTP サーバの設定では、

What is your SMTP server name or IP address? [mail.xx.ritsumei.ac.jp]

と聞かれますので，“smtp.ritsumei.ac.jp”と入力してください。imsetup の全体の流れは次のようになります。

```
% imsetup
Where is your home directory? [/homer/stu0/xx000000]
Where is your Mail directory? [/homer/stu0/xx000000/Mail]
/homer/stu0/xx000000/Mail does not exist. Create it? [yes]
Creating /homer/stu0/xx000000/Mail directory.
Directory /homer/stu0/xx000000/Mail created.
Where is your News directory? [/homer/stu0/xx000000/News]
/homer/stu0/xx000000/News does not exist. Create it? [yes]
Creating /homer/stu0/xx000000/News directory.
Directory /homer/stu0/xx000000/News created.
What is your E-mail address(es)? [xx000000@bkc.ritsumei.ac.jp]
xx000000@ed.ritsumei.ac.jp
What kind of mail spool do you use? (local/POP/IMAP) [local] POP
What kind of POP authentication mechanism? (POP/APOP/RPOP) [POP] POP
What is your POP account name? [xx000000] xx000000@ed.ritsumei.ac.jp
What is your POP server name or IP address? [mail.ed.ritsumei.ac.jp]
pop.ritsumei.ac.jp
Do you want to preserve messages?
0 (delete immediately), -1 (preserve forever),
N > 0 (delete messages after N days) [0]
What is your SMTP server name or IP address? [pop.ritsumei.ac.jp]
smtp.ritsumei.ac.jp
Do you want to use value of Content-Length header for delimitation for
local
mail? (Answer yes if your OS supports Content-Length header like
Solaris 2.x,
otherwise answer no.) [no]
Does your system can detect write errors without fsync(2)? (You can
answer yes,
if your home directory is on local file system, otherwise answer no.)
[no]

Directory /homer/stu0/xx000000/.im created.
Setup /homer/stu0/xx000000/.im/Config.
```

これで、Mew を使用するための初期設定ができました。

9.6.2 Mew の起動と終了

では Mew を使ってみましょう。まず Emacs を実行して、以下のコマンドを入力してください。

M-x mew

すると、Mew のロゴが現れ、図 9.7 のようになります。

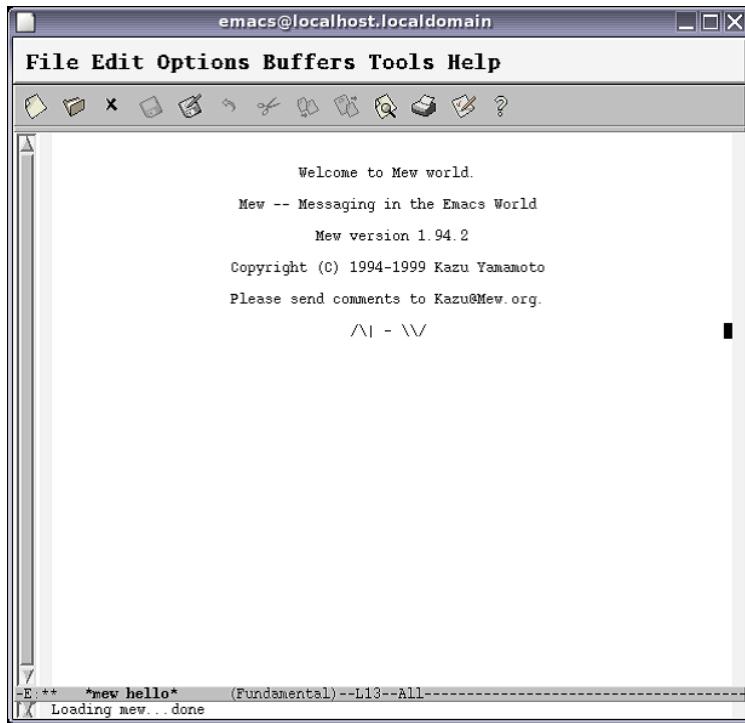


図 9.7 Mew の起動画面

Mew を起動すると、

Enter password :

とパスワードを聞かれるので、UNIX のログインパスワードを入力します。正しいパスワードを入力すると、到着しているメールを表示します。すでに誰かがあなた宛にメールを出していると図 9.8 のようになります。毎回パスワードを入力したくない場合は 9.6.18 節の“パスワードの保存”を見てください。

とりあえずは、Mew を実行できたことを確認して、Mew を終了しておきましょう。Mew を終了するには、先程の画面で

q

と入力すれば終了することができます。

9.6.3 メールを送信する

メールを送信する場合は、最初に Emacs 上で

M-x mew-send

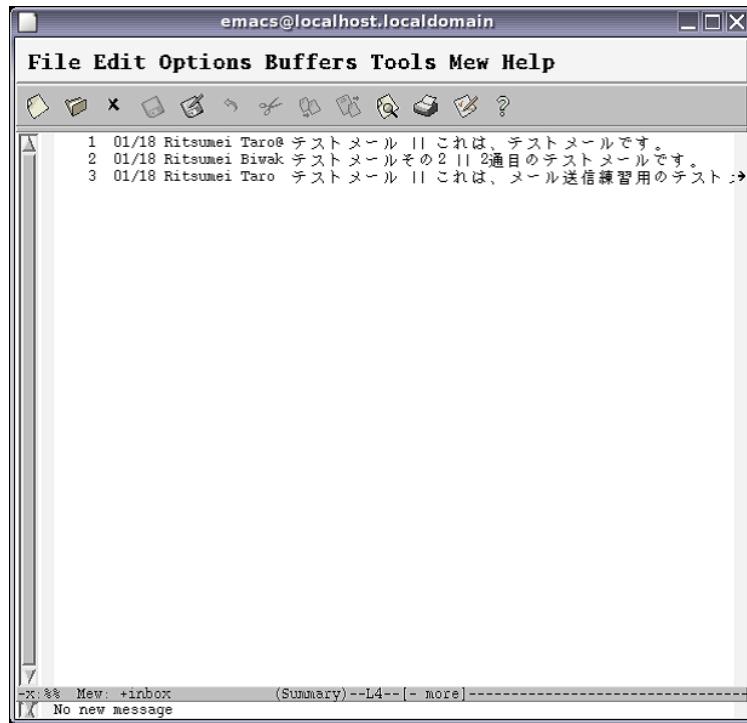


図 9.8 Mew のパスワード入力後の画面 (メールあり)

を入力すると、図 9.9 のような画面になります。この画面は、Draft バッファと呼び、メールを出すために必要な情報(ヘッダ)の部分がすでに書かれています。

ヘッダ部分の To: と Subject: の部分が空白になっているので、ここでまず To: のところに今から出すメールの宛先(メールアドレス)を入力します。

次に Subject: の部分に今から出すメールのタイトルを入力します。今回は自分に出すテストメールですから、Subject は “Test Mail” としてみましょう。

この 2 つを入力すると、後は----以下の部分に本文を書くだけです。好きな文章を入力してみましょう。自分が納得する文章を入力できたら、最後に実際に送りましょう。メールを送るときは、

C-c C-c

を入力すると、

The header was modified. Send this message? (y or n)

と聞かれますので、本当にメールを送信するときは “y” を、やはりメールを送信するのをやめたいときは “n” を入力してください。また、続けてメールを出すときや、内容を少しだけ変えて別の相手にメールを出したいときは

C-u C-c C-c

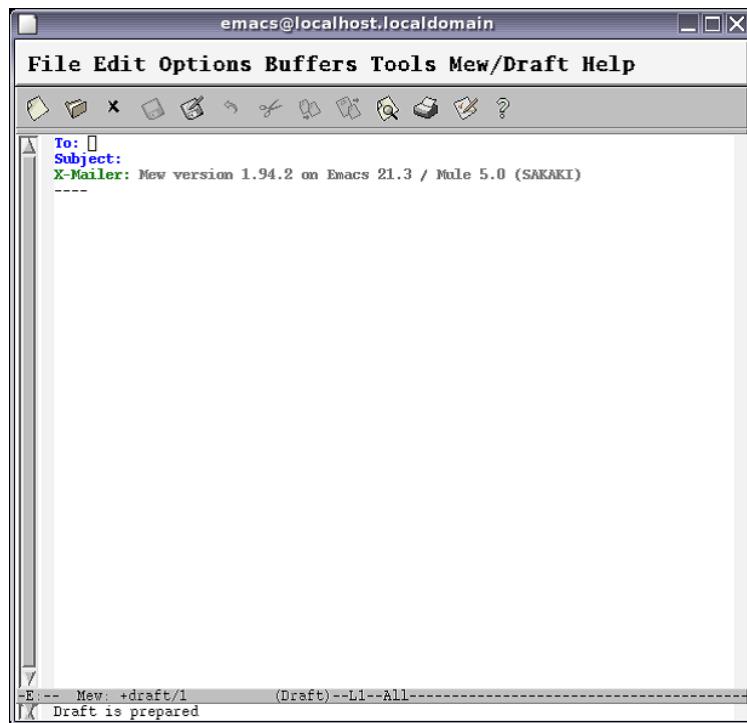


図 9.9 M-x mew-send の起動画面

を入力してください。Draft バッファを消さずにメール送信が行えます。

9.6.4 メールを読む

自分に来たメール読んでみましょう。メールが来ていないようなら、先ほど説明した手順で自分宛てにメールを送ってみましょう。

メールを読むために、Mew を起動しましょう。図 9.8 のようにメールのメッセージがあるはずです。この画面を Summary バッファと呼びます。ここでは主に Summary バッファでのメッセージの読み方について説明します。

Summary バッファでは、カーソル位置のメッセージ（カレントメッセージ）に対して、

- メッセージ内容の表示
- メッセージの削除
- ファイルのセーブ

などをすることができます。Summary バッファのメッセージの表示は 1 行につき 1 通のメールを表していて、その内容は左から

- メッセージ番号
- メッセージが発信された日付
- メッセージの発信者、あるいは自分が出したメッセージの宛先

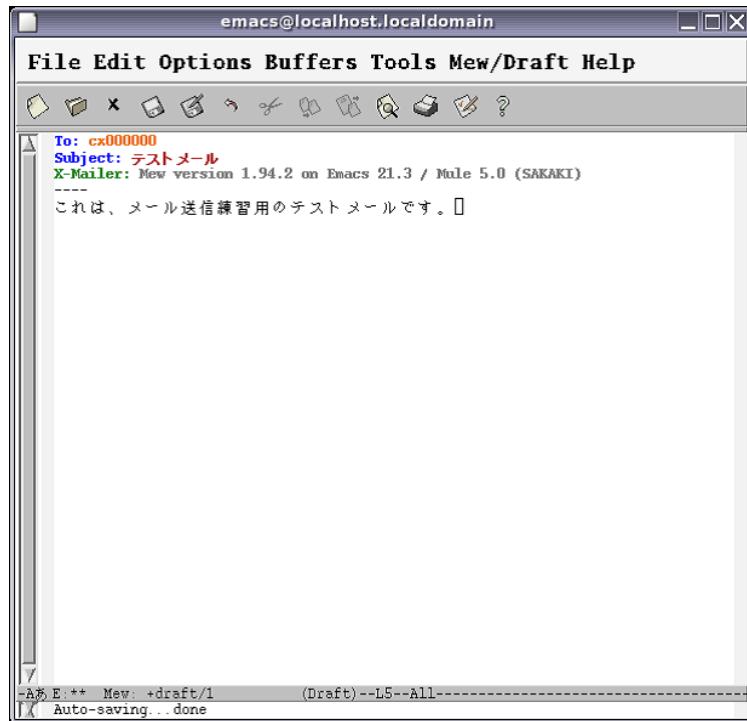


図 9.10 メールの送信

- Subject: の内容
- メッセージの内容の先頭部分

となっています。

Mew を実行すると，新しく届いたメッセージを inbox フォルダに移してメッセージの一覧が表示されます。メッセージの内容を上から順に読んでいくのであれば，

Space

を適宜押すだけです。このコマンドでメッセージを表示し，スクロールさせて次のメッセージに移動することができます。

また，カレントメッセージの内容を強制的に表示するには

. (ピリオド)

を入力してください。これらのコマンドで，図 9.11 のように画面が分割されメッセージの内容が表示されます。分割された下のウインドウに表示されているバッファを，Message バッファと呼びます。また，Mew を使用しているときに，別の新しいメールが来るかもしれません。Mew は実行したときまでに届いているメールの一覧を表示するだけなので，新しく来たメールはこのままでは読むことができません。新しく来たメールを読むためには，

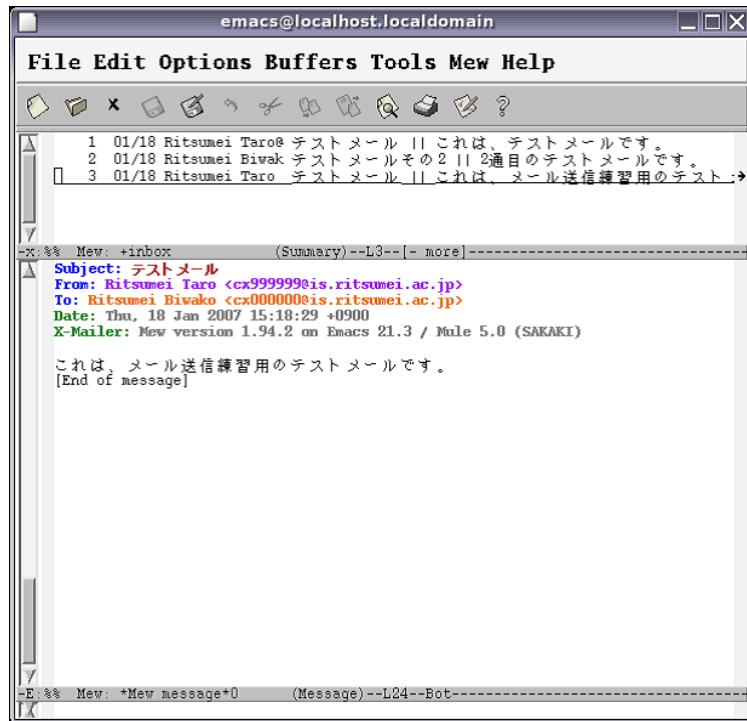


図 9.11 メッセージの表示

i

を入力してください。9.6.18 節の方法でパスワードを保存していない場合は、パスワードを聞かれるので、UNIX のログインパスワードを入力してください。新しく到着したメッセージが Summary バッファに追加されます。

メールを読み終わったら、終了しなければなりません。Mew の終了の仕方は、先にも示したように

q

です。ただし、“q”は Summary バッファで行ってください。

9.6.5 ショートカットでメール送信

メールを出したい場合、“M-x mew-send と入力してください”ということを前に書きましたが、他にも方法があります。Summary バッファで

w

を入力してください。

9.6.6 メッセージの消去

読み終わったメールで“もうこのメールはいらない”というものは、消去することができます。メールを消去するには、まず Summary バッファで消したいメッセージの横にカーソルを合わせて

d

を入力してください。すると図 9.12 のように Summary バッファのメッセージに消去マークがつきます。他にも消去したいメッセージがあれば消去マークをつけましょう。もしも、間違って消去マークをつけて

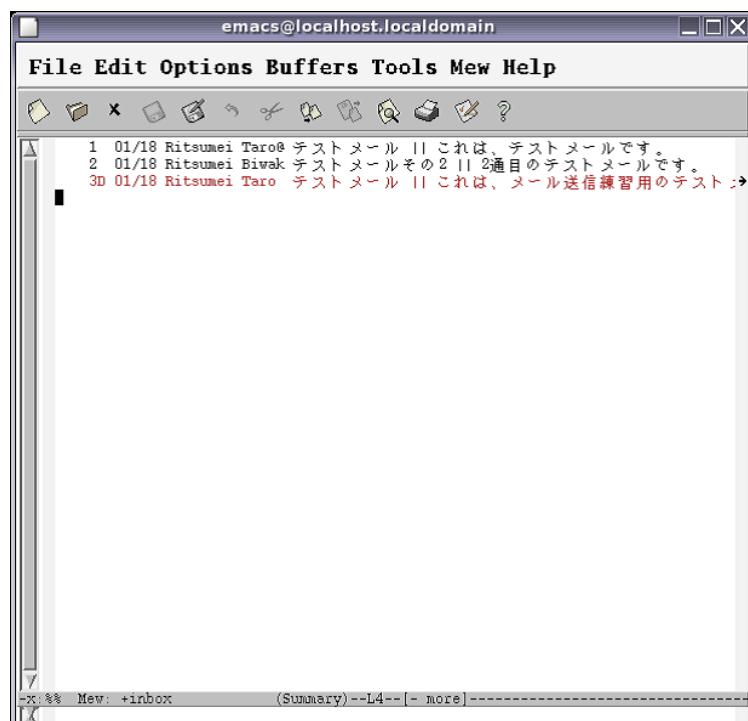


図 9.12 メールの消去 (1)

