

教育用計算機システム使用の手引き

2019 年度版

筑波大学 情報学群 情報科学類

序

この手引きは情報科学類に導入された計算機システムの利用法を中心に、計算機の基本的な利用方法を説明するものです。

注意事項

健康のために

計算機のディスプレイを長時間見つめていると、目の疲労、肩こりなどの症状がでることがあります。長時間（目安としては1時間）連続してディスプレイを見つめながら作業を続けないように注意し、万一症状が現れた場合には、直ちに作業を中止して休憩をとるようにしましょう。

計算機と計算機室の運用

計算機ならびに計算機室の利用にあたっては、次ページ以降の「情報科学類（情報学類）教育用計算機システム利用規定」を遵守することが求められます。利用規定をよく読み、理解した上で利用してください。

質問や問い合わせ

教育用計算機システムは、情報科学類計算機運用委員会が方針を決め、運用・管理しています。また、有志の学生らによる学生管理者（coins-admin）も保守・運用に貢献しています。運用・管理上の方針などについて質問があるときは、計算機運用委員会の教員に問い合わせてください。また、細部に関する質問や指摘については、技術職員や学生管理者に直接問い合わせるか、ニュースやメールなどで問い合わせをしてください。連絡先は、本書10章に記載されています。システムの変更・停止などの情報は、次の計算機システムページに随時掲載されますので参照してください。
<http://www.coins.tsukuba.ac.jp/ce/>

本書の配布方法

本書は、電子ファイルのみで配布されています。PDF形式のファイルを以下のURLからダウンロードすることができます。<http://www.coins.tsukuba.ac.jp/tebiki/2019/tebiki2019.pdf>
PDF形式のファイルを用いることで、調べたい語句の検索などをより容易に行うことができます。有効に活用しましょう。

Dropbox の利用について

たくさんのファイルを保存している心当たりがないのにもかかわらずホームディレクトリの容量の上限を越えてしまう人が多くいます。近年のそうした現象の主な要因として、Dropbox というサービスの同期機能を有効にしているということが挙げられます。

Dropbox は、ファイルをクラウドに保存するサービスですが、Dropbox 公式クライアントはそのファイルをローカルのディスクに保存します。その結果として、多くのファイルが COINS のディスクに保存されてしまい、ディスク容量を圧迫してしまいます。Dropbox との同期をする際にはこれらのリスクに配慮してください。

情報科学類教育用計算機システム利用規定

平成 28 年 3 月
情報科学類計算機運用委員会

1.1 趣旨

情報科学類 (以降では単に「学類」と記す) が運用する教育用計算機システム (以降では単に「システム」と記す) は, 学類の学生, ならびに学類の授業を履修する学生が情報科学に関する知識・スキルを習得することを支援するためのものである。その目的に反しない範囲で, 利用者はシステムを自由に利用してよい。ただし, システムが安全かつ公平に利用されるように, すべての利用者が遵守すべき事項を本規定で定める。

1.2 利用資格と申請

1.2.1 利用資格と期間

次のいずれかに該当する者は, 学類計算機運用委員長の許可を得て, 定められた期間計算機室への入室とシステムの利用ができる。

1. 学類に所属する学生 (在学期間中)
2. 学類担当教職員, 学類の授業を補助する大学院生 (利用可能期間は個別に定める)
3. 情報科学類 (情報学類) を卒業して筑波大学大学院に進学した者 (大学院在学期間中 ただし, 毎年更新手続きを行う事)
4. システムの利用を希望し, 学類計算機運用委員長から承認された者 (利用可能期間は, 利用目的に応じて個別に定める)

1.2.2 利用の申請

システムを利用しようとするものは, 本規定の内容を理解した上で, 所定の申請書を計算機運用委員長に提出し, 許可を得なければならない。

1.2.3 利用の停止

学類計算機運用委員長は, システムの利用者が本規定に反する行為を行ったと確認した場合, 当該利用者のアカウント (もしくは一部の機能) を一定期間停止することができる。

1.3 施設, 設備の利用

計算機室内では、飲食、喫煙は厳禁である（発覚した場合、全計算機室が閉鎖される場合がある）。紛らわしい行為（机の上にペットボトルを置くなど）も避けるべきである。また、濡れた傘の持込や、泥の付いた靴で入室してはならない。計算機を破壊しかねない行為（衝撃、高温、埃、水分を与えるなど）もしてはならない。授業で計算機室を利用しているときでも、空いている端末は履修者以外が利用してもよいが、その場合は授業担当教員の許可を得る。

1.4 アカウント, パスワード, IC カードの管理

利用者は、アカウントを他者に利用させたり、他者のアカウントを利用してはならない。パスワードは容易に類推可能でないものとし、厳重に管理しなければならない（ポストイットやノートなどにパスワードのメモを取っておくなどしてはならない）。計算機室の入室は、利用者が自身の IC カード（学生は学生証、教員は職員証）を用いなければならない（ロックしてあけてもらう、ロックされたので開けてあげる、IC カードを他人と貸借する、などしてはならない）。

1.5 システムの利用

システムを利用するにあたっては、次のことを守ること。

1. システムの AUP(Acceptable Use Policy) を守る。システムの AUP とは、1.1 で述べた利用目的の範囲内でシステムを利用することである。
2. 大規模な計算を行いたい場合は、別に定める「大規模計算に関する規定」に従って利用する。
3. システムのセキュリティを損なわないように務め、システムのセキュリティ上の問題を見つけたときには速やかに管理者（技術職員、学生管理者など）に連絡する。特別な理由によって学類計算機運用委員長の許可を得た場合を除き、悪意のあるプログラム（ウイルス、ワーム、トロイの木馬、攻撃プログラム等）の作成・持ち込み（メール、USB メモリ等）やそれらの意図的な実行、他人のプライバシーや機密を暴くことやそのためのパスワード・暗号化されたファイルの解読、などは特に厳禁である。また、自宅のパソコンを大学のネットワークに接続する場合、そのパソコンの安全確保を確実に行う。
4. ネットワークを通じてアクセス可能な他のシステムを不正に使用したり、運用を妨害するなどの行為はしない。計算機のネットワークケーブルを抜く、自分のパソコンを情報科学類教育用計算機システムのネットワークに有線で接続するなどの行為も行ってはならない。
5. システムの動作に問題があることに気付いた場合は、速やかに管理者（技術職員、学生管理者など）に連絡する。
6. 計算機の利用を終えた後には、必ずログアウトをする。計算機によっては、電源をオフもしくはスリープ状態にする（管理者などからの指示に従う）。スクリーン・ロックを行う時には、次の利用者の利用を妨げてはならない。前の利用者がログアウトしていない場合、代わりにログアウトして利用しても良い。この時、ファイル等を保存しなくても良い。
7. プリンタに出力するにあたっては、プリンタのスイッチ類は触らない。資源保護の観点から、無駄な印刷

は可能な限り避けなければならない。ただし、プリンタが故障しないようにするために、プリンタに入れるのは未使用の用紙のみとする。

1.6 情報発信

システムを利用したインターネットへの情報発信に際しては、次のことを守らなければならない。

1. インターネットに公開する情報には、原則として発信者の氏名を明記し、発信者が責任を負う。
2. 法律や学内規則を遵守し、以下に掲げる情報は発信しない。
 - 他人の名誉を傷つけることを目的としたもの
 - わいせつなもの
 - 著作権法に違反したもの
 - 他人のプライバシー・肖像権を侵害したもの
 - 営利を目的としたもの
 - 特定の政党または宗教団体に係わる活動を目的としたもの
 - その他法律、学内規則、公序良俗に反するもの
3. 自らインターネットへの情報を発信するツールを作成・導入・利用するときはシステム管理者の助言を受ける。なお、本規定でいう「情報発信」とは、次のことをいう。
 - Web ページを開設する
 - ネットワーク・ニュースに投稿する
 - メーリングリストにメールを送る
 - 遠隔会議システムにメッセージを送る
 - 掲示板等へ書き込む
 - その他、上記に類似の行為を行う

参考

システムの利用に当たっては、「筑波大学における情報システム利用のガイドライン」も適宜参照して下さい。なお、次により閲覧可能です。<http://www.u.tsukuba.ac.jp/document/ja/UnivGuideline.pdf>

計算機室の利用について

情報科学類生が使える計算機室は、3C113, 3C205, 3C206 の3部屋があります。各部屋の入り口は、オートロック式になっています。入室の際には、ロックを解除するために入り口のカードリーダーに学生証をかざす必要があります。非接触式の Felica 規格の IC カードなので、財布などに入れたままかざすことができますが、Suica, Pasmo などの他の Felica 規格のカードと一緒に入っていると、干渉して認識されないことがあります。その際は、学生証だけ取り出してかざしてください。学生証を計算機室内に置いたまま外に出ると計算機室に入れなくなるので、十分に注意しましょう。

各部屋の設備は以下の通りです。

表 1: 各部屋の設備

場所	3C113	3C205	3C206
OS	macOS	macOS	Windows/Linux
マシン名	aloe1 ~ aloe96	crocus1 ~ crocus40	dahlia1 ~ dahlia30
プリンタ名	aloe-pr1, aloe-pr2 aloe-pr3, aloe-pr4 color-c113(カラー)	crocus-pr1 crocus-pr2 color-c205(カラー)	dahlia-pr1

また、サーバ・コンピュータや無線 LAN 等の設備は表 2 の通りです。

表 2: サーバ・コンピュータ、無線 LAN 等の設備

サーバ・コンピュータ	VPN サーバ (viola13.coins.tsukuba.ac.jp) WWW サーバ (www.coins.tsukuba.ac.jp) メール受信 (IMAP, POP) サーバ (viola6.coins.tsukuba.ac.jp) メール送信 (SMTP) サーバ (viola6.coins.tsukuba.ac.jp)
無線 LAN SSID 名	AP-C113BGN, AP-C113AN, AP-C206BGN など

iMac の使い方については、1 章を、Windows および Linux の使い方については 5 章を、印刷方法については 1.8 節を参照してください。

情報科学類教育用計算機を利用した大規模計算に関する規定

情報科学類の教育用計算機システムを利用した大規模計算に関する利用規定を以下のように定める。

1. 本学類が保有する教育用計算機について、大規模計算を行ってもよい時間帯を表3のように定める。
この時間帯以外には計算機運用委員会の特別な許可がない限り、教育用計算機を使った大規模計算を行なうことはできない。なお、ここでいう大規模計算とは、使用 CPU 時間 1 時間以上、または、使用仮想記憶 1GBytes 以上のプロセスを、非対話的に実行することをいう。

表 3: 学類計算機での大規模計算可能期間

計算機	大規模計算可能期間
www サーバ viola グループの各サーバ その他管理用コンピュータ	常時大規模計算禁止
aloe, crocus グループ dahlia グループ	休業中（春季，夏季，秋季，冬季） 土曜日，日曜日，祝日および大学が定める休講日 平日の 18:00～8:00 の間

2. ユーザ間の便宜を図るために、大規模計算を行なう場合は、大規模計算作業のお知らせページに大規模計算の計算機資源の利用内容（利用計算機室，利用台数，開始時刻，終了時刻等）を投稿することが望ましい^{*1}。
3. (1) で定める時間外に、何らかの理由で大規模計算を行なうことを希望する場合は、情報科学類計算機運用委員会に申請して許可を求めること。申請は、以下の内容を含む情報を coins-admin@coins.tsukuba.ac.jp 宛てに電子メールで送付すること。
 - 担当者に関する情報（氏名，学籍番号，指導教員，電子メールアドレス，電話番号，所属研究室）
 - 申請理由
 - 計算の内容
 - 希望開始時刻，予想終了時刻，予想 CPU 使用時間，予想仮想記憶使用量

以上の規定を守らない利用に関しては、当該プロセスを計算機運用委員，または、計算機運用委員の委託を受けた者が、強制的に終了させることができる。

^{*1} 投稿には Shibboleth で認証が必要です。

本件に関する質問は情報科学類計算機運用委員会（c-comp@cs.tsukuba.ac.jp）まで.

最終更新日時：2019 年 3 月 30 日

推奨事項

iMac または Linux コンピュータに高い負荷を掛ける事が明らかな作業を行う場合には, `nice` コマンドを使って作業の優先度を下げてください. `nice` コマンドの `n` オプションは 19 を指定してください.

```
$ nice -n 19 コマンド↵
```

`n` オプションは-20(最優先) から 19 まで指定でき, デフォルトは 10 です.

全学計算機システムのメール利用について

本書で説明する「情報科学類教育用計算機システム」は、主に情報科学類生を対象とした計算機システムですが、これとは別に、筑波大学では学群・学類問わず全ての学生・教員が利用できる「全学計算機システム」が存在します。ここでは、この全学計算機システムの Web メールについて説明します。なお、全学計算機システムに関する詳しい説明は、説明ページ (<https://www.u.tsukuba.ac.jp/>) を参照してください。

2017 年度以降に入学した人

Office 365 (<https://cloudmail.u.tsukuba.ac.jp/>) にアクセスして@s.tsukuba.ac.jp ドメインの全てのメールを確認することができます。

2016 年度以前に入学した人

Office 365 の@s.tsukuba.ac.jp ドメインとは別に Active!mail (<https://wmail.u.tsukuba.ac.jp/>) の@u.tsukuba.ac.jp ドメインのメールアカウントを持っています。

利用に関して入学年度別の注意事項等は「全学計算機システムのメールについて」(<https://www.u.tsukuba.ac.jp/email/>) からご確認ください。

学内外を問わず、インターネットに接続されたパソコンであれば、全学計算機システム Web メール (Office 365) を利用して配信メールを確認できます。Office 365 にログインするためには、全学計算機システムと同じユーザ ID とパスワードを使用してください。例えば学籍番号が 209912345 の場合、ログイン ID は「s9912345」となります。

目次

情報科学類教育用計算機システム利用規定	I
1.1 趣旨	I
1.2 利用資格と申請	I
1.3 施設, 設備の利用	II
1.4 アカウント, パスワード, IC カードの管理	II
1.5 システムの利用	II
1.6 情報発信	III
第 1 章 iMac の基本的な使い方	1
1.1 iMac の構成	1
1.2 ログインおよびログアウト	3
1.3 パスワードと統一認証システム	6
1.4 ログイン後の画面	10
1.5 言語環境	13
1.6 ウィンドウシステム	18
1.7 アプリケーションの紹介	21
1.8 印刷	23
第 2 章 コマンド	27
2.1 iTerm	27
2.2 イントロダクション	29
2.3 ディレクトリ構造	30
2.4 ファイルをコマンドラインで操作する	32
2.5 用語の整理	35
2.6 ファイルに関するコマンド	36
2.7 ファイルの中身に関するコマンド	41
2.8 システムに関するコマンド	45
2.9 画像に関するコマンド	46
2.10 プロセスを取り扱うコマンド	46
2.11 コマンドを知るためのコマンド	47
2.12 ワイルドカード	48
2.13 <code>tab</code> キーによる補完	49

2.14	リモートログイン (remote login)	49
2.15	コマンドをさらに使いこなす	50
2.16	その他有用なコマンド	51
2.17	コマンドリスト	53
第 3 章	Emacs	56
3.1	Emacs 基礎知識	56
3.2	実行と終了	59
3.3	編集	60
3.4	Mew	67
3.5	Tips	74
3.6	自宅で Emacs を使うには	77
第 4 章	ブラウザとメーラの使い方	79
4.1	Firefox による Web ブラウジング	79
4.2	Thunderbird による電子メールの利用方法	87
4.3	RainLoop	99
第 5 章	Windows と Linux の使い方	101
5.1	サインインとサインアウト	101
5.2	ウィンドウの基本操作	105
5.3	ファイルの操作	108
5.4	プロセス管理	112
5.5	ホームディレクトリの利用	116
5.6	Windows からの印刷	116
5.7	Windows での日本語入力	117
5.8	Windows コマンドの利用	118
5.9	メール設定	122
5.10	Linux の利用	122
第 6 章	L^AT_EX	123
6.1	L ^A T _E X の概要	123
6.2	L ^A T _E X を使う上での注意	126
6.3	L ^A T _E X による文書生成の流れ	126
6.4	L ^A T _E X に関連するファイルの形式	126
6.5	Hello L ^A T _E X!	127
6.6	L ^A T _E X のコマンドと環境	129
6.7	クラスファイルの指定	131
6.8	本文	132
6.9	表題と著者	132
6.10	見出し	133