しまったら

u

を使ってメッセージの消去の取り消しができます。誤って消去マークをつけたメッセージまで移動してから“u”を入力してください。

不要なメッセージに消去マークをつけたら、

x

を入力してください。図 9.13 のように Summary バッファからそのメールのメッセージが消えたはずです。

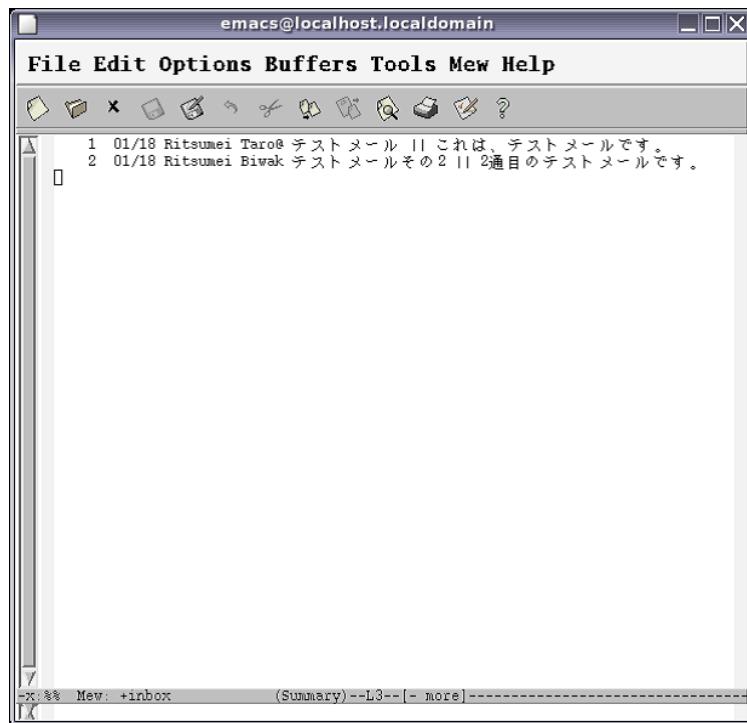


図 9.13 メールの消去 (2)

これで、メールは本当に消えてしまいました、といいたいところですが、実際はメールのファイル自体は残っています。消去したメッセージは `~/Mail/trash` に移動されているのです。つまり、メールを消去するコマンドは実際にはメールを `trash` フォルダに移動させただけなのです。

このことを知らずに、メールを消したつもりになっていると、知らない間に `~/Mail/trash` の下に大量の消去したメールがたまってしまいます。そうならないように、これらの `trash` フォルダのメッセージをもう一度消去します。

`trash` フォルダでもう一度消去する方法は、9.6.16 節の“カレントフォルダの変更”で説明します。

9.6.7 返事を出す

友人からメールが来たのでそれに対する返事が出したい場合は、

a
[REDACTED]

を入力することで、図 9.14 のように `To:` や `Subject:` が自動的に設定された画面が用意されます。また、

A
[REDACTED]

を使うと、図 9.15 のように受け取ったメールの本文を引用できます。こうすることで、相手のメールを参照して返事を書くことができます。

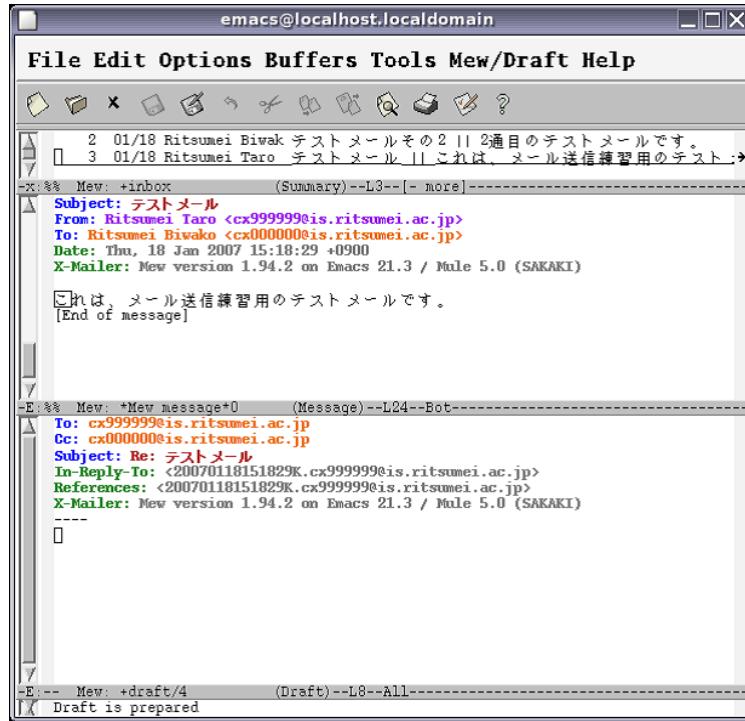


図 9.14 メールの返信

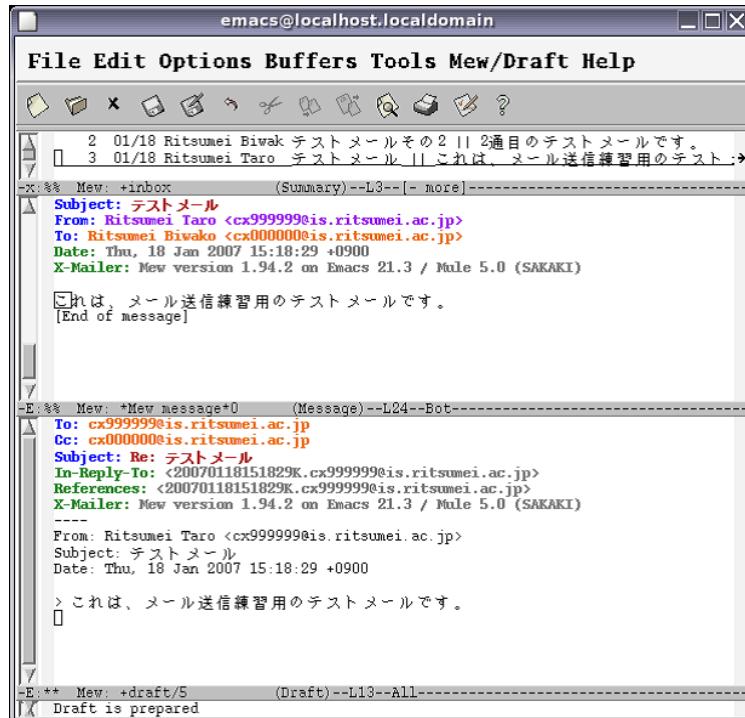


図 9.15 メールの返信(引用)

9.6.8 メールの内容を転送する

受け取ったメールをそのまま他の人に送りたいことがあります。そのときにいちいち写して他の人にお知らせするのは大変です。こんな場合に力を発揮するのが、メールの内容を転送する機能です。

f

を入力してください。すると、図 9.16 のような画面になりますので、宛先や Subject などを入力すれば転送メールを出すことができます。

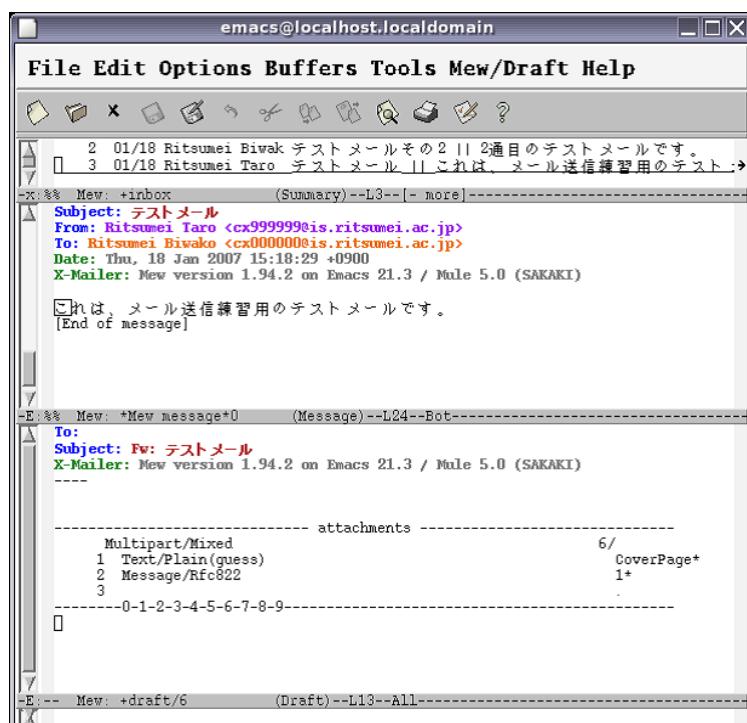


図 9.16 メールの転送

9.6.9 メールの内容をファイルに保存する

実験などでは、先生から重要な指導がメールで送られてくるときがあるでしょう。他にもいろいろな要因で、メールの内容をファイルに保存したいときがあると思います。そのようなときは、

y

を入力してください。そして、自分の好きな名前をつけて保存してください。

9.6.10 Mew 添付ファイル送信

Mew は MIME に対応しているので、画像などのバイナリファイルを添付したメールを簡単に送受信することができます。添付ファイル付きメッセージの事を“マルチパートメッセージ”とも呼びます。以下、“マルチパートメッセージ”として説明します。

9.6.11 マルチパートメッセージの作り方

マルチパートメッセージを送信するには、まず普通にメールを書くときと同じように Draft バッファで、宛先と Subject と本文を書きます。本文が書きあがったら、

C-c C-a

を入力してください。すると図 9.17 のようになります。

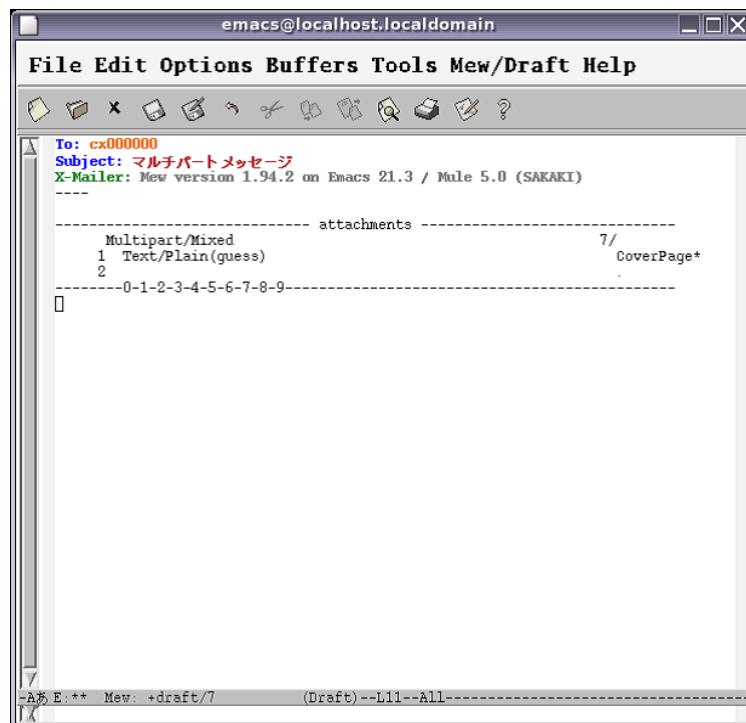


図 9.17 マルチパート (1)

- attachments - より下を添付領域と呼ぶことにします。添付領域では、1 行が 1 つのパートを表していて、その内容は左から

- マーク
- パート番号
- データの型

- ファイル名

となっています。

パート1のCoverPageは本文です。ファイルを添付するには、第2パートにカーソルを合わせて、

を入力してください。すると

と聞かれるので、添付したいファイル名を入力します。入力すると

と聞かれます。ここには、メールを送るときのファイル名を入力してください。何も入力せずにEnterを押せば、同じファイル名になります。すると図9.18のようにデータの型と、ファイル名が表示されます。

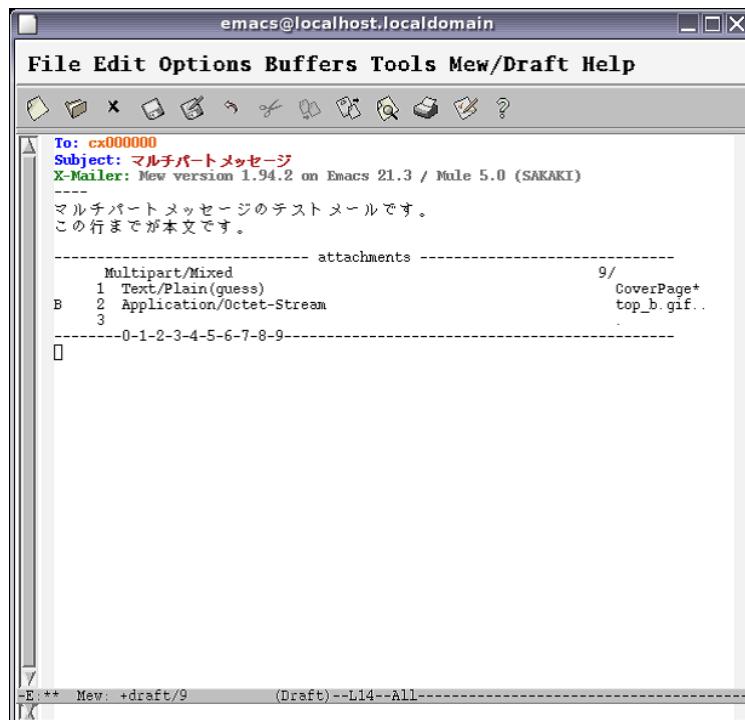


図9.18 マルチパート(2)

ファイルを添付し終わったら、

を入力してください。図9.19のように、メッセージがマルチパートへ変換されます。あとは、普通のメッセージと同じように

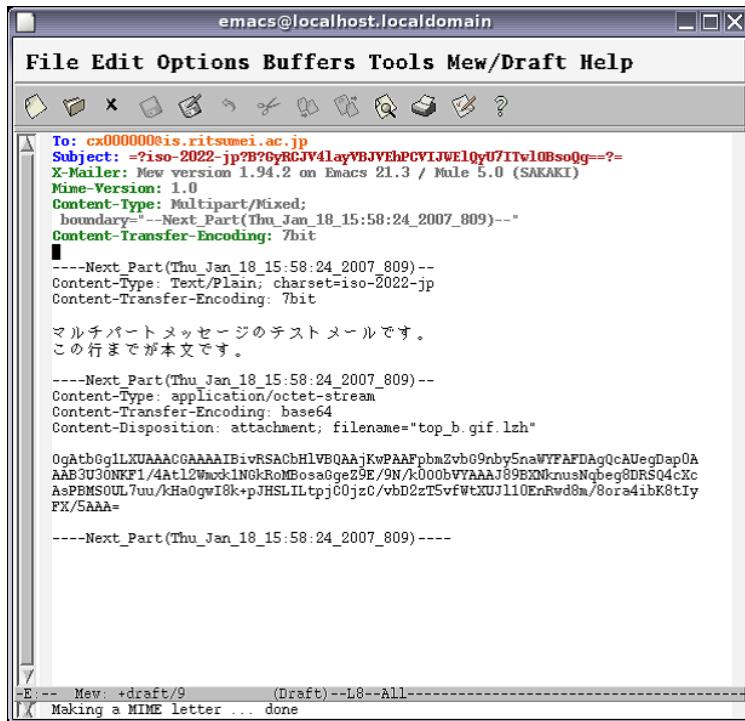


図 9.19 マルチパートへの変換

C-c C-c

で送信されます。

9.6.12 マルチパートメッセージの読み方

マルチパートメッセージの読み方は、普通のメールを読む場合とほとんど変わりません。Mew では、マルチパートメッセージの日付の横に “M” のマークが表示されます。読みたいメッセージにカーソルを合わせて

Space

を入力すると、図 9.20 のように、マルチパート構造が表示されます。

さらに Space を押すと、第 1 パートの本文を表示し、さらに Space を押していくと第 2 パート以降の添付ファイルをそのデータ型に応じて表示します。たとえば、Text/Plain なら Message バッファで、PostScript なら ghostview コマンドで表示されます。

このとき、ワープロ文章などの表示できないファイルが添付されてきたときは図 9.21 のように表示されるので、このようなファイルは

y

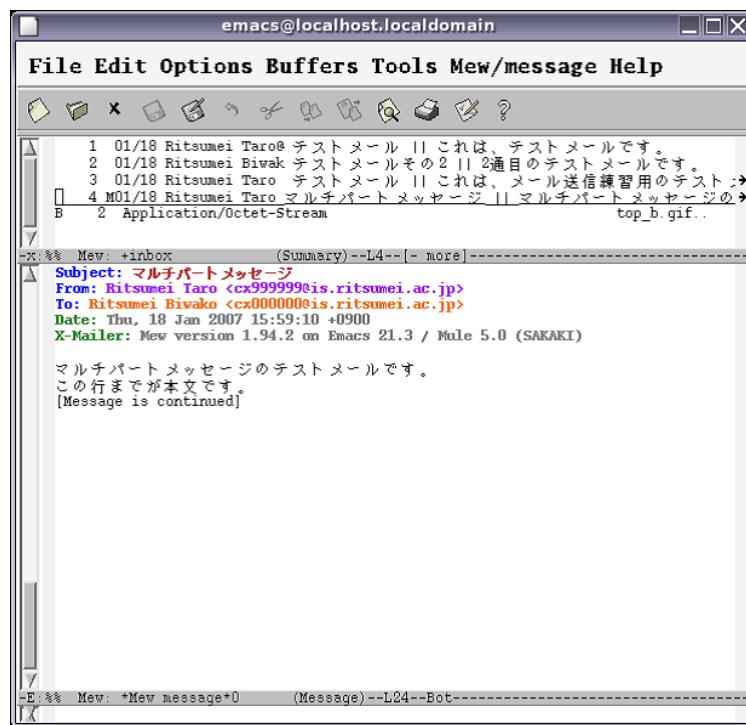


図 9.20 マルチパート構造の表示

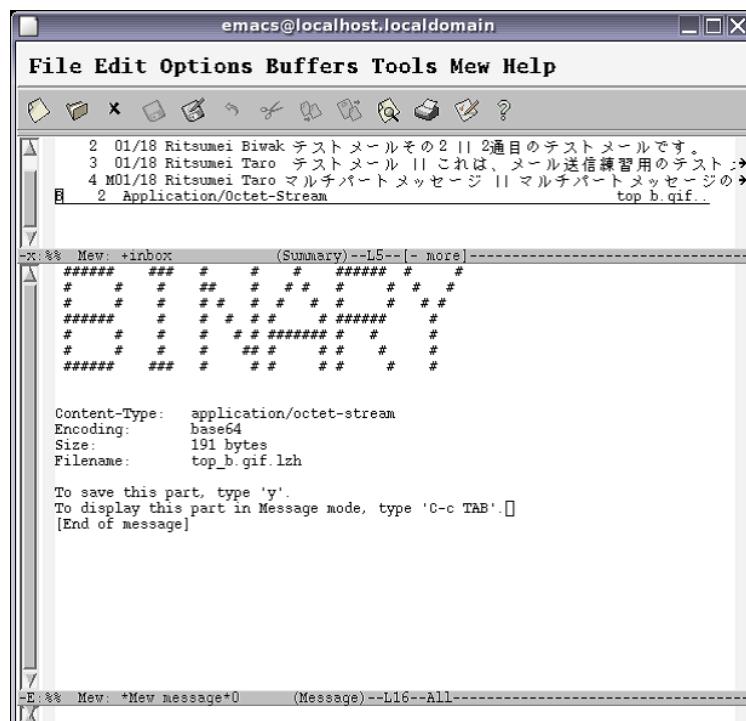


図 9.21 マルチパート（表示できないファイルの場合）

を使って、いったんファイルにセーブして、表示できるところに持っていくください。

9.6.13 Mew のフォルダ機能

Mew では、受け取ったメールを別々のフォルダに収納させたり、フォルダ間の移動ができます。要するに、メールを種類分けしてフォルダに格納しておき、メールの整理整頓ができるということです。