6.11	書体・文字サイズ	137
6.12	改行・改ページ	138
6.13	箇条書き	139
6.14	図	140
6.15	表	144
6.16	脚注	146
6.17	参照	147
6.18	数式	148
6.19	ソースコード	153
6.20	書いた通りに出力する	156
6.21	BIB $\text{\TeX}$ を用いた参考文献	157
6.22	文書の余白	161
6.23	目次の生成	162
6.24	発展的な $\text{\TeX}$ の話題	162
第 7 章	言語処理系	168
7.1	言語処理系とは	168
7.2	Java コンパイラ	168
7.3	C	171
7.4	C++ コンパイラ	179
7.5	FORTRAN コンパイラ	180
7.6	MATLAB	181
第 8 章	VMware Fusion	186
8.1	仮想計算機の概要	186
8.2	仮想計算機に OS をインストールする手順	187
8.3	ゲスト OS を利用する方法	194
8.4	VMware Fusion のより高度な機能	196
8.5	困ったときは	197
第 9 章	ネットワーク	199
9.1	学類計算機資源の利用	199
9.2	無線 LAN サービス	204
9.3	VPN サービスの利用	207
第 10 章	困ったとき	214
10.1	バックスラッシュを入力する	214
10.2	プロセスの強制終了	214
10.3	macOS 環境が固まらないようにするために	216
10.4	プリントアウトができない場合	217
10.5	Web ページ公開について	217

10.6	COINS のシステムがダウンしている場合	217
10.7	Firefox・Thunderbird が起動できなくなった場合	217
10.8	COINS 宛のメールを他のメールアドレスに転送したい場合	218
10.9	ディスククォータに関する警告メールが来てしまった	218
10.10	ブラウザのキャッシュを削除する	220
10.11	quota の上限を大きくしたい	223
10.12	間違えてファイルを消してしまった	223
10.13	各種問い合わせ先	224
索引		225

第 1 章

iMac の基本的な使い方

本章では，情報科学類（COINS^{*1}）の計算機である iMac（macOS Mojave）を利用する上で覚えておく必要がある操作方法について，簡単に説明します．

1.1 iMac の構成

iMac（図 1.1）は本体とマウス（図 1.2）とキーボード（図 1.3）から構成されています．



図 1.1: iMac の本体

図 1.2 はマウスです．ボタンが無いように見えますが，それに相当する場所（ホイールの左右）をクリックすると反応します．初期状態では，マウスの左側をクリックしても右側をクリックしても，同じクリックとして扱われます．macOS で右クリックをする場合には，`control` を押しながらクリックをします．また 1.4.5 節ではマウスの右クリックを有効にする方法を紹介しているので，使いづらい方は設定を変更しましょう．

ホイールは真ん中にある小さな丸い部分で，画面をスクロールするためのものです．スクロール可能な画面でここを上下に動かすことで，画面が上下にスクロールします．

^{*1} College Of INformation Science



図 1.2: マウス



図 1.3: キーボード

図 1.3 はキーボードです。レポートを作成するときやコマンドを打つときなど、キーボードを通して操作することが多いです。普段からキーボードに慣れておくと、打つ速度（タイピング速度）が速くなり効率も良くなります。是非覚えてみてください。

また、よく使うキーの場所を図 1.4 に、その説明を表 1.1 に記します。今後の説明で使っていくのでよく覚えておくようにしましょう。



図 1.4: キーボードの配置

表 1.1: キーボード

表記	キーの位置	名称
	1	エスケープ
	2	スペース
	3	タブ
	4	リターン
	5	デリート
,	6	オプション
	7	コマンド
	8	コントロール
	9	シフトキー

1.2 ログインおよびログアウト

1.2.1 ログイン

iMac を利用するためには、まずログイン (*login*) する必要があります。ここで利用する計算機は、多くの人々の間で共有するものです。一台の計算機を同時に複数の利用者で使うこともあります。そのため、計算機に対して、自分が誰であるのかを知らせる必要があります。これによって、誰が計算機を利用しているか確認できます。また、ファイルなどに対しては所有権を設定することでプライバシーなどの保護をすることができます。ログインとは、自分が計算機を利用する旨を計算機に告知し、計算機を利用する一連の動作をいいます。

図 1.5 は、最初に現れる画面です。ディスプレイの保護と省エネルギーのため、画面が消灯している場合があります。このような場合は、マウスを動かしたり、 を押してみてください。図 1.5 のような画面になるはずです。困ったときは、教員、TA*²または技術職員*³に連絡してください。

図 1.5 は、ブートメニューといい、使用するオペレーティングシステムを選択する画面です。アイコンは、左が macOS Mojave、中央が Windows 10、右が Ubuntu を表しています。クリックすると該当するオペレー

*² Teaching Assistant=授業のサポートをしてくれる先輩

*³ 対応する技術職員は 3E108 にいます。

ティングシステムが起動します。この章では macOS について説明しますので、まず左側をクリックしてみましょう。

次に図 1.6 の名前と書かれているところに、自分のログイン名を入力します。ログイン名は計算機の中での自分の名前にあたるもので、ここでは「s+ 学籍番号の下 7 桁」になります。例えば、学籍番号が 209912345 であれば、ログイン名は「s9912345」になります。自分のログイン名を入力できたら、リターンキー（`return` と書かれたキー）を押します。なお、以後、文字を入力する、もしくは、文字を入れるといった場合には、文字列をタイプし `return` を押すまでの動作を言うことにします。

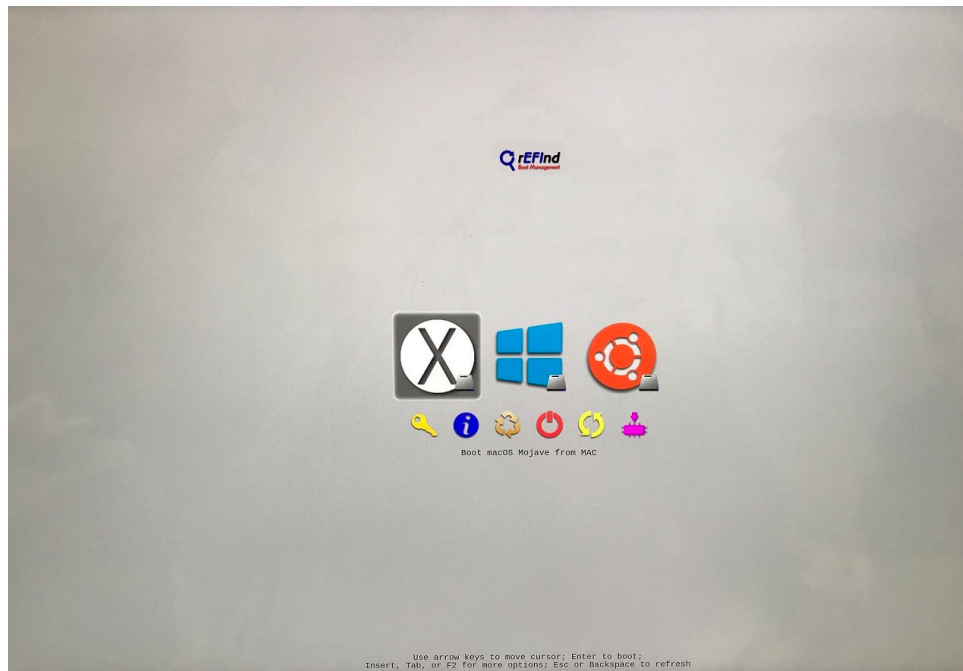


図 1.5: ブートメニュー

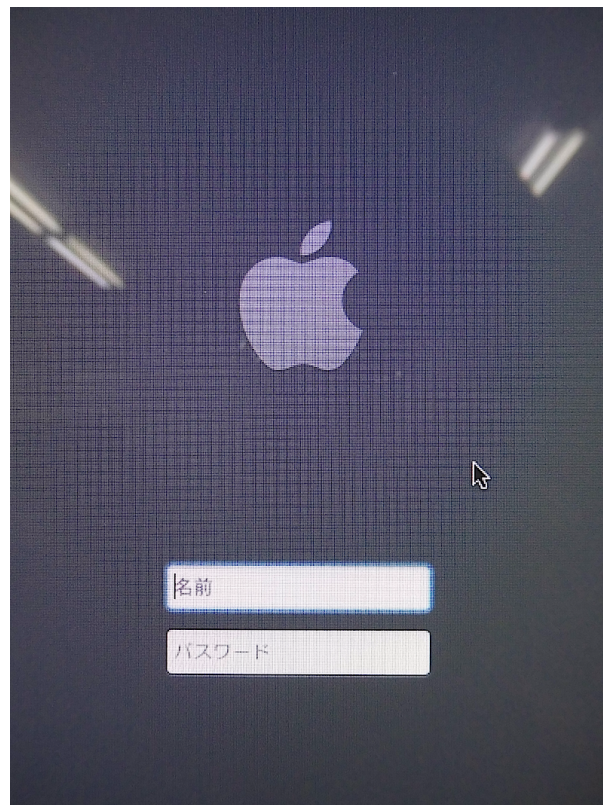


図 1.6: ログイン・認証画面

ログイン名を入力して、**[tab]**を押すとカーソル*⁴は、パスワードの横に移動します。今度は、自分は正当な利用者であることを証明するためにパスワードを入力します。計算機は、相手が誰なのかを目で見たり声を聞いたりする事ができませんから、パスワードを入力してもらうことによって、それを確認します。パスワードについては、1.3 節で述べます。

パスワードを入力する時には、入力した文字は一切画面に表示されません。これは、他人に盗み見をされないようにするためです。ログイン名とパスワードの双方を間違っていなければ、これでログインできます。できない場合は、ログイン名、あるいはパスワードのどちらかが間違っています。ログイン名を今一度確認し、さらに別のパスワードを使っていたり、大文字や小文字等が間違っていたりしないかチェックしてください。Caps Lock がオンになっていると全て大文字で入力されますので、とくにご注意ください。ログイン名あるいはパスワードの入力を何度やっても駄目な場合は、実際のパスワードと入力しているパスワードが異なっている場合がありますので、技術職員に相談してください。

1.2.2 ログアウト

計算機は、皆で共有するものです。使い終わったら、きちんとログアウト (**logout**) しましょう。ログアウトとは、ユーザが計算機の使用を終了する操作です。

*⁴ カーソル (Cursor) 次に入れるべき文字の位置を指示するしるし。一般的には I 字型で示される。

ログアウトによって終了するプロセス^{*5}としないプロセスがあります。エンドユーザが用いるほとんどのプログラムは前者ですが、後者のようなデーモンプロセスと呼ばれる種類のものもあります。ログアウト操作によって、ほとんどのデスクトップアプリケーションは終了しますが、計算機本体の電源は切れません。

ログアウトするには、デスクトップ画面左上のアップルメニューをクリックし、そこから「ログアウト」をクリックします。ショートカットキーは **shift** + **⌘** + **Q** です。この操作を行うと、図 1.7 のような、確認のダイアログが出てきます。ログアウトするのであれば「ログアウト」ボタンを、違う場合は「キャンセル」ボタンをクリックします。

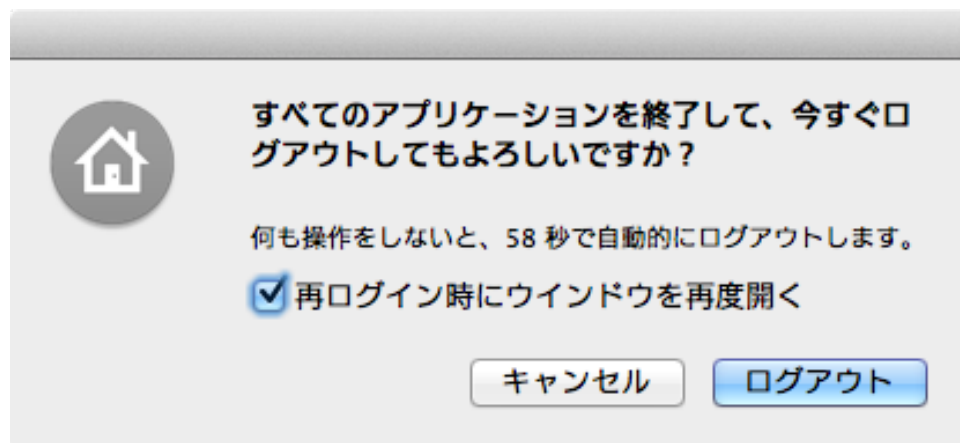


図 1.7: ログアウトダイアログ

ログアウトをすると、図 1.5 の画面に戻ってきます。この画面に戻ったことを確認してから席を離れてください。また、計算機の電源は切らないでください。急に電源を切ると、リモートログインをしている人が作業できなくなったり、作業中のファイルが破損したりします。特に、電源ケーブルを抜いての強制シャットダウンは故障の原因になりますので、絶対に行わないでください。

1.3 パスワードと統一認証システム

1.3.1 統一認証システム

筑波大学内では「筑波大学統一認証システム（以下、統一認証システム）^{*6}」が採用されています。統一認証システム内のシステムであれば、同一のログイン名とパスワードが使用できます。例えば、履修登録などに用いる TWINS というシステムへログインする際は、この統一認証システムのパスワードを用います。COINS も、この統一認証システム内のシステムです。つまり、学類の計算機（COINS）と、統一認証システム内のシステムには、同一のパスワードでログインできます。図 1.8 にこの関係図を示します。

^{*5} コンピュータが実行している作業やプログラムのこと。

^{*6} 統一認証システムに含まれるシステムは <https://account.tsukuba.ac.jp/list.html> で確認できます。

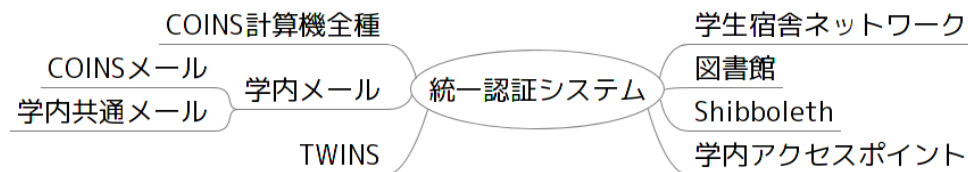


図 1.8: 統一認証システムの関連

また、各システムのログイン名は表 1.2 に示す 3 種類が存在します。例えば学籍番号が 209912345 の場合、「s+ 学籍番号下 7 桁」は「s9912345」となります。「学生証に記載された 13 桁の番号」とは、学生証裏側のバーコード下部に記載された番号の事です。

表 1.2: ログイン名

ログイン名	システム
s+ 学籍番号下 7 桁	COINS のログイン, 全学計算機システム (Office-365 を含む) など
学生証に記載された 13 桁の番号	学生宿舎ネットワーク, 電子図書館システム, 学内アクセスポイント (無線 LAN, utwlan) など
学籍番号	TWINS など

1.3.2 パスワードに関する注意

統一認証システムのパスワードが他人に漏れると、統一認証システムを利用している様々なシステムを不正に利用される恐れがあります。パスワードが漏洩して問題が起きた場合には、パスワードを盗まれた本人が責任を負うことになっています。盗まれにくいパスワードにするため、パスワードは、一般的に次の事項を満たしていることが求められています。