ここでは

- メッセージをフォルダ間で移動させる方法
- カレントフォルダの変更

について説明します。

9.6.14 メッセージの取り込み

先に記述していますが、Summary バッファで “i” コマンドを使用することによって、新たに到着したメールを読み込むことができます。このとき、新たに到着するメールは inbox という名前のフォルダに収納されます。また、カレントフォルダは自動的に inbox フォルダに移動します。

9.6.15 メッセージの移動

全てのメールが inbox に入ったままだと整理整頓に不便です。勉強用や友達用などのフォルダを作成して、メールを仕分けすると便利です。メールを仕分けするときに使用するのがメッセージの移動のコマンドです。

メッセージを移動させるには、Summary バッファ上で移動したいメッセージにカーソルを合わせて



を入力してください。このとき、

Folder name (+from/xx000000): +

のようにフォルダ名を聞いてきますので、カレントメッセージの移動先のフォルダ名を入力してください。このとき、Mew は移動先を推測し（）の中にデフォルト値として表示します。デフォルトのフォルダが希望通りであればそのまま Enter を押すだけです。入力したフォルダがすでに存在していれば、そのまま処理されますが、フォルダが存在しないと

+from/xx000000 does not exist. Create it? (y or n)

のように、フォルダを作成するかどうかを聞かれますので、“y” か “n” を入力して作成するか決めてください。すると、Summary バッファのメッセージの横に “o” のマークがつきます。

注意してほしいのは、上記のコマンドは入力しただけでは、Summary バッファにマークが付くだけであり、実際の処理は行われていないということです。ではどうすればいいかというと、メッセージの消去

のところで出てきた

x

を入力してください。これでメッセージの移動が行われます。また、コマンドを取り消したい場合も同じく“u”コマンドで取り消すことができます。

今は、inboxからの例しか説明していませんが、もちろんinbox以外のフォルダから別のフォルダへ移動することもできます。

9.6.16 カレントフォルダの変更

前の節で、メッセージを別のフォルダに移動させました。それでは、その移動させたメッセージを表示する方法を説明します。

g

を入力してください。移動先のフォルダ名に先ほど、メッセージを移動させたフォルダ名を入力してください。これで、そのフォルダのメッセージの一覧が表示されます。

9.6.6節の“メッセージの消去”で、消去したメッセージは完全に消去することはできませんでした。そこで、メールのメッセージを完全に消去する方法として、trashフォルダに移動して、もう一度消去する方法を説明します。

まずtrashフォルダに移動するために“g”を入力してください。すると

Folder name (+inbox): +

のように、移動先のフォルダ名が聞かれますので、trashと入力してください。

Folder name (+inbox): +trash

入力すると、カレントフォルダがtrashとなり、trashフォルダのメッセージが表示されるので、9.6.6節で説明したように“d”と“x”を使ってメッセージを消去してください。

これで、メッセージのファイルは完全に消去されました。

9.6.17 Mew の設定

Mewを使用するにあたって、知っておくと便利なことが他にもありますので簡単に説明しておきたいと思います。説明するのは以下のことがらです。

- パスワードの保存
- エイリアス (alias)
- シグネチャ (signature)
- メッセージの直接消去

9.6.18 パスワードの保存

メールを受信しようとするたびにパスワードを入力したくない場合は、パスワードを保存することでパスワードの入力を省略できます（この機能は RAINBOW 端末でのみ使えます。また、mew-use-cached-passwd を t にしている場合は無効になります）

パスワードを保存するには Mew が起動している状態で、

```
M-x mew-passwd
```

と入力します。すると、

```
Enter password :
```

と保存するパスワードを聞かれるので入力します。パスワードは `~/.mew/passwd` に保存されるので、管理には気を付けてください。

パスワードを再設定するときは、もう一度 `M-x mew-passwd` を実行してください。

パスワードの保存を解除したいときは、

```
M-x mew-clear
```

を実行してください。

9.6.19 エイリアス (alias)

メールの To: の部分に記入する宛先は、自分と同じサイト（学部）の場合は相手のユーザー名だけを書けばよいのですが、もし異なるサイトのユーザーにメールを出す場合は、

`test-name@test.address.jissai.naiyo.ac.jp`

のような長いメールアドレスを入力しなければなりません。これをいちいち書いたり憶えたりするのは大変です。また、同じ文面を一度に複数の人間に出したいときにも、メールを出す作業を何度も繰り返すのは面倒です。

こんなときにエイリアスと呼ばれている機能が役に立ちます。これは、エイリアスファイルに別名と正式名を書いておくことにより、正式なメールアドレスを指定しなくても別名を指定すれば、ちゃんと正式なメールアドレスでメールを出してくれるというものです。

エイリアスを使用するには、`~/.im/Config` に以下の設定をつけ加えます。

```
AliasesFile=../Mail/aliases
```

そして、実際のエイリアスの定義は `~/Mail` の下に `aliases` という名前のファイルを作成し次のように書きます。

```
test: test-mail@test.address.jissai.naiyo.ac.jp
```

また、一つの別名で複数のメールを出したいときは以下のように書きます。

```
member: kanji,anok,test-mail@test.address.jissai.naiyo.ac.jp
```

以上で設定は終了です。以降は、別名のアドレスを指定するだけでメールを出すことができます。例えば member に出せば、kanji, anok, test-mail@test.address.jissai.naiyo.ac.jp の 3 人にメールを出すことができます。

また、Draft バッファでメールを書くときに To: に

```
To: test
```

と入力したところで TAB を押すと

```
To: test-mail@test.address.jissai.naiyo.ac.jp
```

のようになりますし、

```
To: member
```

と入力したところで TAB を押すと

```
To: kanji,anok,test-mail@test.address.jissai.naiyo.ac.jp
```

のように、アドレスを補完することもできます。

9.6.20 シグネチャ (signature)

メールの最後には自分の署名を記すのが慣例となっています。署名には、本名や連絡先などを書いている人が多いようです。しかし、毎回メールの最後に記入するのは大変骨が折れる作業です。Mew では、メールの最後に自分の署名を記す機能としてシグネチャ機能があります。

シグネチャ機能を使用するには、まず自分の署名を書いたファイルを `~/.signature` のファイル名で用意してください。次に、Draft バッファでメールを書いた後に、

```
C-c TAB
```

を入力してください。するとカーソルの位置に `~/.signature` の内容が挿入されます。たとえば図 9.22 のようになります。

9.6.21 メッセージの直接消去

9.6.6 節で、メールのメッセージは 1 回消去しただけでは実際に消去することはできず、実際のメッセージファイルを消すためには、`trash` フォルダに移動してもう 1 回、地道に消去する方法を紹介しました。

しかし、“1 回消去するだけで、実際にファイルを消去したい” という人は `~/.emacs.el` に以下の 1 行

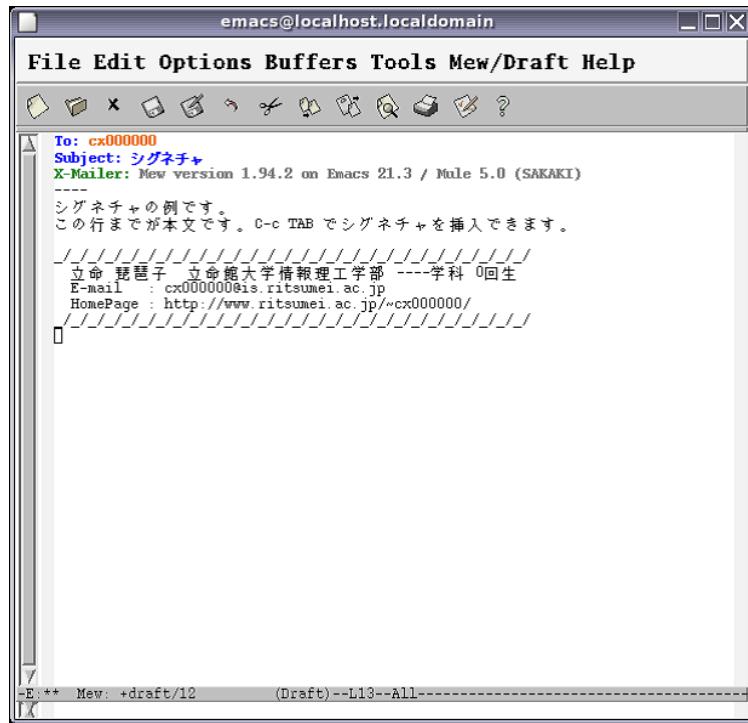


図 9.22 signature をつけた画面

を加えてください。

```
(setq mew-msg-rm-policy 'always)
```

これで，“d”と“x”を使ってメッセージを消去すると，trashへ移動することなく直接メッセージのファイルが消去されるようになります。もちろん，これを設定すると誤って消去してしまったメッセージは，2度と復活しません。

9.6.22 Mew コマンド一覧

Mew には非常に多くの便利な機能がまだまだありますので，興味のある人は勉強のつもりで調べてみてください。この章で扱った Mew のコマンド一覧は表 9.2 のとおりです。

表 9.2 Mew コマンド一覧

| | |
|---------------------------|------------------------------------|
| Mew の起動/終了 | |
| M-x mew | Mew Summary モードを起動 (メールを読む) |
| M-x mew-send | Mew Draft モードを起動 (メールを書く) |
| q | Mew を終了 |
| メッセージの操作 (Summary バッファで) | |
| Space | メッセージの表示 , スクロールアップ , 次のメッセージに移動 |
| . | カレントメッセージを表示 |
| DEL | メッセージのスクロールダウン |
| n | 次のメッセージに移動 |
| p | 前のメッセージに移動 |
| j | 指定した番号のメッセージに移動 |
| i | 新しいメッセージを取り込む |
| d | メッセージの消去 |
| u | コマンドのキャンセル |
| x | コマンドの実行 |
| y | メッセージをファイルにセーブ |
| メールを出すために (Summary バッファで) | |
| w | メールを書くための Draft バッファを用意 |
| a | 現在のメッセージに返答するための Draft バッファを用意 |
| A | 現在のメッセージを引用して返答するための Draft バッファを用意 |
| f | メールを転送するための Draft バッファを用意 |
| メールを書くとき (Draft バッファで) | |
| C-c C-c | メールを送信 |
| C-u C-c C-c | Draft バッファを消さずにメールを送信 |
| C-c TAB | カーソルの位置にシグネチャを挿入 |
| C-c C-a | マルチパートの作成 |
| C-c C-m | メッセージのマルチパートへの変換 |
| c | ファイルのコピー |
| フォルダのコマンド (Summary バッファで) | |
| o | メッセージの移動 |
| g | カレントフォルダの移動 |
| パスワードの保存 (RAINBOW 端末でのみ) | |
| M-x mew-passwd | パスワードを保存 |
| M-x mew-clear | パスワードの保存を解除 |

9.7 Wanderlust

Wanderlust は Emacs 上で動作する IMAP 対応のメールリーダです。また、ニュースリーダにもなるため、メール、ニュースを一括して管理することができます。

9.7.1 初期設定

Wanderlust を初めて使用する場合、Wanderlust 用の設定ファイルを HOME ディレクトリに作成する必要があります。/usr/local/skel にある設定ファイルを、~/ にコピーします。

```
% cp /usr/local/skel/skel.wl ~/.wl  
% cp /usr/local/skel/skel.folders ~/.folders
```

ファイルをコピーしたら、~/wl を Emacs などのテキストエディタで開いてください。そして、wl-smtp-posting-server の値を，“smtp.ritsumei.ac.jp”にしてください。

同様に、elmo-imap4-default-server の値を，“imap.ritsumei.ac.jp”にしてください。

9.7.2 起動と終了

Wanderlust を起動するには Emacs を使っている状態で

```
M-x wl
```

と入力します。すると図 9.23 の画面が表示され、Emacs のミニバッファに

```
Password for IMAP:xx000000/clear@imap.ritsumei.ac.jp:143:
```

のように、メールサーバへ接続するためのパスワードを聞かれるので UNIX にログインするときのパスワードを入力します。毎回パスワードを入力しないようにするには、次のコマンドを入力することでパスワードを保存することができます。

```
M-x elmo-passwd-alist-save
```

保存されたパスワードは ~/elmo にあります。

図 9.23 のような画面を Wanderlust ではフォルダモードと呼びます。Wanderlust を終了するにはこのフォルダモードのときに

```
q
```

と入力します。そうすると本当に終了するかどうかを聞かれるので、“y”と入力します。

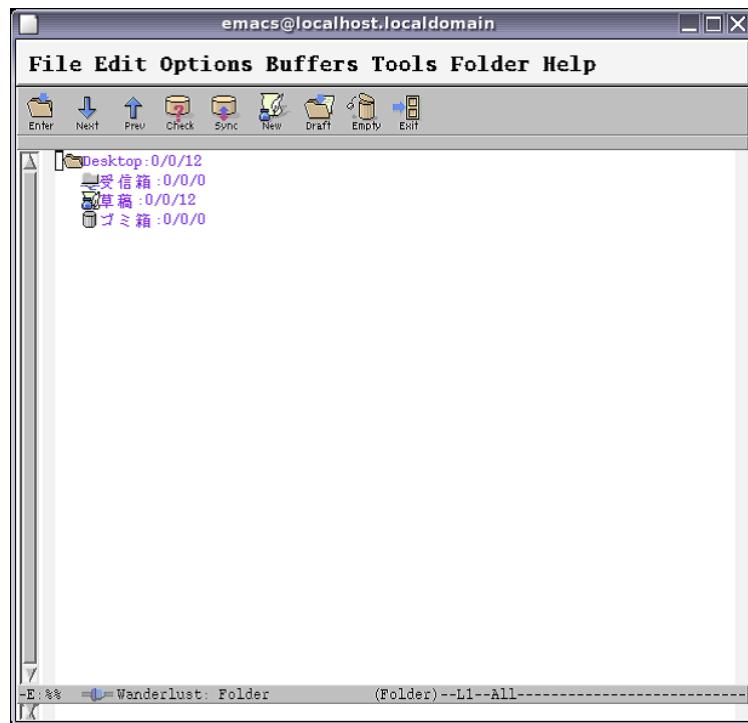


図 9.23 フォルダモード

9.7.3 フォルダモード

フォルダモードでは受信箱、ゴミ箱といったそれぞれの役割をもったフォルダの一覧が表示されます。各フォルダの右側に3つの数字が並んでいますが、左から順にそのフォルダ内の“新規記事数”、“未読記事数”、“全記事数”を表しています。

フォルダの中を覗くにはカーソルの上下により希望のフォルダへ移動し、Enter を入力します。

また、フォルダの内容を更新するには更新したいフォルダにカーソルを合わせて

s

を入力します。

9.7.4 サマリモード

フォルダモードで希望のフォルダを選択するとメール一覧が表示されます。この状態をサマリモードと呼びます。サマリモードからフォルダモードへ戻るには

q

を入力します。メールの内容を読むには読みたいメールにカーソルを合わせて Enter を入力します。また、メールの本文にカーソルを移動させたい場合は

j

を入力します。

9.7.5 添付ファイルの保存

Wanderlust で添付ファイルを保存するには、まず添付ファイルがあるメールを表示、“j”でメール本文にカーソルを移動します。それから、カーソルが保存したいファイル名の上にくるまで“n”を入力します。カーソルが保存したいファイル名の上にきたら

e

を入力し、保存するファイル名を入力すると添付ファイルが保存されます。

9.7.6 メールを書く

Wanderlust 上で、新規メール作成画面にするには

w

受信したメールに返信するには

a

引用文付きで返信するには

A

転送するには

f

と入力します。メールを書いた後、送信するには

C-c C-c

と入力し、送信するかどうかを聞かれるので“y”を入力します。

第 10 章

アプリケーション

10.1 数値解析 Mathematica

Mathematica(マセマティカ)は、数値計算と数式処理のエンジン、グラフィックスのシステム、プログラミング言語の編集、ドキュメントシステム、他のアプリケーションとの高度な接続性をシームレスに統合した数値解析ソフトウェアです。

微分や積分、線形代数、記号演算、他ほとんどの数学の分野で利用可能です。また、強力なグラフィックス機能により、計算結果を簡単に図示できます。ここでは、機能の一部を紹介します。

10.1.1 主な機能

- 数値計算 数値計算を行えます。 ϵ や π などの数学定数も扱えます(図 10.1)



図 10.1 数値計算の結果

- 数式処理や方程式の解答を出すことはもちろん、積分や特殊な記述にも対応しています(図 10.2)
- グラフィックス 精密かつ美麗なグラフィックが瞬時に生成できます(図 10.3)