- パスワードの長さが 8 文字以上。
- 大文字, 小文字, さらに記号や数字を混ぜる。

以下に弱いパスワードの例を挙げておきます。

- 英語, 日本語, その他外国語の辞書にある単語。
- 固有名詞 (人名, 地名など)。
- 単純な文字列 (aaaaaaa, 12345678 等)。
- 7 文字以下の短い文字列。
- キーボードの配列を使った文字列 (qwert, asdfg 等)。
- 上記の単語の綴を逆順にした文字列。
- 上記の単語に記号や数字を付けた文字列。

パスワードを忘れないようにするために、自分の好きな文句の頭文字の途中に数字や記号を挟み込んだものを

パスワードとして使っている人もいます*7。COINS のログインに使用する統一認証のパスワードは、6文字以上8文字以下で設定する必要があります。これを満たさない場合はパスワードの変更は受け付けられず、基準を満たす別のパスワードを使用するように指導されます。パスワードの変更は自由にできるので、キーボード操作になれた時期に必ず変更してください。また、大事なことですが、自分のパスワードをメモしないようにしてください。パスワードのメモを盗まれたり見られたりすると、そのパスワードが有効なアカウントが脅威にさらされます。銀行などのキャッシュカードの暗証番号と同じです。キャッシュカードの暗証番号との違いは、カードがなくても、計算機に触れることができれば、他人になりますことができるという点です。自分のパスワードは、きちんと管理するようにしてください。また、ほとんどの現代的な Web ブラウザはパスワードを記憶することができます。複雑で膨大な数のパスワードをブラウザで一元管理できるたいへん便利な機能ですが、使用する際は必ずマスターパスワードを設定し、パスワードデータベースの安全を確保しましょう。

それぞれのパスワードの変更はそれぞれの Web ページ上で行います。統一認証システムと COINS のパスワード変更は筑波大学統一認証システムのホームページ (<https://account.tsukuba.ac.jp/> 図 1.10) 内の「パスワード変更」をクリックした画面 (図 1.11) で、行う事ができます*8。繰り返すにはなりますが、何かあってからでは遅いのでパスワードは厳重に管理しましょう。

パスワードを変更した後 iMac にログインした際のエラー

macOS ではキーチェーンというパスワード管理システムが搭載されています。このキーチェーンではアカウントのログインパスワードがデフォルトキーチェーン (ログインキーチェーン) のパスワードという設定になっているため、統一認証でログインパスワードが変更されても、キーチェーンのパスワードが変更されてはいないので、エラーが出るようになってしまいます。

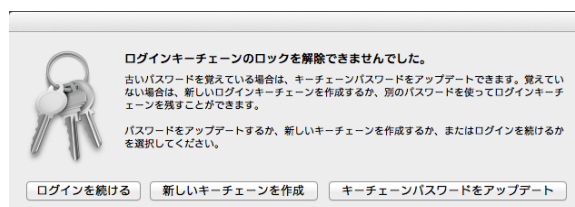


図 1.9: ログイン時のエラー

図 1.9 の表示が出た際には、[キーチェーンパスワードをアップデート] を選択した後、旧パスワードを入力してパスワードの変更を行ってください。

*7 場合によっては、大文字小文字の変更も行う。

*8 UNIX では通常 `passwd` コマンドを使ってパスワードを変更できるが、COINS では統一認証システムとの兼ね合いから使用できません。



図 1.10: 統一認証ホームページ

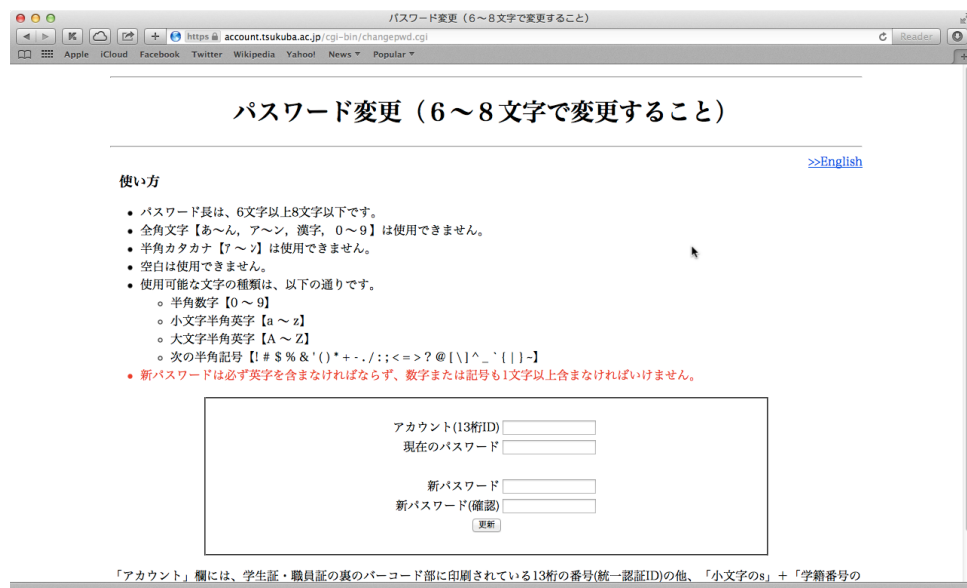


図 1.11: パスワードの変更画面

1.3.3 Shibboleth

Shibboleth とは、認証に必要な Web ページにアクセスする時に認証を行うシステムのひとつです。COINS の一部の Web ページでは、この Shibboleth による認証を行っています。例えば、情報科学類コンピューティング環境（内部情報）の Web ページ (<https://www.coins.tsukuba.ac.jp/ce/>) へアクセスすると、Shibboleth による認証ページ (<https://idp.account.tsukuba.ac.jp/idp/Authn/UserPassword>) へ転

送されます。ここで、ユーザ名とパスワードを入力してから「login」ボタンを押すと、認証が正しければ、元の情報科学類コンピューティング環境（内部情報）の Web ページへ戻ってきます。この際、ユーザ名とパスワードは統一認証システムと同じものを入力してください。

一度 Shibboleth での認証を済ませば、別の Shibboleth による認証を行っている Web ページにアクセスしても、その都度認証を行う必要は無く、認証を省略することができます。これを「シングルサインオン」と言います。

Shibboleth に関する詳しい説明は、説明ページ (<https://idp-test.account.tsukuba.ac.jp/Shibboleth/>) を参照してください。

1.4 ログイン後の画面

図 1.12 は macOS にログインした後に表示される画面を表しています。まずは各部分の名称を覚えましょう。名称を憶えておくことで、トラブルを相談する際に状況を正しく伝えることができます。例えば、「画面の下ボタン」でなく「ウィンドウの下にあるメニュー」と言うことで、どこを指しているのかが具体的に伝わり、相手側が状況を再現する時に役立ちます。



図 1.12: ログイン後の画面

1.4.1 デスクトップ

デスクトップとは、机の上の「作業場所」のようなものです。図 1.12 に示すように、ファイルなどがアイコンと呼ばれる画像で表示されます。より詳しくは、2.5.1 項において説明するホームディレクトリ内の Desktop ディレクトリにおかれたものがデスクトップに現れますが、まずはこのまま読み進めてください。

1.4.2 Dock（ドック）

Dock とは、画面の下部に表示されている、アプリケーションなどのアイコンが並んだ領域の名称です。アイコンは、境界線を境にして、登録済みや実行中のアプリケーションが表示されている部分と、ゴミ箱などの項目が表示されている部分に分かれています。実行しているアプリケーションは図 1.13 のとおりアイコンの下に黒い点が表示されています。1.4.3 項で説明する Finder は常に実行中になっていますが、それ以外の実行しているアプリケーションは右クリックから「終了」を選択すると終了することができます。



図 1.13: Dock の構成

Dock 内に入れたいアイコンは、図 1.14 のように Dock 上にアイコンをドラッグする、もしくは実行中にアイコンを右クリックし、「オプション」内の「Dock に追加」を選択することで Dock に入れることができます。逆に Dock 内から削除したいアイコンは、図 1.15 のように Dock 外にアイコンをドラッグすることで削除できます。



図 1.14: Dock にアイコンを追加する



図 1.15: Dock からアイコンを削除する

1.4.3 Finder（ファインダー）

Finder（ファインダー）とは、ファイルやディレクトリを探したり整理したりすることができるアプリケーションです。図 1.16 は Finder が開かれた状態です。（Dock の  をクリックしても Finder を開くことができます）。

境界線の右側にディレクトリ、ファイル、アプリケーションなどがアイコン形式で並んでいます。これらのアイコンをクリックすることにより、ディレクトリの移動やアプリケーションの利用などができます。左側にはホームディレクトリやデスクトップなどへのショートカットや、過去に利用したファイルの履歴などの機能がついています。

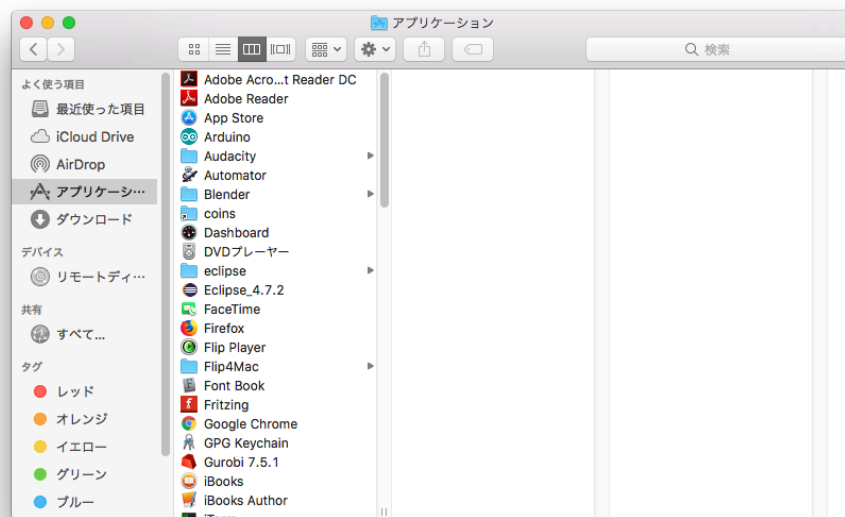


図 1.16: Finder

1.4.4 ゴミ箱

ゴミ箱  は不要になったファイルやディレクトリを一時的に保管しておく特殊な場所（ディレクトリ）です。Dock の右端に表示されているゴミ箱アイコンを右クリックして「ゴミ箱を空にする」を選択すると、ゴミ箱に入っていたファイルやディレクトリはシステムから削除され、それらが使用していたストレージ領域が解放されます。いわば、ゴミを捨てる操作です。

また、ゴミ箱には、メモリーカードなどのリムーバブルディスク^{*9}を取り外す（アンマウントする）ための機能も用意されています。ファイルやディレクトリを捨てる時と同じ要領でリムーバブルディスクのアイコンをゴミ箱へドラッグすると、ゴミ箱のアイコンが図 1.17 のとおり取り出し操作を示すアイコンに変わるので、

^{*9} 現行の iMac には光学ドライブはありません。

そこで手を離すとアンマウントが実行されます。



図 1.17: ゴミ箱によるストレージのアンマウント・取り出し

1.4.5 システム環境設定

「システム環境設定」はマウスやキーボードの設定、文字入力の設定、画面や音量の設定など、macOS に関する様々な設定を行うことができるアプリケーションです。アップルメニューから「システム環境設定」を選ぶ事で、システム環境設定を開く事ができます。

—— マウスで右クリックを使用するための設定 ——

右クリックを有効にするには、まず「システム環境設定」を開きます。次にこの中のマウスをクリックするとマウスの使用例の映像が出てくるので、右クリックに位置する部分を「副ボタン」に変更します。

1.5 言語環境

1.5.1 漢字の変換

図 1.12 のステータスアイコン内にある「あ」や「A」と書かれているボタンをクリックすることで言語を変更することができます。また、キーボードにある「英数」や「かな」ボタンを押すことでも同様の操作は可能です。「英数」を押せば英字入力に、「かな」を押せばかな入力になります。かな入力ではローマ字を打ち込むとかなに変換してくれます。ではここで、かな入力を具体的にやってみましょう。ステータスアイコンが「あ」になっている状態で、jouhoukagakurui とタイプします。

じょうほうかがく^るい

これを （他のキーより少し大きく何も書いてないキー）で漢字に変換します。

情報 科 が 狂 い

と変換されるかもしれませんが、変換結果が正しくないので修正しましょう。文節ごとにアンダーバーが区切れていますので、対象の文節ごとに左右キー（, )で移動することができます。「情報」の部分正しいので、を押して次の文節に移ります。

情報 科 が 狂 い

「かがく^るい」の文節がおかしいので、文節を直します。文節を縮める場合は^{shift}を押しながら, 反対に伸ばす場合は^{shift}を押しながらを押します。「かがく」と「るい」にしましょう。

情報 化学 類

「化学」の変換がちがうので、別の変換になおします。「化学」の部分で  を押すと、変換候補が表示されるので、「科学」を選択します。

情報科学 類

になりましたね。この状態で **return** を押して確定します。このように、目的の漢字が表示されない場合は、文節を短くするか、単語や単漢字ごとに変換します。

1.5.2 ショートカット

macOS に標準でインストールされているかな漢字変換ソフトウェアには、ショートカット（表 1.3）が用意されています。

表 1.3: かな漢字変換ソフトのキーボード操作

 + 	ひらがな「あ」と英字「A」の切り替え
	変換の開始。変換中は下の変換候補へ
shift + 	変換の開始。変換中は上の変換候補へ
control + H	ひらがなに変換
control + K	カタカナに変換
control + L	全角英数に変換
control + :	半角英数に変換
	上の変換候補へ
	下の変換候補へ
	前の文節へ
	次の文節へ
shift + 	文節を縮める
shift + 	文節をのばす

1.5.3 日本語入力的高速化

入力補助の機能である推測候補表示は便利な一方でその処理は重く、快適な入力の妨げになってしまいます。必要がなければ無効化してしまいましょう。設定方法は図 1.18 と図 1.19 を参照してください。



図 1.18: 言語環境の設定

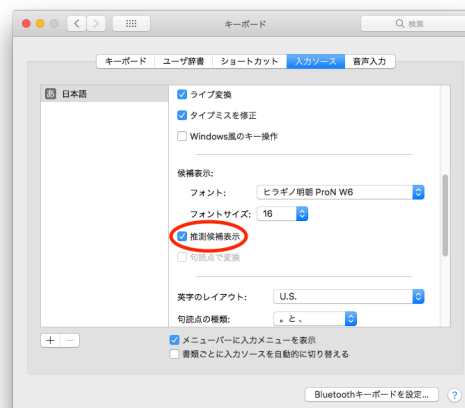


図 1.19: 推測候補表示の設定

1.5.4 Google 日本語入力による日本語と英語の入力

Google 日本語入力は優秀な変換機能を持つメジャーな IME*¹⁰です。計算機室の iMac には、標準ソフト以外の IME として Google 日本語入力インストールされています。デフォルトで設定されている IME を無効にして、Google 日本語入力を有効にするためには、以下の図 1.20 から図 1.23 に示すように、日本語入力と英語入力を設定してください。デフォルトの IME を用いた英語入力で不都合や問題が出た場合は、この設定を試してください。

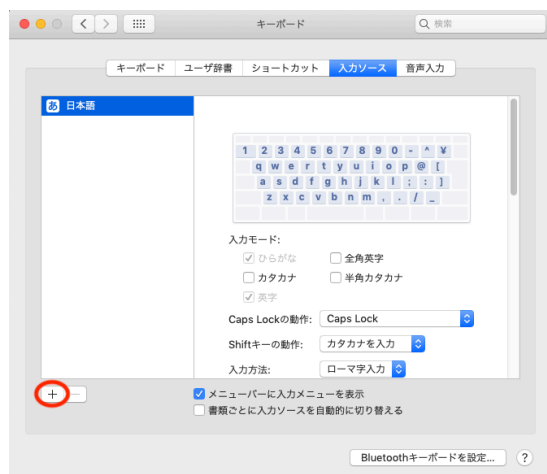


図 1.20: 入力ソースの追加

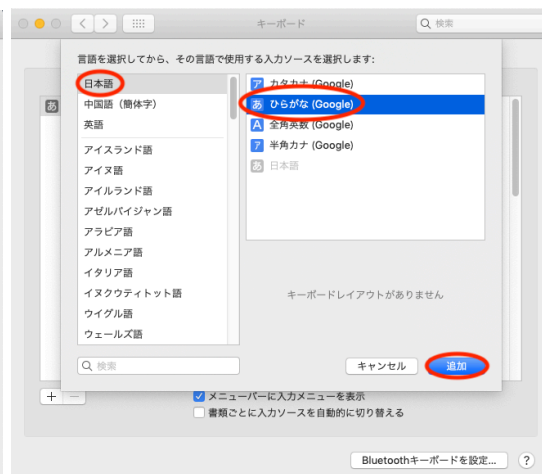


図 1.21: Google 日本語入力 (ひらがな) を追加

*10 Input Method Editor または Engine. キー入力をもとに変換処理などを施すモジュールを指す.