Untitled-1 *

ファイル 編集 セル 書式 入力 カーネル 検索 ウィンドウ ヘルプ

```
In[1]:= Solve[a x^2 + b x + c == 0, x]
Out[1]= {{x -> -(b - Sqrt[b^2 - 4 a c])/(2 a)}, {x -> -(b + Sqrt[b^2 - 4 a c])/(2 a)}}
```

```
In[2]:= Integrate[Log[x] Exp[-x^2], x]
Out[2]= 1/81 Gamma[-2/3] (6 EulerGamma + Sqrt[3] π + 9 Log[3])
```

図 10.2 数式処理の結果

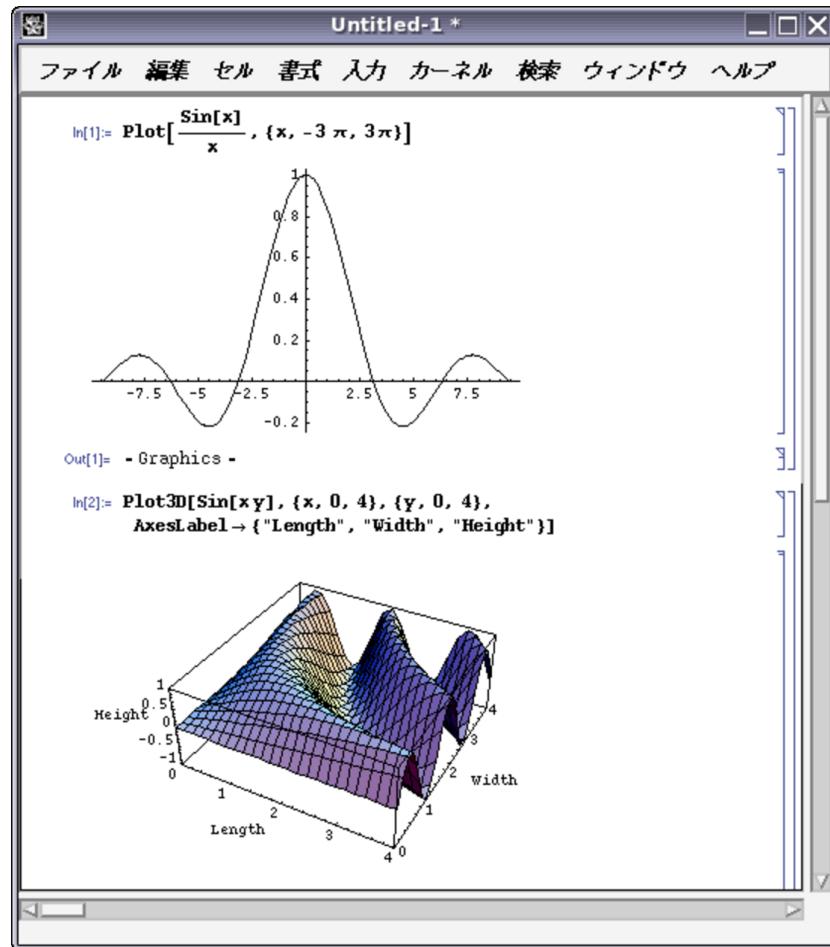


図 10.3 グラフィックスの結果

10.1.2 起動方法

コマンドラインから以下のコマンドを入力することで Mathematica を起動できます。詳しい説明は Help やチュートリアルを参照してください。

```
% mathematica
```

10.2 マルチメディアプレイヤ XMMS

XMMS は Winamp 風の操作性を持った X 上で動作するマルチメディアプレーヤーです。また、ブラウザからストリームコンテンツを閲覧できます。

10.2.1 再生できる主なフォーマット

- オーディオ CD(CD-DA), freedb による CDDB 機能あり
- libmikmod
- MPEG オーディオレイヤー 1, 2, 3(MP3), mpg123 ライブライを使用
- Vorbis
- WAV

10.2.2 起動方法

コマンドラインから以下のコマンドを入力することで XMMS を起動できます(図 10.4)。

```
% xmms filename
```



図 10.4 XMMS の画面

10.3 マルチメディアプレイヤ RealPlayer

RealPlayer はストリームコンテンツを再生するためのツールです。また、ブラウザでストリームコンテンツを閲覧するための plugin としても有名です。

10.3.1 再生できる主なフォーマット

- RM / RA / RAM (RealAudio, RealVideo ストリーム)
- RT (RealText ストリーミングテキスト)
- RP (RealPix ストリーミング画像)

- PNG , GIF , JPG (画像)
- MP3 (MPEG-1/2 Audio Layer-3)
- MPG / MPEG (MPEG Layer1 ビデオフォーマットと Layer2 オーディオフォーマット)
- SMIL , SMI (Synchronized Multimedia Integration Language ファイル)

10.3.2 使用する前に

RealPlayer を使用する前にまず、デスクトップ左上の“アプリケーション”というタブから“デスクトップの設定”の“サウンド”を選び、“全般”タブの“起動時にサウンド・サーバを有効にする”にチェックを入れ、ログオフか再起動を行います。

10.3.3 起動方法

ブラウザで対応形式のストリーミングデータを受信すると、RealPlayer が自動的に起動され、コンテンツを再生します。あるいは、コマンドラインから以下のコマンドを入力することで RealPlayer を起動できます(図 10.5)。もし、“Cannot open the audio device.Another application may be using it.” のエラーが出るなら RealPlayer の“View”タブから“Preferences...”を選び、“Audio Driver Options”の項目の“ESound Support(Linux only,ESD must be running)”の項目にチェックが入っているか確認してください。

```
% realplay filename
```



図 10.5 RealPlayer の画面

10.4 日本語コード変換 nkf

nkf(Network Kanji Filter) はネットワークでメールやニュースの読み書きをするために作られた、文字コードおよび改行コードの変換フィルタです。nkf を利用することで、テキストファイルなどの文字コードや改行コードを容易に変換できます。例えば、Windows 上で作成したテキストファイルを Linux 上で読み込めるように変換できます。

Windows 上で編集したテキストファイル report.tex を Linux 上で編集する場合は次のように入力してください。文字コードを EUC-JP に、改行コードを “LF” に変換します。

```
% nkf -e -Lu --overwrite report.tex
```

*1

Linux 上で編集したテキストファイル report.tex を Windows 上で編集する場合は次のように入力してください。文字コードをシフト JIS に、改行コードを “LF” と “CR” に変換します。

```
% nkf -s -Lw --overwrite report.tex
```

その他にもコマンドラインから Linux 上で編集したテキストファイルの文字コードを変える場合は次のように入力してください。EUC コードで保存されていたファイルを UTF-8 コードに変換します。

```
% nkf -w --overwrite report.tex
```

他にもいろいろな文字コードの変換ができます。調べたいときは man コマンドを使ってください。

注意 - --overwrite オプションを使わずにリダイレクションを使って文字コードを変換するとき、変換前と変換後のファイル名同じにするとファイルの中身が消えてしまうので注意してください。

*1 - --overwrite の - は 2 つ続けて入力してください。

第 11 章

シェルの設定

11.1 シェル変数

シェル変数は、シェルが固有に持つ変数です。tcsh には、特別な意味を持った変数がいくつかあり、これらの変数を設定することで tcsh の動作を変更することができます。

11.1.1 シェル変数の使い方

tcsh のシェル変数は、次のように使用します。

生成 シェル変数を生成するには、次のようにします。

```
% set ignoreeof
```

このようにすると、新しい変数 ignoreeof が生成されます。

代入 シェル変数に値を代入するには、次のようにします。

```
% set history=50
```

このようにすると、変数 history に値 50 を代入します。もし、変数 history が存在しなければ、新たに変数 history を生成して、値 50 を代入します。

参照 シェル変数の値を参照するには、参照したい変数名の前に\$をつけます。

```
% set history=50
% echo $history
50
```

echo は、与えられた引数をそのまま表示するだけのコマンドです。\$history は、tcsh によって変数 history の値である 50 に置き換えられるので、echo 50 が実行されることになり、その結果 50 が表示

されます。

削除 シェル変数を削除するには次のようにします。

```
% unset history
```

これで、変数 `history` が削除され、`history` の値 `$history` が参照できなくなります。

表示 存在するすべてのシェル変数とその値を表示したいときには、次のようにします。

```
% set
```

11.1.2 path (コマンドサーチパス)

シェル変数の中でも特に重要なのが `path` という変数です。変数 `path` はコマンドサーチパスを表します。

コマンドサーチパスとは、あるコマンドを実行する場合に検索するディレクトリのリストのことです。

```
% echo $path  
/usr/bin /bin /usr/ucb /usr/local/bin /usr/X11R6/bin  
/usr/java/jdk1.4/bin /usr/java/jbuilder5/bin
```

コマンドサーチパスを設定するには、次のようにします。

```
% set path=(/usr/bin /bin /usr/ucb /usr/local/bin /usr/X11R6/bin  
/usr/java/jdk1.4/bin /usr/java/jbuilder5/bin)
```

コマンドサーチパスを追加するには、次のようにします。

```
% set path=($path ~/bin)
```

このようにすると、現在設定されているコマンドサーチパスに `~/bin` が追加されます。これで、`~/bin` というディレクトリの中にあるコマンドも実行できるようになります。

コマンドサーチパスに `.` (カレントディレクトリ) を含めると便利そうですが、危険なのでやめましょう。例えば、`/tmp` という誰でも自由に書き込めるディレクトリをカレントディレクトリにしていたときに、悪意ある誰かが `rm` コマンドを `ls` という名前で置いていたら、ファイル名を表示するつもりでそのファイルを消してしまうことになります。カレントディレクトリにあるコマンドを実行するときは、多少面倒でも `./command` と、カレントディレクトリからの相対パスで指定するようにしましょう。

11.1.3 user と home

`user` と `home` は、ログインしたときにシェルによって設定される変数です。`user` は、ユーザー名を表します。また、`home` は、ホームディレクトリを表します。これらのシェル変数が、参照されることはよくありますが、変更されることはありません。

```
% echo $user  
ra012345  
% echo $home  
/homer/stu0/ra012345
```

11.1.4 prompt (プロンプト)

`prompt` は、シェルのプロンプトの文字列を指定する変数です。この変数を変更することによって、シェルのプロンプトを変更することができます。

以下にいくつかのプロンプトの例を示します。

```
% set prompt=> ”  
> set prompt=”$user%”  
ra012345% set prompt=”$user@$HOST%”  
ra012345@ab0sp001% set prompt=”%”
```

```
% set prompt=”%n@%m[%h]%%”  
ra012345@ab0sp001[21]%
```

11.1.5 history と savehist (ヒストリ)

シェルには、ヒストリ機能があります。`tcsh` でヒストリに関する変数としては `history` や `savehist` があります。`history` には、記憶するヒストリ行数を指定します。`savehist` には、`~/.history` に保存するヒストリ行数を指定します。`savehist` をセットすれば、ログアウト時に指定した行数分のヒストリが、`~/.history` というファイルに保存されます。`tcsh` は、起動時に `~/.history` を読んで自分のヒストリリストに取り込みますので、前回ログインしていたときのヒストリを利用することができます。

例えば、記憶するヒストリ行数を 50、また、`~/.history` に保存する行数を 30 にする場合、次のようにします。

```
% set history=50  
% set savehist=30
```

11.1.6 ignoreeof

`ignoreeof` というシェル変数をセットすると、シェルの終了の `Ctrl-d` を無視するようになります。したがって、`logout` コマンド以外でログアウトすることができなくなります。

```
% set ignoreeof  
% ^D  
ログアウトは"logout"を使用してください
```

11.2 環境変数

これまでに説明したシェル変数の他に、Unix には環境変数という変数があります。シェル変数は、シェル自身の動作を変更する目的で使いましたが、環境変数は、シェルが起動するコマンドの動作を変更する目的で使います。環境変数は、次のように使用します。

11.2.1 環境変数の使い方

代入 環境変数に値を代入するには、次のようにします。

```
% setenv EDITOR emacs
```

このようにすると、変数 `EDITOR` に値 `emacs` を代入します。もし、変数 `EDITOR` が存在しなければ、新たに変数 `EDITOR` を生成して、値 `emacs` を代入します。

参照 環境変数の値を参照するには、参照したい変数名の前に\$をつけます。

```
% setenv EDITOR emacs  
% echo $EDITOR  
emacs
```

`$EDITOR` は、変数 `EDITOR` の値である `emacs` に置き換えられるので、`echo emacs` が実行されることになり、その結果 `emacs` が表示されます。

削除 環境変数を削除するには次のようにします。

```
% unsetenv EDITOR
```

これで、変数 `EDITOR` が削除され、`EDITOR` の値 `$EDITOR` が参照できなくなります。

表示 存在するすべての環境変数とその値を表示したいときには、次のようにします。

```
% printenv
```

11.2.2 いろいろな環境変数

主な環境変数は、表 11.1 のようなものがあります。

表 11.1 環境変数

| 変数 | 意味 |
|--------|--|
| PATH | コマンドサーチパス。シェル変数 path の内容が反映される。 |
| HOME | ホームディレクトリ。シェル変数 home の内容が反映される。 |
| USER | ユーザー名。シェル変数 user の内容が反映される。 |
| PAGER | man などが起動するページャを指定。more や less としておくとよい。 |
| EDITOR | ページャ (less など) 等から起動するエディタを指定。emacs や vi としておくとよい。 |

11.3 エイリアス (別名) の設定 (alias, unalias)

いつも使うような長いコマンドをキーボードから打ち込むは面倒なことです。自分でコマンドにエイリアス (別名) をつけることができます。

ファイルのリストを表示する ls コマンドには様々なオプションがあります。ここで、ll というエイリアスを定義してみます。ll とタイプすれば ls -l を実行するようにしてみます。

```
% alias ll 'ls -l'
```

このように、エイリアスの定義は、alias *alias_name* '*command*' とします。また、エイリアス元のコマンドを確認するには alias *alias_name* のようにします。

```
% alias ll  
ls -l
```

実際にエイリアス ll を実行すると以下のようになります。

```
% ll  
合計 12  
-rw-r--r-- 1 ra012345 student 0 Jan 27 09:34 abocado  
(以下略)
```

定義したエイリアスを削除するには、unalias *alias_name* とします。

```
% unalias ll
```

ところで、rm コマンドでファイルを削除してしまうと、そのファイルを二度と復元することはできません。しかし、誤ってファイルを消してしまうことはよくあります。rm コマンドには、-i オプションがあります。-i オプションをつけて rm コマンドを実行すると、本当にファイルを削除するか問い合わせてきます。rm とタイプすれば、rm -i を実行するようにエイリアスを定義してみます。

```
% alias rm 'rm -i'
% rm apple
rm: 'apple' を削除しますか (yes/no)? no
```

しかし、rm *.bak *~ などで 100 個ものファイルを消したいときに、一々確認に答えるのは大変です。このように、一時的にエイリアスを無効化させたいときは、\ を使います。

```
% \rm *.bak *~
```

最後に注意しておかなければならぬのは、定義したエイリアスは現在のシェルを終了してしまうと無効になってしまうということです。シェルの起動時にこれらのエイリアスを設定するには、~/.cshrc という、シェルの起動時に読み込まれる設定ファイルなどに書く必要があります。このことについては、後ほど触れます。

今までに説明したことを表 11.2 へまとめます。

表 11.2 エイリアス

| コマンド | 説明 |
|--|--|
| alias | 現在定義されているエイリアスの一覧を表示する |
| alias <i>alias_name</i> | エイリアス名 <i>alias_name</i> の定義を表示する |
| alias <i>alias_name</i> ' <i>command</i> ' | <i>command</i> に <i>alias_name</i> という別名を付ける |
| unalias <i>alias_name</i> | エイリアス <i>alias_name</i> を削除する |

11.4 .cshrc と .login

tcsch は、ログイン時に ~/.cshrc と ~/.login の 2 つのファイルを読み込んで実行します (~/.cshrc を実行した後に、~/.login を実行します)*1。 ~/.cshrc や ~/.login では、通常はシェル変数や環境変数の設定およびエイリアスの定義などを行います。 ~/.cshrc は、tcsch が起動されるたびに実行されますが、~/.login の内容はログイン時に一回だけ実行されます。シェル変数や環境変数のセットおよびエイ

*1 なお、~/.tcshrc があれば、~/.cshrc の代わりに ~/.tcshrc を実行します。

リアスの定義などは、シェルを起動するごとに設定を行うので、`~/.cshrc` に書いておくとよいでしょう。

RAINBOW 環境では、あらかじめ`~/.login` と`~/.cshrc` が用意されています。ここでは環境設定の例として、それらの用意されているファイルを用いて解説します。

11.4.1 .login

まず `.login` です。

```
#  
# Ritsumeikan Univ. default .login  
  
#  
  
# 以下の 3 行は、RAINBOW の system 環境設定を行うための記述なので  
# 削除しないで下さい。  
# 修正が必要な場合は個人の責任のもと、必要な環境変数のみ設定  
# 変更を行ってください。  
#  
if (-f /homer/stu1/skel/default.login) then  
    source /homer/stu1/skel/default.login  
endif
```

`default.login` へのパスは、ユーザーによって異なりますので、ここで記述されてるパスと異なっていても問題はありません。ただし、文頭に記述されている通り、`if` からの 3 行は RAINBOW のシステムに合わせた環境設定を行うための記述なので、削除しないでください。通常 `.login` を変更する必要はありません。

設定ファイルでは、`#`から始まる行はコメントと見なされ、読み飛ばされます。設定を書き足す際は、あとで見て分かりやすいように、説明、注釈などをコメントとして書くようにするとよいでしょう。

`if` から始まる部分は、シェルスクリプトという言語で書かれており、シェルスクリプトを使うと設定ファイル内にプログラムを書くことができます。つまり、特定の条件のときだけ適応する、環境に合わせて振り分けるといった、高度な設定を行うことができます。