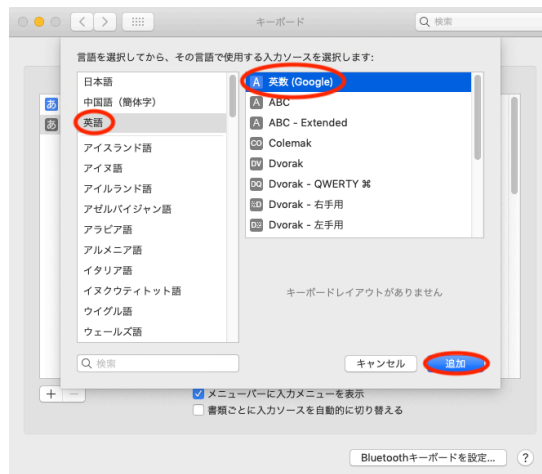


図 1.22: Google 日本語入力（英数）を追加

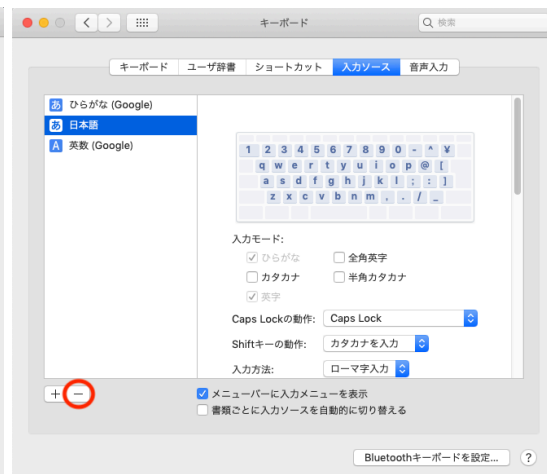
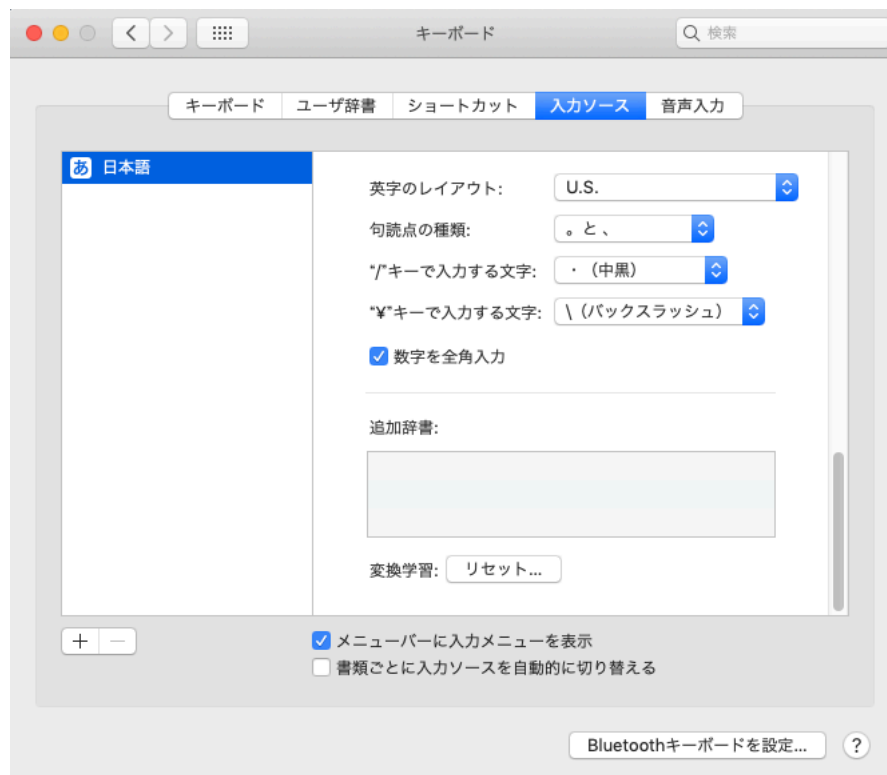


図 1.23: 入力ソースの削除

1.5.5 バックスラッシュを入力する

バックスラッシュ (`\`) は、情報科学類の macOS の標準設定では円マーク (`¥`) として入力されています。入力の設定を変更することで、`¥` を押すと `\` が入力できるようになります。

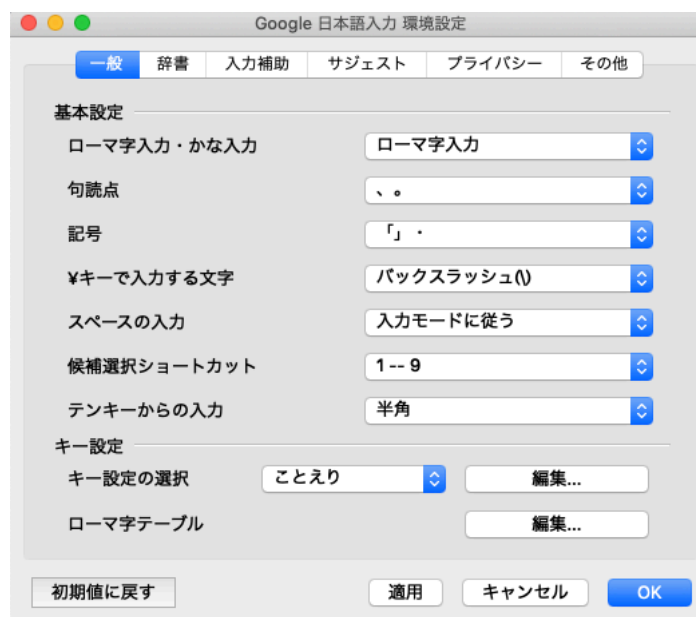
メニューバーの右上の「A」または「あ」をクリックしメニューを表示し、「日本語」環境設定を開く... を選びます。出てきたウィンドウの「入力ソース」>「¥」キーで入力する文字 から、`\` (バックスラッシュ) を選びます (図 1.24)。

図 1.24: `\`を入力するための設定（デフォルト IME）

また、`option` + `¥` で `\` を入力することもできます。

Google 日本語入力を使う場合にも、同様の設定を行います。

Google 日本語入力を選択している状態で、メニューバーの右上の「A」または「あ」をクリックしてメニューを表示します。その中の「環境設定...」を選び、表示されたウィンドウの中の、「一般」>「¥キーで入力する文字」から、「バックスラッシュ (\)」を選択します（図 1.25）。

図 1.25: `\`を入力するための設定 (Google 日本語入力)

1.6 ウィンドウシステム

各アプリケーションはそれぞれひとつ以上のウィンドウを持ち、ユーザーである皆さんはウィンドウに対してマウスやキーボードで操作をすることができます。これらのウィンドウを管理するシステムが、Aqua（アクア）と呼ばれるウィンドウシステム（*window system*）です。ユーザーはウィンドウを切り替えることで、複数のアプリケーションを並行して利用することができます。

一方、ウィンドウの大きさや配置などの外観は、ウィンドウマネージャと呼ばれるアプリケーションが管理します。macOS ではウィンドウ枠の機能が統一されており、例えばウィンドウを閉じたり一時的に縮小表示したりといった利用頻度の高い操作のために、ウィンドウ枠の左上には3種類のボタン（閉じる、最小化、ズーム）が配置されています。またタイトルバーをダブルクリックすると Dock に収容されます。ウィンドウの最小化を行うと、Dock に収容されるということです。

1.6.1 マウス操作の基本

計算機に対して指示をする場合の多くは、マウスを使って行います。マウスの操作に関していくつか覚えておきたい用語がありますので解説します。

マウスポインタ (mouse pointer)

マウスの動きにあわせて画面上で移動する記号のことです。一般的に矢印の事が多いですが、別の形になることもあります。マウスポインタは位置を指示だけでなく、その時点でのウィンドウ・マネージャの状態も表現しています。I 字型のポインタの場合、その場所には文字があるか文字が入力できることを意味し、時計や歯車の形であれば、現在処理を行っていて入力を受け付けていないことを意味しています。

クリック (click)

マウスのボタンを一度だけ押す操作のことです。一般にクリックとだけ言う場合は一番左のボタンを押す動作を指します。ウィンドウ・マネージャ上での大抵の作業はクリックで行えます。macOS における副ボタンのクリック（コンテキストメニューの表示、右クリック）は、`control` を押しながらクリックすることで行えます。

iMac に接続されているマウスは左側と右側でボタンが分かれているため、設定を変更することで（1.1 節参照）右クリックを右側のボタンに割り当てることもできます。真ん中にある小さな球体はホイールアップ／ホイールダウンで使用するホイールです*¹¹。

計算機が処理に時間を取られている状態では、クリックしてもすぐに反応しない場合があります。このような場合であっても、連続して何度もクリックせずに反応があるまで待ってから次の操作に移ってください。

ダブルクリック (double click)

クリックを連続してすばやく 2 回行う操作のことです。

ドラッグ (drag)

ドラッグという手で物をつかむのに相当する動作と、ドロップといい手から物を離すのに相当する操作をまとめてドラッグ・アンド・ドロップと呼びます。ドラッグは、マウスのボタンを押したままマウスを動かすことで行います。マウスのボタンを離す動作がドロップに相当します。ウィンドウの大きさを変更する時などに利用します。

ホイールアップ／ホイールダウン (wheel up / wheel down)

ブラウザ等を使っていると、ウィンドウに入りきらない高さの画面が表示されることがあります。その時には横にスクロールバーが表示されますが、そのバーをドラッグしなくてもホイールアップ／ホイールダウンだけで画面の上下ができます。たいていのウィンドウに使えます。

1.6.2 ウィンドウに対する操作

ターミナルなど、アプリケーションを実行すると画面にウィンドウが開きます。図 1.26 はターミナルのウィンドウの例です。ターミナルは端末エミュレータと呼ばれるものです。この端末エミュレータ上で動作するシェルと呼ばれるコマンドを介して他の様々なコマンドを実行します。

*¹¹ 3 ボタンマウスが接続されている場合もあります。

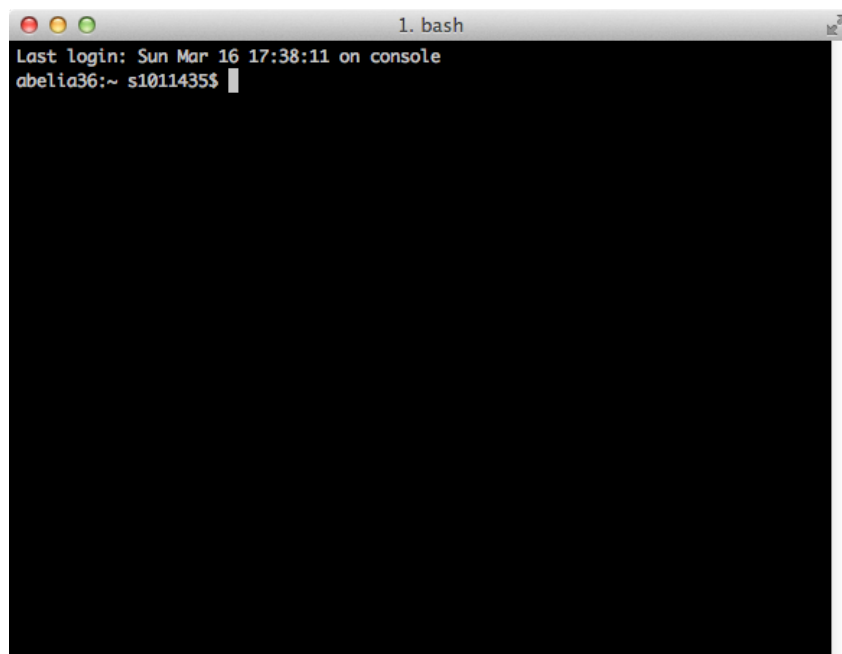


図 1.26: 端末エミュレータ (iTerm)

ウィンドウ・メニュー

ウィンドウをクリックすると、そのウィンドウがアクティブ状態^{*12}になり、デスクトップ画面のメニューバーが、アクティブ・ウィンドウのウィンドウ・メニューに切り替わります。このメニューから、アクティブ・ウィンドウに対する様々な操作を行うことができます。

ウィンドウを閉じる（閉じるボタン）

タイトルバーの左上に赤・黄・緑の3つのボタンがあります。左側のボタン（赤）を押すと、ウィンドウを閉じることができます。ただし、そのウィンドウで表示されていたアプリケーションは終了していないことに注意してください。終了するには、アプリケーションのメニューバーか Dock のコンテキストメニューから終了を選択します。ショートカットキーは **⌘** + **Q** です。

アイコン化（最小化ボタン）

真ん中にあるボタン（黄）をクリックすると、そのウィンドウはアイコン化され Dock に収納されます^{*13}。収納されたアイコンをクリックすると、もとのウィンドウの状態に戻すことができます。

ズームボタン

右のボタン（緑）をクリックすると、ウィンドウの大きさを自動的に適切な大きさに変更します。その部分を再度クリックすると、元の大きさに戻すことができます。

^{*12} ウィンドウがユーザのインタラクションを受け付ける状態になること。

^{*13} **[F11]** キーを押すと表示中の全ウィンドウがアイコン化されます。

ウィンドウの大きさを任意に変更する

大きさを変更させたいウィンドウの四隅にマウスポインタを移動させると、境界線のところでポインタの形が変化します。この状態でドラッグすることによってウィンドウの大きさを変更することが可能になります。

1.6.3 キーボードのショートカット

キーボードは文字や数値を打ち込むだけでなく、色々な機能があります。特にマウスの右クリックをした時や、メニューバーから辿っていくとできる機能が多いのですが、それらをまとめて表 1.4 に記載しておきます。 +  と記載されている場合は「 キーを押しながら  キーを押す」ことを意味します。

表 1.4: ショートカットキーのまとめ

動作	キー
コピーする	 + 
貼り付け	 + 
切り取り	 + 
元に戻す	 + 
全て選択	 + 
ウィンドウを閉じる	 + 
上書き保存	 + 
検索	 + 

今回は表 1.4 のとおり記載しましたが、これ以外にも便利な機能があるので調べてみるのもよいでしょう。また、この他にもスクリーンキャプチャ機能というものがあります。これは、macOS 特有の機能でスクリーンショット撮影してくれる（ウィンドウの画面を撮影して保存してくれる）機能です。 と  と  を同時に押した後に、キャプチャしたい画面にマウスを合わせて、 を押します。すると、カメラのアイコンが出てくるので、キャプチャしたい画像をクリックすると、スクリーンショットがデスクトップに保存されます。また、その際に「スクリーンショット〇〇」（〇〇には数字が入ります）というファイル名で保存されます。

1.7 アプリケーションの紹介

ここでは、よく使う macOS 用のアプリケーションの紹介などをします。アプリケーションは `/Applications/` 内にあります。また、前述の図 1.16 のように、Finder の左側のサイドバーからアプリケーションを選択することでも、アプリケーションのあるディレクトリが見られます。