11.4.2 .cshrc

次に `.cshrc` を解説していきます。`.cshrc` は長いので、少しづつ順に見ていきます。

```
# Ritsumeikan Univ. default .cshrc
#
# 以下の3行は、RAINBOW の system 環境設定を行う為の記述なので、
# 削除しないで下さい。
# 修正が必要な場合は個人の責任のもと、必要な環境変数のみ設定
# 変更を行ってください。
if (-f ~skel-gr9/default.cshrc) then
    source ~skel-gr9/default.cshrc
endif
```

ここも先ほどと同様、RAINBOW のシステムに合わせた環境設定を行うための記述なので、削除しないでください。

```
# 以下に OS 每のケース判別テンプレートを添付します。
#
#         switch ($OSTYPE)
#             case "sun":
#                 breaksw
#             case "hp":
#                 breaksw
#             case "linux":
#                 breaksw
#             case "other":
#                 breaksw
#             endsw
#
# もし、OS が linux の場合のみ自分のホームにある bin ディレ
# クトリに path を通したい場合は、例えば以下の様に設定し、既
# に設定されている path 環境変数を消さない様に注意して下さい。
#
#     EX)
#         switch ($OSTYPE)
#             case "linux":
#                 set path=( $path $HOME/bin )
#                 breaksw
#             endsw
```

ここは、利用している OS の種類に合わせて設定を変更する例が書かれています。全体がコメントアウトされているので、このままでは動作しません。

もし複数の OS を利用していて、特定の OS の場合のみ設定を変更したい場合は、switch から endsw までのコメントを外し、例のように、case 内を変更してください。

```
# Default
umask 022
if ( $?prompt ) then
    set prompt = "% "
    set history=100
    stty erase ^H kill ^U intr ^C
endif
```

ここには、標準的な設定が書かれています。

Default の部分の先頭にある umask はファイル生成時のデフォルト権限を設定しています。この設定では、新規生成のファイルは自分以外には書き込みができないモードになります。読み取りや実行もさせたくない場合は 077 にしてください。

次の if 文では、プロンプトの設定やヒストリの設定が行われています。プロンプトの変更はこの set prompt で行います。

デフォルトの ~/.cshrc は以上です。その他、エイリアスの登録や環境変数の設定は、この .cshrc に書くとよいでしょう。エイリアスの登録の具体例を挙げます。

```
# ll をタイプすると ls -l を実行する
alias ll 'ls -l'
```

エイリアスや環境変数の設定が長くなるようなら、設定を別ファイルにしておくと分かりやすいでしょう。tcsh では、これらのファイル名は特に決まってないので、自分で分かりやすい名前をつけてください。source を用いることで、自分で作成した設定ファイルの設定を利用することができます。

source を用いた設定の具体例を挙げます。まず、自分のホームディレクトリに、設定ファイル（仮に .alias とします）を作成し、そのファイルに上で説明したエイリアスの登録の具体例を記述します。次に、.cshrc に以下の記述を追加します。

```
# エイリアスの設定の追加を指定する
source ~/ .alias
```

このように、デフォルトの .login , .cshrc には、最低限の設定のみ用意されています。RAINBOW 環境をより使いやすいものにするために、この章を見返しながら、.cshrc に設定を書き加えてみましょう。

11.5 ログインシェルの変更

ログインシェルとは、ログイン時に起動されるシェルのことです。通常は、ログインシェルとして tcsh が設定されています。このログインシェルは、他のシェルに変更することができます。ログインシェルには、自分のお気に入りのシェルを設定しておくとよいでしょう。ここでは、その変更方法を説明します。

ログインシェルの変更を行うには、ldapchsh コマンドを用います。

```
% ldapchsh
Changing LDAP account information for ra012345.
Please enter password:
```

まず、パスワードを入力してください。ログインシェルを変更しているのが本人であるかどうかを確かめます。これは、自分が少し席を離れたときに、他人が勝手にログインシェルを悪意あるコマンドに変えてしまうのを防ぐためです。なお、パスワードは画面には表示されません。

```
% ldapchsh
Changing LDAP account information for ra012345.
Please enter password:

Changing login shell for ra012345.
To accept the default, simply press return.
To use the systems default shell, type the word 'none'.
Login shell [ /bin/csh]:
```

ここで、新しいログインシェルを何にするのか聞いてくるので入力します。例えば、ログインシェルを bash に変更したい場合、/usr/local/bin/bash と入力します。

```
% ldapchsh
Changing LDAP account information for ra012345.
Please enter password:

Changing login shell for ra012345.
To accept the default, simply press return.
To use the systems default shell, type the word 'none'.
Login shell [ /bin/csh]: /usr/local/bin/bash
The login shell has been changed.
```

すべての入力に間違いがなければ、このようにログインシェルを変更できます。これで、次のログインからは、変更したシェルが起動するはずです。

11.6 その他のシェル

UNIX の世界では様々なシェルがあります。今まで紹介してきた `tcsh` の他には、強力な補完機能を備えた `zsh` や Linux における標準のシェル `bash` などが広く使われており、それぞれ便利な機能を備えています。しかし、`zsh` と `bash` は `sh` 系のシェルなので、今まで説明してきた `tcsh` とは、ずいぶん感じが違います。どのシェルを使うかはユーザーの好みによります。みなさんも手に馴染むシェルを見つけてください。

第 12 章

GNOME

12.1 GNOME とは

GNOME (GNU Network Object Model Environment)^{*1} は, GNU プロジェクトの一環である GNOME プロジェクトによって開発がすすめられている統一デスクトップ環境です. GNOME プロジェクトは, フリーなシステムのための完全にフリーなデスクトップ環境を創造するために誕生し, ユーザーフレンドリーなアプリケーション群や使いやすいデスクトップを提供することを目的としています.

GNOME は他の GNU プログラムと同様に近年の全ての UNIX ライクなオペレーションシステム上で動くよう設計され, またユーザーが自由に入手・修正・配布する権利を与えられています.

^{*1} GNOME は, “ゲノーム”と読まれることが多いようです.

12.2 GNOME の各部説明

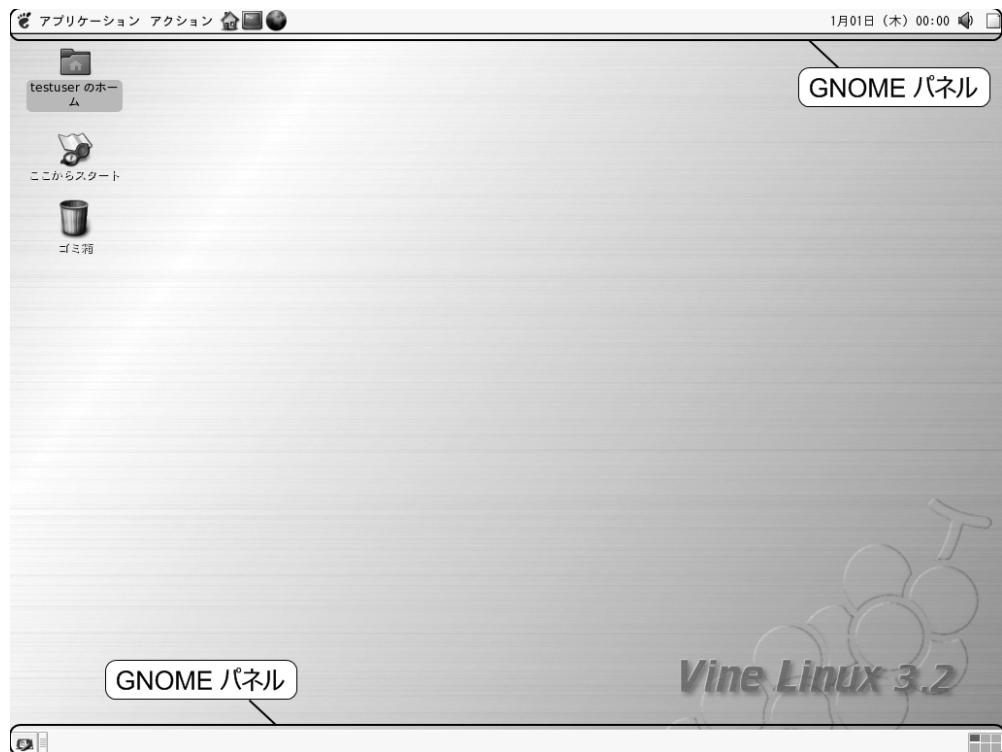


図 12.1 GNOME の画面

GNOME パネル

GNOME パネルはユーザーインターフェースの核になる部分で、メニュー・バー やユーザーが自由に編集できるユーザーメニュー、アプリケーションの起動を簡易化するアプリケーションランチャ、GNOME アプレット（後述）などの格納場所であり、ユーザーはこれらを自由に追加および削除できるようになっています。

12.2.1 GNOME パネルの各部説明

次に、GNOME パネルの各部について説明していきます。パネルの内容は変更できるため、図のものは一例です。



図 12.2 GNOME パネル（上部）

左からメインメニューとなるアプリケーションおよびアクション,Nautilus , GNOME 端末 , Firefox , 時計アプレット , 音量コントローラ , ウィンドウメニューとなっています.

メニュー・バー

メニュー・バーは GNOME のほとんどすべての機能へのアクセスを提供するメニューツールです.

メニュー・バーの主な中身はアプリケーションメニュー , アクションメニューなどからなっています. 各サブメニューはさらにサブメニューを持ち , 豊富な機能を提供しています.

Nautilus (図 12.3)

Nautilus(ノーチラス) はファイルやヘルプ , 各種設定 , ウェブリソースなどへのアクセス機能を統合し , ブラウザ感覚で扱えるアプリケーションです.

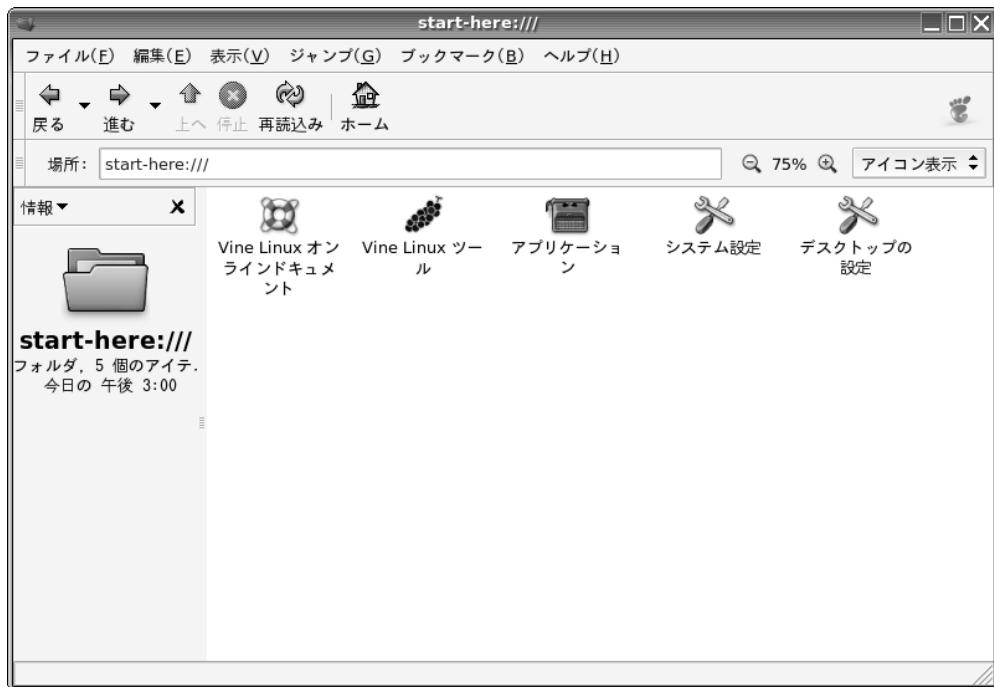


図 12.3 Nautilus

GNOME 端末

上記に挙げた GNOME 標準のターミナルエミュレータです.GNOME は多様な機能を提供していますが , UNIX を使用する上でシェルコマンドの入力による操作は必須となりますので , 最も使用されるアプリケーションになるでしょう.

Firefox

インターネットコンテンツを閲覧する際の代表的なアプリケーションです. 詳細については , 9.3 節などを参照してください.

時計アプレット

現在の時刻を表示するアプレットです。また、時刻や日付をコピーすることもできます。

音量コントローラ

音量の調節を行います。

ウインドウメニュー

ウインドウメニューは、全てのワークスペース内で起動しているアプリケーションを表示します。ボタンをクリックするとリストが表示されます。使用したいアプリケーションをクリックすると、他のワークスペースにある場合でも、自動的にそのワークスペースに移動し、アプリケーションがアクティブな状態になります。

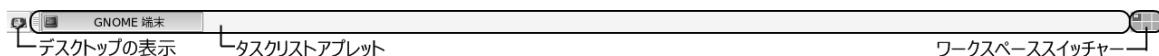


図 12.4 GNOME パネル(下部)

左からデスクトップの表示、タスクリストアプレット、ワークスペーススイッチャーとなっています。

デスクトップの表示

表示されているウインドウを全て最小化し、デスクトップを表示します。もう一度押すと、先ほど表示されていた全てのウインドウを表示します。

タスクリストアプレット

タスクリストアプレットは、表示されているワークスペース内で起動しているアプリケーションを表示します。多数のアプリケーションウインドウが開いていたり、最小化されたアプリケーションがある場合でも、タスクリストのアプリケーションをクリックすることでアクティブ状態にすることができます。

ワークスペーススイッチャー

GNOME が提供するデスクトップ画面は一つではありません。仮想的に複数のワークスペースが提供されており、複数の異なる作業を行なう場合に画面を分けて使用すると便利です。ワークスペーススイッチャーはこれらの仮想デスクトップの切り替えを行います。

12.2.2 GNOME アプレット

GNOME が提供する“アプレット”とは GNOME パネル上で動く小さなアプリケーションのことです。先ほどのデスクトップ例の中でも紹介しましたが、ここではさらにその他のアプレットを紹介します。

スクリーンショットアプレット

デスクトップ画面全体や、特定のウィンドウのスクリーンショットを撮影するためのアプレットです。

付箋紙アプレット

デスクトップ付箋紙の生成、表示、管理を行います。

この他にもさまざまなアプレットが用意されています。

12.2.3 GNOME パネルをカスタマイズする

GNOME のデスクトップおよびパネルのカスタマイズは、一部を除いてユーザーが自由に行なうことができます。

GNOME パネルのメニュー・バーに登録されているアプリケーションやアプレットを追加する場合は、メインメニューからたどって該当するアプリケーションアイコンを右クリックし、現われるポップアップメニューから“ランチャをパネルに追加する”を選択してクリックします。

例として、GNOME 標準のテキストエディタである GNOME テキストエディタ (gedit) を追加してみましょう。まず、アプリケーションをクリックして、“アクセサリ”サブメニューを選択すると、“GNOME テキスト・エディタ”と書かれたアイコンが現われたはずです。このアイコンを右クリックし、“ランチャをパネルに追加する”を選択してください。図 12.5 のように GNOME パネルに gedit のアイコンが表示されましたか？追加されたアイコンをクリックし、実際に gedit が起動するか確認してください。

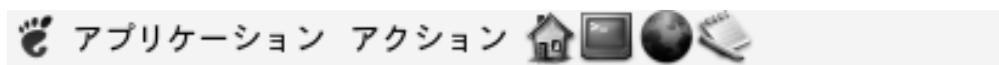


図 12.5 gedit が追加された GNOME パネル

また、登録されていないアプリケーションでも、GNOME パネルを右クリックし、“パネルに追加”、“ランチャ”を選択して表示されるランチャウインドウを設定することで、GNOME パネルに追加することができます。

パネルに追加されたアイコンは、そのアイコンを右クリックし“パネルから削除”を選択することで簡単に削除できます。

12.3 GNOME 上での日本語の扱いについて

GNOME では , GNOME 端末などで日本語入力ができるようになっています.GNOME の標準として採用されている日本語入力システムは,SCIM です. 実際の入力のしかたは, 全角/半角キーを押すことで日本語が入力できる状態になります.

12.4 GNOME で困ったときは (ヘルプシステム)

GNOME を使っていて , 分からないことや , 困ったことがあったときには ヘルプシステム が役立ちます.

ヘルプシステムには , GNOME やその他のアプリケーションについての解説が詳しく載っています. ヘルプシステムを起動するには , “アプリケーション” から “GNOME ヘルプ・システム” を選択します. するとウインドウが出てくるので , 目次から調べたい内容を選びます.