1.7.1 Microsoft Office

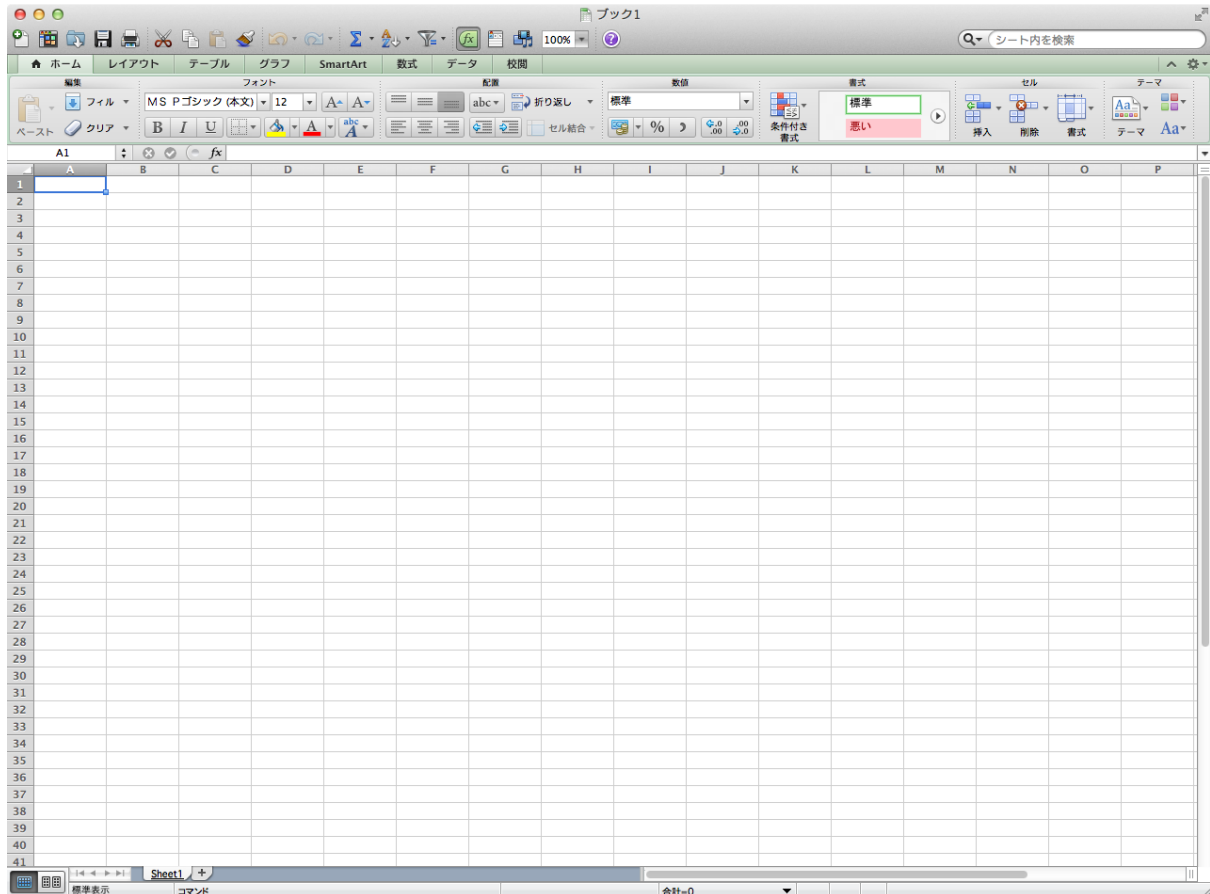


図 1.27: Microsoft Excel の画面

Microsoft Office は、表計算ソフトやワープロソフトなどを取り揃えた、総合オフィスソフトウェアです。主に、以下のようなソフトがあります。

- Microsoft Word（ワードプロセッサ）
- Microsoft Excel（表計算）
- Microsoft PowerPoint（プレゼンテーション）

Microsoft Office のソフトは、Excel で作成した表を Word で作った文章に貼付けるなど、データのやりとりがスムーズに行えます。また、Word と Excel の詳しい使い方については、共通科目「情報(実習)」の手引きに書かれています。全学生に配布されるほか、全学計算機システムの Web ページ (<http://www.u.tsukuba.ac.jp/>) からダウンロードできるので、そちらを参照してください。

1.7.2 Preview.app

Preview.app は、画像や PDF 文書などを表示するアプリケーションです。また、Word や Excel などの印刷プレビューも表示されます。このアプリケーションを指す言葉として、「プレビュー」と呼ぶこともあります。

1.7.3 LibreOffice

LibreOffice とは、オープンソースの総合オフィスソフトウェアです。Microsoft Office とよく似た機能を持ち、ある程度の互換性があります^{*14}。

1.8 印刷

ここでは、macOS からの印刷について説明します。Windows からの印刷は、5.6 節を参照してください。

1.8.1 印刷

各アプリケーションのメニューバーの「ファイル」>「プリント」を選ぶことによって、現在見ているページを印刷することができます。プリント設定の画面は、使用しているアプリケーションによって若干変わりますが、ここでは Preview.app からの印刷について説明します。Preview.app のプリント設定の画面は 2 種類あり、図 1.28 はより単純な方を示しています。また、図 1.29 は、図 1.28 の「詳細を表示」をクリックして得られる、より高度な設定を行う画面です。各部の説明については、以下の通りです。

^{*14} 図やフォントなどは互換性を失いやすいので注意が必要です。



図 1.28: プリント設定

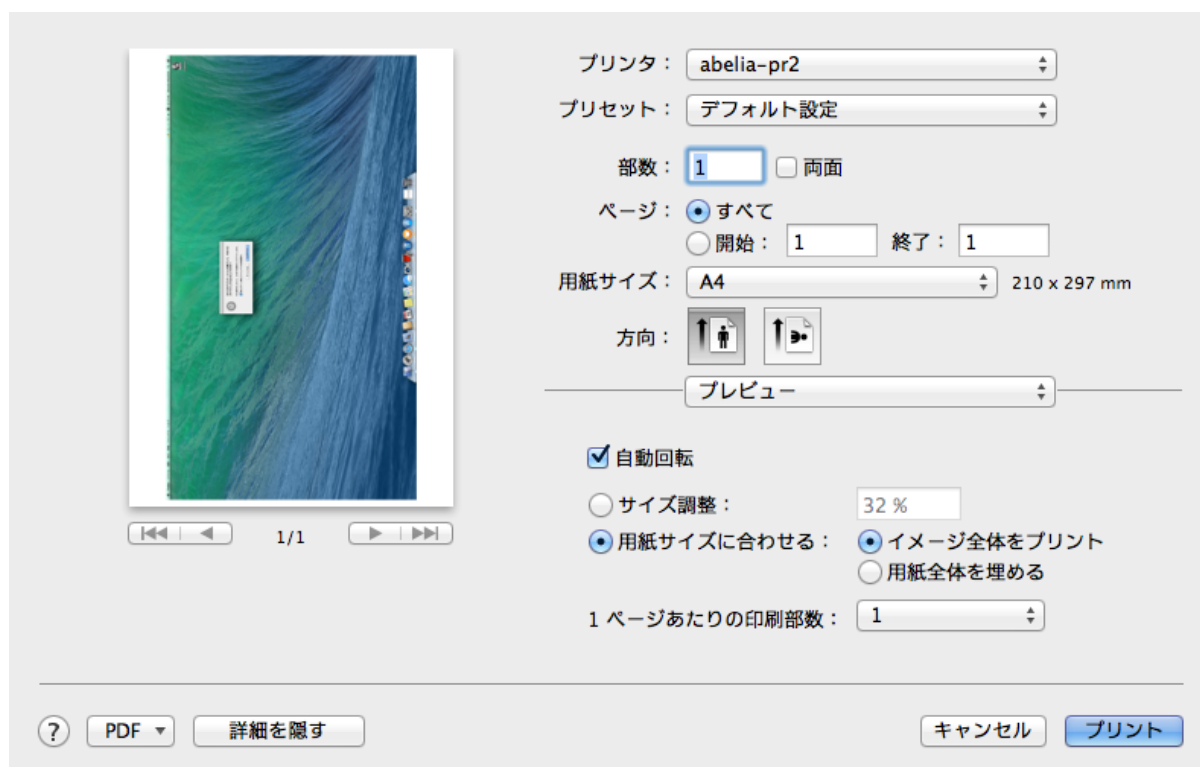


図 1.29: プリント設定（高度）

プリンタ 印刷するプリンタを指定します。

プリセット 印刷の設定を選びます。

部数 印刷する部数です。

両面 両面印刷の指定です。

用紙サイズ 用紙サイズを指定します。COINS の計算機室では、A4 のみ対応しています。

印刷を行う前には、必ずプレビュー画面を確認して、印刷の内容や設定が間違っていないことを確認してから印刷するようにしましょう。

各計算機室のプリンタの台数およびプリンタの名前は表 1 を参照してください。3C113, 3C205, 3C206 で利用可能なプリンタを指定することができますので、自分の位置や用途に合わせて適切なプリンタを選択してください^{*15}。

1.8.2 プリント枚数の上限

個人が印刷できる枚数の限界は、1 月で 200 枚です。それを超えると印刷できなくなります。もし、正当な理由があってそれを超えてしまう場合は技術職員室（3E108）に相談してください。ただし、単純に枚数によって制限がかかるのではなく、カラー印刷だと 1 枚でも、2 枚分として換算されます。また、両面印刷は 1 枚分として換算されるので、カラー印刷を避け、両面印刷にすれば印刷制限に引っ掛かりにくくなります。た

^{*15} 3C113 で 3C205 や 3C206 のプリンタに出力させることも可能です。

とえ印刷制限に余裕があっても、「両面白黒印刷」にご協力ください。

1.8.3 プリント枚数の確認

印刷枚数については、以下の方法で確認できます。

- ホーム画面のカウンタを見る。
- <https://viola4.coins.tsukuba.ac.jp:9192/user> にアクセスする。（http ではなく https であることに注意！）
 1. 自分のユーザ ID とパスワードを聞かれた場合には入力する。
 2. 現在の印刷枚数の状況が表示される。

いざというときに上限に引っかからないよう、定期的を確認しましょう。

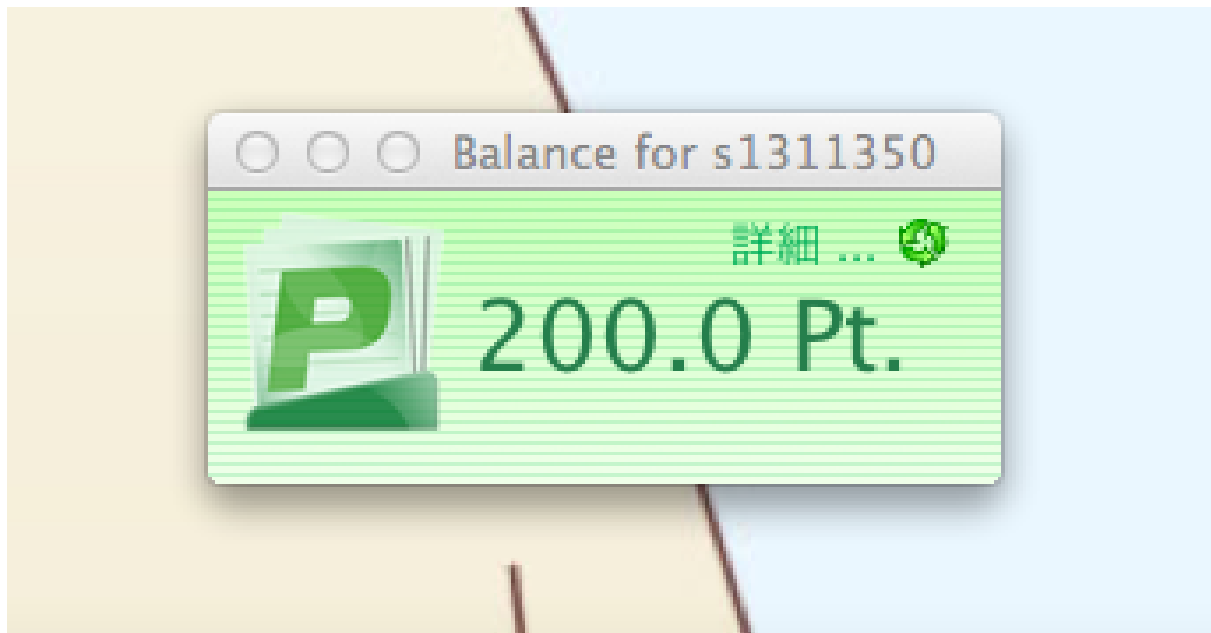


図 1.30: ホーム画面にある印刷可能枚数のカウンタ

第 2 章

コマンド

ここでは、UNIX をコマンドで操作する方法を、UNIX の一つである macOS を使用して学びます。

2.1 iTerm

iTerm は豊富な機能を持ったターミナルエミュレータです (図 1.26)。デスクトップの下の方にある Dock から、iTerm を立ち上げてみましょう。メニューバーの **iTerm** > **Preferences...** から **Profiles** タブ (図 2.1) でプロファイル^{*1}の作成や、プロファイルの文字コードやフォント設定などができます。以下に Profiles タブ内で主に設定しておくべき設定と設定方法を載せます。

^{*1} ターミナルを開くときの設定を名前付きでまとめておくものです。

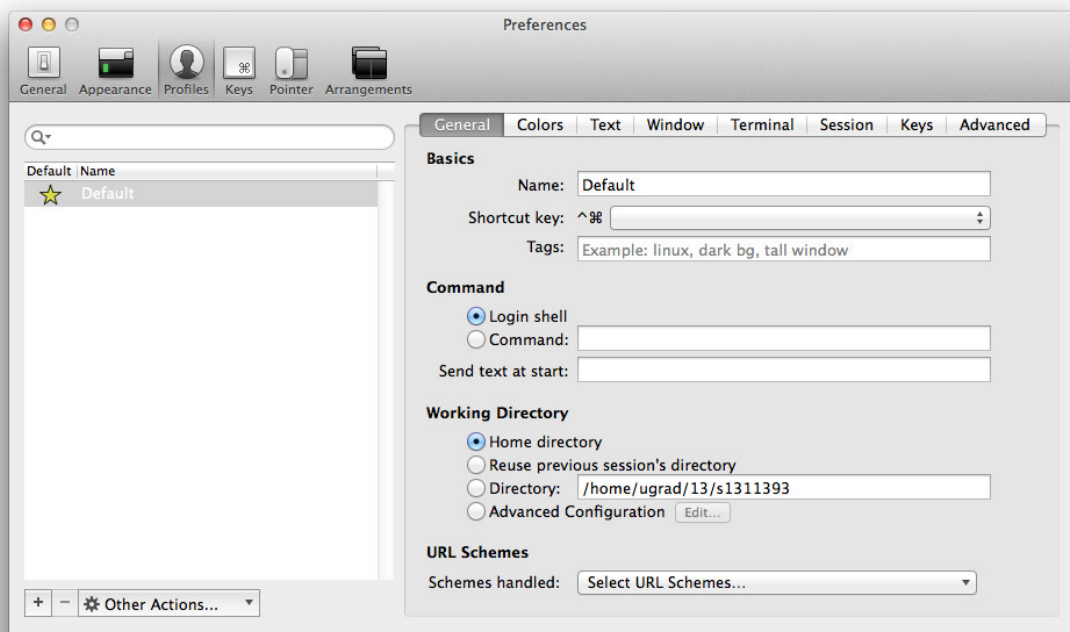


図 2.1: Profile タブ

プロファイルの追加 プロファイルを追加したい場合はウィンドウ左下の **+** を選択すると追加されます。

文字コード **Terminal** タブ内の Character Encoding と書かれている箇所のドロップダウンリストから utf-8 や euc-jp などが設定できます。

端末 (\$TERM) の設定 **Terminal** タブ内の Report Terminal Type と書かれている箇所のドロップダウンリストから xterm-256color などが設定できます。