第 13 章

困ったときには

13.1 トラブルシューティング

13.1.1 UNIX 基礎編

- Q. ログインできません.
- A. まず、ユーザー名やパスワードの入力ミスでないか再確認して下さい。大文字/小文字は区別されます。また、Caps Lock は解除しておいてください。
- Q. ログインしてもすぐにログイン画面に戻ってしまいます。
- A. ホームディレクトリの quota 制限 (ディスクの容量制限) に引っかかっている可能性があります。コマンド行ログインをして、不要なファイルを削除して再度ログインしてください。現在自分がどれだけの容量を使っているのか確認するには, du -hs を入力してください。わからない人は, RAINBOW STAFF に対応を依頼してください。なお、各ユーザーのホームディレクトリの容量については、その他、知っておくべきこと-学部、回生別のホームディレクトリの容量の節を参照してください。
- Q. 画面操作ができなくなりました。
- A. 様々な原因が考えられます。ケースバイケースで原因や解決方法は異なります。ここでは実行されているプロセスが何か悪さをしている場合の対処方法について紹介します。この場合、他のコンピュータから当該コンピュータにログインして、原因となっているプロセスを終了させる必要があります。おかしくなったコンピュータを ab0sp001, 新たにログインしたコンピュータを ab0sp002 として具体的な対処方法を示します。まずおかしくなったマシンに ssh コマンドを使って他のコンピュータからログインします。次に, ps コマンドで、実行されているプロセスの一覧を表示させます。ここで得られたプロセス ID (PID) を使ってプロセスを強制終了 (kill) します。以下に例を示します。*¹

*¹ ps コマンドのオプションは OS の種類によって異なります。

```
% ssh ab0sp001          ab0sp001 にリモートログインする
Password:
% ps -u ra012345    ユーザー名    自分のプロセス一覧を表示する
  PID TTY      TIME CMD
 2430 pts/1    0:00:01 csh
 2449 pts/1    0:00:00 ps
 2457 pts/6    0:00:01 xterm
.
.
.
% kill 2457          xterm を終了させる
% exit
```

上記の方法で終了しなかったプロセスに関しては、次のように kill コマンドに -KILL オプションをつけると、プロセスを強制終了することができます。

```
% kill -KILL PID
```

- Q. ウィンドウが変になってしまって、キー入力を受けつけてくれません。どうしようもないのでもう消してしまいたいです。
 - A. xkill というコマンドで強制的に消す方法があります。GNOME 端末からこのコマンドを実行すると、マウスカーソルのアイコンが変わるので、そのまま消したいウィンドウをクリックしてください。
- Q. 誤ってファイルを消してしまいました。
 - A. rm コマンドで削除した場合は、削除の取消はできないため、もう一度作り直すしかありません。Nautilus 上でゴミ箱へ移動するで削除した場合は、ゴミ箱の中に残っています。予防策の 1 つとして、rm コマンドを使うときに -i オプションをつけるというのがあります。このオプションにより、rm コマンドはファイル一個一個について、消すかどうかユーザーに確認を求めてきます。
また、日頃から自分でバックアップを取ることをお薦めします。なお、バックアップについてはバックアップの章を参照してください。
- Q. ゴミ箱に捨てたものは完全に消えていないと言われました。
 - A. ゴミ箱に捨てたものは、ゴミ箱に移動しただけで、完全には削除されていません。完全に削除するには、ゴミ箱を開いてゴミ箱を空にするを実行してください。
- Q. GNOME 端末で、シェルのプロンプト（“%” の文字）が返ってきません。
 - A. 前回実行したコマンドをバックグラウンドで（“&” を最後につけて）起動しなかったのでしょうか。簡単な解決方法は、そのプロセスを終了させればプロンプトが表示されます。強制的に終了させたい場合には、プロンプトを表示させたいターミナル上で Ctrl-c を押すことで、プロセスが終了しプロンプトが表示されます。あるいは、Ctrl-z を押してプロセスを一時停止させた後 bg コマンドを実行すれば、バックグラウンドでの実行に切り替えることができます。なお、bg コマンドについてはシェル-ジョブの操作の節を参照してください。
- Q. GNOME 端末で漢字が化けてしまいます。

- A. GNOME 端末の文字コードの設定が間違っている可能性があります。メニューの [端末]-[文字コード] より EUC-JP または UTF-8 を選択してください。
- Q. GNOME 端末で訳の分からぬ漢字だらけになってしまいました。
- A. GNOME 端末をリセットしましょう。メニューの [端末]-[リセット] でリセットできます。
- Q. リモートログインしたマシンで X のアプリケーションが動きません。
- A. リモートログインをして X アプリケーションを動かすには ssh を用います。以下に ab0sp001 にリモートログインして，Emacs を起動させる場合の例を示します。

```
% ssh -X ab0sp001  
Password:  
% emacs
```

ab0sp001 にリモートログインする

- Q. コマンドの使い方を調べたいです。
- A. たいていのコマンドにはオンラインマニュアルが用意されています。オンラインマニュアルを使用するには man というコマンドを用います。シェルに man command 名 のように入力して利用します。
- Q. シェルを変更したいです。
- A. 通常はログインシェルとして tcsh が設定されています。ログインシェルを変更するには, ldapchsh というコマンドを用います。なお, ldapchsh についてはシェルの設定-ログインシェルの変更の節を参照してください。

13.1.2 Emacs 編

- Q. ファイルに保存をしようとすると, IO error writing :ディスク使用量制限を超過しましたというエラーがれます。
- A. ホームディレクトリの quota 制限 (ディスクの容量制限) に引っかかっています。容量制限を越えて保存しようとしたために、保存できなかったというエラーです。不要なファイルを削除して、空き容量を増やせば保存することができます。現在自分がどれだけの容量を使っているのか確認するには、du -hs を入力してください。わからない人は、RAINBOW STAFF に対応を依頼してください。なお、各ユーザーのホームディレクトリの容量については、その他、知っておくべきこと-学部、回生別のホームディレクトリの容量の節を参照してください。
- Q. ファイルの保存をする前に Emacs が異常 (強制) 終了してしまいました。ファイルは無事でしょうか。
- A. オートセーブ機能によりファイルが保存されている場合があります。

編集対象のファイルがあるディレクトリの #ファイル名# がそのファイルにあたります。Emacs 上で M-x recover-file を使ってオートセーブされた時点のファイルに復元することができます。

M-x recover-file

なお、Emacs のオートセーブについてはスクリーンエディタ-Emacs-ファイル編集に伴う Emacs の基本操作の節を参照してください。

Q. 正しいコマンドを入力しても正常に動作しません。

A. いろいろな原因が考えられます。例えば、間違って何らかのモードになっているかもしれません。

取りあえず、一度 Emacs を起動し直してください。それでも同じなら、コンピュータを変えるか、もしくは個人設定の再編集を行ってください。デフォルトの Emacs を起動する方法は以下の通りです。

```
% emacs -q
```

Q. ファイル編集時に必要な文字まで消してしまいました。元に戻す機能はないですか。

A. 元に戻したいわけですね。undo しましょう。C-/ (Ctrl を押しながら / 入力) か、C-x u です。何度も繰り返せば、どんどん初期状態に近づいていきます。

Q. 英語以外の外国語を編集したいのですが。

A. Emacs は、言語環境を切り換えることで多言語に対応しています。

モード切換えの例：ドイツ語の場合

まず Emacs 上で以下のコマンドを入力します。

```
M-x set-language-environment
```

すると切換えたい言語環境を尋ねてくるので、German と入力します。

```
Set language environment (default, English): German
```

これで、ドイツ語モードになりました。以降は C-¥ でドイツ語を入力できるようになります。ウムラウトなどは、文字の組合せによって入力します。詳しくはヘルプ (M-x quail-help) を参照してください。

Q. キー入力ができないのです。

A. 非常に漠然とした質問で原因もいろいろと考えられますが、Emacs だけにキー入力ができないのであれば、取りあえず C-g を入力して操作を終了してみてください。

Q. 文字が化けてしまいました。

A. 文字コードが違っています。次のコマンドをファイルを見る前に入力してください。

```
C-x RET f コード名
```

コード名がわからない場合は、Space を入力すると一覧が表示されます。詳しくは、UNIX における日本語入力-日本語文字コードの節を参照してください。

13.1.3 メール編

Q. メールを送ったのにエラーメッセージとともに戻ってきました。

A. この場合以下のような原因が考えられます。

case1) 送り先のメールアドレスが間違っていますか？

対処法 1 もう一度宛先を確認してください。特にスペルミスに気をつけてください。

case2) エイリアスの設定の方法がおかしいのでは？

対処法 2 エイリアスの設定部分で宛先を間違えていないかもう一度調べてください。

case3) 上の 2 つともおかしくない場合は、メールのシステムがおかしくなっている場合も考えられます。

対処法 3 メールのヘッダを付添して最寄りの RAINBOW STAFF に相談してください。

Q. Mew を使っていて、間違って大事なメールを消してしまいました。

A. Mew の初期設定では、メールを削除しても trash フォルダに移動するだけで、実際には削除されません。メール-Mew の使い方-メッセージの消去の節を参照してください。

Q. 正しい操作をしているはずなのに、メールを取り込めない。メールを送れない。

A. 本当に操作に間違いがないのであれば、ホームディレクトリの容量制限にひっかかって、メールを取り込んだり、書きかけのメールを保存する draft が作成できないかもしれません。いらないファイルを消してから、もう一度試してみてください。

13.1.4 デバイス編

Q. PDF ファイルを印刷できない。

A. lp コマンドで印刷していませんか。PDF ファイルは Adobe Acrobat Reader を利用して印刷する必要があります。詳しくは印刷-PDF ファイルの閲覧と印刷の節を参照してください。

Q. LATEX ファイル (DVI ファイル) を印刷できない。

A. lp コマンドで印刷していませんか。LATEX ファイルは dvips コマンド、または DVI プレビューをを利用して印刷する必要があります。詳しくはアプリケーション-LATEX による文書作成環境の節を参照してください。

Q. tar でフロッピディスクに取ったアーカイブをリストアしようとしたのですが、もとに戻りません。

A. きちんとフロッピディスクに保存されていないか、もしくは相対パスで保存されたため、自分の予期せぬ所にリストアされている可能性があります。

まず、tar コマンドを用いてフロッピディスクの中身を確認しましょう。tar コマンドの詳しい使い方はバックアップの章を参照してください。中身の情報が表示されたならきちんと保存されていると思われます。次に表示された情報を見て保存されているファイルが相対パスで指定されているか、絶対パスで指定されているかを確認してください。絶対パスで保存されている場合はそこで指定されているパスの場所にリストアされているはずなのでもう一度そのディレクトリを確かめてください。

Q. 家の PC で作ってきたテキストファイルを持ってきました。Emacs 等で再編集したいのですがどうすればよいでしょうか。

A. 家の PC からフロッピディスクや CDROM などのメディアで持ってきたデータを UNIX 上に移したいわけですね。mtools コマンドや mount コマンドを利用してください。詳しくは、バックアップの章を参照してください。

13.1.5 L^AT_EX 編

- Q. L^AT_EX ファイル (DVI ファイル) を印刷したい.
- A. xdvi コマンドで DVI プレビューを起動して , Print ボタンから印刷することができます. なお , DVI プレビューについてはアプリケーション-L^AT_EX による文書作成環境の節を参照してください.
- Q. 特定のページだけ印刷したい.
- A. xdvi コマンドで DVI プレビューを起動して , 印刷したいページを表示します. そして , Print ボタンを押し , 出てきたダイアログの Print ボタンを押して Current Page を選択すれば , 表示されているページのみを印刷することができます. なお , DVI プレビューについてはアプリケーション-L^AT_EX による文書作成環境の節を参照してください.
- Q. L^AT_EX ファイル (DVI ファイル) を PDF ファイルに変換したい.
- A. dvipdfmx コマンドで変換することができます.
- 例えば , *file_name.dvi* という L^AT_EX ファイルを PDF ファイルに変換したい場合 ,

```
% dvipdfmx file_name.dvi
```

となります.

- Q. tex ファイルから dvi ファイルに変換する際に途中で処理が止まってしまいます.
- A. L^AT_EX の構文にそぐわない箇所があると , L^AT_EX は処理を中断し , ユーザーの判断を待ちます. エラーとなったときのプロンプトには” *” と” ? ” の2種類があります.
- * は , 指定されたファイルが見つからない場合や , 終了コマンドが指定されていなかった場合のプロンプトです. このときは , 正しいファイル名や必要なコマンドを入力するか , 強制的に終了させます. 強制的に終了させるには , Ctrl-d を入力してください.
- ? は , タイプミスなどによる命令の間違いなどの場合のプロンプトです. この場合は , h^{*2} , e^{*3} , x^{*4} , Enter^{*5} などをタイプすることで対処します.
- Q. tex ファイルのソースを変更したのにプレビューで確認すると変更箇所が反映されません.
- A. tex ファイルを変更しただけでは , プレビューに反映されません. もう一度 , コンパイルをしてください. 変更があれば , その都度コンパイルをしないといけませんので注意してください.
- Q. tex ファイルに間違いが見当たらないのに , エラーがかえってきます.
- A. tex ファイルの漢字コードを疑ってください. tex ファイルを書いた環境が UNIX であれば , 漢字コードがだいたい EUC であるためうまくいきますが , Windows で書いた場合 , シフト JIS である可能性があります. シフト JIS のまま , コンパイルしようとしても漢字コードが違うためコンパイルできません. 漢字コードを EUC に変換する方法については , UNIX における日本語入力- 日本語文

^{*2} 簡単なヘルプを表示する.

^{*3} エディタが起動され , エラー行にジャンプする. 修正後 , エディタを終了するとその位置より処理が実行される.

^{*4} 処理を中断する.

^{*5} そのエラーを無視して , 次に進む.

字コードの節を参照してください。漢字コードを変換したら、もう一回コンパイルしてみましょう。

13.2 その他 Q&A

Q. 画像のフォーマットを変換したいのですが。

A. 画像形式の変換には `convert` コマンドを使います。例えば、`file_name.eps` という EPS ファイルを、`file_name.bmp` という BMP ファイルに変換したい場合は次のように入力します。

```
% convert file_name.eps file_name.bmp
```

`convert` コマンドを使えば、画像形式の変換だけでなく、解像度の処理、画像の大きさの変更もできます。詳しくは、`man convert` を参照してください。

Q. レポート作成の際に実験結果の画像をプリントアウトしたり、実行過程をプリントアウトしたりしたいのですが。

A. デスクトップ全体のイメージを画像ファイルとして保存するにはキーボード右上の Print Screen キーを押します。また、1つのウインドウのみのイメージを画像ファイルとして保存するには `xwd` コマンドを用います。例えば、`file_name.eps` というファイル名でウインドウのイメージをファイルに保存したい場合、

```
% xwd > file_name.xwd
```

を実行すると、マウスカーソルのアイコンが変わるので、そのままキャプチャしたいウインドウをクリックしてください。これで XWD(X-Windows Dump image data) という特殊なフォーマットで保存されます。次に

```
% convert file_name.xwd file_name.eps
```

を実行して、XWD ファイルを EPS ファイルに変換します。

実行過程をコマンド入力ごとファイルに残すには `script` コマンドがあります。使用方法を以下に示します。

```
% script
スクリプトを開始しました、ファイルは typescript です
%
.....
% exit
スクリプトを終了しました、ファイルは typescript です
%
```

これで script を実行してから exit が入力されるまでのログが typescript ファイルに残ります。ただし、script はラインフィードやバックスペースも含め、すべてをログファイルに書き込みます。画面の表示の通りに実行過程をプリントアウトしたい場合は、実行結果を Emacs にコピー & ペーストしてください。コピー & ペーストの方法は GNOME 端末上に表示されている実行結果を選択し、選択された状態のまま Emacs に切替えて C-y で貼り付けることができます。

- Q. 画像を表示させたいのですが。
- A. ImageMagick という画像ファイル表示ツールがあります。display というコマンドを用います。なお、display については図・グラフの作成-画像を表示/変換-ImageMagick の節を参照してください。
- Q. 各ディレクトリがどのくらいの容量なのか知りたいのですが。
- A. du コマンドを用いることによって調べることができます。

```
% du directory
```

上記のように入力すると、directory で指定したディレクトリ以下の全てのディレクトリについてそれぞれの使用量を得ることができます。また、-s オプションをつけることによって指定したディレクトリ以下の総使用量だけを得ることもできます。注意して欲しい点は、du コマンドの出力は block 単位であるということです。1 block は 512, 1024, 2048 byte などのどれかなので、使用している OS ではどのサイズかを man du で確認してください。

- Q. ファイルが消えてしまいました。自分で消した覚えはありません。
- A. いろいろと原因が考えられますが、セキュリティに焦点を当てて回答します。消えてしまったファイルのアクセス許可は適切に設定されていましたか。他人への w(write) の許可属性は不許可になっていましたか。誰もいじらないファイルが消えることはまず考えられません。そのため、ファイルが消えてしまうということは自分のセキュリティ管理が行き届いていなかったことに原因がある可能性が高いのです。UNIX の基本- ファイルシステム- ファイルへのアクセス許可の変更の節を参照して個人のファイルのセキュリティの管理をきちんと行ってください。
- Q. 起動後の環境が違ったり、コマンドがきかないなどの症状がでます。
- A. ファイルサーバが故障していてホームディレクトリがマウントされていない場合などに、この様な症状がでます。最寄りの RAIBOW STAFF に相談してください。
- Q. 個人ホーム領域で Web ページ (ホームページ) を作りたいのですが。
- A. 学生・院生が個人のホーム領域で作成している Web ページは学内 (RAINBOW ネットワーク) からのみ閲覧可能です。但し、授業や研究目的に限り、学部長・研究科長の申請により学外公開が可能になります。学外公開の必要がある場合は、所属学部・研究科に申し出てください。
- Q. ファイルを持って帰りたいのですが、サイズが大きくて一枚のフロッピディスクに収まりません。どうすればよいでしょうか。
- A. サイズの大きなファイルをフロッピディスクに入れて持ち帰りたい場合、ファイルを一時的に分割すれば可能です。詳しくはバックアップの章を参照してください。

13.2.1 質問の仕方

RAINBOW を利用していると，Q&A を見ても自分の知りたいことが載っていない，どうしても自分ひとりでは解決ができない，という場合がでてくると思います。この様な場合，より詳しい人にたずねることも問題を解決する手段の 1 つです。RAINBOW にはそれら質問を受ける手段を以下のようにいくつか提供しています。

- よくある質問 Q&A Web ページ
(<http://www.ritsumei.ac.jp/acd/mr/i-system/faq/index.html>)
に似た質問がないか調べてみる。
- RAINBOW サービスセンターのスタッフに直接尋ねる。

何でも他人に聞いてばかりというのもあまりよろしくありませんが，初心者が“これって聞いたら恥ずかしいことなんだろうか。”などと考えて質問するのを躊躇するのはもっとよくないことです。自分なりに考えてみて，自分なりに調べてみてそれでもわからなければ，どんどん質問しましょう。メールでの質問は，受ける側も手の空いた時間に回答することができます。

質問をする場合には幾つかの事柄に注意しなければなりません。まず，質問の内容を整理して的確な状況を相手に伝えることです。初心者からの質問に，“インターネットがしたいのですが，どうやるんですか。”なんていう質問がよくありますが，受け手にしてみれば何が言いたいのか良くわかりません。この場合，“Firefox などのブラウザを使って，Web ページを参照したいのかな?”などと推測で話を進めなければなりません。これでは回答側も即答しかねてしまいます。この場合，Firefox とインターネットの違いが判らないならば，画面の描写でもなんでも構いません，もっと細かく質問の内容を書くことができるはずです。