フォントと大きさ **Text** タブ内の Regular Font と Non-ASCII Font セクションにある **Change Font** ボタンからそれぞれ変更できます。フォントや ASCII について理解していない場合は両方を同じ日本語対応のフォントに変更すると良いでしょう。

端末のカラースキーム **Colors** タブから表示色を細かく設定することができます。ですが細かく設定するのは手間がかかるため、**Colors** タブ内の **Color Presets...** ドロップダウンリストから好みの既存の設定をインポートして使うと良いでしょう。

このような設定以外にも沢山の細かな設定が行えるので iTerm をよく使う方は調べてみると良いでしょう。また、iTerm は講義でもよく使うので自分のプロファイルを作ってデフォルトに設定しておくとも良いでしょう。

2.2 イン트로ダクション

2.2.1 Hello, world

こんな感じの、味気ない表示がウィンドウの上の方にでていますか？*2

```
abelia36:~ s1611xxx $ █
```

ここで、`[e]`と打つと、ちゃんと `e` と表示されると思います。

とにかく最初はなにがなんだかわからないかもしれませんが、習うより慣れろですから、とにかく `echo Hello, world` と打ってみましょう。こんな見ためになればとりあえず OK です。

```
$ echo Hello, world █
```

`$` はもともと出力されていますから、`e` から入力すればよいです。

さて、ここで `[↵]` を叩いてみましょう。下のようになれば成功です。

```
$ echo Hello, world↵  
Hello, world  
$ █
```

さて、今、何をしたのでしょいか。

2.2.2 コマンドと引数

実は、今、コマンドというものを実行しました。`$` から最初の空白までがコマンド名です。

この場合、`echo` がコマンド名です。

そして、最初の空白以降は引数です。引数というのは何でしょうか。例えば、`echo` というのは様々な文章を出力できる汎用的なコマンドです。ですから、`Hello` のほかにも `bye` や `Stand up Workers` など様々なことを出力することができます。そこで、コマンドを実行する時には具体的に何をすればいいのか教える必要があります。その教える内容というのが引数で、コマンドに続けて書くことで、コマンドに指示することができます。

2.2.3 コマンドの基本

さて、コマンドは様々なことができると言いました。それでは、具体的にどのように、何をすることができるのでしょうか。なぜ、わざわざ iTerm からコマンドを利用するのでしょうか。

*2 これ以降、`$` の前は本筋とは関係ないので省略します。これと `$` はあわせてプロンプトと呼ばれます。プロンプトはコマンドを受け付けることを示すために、コンピュータが出力します。このメッセージを改造して、時間や、これから説明するカレントディレクトリ、実行しているマシンの名前やユーザ名などを表示させたり、コマンドの実行が成功したか失敗したかによって、対応する顔文字を表示させたりする人もいます。かわいいプロンプトを使っている人を見かけたら、どういう風になっているか声をかけて聞いてみましょう

コマンドは、今やってみたとおり、文字をベースにコンピュータに下す命令です。コンピュータは、プロンプトから  の入力までの文字列をコマンドとコマンドへの引数であると解釈して命令を実行します。文字で命令を書くことで、手順書のようにコンピュータに何をすればいいのあらかじめ教えておくことができます。そうすることで、時間のかかる操作や何度も繰り返す単調な操作を、人間の手間をかけることなく行うことができます。

コマンドを使いこなすことで、たとえば、1000 枚の写真と 200 個の文書が含まれている 100 個の ZIP ファイルを全部インターネットからダウンロードして開き、写真だけをすべて 50% 縮小して 1 つの ZIP にいれ、文書は特定の語を含むファイルだけ抜きだし 1 つのテキストファイルにまとめて友達のコンピュータに送信するといった、マウスを使って操作していると気の遠くなるようなことが簡単にできるようになります。

それには、この章で説明する内容を越えた知識が必要となりますが、本書では最初のステップを踏み出す手引きをします。echo コマンドではファイルを必要としませんでしたが、上の例のようなファイル操作を行うためには、まずファイルについて知る必要があります。

2.3 ディレクトリ構造

ここでは操作の対象となるファイルとディレクトリについて説明します。まず身近にあるファイルについて考えてみましょう。一つファイルを想像してみてください。たとえば、この手引きの PDF ファイルでもいいかもしれません。それを念頭にちょっと話を進めていきます。

UNIX では、ファイルに英文字・数字・いくつかの記号からなる名前 (ファイル名) をつけて管理します。そして、ファイルをまとめるものとしてディレクトリ (*directory*) というものがあり、このディレクトリの中に、ファイルとディレクトリを格納することができます。

ちょうど、入学式でもらったファイルをまとめておくと、あとあと困らないので、バインダーに閉じたり^{*3}、ひとところに纏めて置いたりしたと思いますが、それと同じようなことが、人間が使いやすいようにするためにコンピュータでもできるようになっているわけです。

そしてこういったお約束ごとがあるので、ひとつのコンピュータに複数のファイルを分類して入れることができるというわけです。

なお、1 つのディレクトリ内に同じファイル名のファイルを 2 つ以上格納することはできません。ただし、図 2.2 のように、別のディレクトリにあるファイル同士はファイル名が同じでも共存可能です。

^{*3} ただし、コンピュータの中ではバインダーの中にバインダーを閉じるといったことは容易に行えます

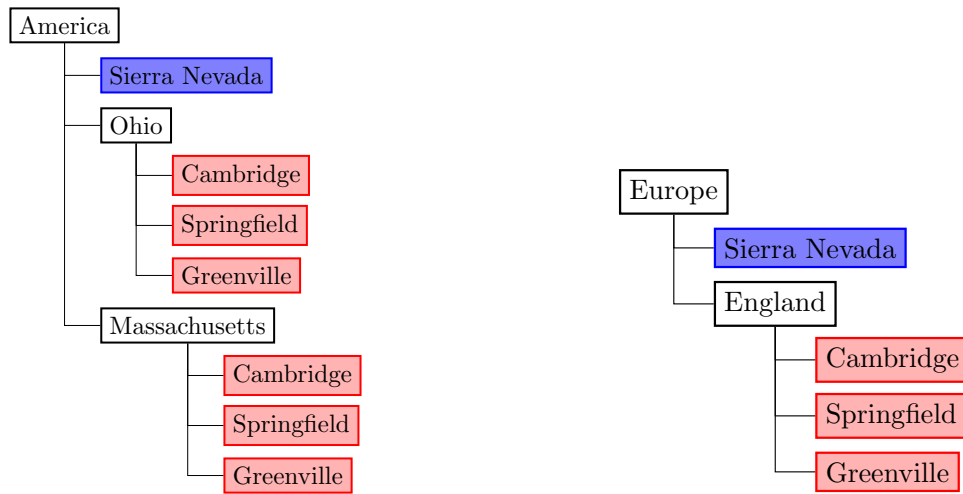


図 2.2: ファイル名が同じでも、ディレクトリが違えば共存できる

こういった構造や、それを支えるシステムをディレクトリ構造、と呼び習わすことになっています。そういうわけで、UNIX のディレクトリ構造は、図 2.3 のように、ディレクトリにファイルを入れることができます。

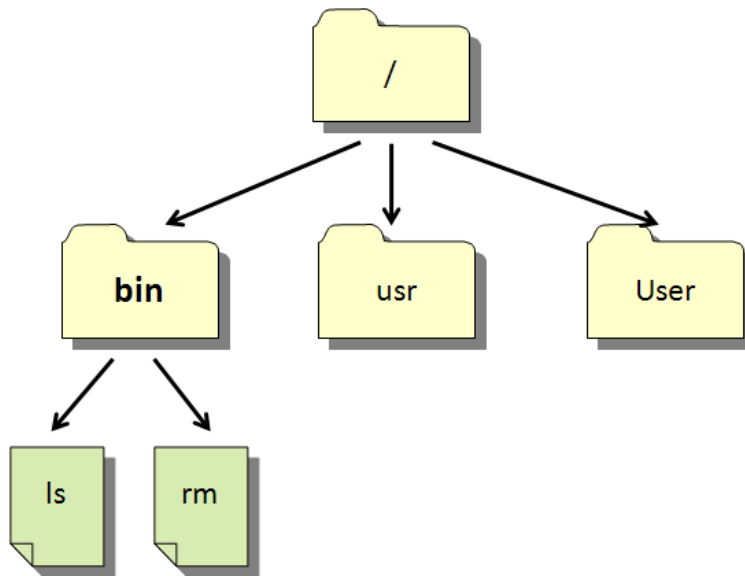


図 2.3: ディレクトリ構造の例

ディレクトリ構造とは、こういったファイルやディレクトリを扱うシステムと構造の事です。ディレクトリをさらにディレクトリに入れることができるので、図 2.3 のような入れ子構造を作ることができます。

図 2.3 の例では、まず「/」で表される「木の根^{*4}」にあたるルートディレクトリというものがあります。

^{*4} ディレクトリ構造は樹形図で表されることが多いため、しばしば木という表現が用いられ、親ディレクトリのないディレクトリを根（あるいは *root*/ルート）と呼びます。macOS のターミナルや Windows のコマンドプロンプトでは、その名の通り *tree* というコマンドがあり、カレントディレクトリ（後述）をルートとしたディレクトリ構造が見られます。

これから皆さんが触れていく UNIX では、1つのコンピュータはただひとつのディレクトリツリーをもち、接続されたドライブやネットワーク上のディレクトリ構造もすべてこのディレクトリツリーに参加することになっています。

つまり、ルートディレクトリとは、コンピュータからみえるすべてのファイルとディレクトリの根っこにあたるものです。これより上のディレクトリはありませんし、ここからたどれないディレクトリ構造はありません。UNIX のディレクトリ構造は「/」というディレクトリからなる唯一のディレクトリツリーを持ち*5、すべてのディレクトリの“根”になっています。

さて、図 2.3 のルートディレクトリには、その中に “bin” “usr” “User” という 3 つのディレクトリがありますね。ディレクトリ “bin” の中にはさらに、“ls” “rm” というファイルがあります。これらは実はファイルであるだけでなく、プログラムでもあります。プログラムはファイルなのです。

2.4 ファイルをコマンドラインで操作する

2.4.1 ls - 一番有名なコマンドを使ってみる

とにかく、ls /と打ってみましょう。最初の\$はもう画面にでているはずなので打たないこと、最後に`↵`を押すことを忘れずに。

```
$ ls /  
Applications  Pictures  Volumes  etc      net      sbin      usr  
Library       System   bin      home     opt      tmp       var  
Network       Users    dev      mach_kernel private  traces.log
```

このコマンドは、引数にわたされた文字列をディレクトリ名と解釈して、そのディレクトリの内容を表示する命令です。では、“/”とはどんなディレクトリかといいますと、2.3 節で述べたルートディレクトリのことです。いちいちルートディレクトリと打つとかったるいですし、打ち間違いはバグの元ですから、短く“/”という名前になっているのです。そのコマンドの結果が、上のリスト 2 行め以降の内容になります。

次に、単にlsと打つとどうなるでしょう。

```
$ ls
```

なにか出てきたかもしれませんが、何も表示されなかったかもしれません。あなたがもう何かファイルを作っていた場合にはそれが表示されるのです。よく分からないと思うので、ちょっと手習い用のディレクトリを掘って、そこですこし実験してみましょう。

*5 みなさんにとって馴染み深いかもしれない Windows では、ハードディスクは C:\、USB メモリは Z:\、といったようにディレクトリツリーが別れており、いわば森のような構造になっています。このような構造をとっている理由としては、ハードディスクが高価だった頃の名残で、S 起動用のフロッピー、アプリケーション起動用のフロッピーや保存用のハードディスクなど、媒体を分けていたためです。

2.4.2 最初の手習い - ディレクトリを掘るということ

まず、`mkdir erste`と打ってみましょう。これは、ディレクトリを作成するコマンドです*⁶。すでに同名のディレクトリがあった場合は、別の引数を渡しましょう*⁷。

Listing 2.1: ディレクトリが正しく作成された場合

```
$ mkdir erste↵  
$ █
```

Listing 2.2: ディレクトリがすでに存在したので重複して作成できなかった場合

```
$ mkdir erste↵  
mkdir: erste: File exists  
$ █
```

もういちど`ls`してみると、前に`ls`したときと比べて、今掘ったディレクトリが新しく登場していると思います。

このコマンドは、ディレクトリを掘るコマンドだと言いました。これはスコープのようなもので、新しくディレクトリを掘ることはできても、その中に入るとのことまでは自動ではやってくれません*⁸。

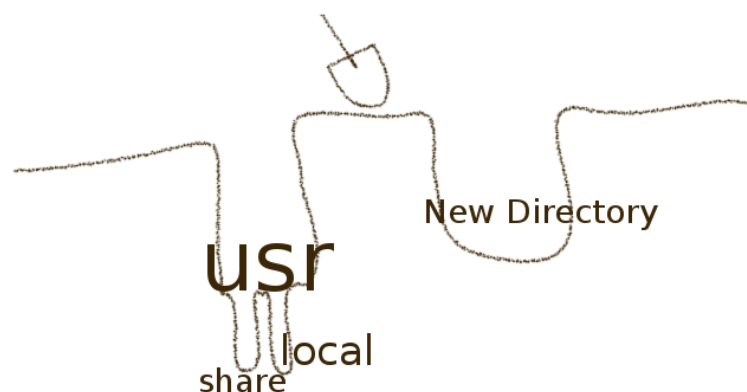


図 2.4: スコープのあるところで作業をする

*⁶ MaKe DIRectory の略です

*⁷ `mkdir zweite`など打ってください

*⁸ これは、一度にたくさんディレクトリを作るときなどに便利です。このように、一つのコマンドがあれやこれやお節介を焼くべきではない、言われたことだけちょうど行うにとどめるべきだ、というプログラムに関する哲学に基づいて、一つのコマンドごとにだいたいざっくりと一つの機能だけに絞り込んで UNIX のコマンドは作られています

そこで、次に自分でその穴に入り、作業をするディレクトリを変える必要があります。

2.4.3 自分について知る

ところで、よく文中にでてくる「作業」とはどういう意味でしょう。実はこうしてコマンドを打つとき、あなたもファイルと同様ルートディレクトリから辿れるある場所に立って、そこからコマンドを打っているのです。その場所を知るために、`pwd` というコマンドを使って調べてみます。

```
$ pwd↵
```

`/home/ugrad/s1911xxx` というようなあるディレクトリが表示されるかもしれませんが、もしかしたら別のところにいるのかもしれませんが、どこにいるのかに関わらず、ある場所が存在して、そこに立って足元にディレクトリという穴を掘ったところです。作業をするときは、必ずシステムの中のどこかにいてそこから作業をしているということを忘れないでください。

2.4.4 穴に落ちる

ということで、現在は、自分の居場所について知り、そして、足元にディレクトリを掘ったところでした。今度は、そのディレクトリに入ってみようとおもいます。

Listing 2.3: 先ほど作成したディレクトリが `erste` だった場合

```
$ cd erste↵  
$ █
```

`cd`^{*9}は、自分の作業をするディレクトリを変えるコマンドです。もう一度`pwd`すると、「穴に落ちているということ」がわかると思います。

2.4.5 ファイルを作る

さて、ここで`ls`してみてください。おそらくまだ何もありません。そこで、新たにファイルを作ってみようと思います。

```
$ touch test↵  
$ █
```

もう一度、単に`ls`と打ってみましょう。

`test` というファイルが追加されていると思います。

さらにいろんなコマンドを下で解説していますが、まだよく理解していない機能を試すときは、重要なファイルを誤って消さないように、こうしてテスト用のディレクトリを作ってしまうとよいでしょう。

^{*9} Change the working Directory の略です

2.5 用語の整理

ここで、ディレクトリ構造の話をするときによく出てくる用語を紹介します。

コマンド (command)