トラブルが発生したときの質問も，

- 何をしていて，どの様な状態になったか。
- どんなプログラムを走らせていたか。
- こういうことをしたが，状態は変わらなかった。
- 出ているエラーメッセージ。
- トラブルが発生した時間，使用していたコンピュータ。

など，できるだけ細かく状況を説明すれば，答えるほうも回答が容易になります。

エラーメッセージなどはログを取っておいてメールに添付しておくと良いでしょう。script コマンドなどを使うと簡単にログが取れます。

自分にとってわかり切っているような情報は結構見落しがちです。質問は相手の立場になって考えましょう。受け手，読み手が答えたくなるような質問だと的確な回答が得られる可能性が高くなります。

付録

付録 A リモートアクセス

研究室などでコンピュータを使っていると、RAINBOW のコンピュータ（ホスト）とやり取りしたいことがでてくるかと思います。この章では、ネットワーク上でできる様々なやり取りとその方法について説明しています。

付録 A.1 基礎事項

まずははじめに、この章で使われる用語の説明をします。

ホスト

ネットワークにつながれているコンピュータのことを、ホストマシン あるいは単に ホスト と呼びます。ホストにはそれぞれ名前がついていて、それを ホスト名 と呼びます。

今、自分が使っているコンピュータのホスト名は、`hostname` を使って調べることができます。

```
% hostname
ab0sp001
```

今、ネットワークを通じて他のマシン上で作業したいとします。そのとき、遠隔操作される マシンをリモートホスト、遠隔操作をするホスト（今使っているマシン）を ローカルホスト と呼びます。

そして、リモートホストにログインすることを リモートログイン と呼びます。リモートログインの具体的な方法は付録 A.2 節を参照してください。

また、離れたホスト間でファイルのやりとりもできます。ファイルの転送には主に FTP(File Transfer Protocol:ファイル転送プロトコル) というものが使われ、FTP クライアントを利用することによって実現されます。ファイル転送の具体的な方法は、付録 A.3 節を参照してください。

クライアント/サーバ

クライアント は、サーバ に対して何らかの要求を出し、サーバはその要求を受けて仕事をします。このような協調の方式をクライアント/サーバ 方式と呼びます。一般的に、サーバが 1 つでクライアントが複数という形を取ることが多く、これによってデータの共有や一元管理を行います。そのため、クライアントとサーバは別々のホストであることが多いですが、同一のホストでも構いません。

X Window System は、画面表示をクライアント/サーバ方式で行っています。X Window System では、グラフィック画面を制御するサーバは各マシンに 1 つだけ動くことができます。これを X サーバと呼びます。

グラフィック画面を直接制御できるのは、この X サーバだけです。そのため、他のプログラムは、直接、画面にグラフィックを表示することはできません。しかし、他のプログラムは、X サーバに“依頼”して、X サーバにグラフィックを表示してもらいます。この依頼には X プロトコル というものが使われます。

通常、グラフィックベースのアプリケーション（X クライアント）を動かすとき、表示の要求をローカルホストの X サーバに送り、ローカルの X サーバがグラフィックを表示します。そこで、リモートホストで X クライアントを動かすときに、表示の要求をローカルホストの X サーバへ送ることにより、ローカルホストの画面に表示を行うことが可能です。

この機能について詳しくは、付録 A.4 節を参照してください。

付録 A.2 リモートログインとリモートシェル

リモートホストにログインすると、自分の利用している端末がリモートホストの端末として利用できるようになります。表示されるプロンプトもリモートホストのものですし、入力するコマンドもリモートホストで実行されます。また、リモートホストにログインすることなしに、直接リモートホストにコマンドの実行を指示することもできます。

ここでは、これらを行なう手段として、ssh, telnet というコマンドについて説明していきます。

ssh (Secure Shell)

ssh は、リモートホストにログインしたり、リモートホストでコマンドを実行するためのコマンドです。ホスト名が *hostname* であるホストにリモートログインするときは、次のように入力します。

```
% ssh hostname
```

ここでは、例として ab0sp001.bkc.ritsumei.ac.jp というホストへリモートログインしてみます。

```
% ssh ab0sp001.bkc.ritsumei.ac.jp
The authenticity of host 'ab0sp001.bkc.ritsumei.ac.jp (172.24.21.1)' can't be established.
RSA key fingerprint is 19:47:65:9b:03:9f:0e:71:1c:14:23:1e:c8:0b:66:7b.
Are you sure you want to continue connecting (yes/no)?
```

そのホストに対して ssh を使って初めてログインするときは、上記のように質問されます。二度目以降には、上記の質問は出ません。ここでは、ひとまず yes と入力してください。

```
% ssh ab0sp001.bkc.ritsumei.ac.jp
The authenticity of host 'ab0sp001.bkc.ritsumei.ac.jp (172.24.21.1)' ca
n't be established.
RSA key fingerprint is 19:47:65:9b:03:9f:0e:71:1c:14:23:1e:c8:0b:66:7b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ab0sp001.bkc.ritsumei.ac.jp,172.24.21.1' (R
SA) to the list of known hosts.
ra012345@ab0sp001.bkc.ritsumei.ac.jp's password:
```

次に、パスワードを入力します。このとき、入力したパスワードは画面に表示されないことに注意してください。

```
% ssh ab0sp001.bkc.ritsumei.ac.jp
The authenticity of host 'ab0sp001.bkc.ritsumei.ac.jp (172.24.21.1)' ca
n't be established.
DSA key fingerprint is 19:47:65:9b:03:9f:0e:71:1c:14:23:1e:c8:0b:66:7b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ab0sp001.bkc.ritsumei.ac.jp,172.24.21.1' (D
SA) to the list of known hosts.
ra012345@ab0sp001.bkc.ritsumei.ac.jp's password:
Last login: Tue Feb 13 20:21:42 2007 from ab0sp002.bkc.ritsumei.ac.jp
%
```

これで、*ab0sp001.bkc.ritsumei.ac.jp* にリモートログインできました。ホスト名を *hostname* で確認してみてください。

リモートホストでひとつだけコマンドを実行したいという場合には、わざわざログインする必要はありません。ホスト名の後にコマンドを指定すると、そのコマンドをリモートホスト上で実行できます。

```
% ssh hostname command
```

ssh には、様々な便利なオプションがあります。例えば、リモートホストでのユーザー名がローカルホストでのユーザー名と異なる場合は、-l オプションでユーザー名を指定することができます。

```
% ssh -l username hostname
```

ssh コマンドでは、(ほぼ) 全てのデータを暗号化して通信を行います。そのため、インターネットのような、誰がどこで通信内容を覗き見しているかわからないような環境でリモートログインをするときの事実上の標準となっています。セキュリティを意識する方は、後述の *telnet* より *ssh* を利用することをお勧めします。

telnet

telnet は、リモートホストにログインするためのコマンドです。*telnet* でリモートログインすると、自分の利用している端末がリモートホストの端末として利用できるようになります。表示されるプロンプト

もリモートホストのものですし、入力するコマンドもリモートホストで実行されます。

例えば、ホスト名が *hostname* であるホストにログインする場合は、次のようにします。

```
% telnet hostname
```

それでは実際に練習として `remote.ritsumei.ac.jp` への接続を `telnet` で行ってみましょう。

```
% telnet remote.ritsumei.ac.jp
Trying 172.24.21.2...
Connected to in3rp001.bkc.ritsumei.ac.jp.
Escape character is '^]'.

Vine Linux 3.2 (Ducru Beaucaillou)
Kernel 2.4.33-rits1 on an i686
login: ra012345 ← ユーザー名
Password: ??????? ← パスワード (入力した文字は画面に表示されません)
Last login: Tue Feb 13 21:37:41 from ab0sp001.bkc.rit
%
```

付録 A.3 コンピュータ間でのファイル転送

ここで触れるファイル転送は、リモートホストにあるファイルをローカルホストにコピーしてきたり、ローカルホストのファイルをリモートホストにコピーしたりすることを呼びます。

研究室のホストにあるファイルを RAINBOW のホーム上に持って来たり、その逆のことをするときに使います。また、インターネット上で公開されている様々な資源（プログラムのソースなど）入手する際にも、ファイルをこちらに転送することになります。

ここでは、このファイル転送の一般的な方法について説明します。

`ftp`

ファイル転送の手段として、`ftp` というコマンドを用いる方法があります。`ftp` を起動すると、“`ftp>`”とプロンプトが表示されるので、コマンドを使ってファイル転送を行います。

ここで、`ab0sp001`（ユーザー名：`ra012345`）に `ftp` でアクセスしてみましょう。

```
% ftp ab0sp001
Connected to ab0sp001.bkc.ritsumei.ac.jp (172.24.21.2).
220 ProFTPD 1.3.2c Server (RAINBOW) [172.24.21.2]
Name (ab0sp001.bkc.ritsumei.ac.jp:ra012345): ra012345      ユーザー名
331 Password required for ra012345.
Password: ???????      パスワード (入力した文字は画面に表示されません)
230 User ra012345 logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

ここで，“?”と打つと，使えるコマンド一覧が出てきます。UNIXの基本的なコマンドは一通り使えます。それ以外に ftp でよく使用するコマンドは get と put と bye です。

“ftp>”の状態で，ls や cd などを使い，転送したいファイルのあるディレクトリに移動してください。持ってきたたいファイルが *a.txt* だったとすると，次のようなコマンドでファイルを持ってくることができます。

```
ftp> get a.txt
```

また，ftp を実行したディレクトリ内（ローカルホスト）の *b.txt* というファイルをリモートホストに送りたい場合は，

```
ftp> put b.txt
```

で，送れます。

単に ftp とだけ入力した場合，この状態ではリモートホスト名を指定していないため，ファイル転送を行なうことはできません。ここで open コマンドを入力することで，リモートホストを指定できます。

```
% ftp  
ftp> open ab0sp001.bkc.ritsumei.ac.jp  
Connected to ab0sp001.bkc.ritsumei.ac.jp (172.24.21.2).  
220 ProFTPD 1.3.2c Server (RAINBOW) [172.24.21.2]  
Name (ab0sp001.bkc.ritsumei.ac.jp:ra012345):
```

引き続いて，他のリモートホストとファイル転送をしたい場合は，現在の接続を終了させるために，close を入力した後で，open コマンドを利用します。

```
ftp> close  
221 Goodbye.  
ftp> open ab0sp002  
Connected to ab0sp001.bkc.ritsumei.ac.jp (172.24.21.2).  
220 ProFTPD 1.3.2c Server (RAINBOW) [172.24.21.2]  
Name (ab0sp001.bkc.ritsumei.ac.jp:ra012345):
```

ファイル転送が終わったら，最後に bye で ftp を終了します。

```
ftp> bye  
221 Goodbye.  
%
```

基本はこれだけですが，色々便利な機能がついているので，興味のある人はオンラインマニュアルを読んでください。

ncftp

インターネット上には、様々なところで開発されたプログラムのソースなどのインターネット資源を収集した、Anonymous FTP サイト というものがあります。anonymous とは匿名という意味で、誰でもアクセスできるようになっています。Anonymous FTP サイトはインターネット上に無数にあります。ncftp は、Anonymous FTP 向きの高機能なコマンドです。特徴として

- カレントディレクトリが表示される。
- TAB を押すことで、ファイル名を補完することができる。
- Ctrl-p, Ctrl-n や 矢印キーの上、下でヒストリ機能が使える。

ncftp を利用するために、ncftp を起動後、

```
ncftp / > set passive 0
```

と入力してください。これで、ncftp が利用するための設定ができました。この設定は、一度行なうと保存されるので 2 回目以降は行なう必要はありません。

ncftp は、Anonymous FTP サイト以外でも利用可能です。そうしたい場合は、オプション -u をつけてます。これにより、Anonymous FTP ログインを自動で行わないようになります。

```
% ncftp -u ra012345 ab0sp001.bkc.ritsumei.ac.jp
```

というように使います。すると、

```
Password required for ra012345.
```

と表示されますので、パスワードを入力してください。ちなみに、パスワードを入力する場合、パスワードの文字の代わりに画面には “*” が表示されるので、注意してください。

scp

scp は、cp と同じような感覚でホスト間のファイルコピーができるコマンドです。あるホストから、*a.txt* というファイルを *hostname* にファイル転送したい場合は、以下のようにします。

```
% scp a.txt username@hostname:
```

なお、コピー先ホストのユーザー名がコピー元ホストのユーザー名と異なる場合は、ホスト名の前に *username@* をつけることで、ユーザー名を指定することができます。

次に、*hostname* の *txt* というディレクトリにある *b.txt* というファイルを、今自分のいるホストに持てきたいときは以下のようにします（例では、*hostname* でのユーザー名が *username* であるとします）。

```
% scp username@hostname:txt/b.txt .
```

付録 A.4 リモートの X クライアントを利用する

ここでは、リモートホストで動かしたい X クライアントをローカルホストで表示する方法を説明します。

ssh X フォワーディング

通常、ssh でリモートログインしてから X クライアントを起動すると、

```
% ssh ab0sp001.bkc.ritsumei.ac.jp
ra012345@ab0sp001.bkc.ritsumei.ac.jp's password:
Last login: Tue Feb 13 21:44:12 from from ab0sp002.bkc.ritsumei.ac.jp
% xeyes
Error: Can't open display:
```

と表示されて実行できません。リモートログイン先で X クライアントを実行したい場合には、リモートログインするときに -X オプションを指定して ssh を実行してください。

```
% ssh -X ab0sp001.bkc.ritsumei.ac.jp
ra012345@ab0sp001.bkc.ritsumei.ac.jp's password:
Last login: Tue Feb 13 21:41:17 from ab0sp002.bkc.ritsumei.ac.jp
% xeyes
```

付録 A.5 その他のコマンド

finger

このコマンドはユーザーログイン情報を見るコマンドです。

```
% finger taro
```

とすれば、全ての *taro* さんの情報を出力してくれます。

もし、自分のホームディレクトリに .plan や .project ファイルを作つておくと誰かが finger したときに、それを出力してくれます。

また、

```
% finger @hostname
```

とすると、*hostname* にログインしているユーザーのユーザー名などが表示されます。

付録 B バックアップ

バックアップとは、大切なファイルをフロッピディスクなどに記録し、保存しておく作業のことです。

作業ミスや、コンピュータの異常などによってファイルが消えてしまうことはよく起こります。このとき、バックアップがあればデータを取り戻すことができます。

また RAINBOW では、個人で使用できるハードディスクの容量には制限があり、この容量を越えるとログインできなくなるなどの不具合が発生します（付録 D.2 節参照）。よって、すぐに使用するファイル以外はフロッピディスクにバックアップしハードディスク上から削除する必要があります。

さらに、RAINBOW のユーザーの中には自宅にパソコンを所有している人も多いと思います。“学校でやった作業の続きを家でやりたい”などというときには、フロッピディスク等にバックアップを取って家にデータを持って帰り、家のパソコンで作業の続きをすることも可能です。

それでは、以下にユーザーのためのバックアップ方法を示します。

付録 B.1 MS-DOS ファイルシステムのフロッピディスク

UNIX と自宅にあるパソコン（MS-DOS、Windows、MacOS など）とでデータをやりとりする場合、MS-DOS ファイルシステム^{*6} のフロッピディスクを用いることになります。

ここでは、UNIX で MS-DOS ファイルシステムのフロッピディスクを取り扱う方法を説明していきます。

初期化

一度も使ったことのないフロッピディスクを使う場合、初期化（フォーマット）と呼ばれる作業を行う必要があります。なお、データの入っているフロッピディスクを初期化してしまうとデータが消えてしまうので、この作業は慎重に行ってください。

```
% fdformat /dev/fd0H1440
% /sbin/mkfs -t vfat -c /dev/fd0H1440
```

マウント

フロッピディスクにアクセスする場合、通常 マウント と呼ばれる作業を行います^{*7}。フロッピディスクのマウントを行うには、mount コマンドを使用します。

```
% mount /mnt/floppy
```

マウントを行うと、/mnt/floppy とフロッピディスクの中身が同じものとなります。

^{*6} FAT 形式 と呼ばれることもあります。

^{*7} 後述の mtools を使用すれば、マウントせずにフロッピディスクにアクセスすることができます。

例えば、

```
% cp ~/test.txt /mnt/floppy
```

とすれば、ホームディレクトリにある *test.txt* がフロッピディスクにコピーされることになります。
フロッピディスクを取り出すときは必ず、

```
% umount /mnt/floppy
```

を実行しアンマウントした後に取り出してください

mtools

mtools を使用すると、マウントすることなしに MS-DOS 形式のファイルを扱うことができます。Windows などに慣れていると、ついアンマウントせずにフロッピディスクを抜いてしまいがちです。そのような方には、**mtools** の利用をお勧めします。

では、**mtools** のコマンドの 1 つである **mcopy** を使って、ホームディレクトリにある *test.txt* を 2HD 形式のフロッピディスクに保存してみましょう。

```
% mcopy ~/test.txt a:
```

この例で、一番最後についている *a:* はドライブ名と呼ばれ、フロッピディスクの形式により表 1 のドライブ名を使います。

表 1 mtools でのドライブ名

| ドライブ名 | フロッピディスクの種類 |
|-------|-------------|
| a: | 2HD |
| b: | 2DD |

また、2HD 形式のフロッピディスクに保存してある *test.txt* を UNIX のホームディレクトリにコピーする場合は次のようにします。

```
% mcopy a:test.txt ~/
```

このように、フロッピディスク内のファイルを指定する場合は“ドライブ名:ファイル名”と指定します。**mtools** では、アンマウントをする必要はありません。フロッピディスクドライブのアクセスランプが消えたのを確認して、フロッピディスクを取り出してください。

以下に、**mtools** のコマンドのうち主なものを紹介します。詳しくはオンラインマニュアルを参照してください。

mattrib MS-DOS ファイルの属性を変更する。
mcd フロッピディスク上の MS-DOS ディレクトリを移動する。
mcopy フロッピディスク上の MS-DOS ファイルを UNIX へコピーする。もしくは UNIX のファイルを MS-DOS ファイルとしてフロッピディスク上にコピーする。
mdel フロッピディスク上の MS-DOS ファイルを削除する。
mdir フロッピディスク上の MS-DOS ディレクトリを表示する。
mformat 物理フォーマットされたフロッピディスクを MS-DOS 形式に論理フォーマットする。
mlabel フロッピディスク上に MS-DOS volume label を作成する。
mmd フロッピディスク上に MS-DOS ディレクトリを作成する。
mrd フロッピディスク上の MS-DOS ディレクトリを削除する。
mren フロッピディスク上の MS-DOS ファイルをリネームする。
mtype フロッピディスク上の MS-DOS ファイルの内容を表示する。

付録 B.2 MS-DOS ファイルシステムの USB メモリ

データのやりとりには、USB メモリもご利用になれます。ここでは、UNIX で MS-DOS ファイルシステム^{*8}の USB メモリを取り扱う方法を説明していきます。

マウント

USB メモリは、初期状態ですでにフォーマットされています。USB メモリのマウントを行なうには、**usb-mount** コマンドを使用します。

```
% usb-mount
```

実行すると /mnt/usb[1-4] のいずれかに USB メモリの中身が表示されます。

USB メモリを取り出すときは必ず、

```
% usb-umount
```

を実行しアンマウントした後に取り出してください。

付録 B.3 UNIX ファイルシステムとしてのバックアップ

UNIX ファイルシステムのフロッピディスクを取り扱う場合、使い始める前には **mount** コマンド、取り出す前には **umount** コマンドを実行する必要があります。

各コマンドの実行方法は MS-DOS ファイルシステムのマウント方法（付録 B.2 節）と同じです。

^{*8} FAT 形式と呼ばれることがあります。

UNIXではtarコマンドを使用してバックアップを取ったり、そのリストアを行ったりすることができます。tarコマンドは複数のファイル（または、ディレクトリ）をアーカイブと呼ばれる1つのファイルにまとめるコマンドです。

ここでは直接フロッピディスクにアーカイブを保存する方法のみを示しますが、デバイスファイル（/dev/fd0）の代わりにファイル名（test.tar.gzなど）を入力することにより、アーカイブを通常のファイルとして保存することもできます*9。

- バックアップする場合

```
% tar cvf /dev/fd0 test.txt
```

と入力すると、ファイルtest.txtをフロッピに落とす*10ことができます。また、

```
% tar cvf /dev/fd0 filename1 filename2 ...
```

と入力すると、複数のファイルを落とすことができます。

ディレクトリについても同様ですが、注意すべき点が1つあります。引数にファイル名の代わりにディレクトリ名を指定すると、指定されたディレクトリ以下のファイルを保存することができるのですが、このときディレクトリのパスを絶対パスではなく相対パスで書くことをお薦めします。これはリストア（復旧）する時関わってくることなのですが、絶対パスで落としてしまうとリストア時に、絶対パスで指定された場所にしか戻すことができないからです。

以下にディレクトリごとバックアップを取る場合の例を示します。

```
% tar cvf /dev/fd0 ./dirname
```

- バックアップファイルをフロッピから取り出す場合

```
% tar xvf /dev/fd0
```

と入力するとフロッピに入っているファイル全てがカレントディレクトリに展開されます。特定のファイルだけを取り出す場合は、

```
% tar xvf /dev/fd0 filename1 filename2 ...
```

と、引数に取り出したいファイル名を列挙してください。ディレクトリの場合も同様です。

（注意）ディレクトリを展開するときアーカイブが絶対パスか相対パスか気をつけるようにしてください

*9 通常のファイルを扱う方法については、オンラインマニュアルを参照してください。

*10 バックアップを主な目的として、別の媒体などにコピーをしたりすることをこのように表現することがあります。

さい。

- アーカイブ(フロッピディスク)の内容を見る場合

アーカイブの内容(ここではフロッピディスクの内容)を見る場合、

```
% tar tvf /dev/fd0
```

としてください。画面にアーカイブの内容が表示されます。

付録 B.4 サイズの大きなファイルのバックアップ

ファイルを圧縮する

UNIX上では、compressやgzipといったコマンドを利用してファイルを圧縮し、ファイルサイズを小さくすることができます。

- compressコマンドを用いる方法

(以下では圧縮したいファイルを *filename* とします)

```
% compress filename
```

圧縮されたファイルは、ファイル名の最後に .Z がつき、*filename.Z* という名前になります。
compressで圧縮したファイルの復元には、uncompressコマンドを用います。

```
% uncompress filename.Z
```

- gzipコマンドを用いる方法

(以下では圧縮したいファイルを *filename* とします)

```
% gzip filename
```

圧縮されたファイルは、ファイル名の最後に .gz がつき、*filename.gz* という名前になります。
gzipで圧縮したファイルの復元は、gzipコマンドを -d オプションをつけて使用します。

```
% gzip -d filename.gz
```

分割してフロッピディスクに保存する

ファイルサイズが大きい場合、例えば1つのファイルがフロッピディスク1枚に入り切らない場合のバックアップ方法について説明します。この場合、1つのファイルを数枚のフロッピディスクにわけて保存することになりますが、ここでは2種類の方法を示します。

- tar のマルチボリュームオプションを用いる方法

RAINBOW にインストールされている tar コマンドは、一般に GNU tar と呼ばれる GNU バージョンのものです。GNU tar には M (マルチボリューム) というオプションがあり、これを使用すると、1つのアーカイブを複数のフロッピディスクに分割して保存できます。

以下に具体的な使用例を示します。なお、ここで示すファイル *sample.gz* は 2Mbytes の容量があるものとします。

```
% tar cvfM /dev/fd0 sample.gz  
ボリューム # 2 ('/dev/fd0') を準備します。リターンキーを押してください:
```

上のようなメッセージが表示されたら、フロッピディスクを入れ換え、Enter を入力してください。これで、*sample.gz* は 2 枚のフロッピディスクに保存されました。

このアーカイブを元に戻す場合は、次のようにしてください。

```
% tar xvfM /dev/fd0  
ボリューム # 2 ('/dev/fd0') を準備します。リターンキーを押してください:
```

メッセージが表示されたら、先程と同様に、指示通りフロッピディスクを入れ換えて、Enter を押してください。

- split コマンドを用いる場合

split コマンドはサイズの大きなファイルを 1 枚のフロッピディスクに入り切る容量の複数のファイルにあらかじめ切り分けるコマンドです。またファイルの分割、連結は ASCII 文字列のファイルの方が扱い易いので、ここでは、バイナリファイルをエンコード・デコードするコマンドも合わせて紹介します。

例として *sample.gz* というバイナリファイルに圧縮されている 2MBytes のファイルを扱います。まず、バイナリを ASCII 文字列に変換（エンコード）します。変換には uuencode というコマンドを用います。コマンドの使用法は次の通りです。

```
% uuencode input_file file_label > output_file
```

最後の *file_label* とは何かと思うでしょうが、通常は特に気にする必要はありません。ファイル名と同じものを指定してください。*output_file* で指定するファイル名は出力されるファイルの名前です。出力ファイルに *sample.gz.uucd* とした場合は以下のようになります。

```
% uuencode sample.gz sample.gz > sample.gz.uucd
```

この結果 ASCII 文字列に変換された *sample.gz.uucd* ができるがります。

(注意) ASCII 文字列に変換されるとファイルの容量が大きくなります。

次に split コマンドを用いてファイルを分割します。以下のような形式で、コマンドを実行してく

ださい。

```
% split -行数 input_file output_file
```

行数はデフォルトで 1,000 行となっています。1,000 行で約 60KB と考えてください。*output_file* を指定しないと出力ファイル名は *x* となり *aa*, *ab*, *ac*, ... を順番に付加したファイルがつくれます。

具体的には以下のように入力を行います。

```
% split -10000 sample.gz.uucd sample.
```

結果は、*sample.aa*, *sample.ab*, *sample.ac*, *sample.ad* という 600KB のファイルが 4 つできあがることになります。

分割したファイルを再び利用するために、各ファイルを結合し元に戻さなければなりません。各ファイルは、uuencode を用いて ASCII 文字列 に変換されたものなので、ここでいうファイルの結合とはそれらの各ファイルを順序通りに 1 つのテキストファイルにすることです。つまり、split を行う前のファイルを生成することです。cat コマンドを用いた上記 *sample.gz.uucd* ファイルの結合の例を以下に示します。

```
% cat sample.aa sample.ab sample.ac sample.ad > sample.gz.uucd
```

上記のように cat コマンドの引数に結合したい各ファイルを順序通りに与えると、結合した結果をリダイレクションで指定したファイルに出力します。ここで *sample.gz.uucd* は出力ファイル名です。

そして得られた *sample.gz.uucd* を次のように uudecode コマンドを用いて変換することによって元のバイナリファイルに復元することができます。

```
% uudecode sample.gz.uucd
```

上記の結果、再び *sample.gz* というファイルが生成されます。これが復元された元のファイルです。

付録 B.5 その他の記憶媒体でのバックアップ方法

最後に、CD-R を用いたバックアップ法と GUI からのマウント方法を紹介させていただきます。

CD-R をコンピュータに挿入し、デスクトップの何もない所で右クリックを行うと図 1 のようなメニュー画面が現れます。そのメニュー画面からディスクを選択し、CD-R を選択します。図 3 のような媒体に応じたアイコンが表示されればマウント成功です。同様に、フロッピディスクの場合においてもこの方法でマウントができます。

CD-Rなどのバックアップが終了し、コンピュータから取り出すときには、デスクトップ上にある図3のような記憶媒体のアイコン上で右クリックをし、図2のメニュー画面から“取り出し”を選択します。デスクトップ上から記憶媒体のアイコンが消えたらメディアの取り出しができる状態になります（これをアンマウントと呼びます）。

アンマウントせずにメディアを取り外した場合、そのメディアが破損する恐れがありますので、メディアを取り出す際には必ずアンマウントを行ってから取り外すようにしてください。

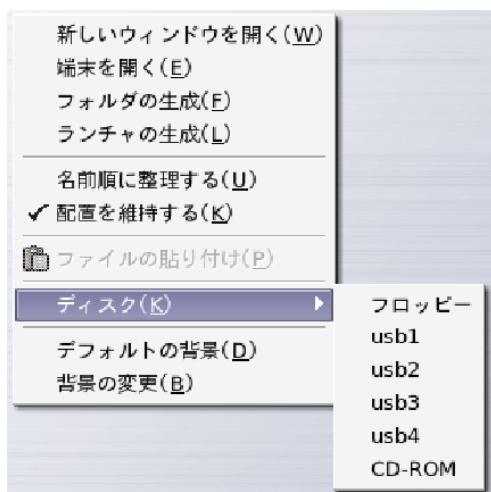


図1 マウント



図2 アンマウント



図3 アイコン

付録 C ソフトウェア一覧

RAINBOW の情報教室に導入されているソフトウェアを一覧にしてまとめておきます。但し、バージョンは各教室によって若干異なることがあります。最新のバージョンは、rpm コマンドで確認してください。

ソフトウェア一覧 (2010 年 2 月時点)

| ソフトウェア | バージョン | 説明 |
|-------------|---------------------|--------------------------|
| Anthy | 7500b | かな漢字変換ソフトウェア |
| Eclipse | 3.5.0 | 統合開発環境 |
| Firefox | 2.0.0.20 | WWW ブラウザ |
| FD | 2.08c | ファイル操作ユーティリティ |
| GNOME | 2.4.1 | GNOME デスクトップ環境 |
| GNU Emacs | 21.3 | テキストエディタ |
| GnuPG | 1.2.6 | PGP 互換ファイル暗号化ツール |
| ImageMagick | 6.0.8.3 | 画像表示/編集ソフトウェアパッケージ |
| JDK | 1.3.1 ,1.4.2 ,1.5.0 | Java 2 SDK (開発キット) |
| Mew | 1.94.2 | メールリーダ |
| Mupad | 2.0 | 数式処理ソフトウェア |
| OpenSSH | 3.9p1 | 暗号化リモートログインソフトウェア |
| OpenSSL | 0.9.7d | 暗号ユーティリティ |
| PVM | 3.4.3 | 並列処理ソフトウェア |
| Perl | 5.8.2 | Perl 言語インタプリタ |
| Python | 2.3.4 | Python 言語インタプリタ |
| Ruby | 1.8.1 | Ruby 言語インタプリタ |
| SKK | 11.6 | かな漢字変換ソフトウェア |
| Tgif | 4.1.43 | 図表作成ツール |
| XOrg | 6.8.2 | X Window System |
| a2ps | 4.13b | テキストファイル印刷ツール |
| attendance | 1.1 | 出欠確認システム (教員用端末のみインストール) |
| autoconf | 2.59 | 開発支援ツール |
| automake | 1.8.5 | 開発支援ツール |
| bc | 1.06 | 数値計算ソフトウェア |
| binutils | 2.14.90.0.7 | 開発ツールパッケージ |

ソフトウェア一覧 (2010 年 2 月時点) (続き)

| ソフトウェア | バージョン | 説明 |
|-------------|----------|-------------------------------|
| bison | 1.875 | 開発ツール |
| cups | 1.1.22 | プリンタユーティリティ |
| dvips | 5.94 | dvi から PostScript への変換ソフトウェア |
| flex | 2.5.4a | 開発ツール |
| g77 | 3.3.2 | FORTRAN 77 コンパイラ |
| gawk | 3.1.3 | AWK 言語インタプリタ |
| gcc | 3.3.2 | C コンパイラ |
| gdb | 5.2.1 | デバッガ (開発ツール) |
| ghostscript | 7.07 | PostScript インタプリタ (印刷ツール) |
| glibc | 2.3.3 | GNU C ライブラリ |
| gnuplot | 4.0.0 | グラフ作成ツール |
| gpc | 2.1 | Pascal コンパイラ |
| gprolog | 1.2.17 | Prolog 言語処理系 |
| gtk2 | 2.4.13 | X Windows System 用グラフィクスライブラリ |
| gttypist | 2.7 | タイピング練習ソフトウェア |
| gv | 3.6.2 | PostScript , PDF 閲覧ツール |
| gzip | 1.3.2 | ファイル圧縮ツール |
| less | 358 | テキストファイル閲覧ツール |
| lha | 1.14i | LZH ファイルアーカイバ (ファイル圧縮ツール) |
| lynx | 2.8.6 | テキストベース WWW ブラウザ |
| make | 3.80 | プログラム管理ツール |
| mtools | 3.9.9 | フロッピーディスク操作ツール |
| ncftp | 3.1.9 | FTP クライアント |
| netpbm | 10.18.12 | 画像ファイル操作ユーティリティ |
| nkf | 2.04 | 日本語文字コード変換ユーティリティ |
| openMotif | 2.2.3 | Motif 互換グラフィクスライブラリ |
| pLaTeX | 2e | 文書処理システム |
| psutils | 1.17 | PostScript ユーティリティ |
| sed | 4.1.2 | テキストファイル操作ユーティリティ |
| tar | 1.14 | ファイルアーカイバ |
| tcl | 8.4.6 | TCL 言語インタプリタ |
| tcsh | 6.14 | シェル |
| twm | 6.8.2 | X Window System ウィンドウマネージャ |
| unzip | 5.50 | ZIP ファイル展開ツール |

ソフトウェア一覧 (2010 年 2 月時点) (続き)

| ソフトウェア | バージョン | 説明 |
|------------|------------|---------------------------|
| vim | 6.3.82 | テキストエディタ (改良版 vi クローン) |
| wget | 1.9.1 | ファイルダウンロードツール |
| xdvik | 22.40y1_jp | DVI ファイル閲覧ツール |
| xfs | 6.8.2 | X フォントサーバ |
| xgraph | 12.1 | グラフ描画ツール |
| xloadimage | 4.1 | 画像表示ツール |
| xmgr | 4.1.2 | グラフ作成ツール |
| xpaint | 2.5.7 | 画像エディタ |
| xxgdb | 1.12 | デバッガ |
| zip | 2.3 | ZIP ファイルアーカイバ (ファイル圧縮ツール) |

付録 D その他 , 知つておくべきこと

RAINBOW のコンピュータを扱う上で , これまでの章で述べられたこと以外で , 知つておいて欲しいことをまとめます . 以下で述べる内容は , 2010 年 2 月現在での情報です . 変更されることもありますので , あらかじめそのことも念頭において読んでください .

付録 D.1 リモートで利用可能なホスト

研究室や自宅などから , RAINBOW のホストにリモートログインしたいことがあると思います . 以下のホストが利用可能なホストとして公開されています .

remote.ritsumei.ac.jp (全学部共通)

但し , これらは多くの人が利用するホストですので , 暴走プロセスが生じた場合は , 必ず kill し , 消えたのを確認してからログアウトしてください . 消さずに放置しておいた場合は , そのままホストが再起動されるまで暴走プロセスが残り続けて負荷となり , 他の人の迷惑になります . プロセスが消えない場合は , 最寄りの RAINBOW STAFF に相談してください .

付録 D.2 学部 , 回生別のホームディレクトリの容量

RAINBOW では , 各ユーザーに quota 制限 (ディスクの容量制限) が設けられていますが , メールなどを長い間溜め続けて放置しておくと , この容量を越えてしまうことがあります . 容量制限を越えると , RAINBOW を利用する上でさまざまな障害が発生します . こまめにホームディレクトリの容量を確認するようにしてください . ホームディレクトリの容量を確認する手段として , du コマンドが利用できます .

表 2 ディスクの制限容量 (2010 年 2 月時点)

| | |
|------------|-------|
| 学部学生 (全学部) | 50MB |
| 大学院生 | 100MB |
| 教員 | 200MB |