コマンドは、コンピュータに対する命令です。UNIX システムには、たくさんのコマンドが搭載されていて、自分で増やすこともできます。コマンドを発行する時は、端末、端末エミュレータなどと呼ばれるソフトウェア (本章では iTerm に限定して解説してあります) を用いて文字を使ってコマンドを発行します。

引数 (argument)

コマンドに、動作の詳細を教えるものです。コマンドに続けて書きます。引数の数はいくらでもいいのですが、認識できる限界や順序、書式はコマンドごとに限界があります。

ディレクトリデリミタ (directory delimiter)

ディレクトリの名前と名前をわけるものです。/`の下にある bin に入っている ls というファイルを指すには/bin/ls と表記します。ルートディレクトリ (2.3 節参照) をさす"/"とデリミタの"/"は同じ文字ですが、意味が違うことに注意しましょう。同じ文字である必然性はないので、システムによっては違う文字になっていますが、UNIX ではどちらも"/"という文字を使うという約束になっています。同じ文字なら同じ意味だと思いついてしまうと、理解の妨げになるので注意しましょう。`

カレントディレクトリ (current directory)

「自分が今作業をしているディレクトリ」のことです。たとえば、図 2.3 の中の「bin」というディレクトリで作業をしている場合、「カレントディレクトリは/bin である」といいます。

パス名 (path name)

ここでは、あるディレクトリやファイルの場所を表す文字列を意味します。たとえばディレクトリデリミタの例に出てきた"/bin/ls"はパス名の例です。

相対と絶対パス名

力学には相対座標と絶対座標という言葉がありますが、UNIX においてもあるディレクトリやファイルまでの経路を表す方法に絶対パス名を記述する方法と相対パス名を記述する方法があります。絶対パス名はルートディレクトリから中継する全てのディレクトリを記述したものです。bin ディレクトリ以下の ls を指定するには下記のように指定します。

```
$ /bin/ls↵
```

これに対して相対パス名とは、カレントディレクトリを基準とした経路を表すものです。図 2.3 において bin のディレクトリにて作業をしているとします。ここで bin ディレクトリ以下の ls を指定するには以下のよう

```
$ ./ls↵
```

ここで「`.`」はカレントディレクトリを表します。

ファイルの指定を「`/`」記号から始めると、それは「絶対パス名」とであるとみなされます*10。「`/`」以外の文字始めると、それは「相対パス名」とであるとみなされます。

(ファイル名の) 拡張子 (extension)

ファイル名は (文字種の制限を守る限り) 自由に付けることができますが、ファイル名の最後に、ファイルの種類を表わす文字列を「`.`」で区切って書くことがあります。これを拡張子といいます。たとえば、Java 言語のプログラムが書かれたファイルには「`.java`」、音楽は「`.mp3`」や「`.ogg`」といったものです。

2.5.1 ホームディレクトリ (home directory) について

ユーザには、各自が自由に使える「ホームディレクトリ」が割当てられています。みなさんの実際の作業は、各自のホームディレクトリの下で行います。ホームディレクトリは、「`~`」(チルダ) 記号で表わされます。たとえば「自分のホームディレクトリにある『Report』というディレクトリの…」というようなときに「`~/Report/...`」のような使いかたをします。

2.5.2 ファイルのパーミッション (permission)

各ファイル・ディレクトリは、ファイルのアクセス権を設定するための「パーミッション (permission)」という属性を持っています。パーミッションについては、「`ls` コマンド 2.6.1」「`chmod` コマンド 2.7.7」の所でもう少し詳しく述べます。ファイルのパーミッションの設定には気をつけてください。設定を間違えると、自分のメールを他人に読まれたり、大事なファイルが他人に消されたりということが起こり得ます。ファイルシステムについてのより詳しい説明については「UNIX Super Text」など、UNIX についての本がたくさん出ていますので、それらを参照してください。

2.6 ファイルに関するコマンド

2.6.1 `ls` コマンド

自分が現在作業をしているディレクトリ (カレントディレクトリといいます) にあるファイルの名前を表示します。

```
$ ls↵
dir1 file1 file2
$ █
```

この例では、3 個のファイルおよびサブディレクトリの名前が表示されました。また、コマンドの後に、ディレクトリ名を指定すると、指定したディレクトリにあるファイルの名前を表示します。

*10 電話番号が 0 から始まると市外局番になるようなものです。ファイルに至る経路を全て記述することから、「フルパス」とも呼ばれます。

ls コマンドに、「-l」(小文字の L) というオプションを付けることにより、それぞれのファイル・ディレクトリについての詳しい情報を見ることができます。

```
$ ls -l
total 245
drwxr-xr-x 2 johotaro ugrad      48  1 25 00:45 dir1
-rw-r--r-- 1 johotaro ugrad    7377  1 25 00:44 file1
-rw----- 1 johotaro ugrad  239914  1 25 00:44 file2
$
```

ここで、ファイルのパーミッションの表示について説明します。左から、「ファイルのパーミッション情報」「リンク数」「ファイル所有者」「ファイル所有グループ」「ファイルサイズ」「ファイルの最終更新日時」「ファイル名」を示しています。例として、上の file1 に関して見てみます。

```
-rw-r--r-- 1 johotaro ugrad 7377  1 25 00:44 file1
```

いちばん左に `-rw-r--r--` と表示されていますが、これがそのファイルのパーミッションの表示です。この 10 桁の文字列のうち、一番左の 1 文字は、そのファイルの種類を示しています。これが「-」なら通常のファイル、「d」ならディレクトリ、「l」なら、シンボリックリンクです^{*11}。残りの 9 桁が、実際のアクセス許可情報を持っています。この 9 桁の表わす情報は、表 2.1 のようになっています。

表 2.1: 9 桁の文字列の意味

左側 3 桁	ファイルのオーナーの持つ権利
中央 3 桁	グループの持つ権利
右側 3 桁	その他の人の持つ権利

それぞれの 3 桁が、「読み」「書き」「実行」についての許可を表わしています (表 2.2) ^{*12}。

表 2.2: 文字の意味

左の桁が r	読むことが許可されている。
中央の桁が w	書き込みが許可されている。
右の桁が x	実行 (ディレクトリに対しての場合は、探索) が許可されている。
- の表示	許可されていない。

これらの表から、先程の例のファイルは、「オーナーは読み・書きができ、グループの人とその他の人は読むことのみできる」ということがわかんと思います。

^{*11} Windows のショートカットのようなものです。他にもありますが、それらは man で調べてみてください。

^{*12} Windows などでは.COM や.EXE という拡張子が実行ファイルになりますが、UNIX ではそのファイルが実行ファイルかどうかはこの実行パーミッションによって識別されます。

今までの例では、「ドットファイル*¹³」を見ることができません。ドットファイルを見るときには、「-a」という省略可能引数*¹⁴ を付けます。また、「-F」オプションをつけると、ディレクトリ名の末尾に/が付いて判別しやすくなります。日本語のファイル名が正しく表示されないときは「-v」をつけると正しく表示されるかもしれません*¹⁵。これらの他、ls コマンドには多数のオプションがありますが、それらの詳細は「man ls」として、ls コマンドのマニュアルを参照してください。

2.6.2 mkdir コマンド

mkdir コマンドは、新しくディレクトリを作成するコマンドです。mkdir `ディレクトリ名` のようにして使います。

```
$ ls -l↵
total 10
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 00:10 dir1
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file1
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file2
$ mkdir dir2↵
$ ls -l↵
total 18
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 00:10 dir1
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 10:44 dir2
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file1
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file2
$ █
```

2.6.3 rmdir コマンド

rmdir コマンドは、空のディレクトリを削除するコマンドです。rmdir `ディレクトリ名` のようにして使います。

```
$ ls -l↵

total 18
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 00:10 dir1
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 10:44 dir2
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file1
-rw-r--r--  1 johotaro  ugrad    12  2 15 00:10 file2

$ rmdir dir2↵
$ ls -l↵
total 245
drwxr-xr-x  2 johotaro  ugrad  4096  2 15 00:10 dir1
```

*¹³ 「.」記号ではじまるファイル・ディレクトリのことです。設定ファイルなどによく使われます。

*¹⁴ この場合、通常の引数と違って動作の詳細を変更する引数です。オプションともいいます

*¹⁵ それでも正しく表示されないときはターミナルの設定 (`iTerm` ≫ `Preferences` ≫ `Terminal`) 内の `Character Encoding` で文字セットエンコーディングを変更してみてください。

```
-rw-r--r--  1 johotaro  ugrad    12   2  15  00:10 file1
-rw-r--r--  1 johotaro  ugrad    12   2  15  00:10 file2
$ █
```

2.6.4 cp コマンド

cp コマンドは、ファイルのコピーを行なうためのコマンドです。あるファイルを、別のファイル名にコピーしたい場合、cp コピー元ファイル名 コピー先ファイル名のようにします。

```
$ ls -l↵
file1
$ cp file1 file2↵
$ ls↵
file1 file2
$ cp file2 file3↵
$ ls↵
file1 file2 file3
$ █
```

コピー先ファイル名にディレクトリ名を指定すると、そのディレクトリの中にファイルのコピーが作られます。またこの場合には、コピー元ファイル名を複数指定することができます。

```
$ mkdir dir1↵
$ ls dir1↵
$ ls -F↵
dir1/ file1 file2
$ cp file1 file2 dir1↵
$ ls dir1↵
file1 file2
$ █
```

2.6.5 mv コマンド

mv コマンドは、cp コマンドとは異なり、「ファイルの移動」「ファイル名の変更」を行なうコマンドです。mv 移動元ファイル名 (複数可) 移動先ディレクトリ名あるいは mv 旧ファイル名 新ファイル名のような書式で使います。

```
$ ls -F↵
dir1/ file1          ←ファイル1つに、ディレクトリ1つ。
$ ls dir1↵
    ←dir1 の中にファイルはない。
$ mv file1 file2↵
$ ls -F↵
dir1/ file2          ←ファイル名が変わった。
$ mv file2 dir1↵
$ ls -F↵
dir1/
$ ls dir1↵
```



```
file2          ←ファイルが移動した。
$ █
```

2.6.6 rm コマンド

rm コマンドは、ファイルの削除をするためのコマンドです。rm **ファイル名 (複数可)** のようにして使います。また、rm -r **[ディレクトリ名]** のようにすると、指定したディレクトリの中のファイル全て、および、そのディレクトリ本体をすべて削除します。「-i」オプションをつけて実行すると、削除するファイルについて、それぞれ本当に削除するかどうか聞いてきます。この質問には、y (yes) か n(no) で答えてください。

```
$ ls -F↵
dir1/ file1    ←ファイル1つに、ディレクトリ1つ。
$ ls dir1↵
file2          ←ディレクトリ内に、ファイル1つ。
$ rm -r dir1↵
$ ls↵
file1          ←ディレクトリが削除された。
$ rm -i file1↵
remove file1?   ← y と答えると
$ ls↵ ←ファイルが削除された。
$ █
```

rm コマンドを使用する際は、誤って必要なファイル・ディレクトリを削除しないように注意しましょう。特に、rm -r * のようなコマンドは要注意です。このコマンドはカレントディレクトリ、およびそれより下にある全てのファイルおよびディレクトリを消去するコマンドです。つまり、ディレクトリ単位でざっくり消してしまえるので便利といえ便利ですが、消してしまった後に必要となると、復元できなくなってしまうこともあります。削除する前に本当に削除してしまっても大丈夫なのかチェックしてから削除しましょう。

2.6.7 cd コマンド

カレントディレクトリを移動するコマンドです。cd **ディレクトリ名** のようにして使います。ディレクトリ名を省略した場合は、「自分のホームディレクトリへの移動」となります。また、「cd -」とすると移動する前のディレクトリに戻ります。

2.6.8 pwd コマンド

今いるディレクトリの絶対パス名を確認できるコマンドです。cd コマンドであっちこっちに行っているとシンボリックリンクなどに惑わされて、今自分がどこにいるか判らなくなることがよくあります。そのようなときに pwd コマンドを使うと、自分のいるディレクトリがわかります。

```
$ pwd↵
/home/ugrad/99/s99xxxxx
$ █
```

2.7 ファイルの中身に関するコマンド

2.7.1 cat コマンド

ファイルの内容を画面に表示するコマンドです。cat ファイル名 のようにして使います。そのファイルがテキストファイル (通常の文字からなるファイル) ならば、その内容を読むことができます。ファイル名を複数指定すると、指定した順番にファイルを続けて表示します。これを利用して、複数のファイルを連結する際にも使うことができます。

```
$ cat file1↵
This is test 1.
$ cat file2↵
This is test 2.
$ cat file1 file2 > file3↵
$ cat file3↵
This is test 1.    ←file1, file2の内容がfile3に入っている.
This is test 2.
$ █
```

2.7.2 lv コマンド

cat コマンドでは、1画面を越える長さのファイルを読もうとすると、ファイルの最初の方がスクロールして画面から消えてしまい、内容が読めません。長いファイルを読むときに、1画面ずつ読むためのコマンドがlv^{*16}です。ここでは、lvのキー操作を簡単に表2.3にまとめておきます。

lvは、「文書・文章の入ったファイルを表示し、その内容を(人間が)読む」ことを目的としたコマンドですが、catについては、ファイルを表示するためだけでなく、ファイルの内容を連結するためにも使われます。

2.7.3 head コマンド

head コマンドは、指定したファイルや標準入力の最初の10行を表示するコマンドです。-n 数字 オプションを付けることにより、先頭から任意の行数を表示することができます。

```
$ cat file1↵
line 1.
line 2.
line 3.
line 4.
line 5.
line 6.
$ head -n 3 file1↵
line 1.
line 2.
```

^{*16} こういうコマンドを「ページャ」ともいいます。

表 2.3: lv コマンドのまとめ

キー	機能
q	lv を終了します.
j	1 行読み進みます (ファイルの後方に向かってスクロールします).
k	1 行戻ります (j の逆です).
f,(space)	1 画面読み進みます.
b	1 画面戻ります (f の逆です).
g,>	先頭に移動します.
G,>	末尾に移動します.
/ パターン	パターンで示された文字列の検索を, ファイル後方に行ないます.
? パターン	パターンで示された文字列の検索を, ファイル前方に行ないます.
n	直前の検索を後方に向かって繰り返します.
N	直前の検索を前方に向かって繰り返します.

```
line 3.
$ █
```

2.7.4 tail コマンド

tail コマンドは, 指定したファイルや標準入力最後の 10 行を表示するコマンドです. head コマンドと同様に, `-n` オプションを付けることにより, 最後から任意の行数を表示することができます.

```
$ tail -n 4 file1↵
line 3.
line 4.
line 5.
line 6.
$ █
```

2.7.5 grep コマンド

grep コマンドは, ファイルや標準入力に対して指定した文字列の検索を行い, その文字列が含まれる行を表示するコマンドです.

```
$ cat file2↵
tsukuba
kenkyugakuen
bampakukinenkoen
midorino
miraidaira
moriya
$ grep ba file2↵
```

```
tsukuba
bampakukinenkoen
$ █
```

2.7.6 sort コマンド

sort コマンドは、ファイルや標準入力に対して、行単位でのソートを行うコマンドです。

```
$ sort file2.txt
bampakukinenkoen
kenkyugakuen
midorino
miraidaira
moriya
tsukuba
$ █
```

2.7.7 chmod コマンド

chmod^{*17} コマンドは、ファイルのパーミッション 2.5.2 を変更するためのコマンドです。chmod **パーミッション変更シンボル** **ファイル名** という書式で使います。パーミッション変更シンボルは、「パーミッションをどのように設定するか」を表わす文字列で、「誰に対してのパーミッションを変更するのか (u,g,o あるいは a の記号を用います)」「許可を与えるのか、許可を取り消すのか (+, - あるいは = の記号を用います)」「書きこみ、読みこみ、実行許可のうちのどれに対してか (r,w あるいは x の記号を用います)」を指定する 3 つの部分からなります。表 2.4 に、例を挙げます。

表 2.4: 管理者権限の変更

chmod u+w file1	file1 に対して「オーナーによる書き込み許可」を出す。
chmod g+r file2	file2 に対して「グループによる読み込み許可」を出す。
chmod o-x file3	file3 に対して「他の人による実行許可」を取り消す。
chmod a+rx file4	file4 に対して「すべての人に対する読み込み・実行許可」を出す。
chmod u=rw file5	file5 に対するオーナーの権利は「読み・書きは許可、実行は不許可」とする。

自分に対するアクセス許可がないファイルをアクセスしようとしてもできません。以下の例のように、エラーになります。

```
$ ls -l
total 245
drwxr-xr-x 2 johotaro ugrad 48 1 25 00:45 dir1
```

^{*17} CHange MODE の略です。

```
-rw-r--r-- 1 johotaro ugrad      7377  1 25 00:44 file1
-rw----- 1 johotaro ugrad    239914  1 25 00:44 file2
$ chmod u-r file1↵
$ ls -l↵
total 245
drwxr-xr-x 2 johotaro ugrad       48  1 25 00:45 dir1
--w-r--r-- 1 johotaro ugrad      7377  1 25 00:44 file1
-rw----- 1 johotaro ugrad    239914  1 25 00:44 file2
$ touch file1↵
file1: Permission denied
$ █
```

ファイルのパーミッションを、記号でなく数値で直接設定するやり方もあります。これは $r = 4, w = 2, x = 1$ として、オーナー、グループ、他の人の順にその和を並べるやりかたです 2.5。慣れると一度に全てのパーミッションの設定ができて便利です。

```
$ ls -l↵
total 245
drwxr-xr-x 2 johotaro ugrad       48  1 25 00:45 dir1
-rw-r--r-- 1 johotaro ugrad      7377  1 25 00:44 file1
-rw----- 1 johotaro ugrad    239914  1 25 00:44 file2
$ chmod 666 file1↵
$ ls -l↵
total 245
drwxr-xr-x 2 johotaro ugrad       48  1 25 00:45 dir1
-rw-rw-rw- 1 johotaro ugrad      7377  1 25 00:44 file1
-rw----- 1 johotaro ugrad    239914  1 25 00:44 file2
$ █
```

file1 に注目すれば、わかりやすいと思います。ここで途中（6 行目）で出てきた「666」という数値について説明を加えておきます。左の数値は所有者の権限を、真ん中の数値はグループの権限を、右の数値はその他（所有者でもグループでもない）の人の権限を表します。数値は 4 が読み出し許可、2 が書き込み許可、1 が実行許可を表し、例えば 6 なら $4+2$ で「読み出し許可 + 書き込み許可」という意味になります。つまり、今回出てきた 666 であれば「所有者にもグループにもその他にも、読み出しと書き込みを許可する」ということになります。数値の詳細に関しては表 2.5 を参照してください。

表 2.5: 管理者権限を表す数値の詳細

数値	内容
4	読み出し許可
2	書き込み許可
1	実行許可
7 (4+2+1)	読み出し許可+書き込み許可+実行許可
6 (4+2)	読み出し許可+書き込み許可
5 (4+1)	読み出し許可+実行許可
3 (2+1)	書き込み許可+実行許可

2.7.8 open コマンド

open コマンドとは、ファイル・ディレクトリ・URL などを開くコマンドです。Finder などの GUI 環境における、ファイルのダブルクリックに相当します。

```
$ open file1↵
↑ これでfile1というファイルが開かれた。
$ open .↵
↑ これでカレントディレクトリがFinderで開かれた
```

2.8 システムに関するコマンド

2.8.1 quota コマンド

自分がどれくらいディスクを使用しているかを調べるコマンドです。

```
$ quota -v↵
Disk quotas for user s1311350 (uid 5621):
   Filesystem  1K blocks      quota    limit  grace  files   quota  limit  grace
    /home       2456883   3072000  3584000      84863      0      0
$ █
```

この例だと、使用量 (blocks) が 2456883 キロバイト、制限 (quota) が 3072000 キロバイトとなっています。

2.8.2 du コマンド

ディスクの使用量を表示するコマンドです。指定したファイルと、それをルートとする階層中にある全ディレクトリのディスク使用量を表示します。ここでは、カレントディレクトリ下の各ファイル、または、ディレクトリごとのディスク使用量の一覧を表示しています。「-h」によってディスク使用量を人間の読みやすい単位に変換し、「-s」によって各ディレクトリのディスク使用量の総計のみを表示しています。

```
$ du -h -s *↵ ← ‘*’ は後述するワイルドカード
12K    dir1
```

```
44M      dir2
200M     dir3
$ █
```

2.9 画像に関するコマンド

2.9.1 convert コマンド

convert コマンドは、画像のサイズや形式を変換するコマンドです。非常に多くの機能を持っていて、すべてを網羅することはできないので、ぜひ各自で調べてみてください。

```
$ convert piture1.png picture2.eps↵ ← PNG形式をEPS形式に変換
$ convert -resize 50% piture1.png picture3.png↵ ← サイズを縦横ともに50%に縮小
$ convert -resize 512x512 piture1.png picture4.png↵
↑ 縦横比を保ったまま、512x512のサイズに収める
$ convert -resize 512x512 piture1.png picture5.png↵
↑ 縦横比を保たずに、512x512のサイズに変換
$ █
```

2.10 プロセスを取り扱うコマンド

Unix でプログラムを作ったり作業をしたりしていると、「プログラムが止まらなくなった」「他のウィンドウからプログラムを停止させたい」という状態になることがよくあります。そのようなときに用いるコマンドを説明します。

2.10.1 ps コマンド

ユーザが今どのようなプロセス（プログラム）を動かしているのか調べるためのコマンドです。

```
$ ps↵
  PID TTY          TIME CMD
 5978 pts/8        00:00:00 bash
29522 pts/8        00:00:00 ps
$ █
```

ps **-U** **ユーザ名** のようにすると、そのユーザが動かしているプロセスの一覧が表示されます。一番右に表示される文字列が実行されているコマンドの名前で、一番左の数字がそれぞれのプロセスに与えられているプロセス ID といわれる番号です。UNIX で同時に動いている多くのプログラムは、このプロセス ID という番号で識別されます。ps a とした場合、全ユーザが動かしているプロセスの一覧を表示できます。

他にも、ps l とすると、プロセスについてのより詳しい情報が得られます。ps u とすると、プロセスの実行開始時間を同時に得ることができます。そして、ps x とすると、端末の直接の制御下でないプロセスも表示することができます。

2.10.2 kill コマンド

主に、プロセスを強制終了させるために使われるコマンドです。kill `プロセス ID` と実行すると、そのプロセスを強制終了します。これで終了できないプロセスには、kill `-KILL` `プロセス ID` が有効かもしれません。

```
$ ps↵
  PID TTY          TIME CMD
 5978 pts/1        00:00:00 bash
29621 pts/1        00:00:00 sleep
29522 pts/1        00:00:00 ps
$ kill 29621↵
$ ps↵
  PID TTY          TIME CMD
 5978 pts/8        00:00:00 bash
29522 pts/8        00:00:00 ps
$ █
```

2.11 コマンドを知るためのコマンド

2.11.1 man コマンド

本手引きに書かれている説明はごく簡単かつ基本的なもので、細かいコマンドまでは書かれていません。疑問点などがあったり、使用法を忘れてしまった場合や、そのコマンドの詳しい使い方や仕様を知りたいときのために、man というコマンドが用意されています。これは、

```
$ man ls↵
```

のように用います。これを実行すると、そのコマンドのマニュアルが画面に表示されます。このマニュアルは、表 2.6 のように章が分けてあります。同じ名前の別のマニュアルが、2 つ以上の章に入っていることがあ

表 2.6: 章建て

1 章	コマンド
2 章	システムコール
3 章	ライブラリ関数
4 章	デバイスファイル
5 章	ファイルフォーマット
6 章	ゲーム
7 章	その他
8 章	システム管理コマンド
9 章	カーネルルーチン
10 章	その他 (2)

ります。その場合は、`man` `章番号` `コマンド名` のように、章を指定して、自分の見たいマニュアルを見て下さい*18。

また、`man` `-k` `キーワード` のようにすると、キーワードに関係のあるマニュアル (manual) の項目名が表示されます。`man` コマンドのより詳しい使用法は、

```
$ man man↵
```

のように入力すれば、見ることができます。UNIX を使いこなすためには、*man* を見る習慣をつけることが大切です*19。

2.11.2 info コマンド

`info` コマンドでもコマンドについての説明を見ることができます。一部のコマンドの `man` コマンドの説明では、`info` コマンドも参照することを指示される場合があるのでそういった場合に `info` コマンドを使うと良いでしょう。なお、後述する Emacs の `info` コマンドとは別物です。

```
$ info ls↵
```

のように用います。

2.12 ワイルドカード

ワイルドカード*20は様々なパターンにマッチし、適宜置換されます。似たような名前のファイルや特定の規則に基づいた名前のファイルを扱うときに便利です。

```
$ ls↵
a.c    b.c    c.c    d.c    a.java  b.java  c.java  d.java  report.tex
$ ls ?.c↵
a.c    b.c    c.c    d.c
$ ls a.*↵
a.c    a.java
$ ls [ab].c↵
a.c    b.c
$ ls [^ab].java↵
c.java  d.java
$ ls [^ab].*↵
c.c    d.c    c.java  d.java
$ ls [c-z]*.[d-z]*↵
d.java  report.tex
$ █
```

*18 マニュアル内でよくある、「ls(1)」などのようなコマンド名のあとに括弧内に数字が書かれた表記は、括弧内の数字が章を表わしています (この場合は「1 章の ls」コマンド)。

*19 英語で画面が埋め尽されてうんざりすることがあると思いますが、そう難しい英語ではないので頑張って読みましょう。

*20 正式には“ファイル名の置き換え”，あるいは“パス名の展開”ですが，一般に“ワイルドカード”という名称が浸透しています。

- `*`
0文字以上の任意の文字列にマッチします。
- `?`
任意の1文字にマッチします。
- `[pat]`
`pat`の中のいずれかの1文字にマッチします。‘`^`’を`pat`の先頭につけることでその補集合にマッチします。
`pat`には他にも特殊な表現を用いることができます。

表 2.7: `pat` の例

<code>abc</code>	a, b, c のいずれか
<code>a-c</code>	上記と同じ
<code>a-z</code>	小文字のアルファベット 1 文字
<code>A-Z</code>	大文字のアルファベット 1 文字
<code>0-9</code>	数字 1 文字
<code>a-zA-Z</code>	大小を問わずアルファベット 1 文字
<code>0-9a-zA-Z</code>	全ての半角英数字
<code>^0-9a-zA-Z</code>	全ての半角英数字以外

2.13 `tab`キーによる補完

入力の途中などに`tab`キーを押すことで、コマンド名やファイル名などを自動で補完してくれます。

例えば`gatherheaderdoc`というコマンドを実行しようと“gather”まで入力したが、それ以降を覚えていなかった場合、`tab`キーを押してみます。

```
$ gathertab
$ gatherheaderdoc█
```

残りの部分が補完されました。

また、ファイル名の補完もできます。`ls b`まで入力して`tab`キーを押すと、`b`から始まるファイル名が候補として出力されます。

```
$ ls btab
bin/          bitbucket/  bookmarks/  b.xx
$ █
```

候補が一意に定まらない場合、候補を出力するだけにとどまります。

2.14 リモートログイン (remote login)

ネットワークに接続された他の計算機の資源を利用することができます。

2.14.1 ssh コマンド

ネットワークに接続された他の計算機の資源を利用することができます。これをリモートログインといいます。詳しい説明や COINS 計算機に学外から接続する場合の注意点は、9.1.3 を参照してください。ここでは、リモートログインに関するコマンドの使い方についてのみ説明します。

ssh は他の計算機にログインするためのコマンドです。書式は、ssh ログインしたい計算機名 です。初回接続時には Are you sure you want to continue connecting (yes/no)? というメッセージが表示されます。https://www.coins.tsukuba.ac.jp/ce/pukiwiki.php 以下にある coins システムの主なホストの公開鍵のハッシュ値一覧からログイン先のハッシュ値（MD5 値もしくは SHA256 値）があるかどうか確認し、問題がなければ yes と入力します。

ssh 以後のコマンド実行などは、ssh した先の計算機で行われます。ログアウトすれば、元の計算機での作業に戻ります。

```
$ ssh crocus01.coins.tsukuba.ac.jp*21↵
The authenticity of host 'crocus01.coins.tsukuba.ac.jp (130.158.86.193)' can't be
established.
RSA key fingerprint is 2e:de:68:32:7a:86:bc:65:f5:ba:37:04:be:6f:e1:35. ←MD5値
Are you sure you want to continue connecting (yes/no)? yes
Password:パスワードを入力する。これは表示されない
crocus01:~ $ █
crocus01:~ $ logout↵
Connection to crocus01 closed.
↑ 接続を切ったというシステムからのメッセージ。
$ █← 元の計算機に戻った。
```

ssh コマンドに -l ログイン名 というオプションを付けるか、計算機のアドレスの前にユーザ名と@をつけると、指定したログイン名のユーザとしてログインできます。たとえば、学内の Wi-Fi 等からログインする場合はこのようになります。

```
$ ssh s1411xxx@crocus01.coins.tsukuba.ac.jp
```

2.15 コマンドをさらに使いこなす

2.15.1 標準入力、標準出力

通常、シェル上で実行するコマンドの入力はキーボードから行い、ディスプレイに出力されます。これをそれぞれ標準入力、標準出力といいます。

^{*21} 情報科学類の iMac の macOS にインストールされている ssh はバージョンが古く、MD5 値が標準で出力されますが、最近の ssh コマンドは SHA256 で表示されます。上記 URL には両方記載されているので、自分のクライアントに表示された方と合っているか確認してください。これにより ~/.ssh/known_hosts にログイン先の情報が保存され、次回以降はこのメッセージが表示されなくなるので、上記のオプションは必要なくなります。

リダイレクト

リダイレクトとは、標準入力先または標準出力先を変更する機能です。記号 `>` で、標準出力を指定したファイルに切り替えて、指定したプログラムを実行します。記号 `<` で、標準入力を指定したファイルに切り替えて、指定したプログラムを実行します。

```
$ ls -l > file1↵
$ cat file1↵
total 9
drwxr-xr-x  2 johotaro ugrad  4096  2 15 00:10 dir1
-rw-r--r--  1 johotaro ugrad     0  2 17 01:13 file1
-rw-r--r--  1 johotaro ugrad    12  2 15 00:10 file2
$ █
```

この例では、`ls -l`の実行結果を、画面ではなく `file1` に出力しています。

パイプ

パイプとは、複数のプログラムを同時に実行したり、プログラムの標準出力を、次に実行するプログラムの標準入力につなぐ機能です。記号 `|` で複数のコマンドをつないで使います。左側に記述したコマンドが先に実行されます。

```
$ ls -l | head -n 3↵
total 9
drwxr-xr-x  2 johotaro ugrad  4096  2 15 00:10 dir1
-rw-r--r--  1 johotaro ugrad     0  2 17 01:13 file1
$ █
```

`ls -l`はファイル・ディレクトリの情報を出力するコマンドで、`head -n 3`は入力されたデータのうち、先頭3行を表示するコマンドです。`ls -l`の出力先を `head -n 3` の入力先につなぐことにより、ファイル・ディレクトリの情報のうち、先頭3行を表示しています。

2.16 その他有用なコマンド

2.16.1 alias コマンド

このコマンドを用いると、良く使うコマンドやオプションに別な名前を付けることが出来ます。`alias` `自分で付けたい名前` = '`コマンド列`' で定義します。

```
$ ls↵
dir1  file1  file2
$ alias lls='ls -l -F'↵
$ lls↵
total 245
drwxr-xr-x  2 johotaro ugrad    48  1 25 00:45 dir1
```

```
-rw-rw-rw-  1 johotaro ugrad      7377  1 25 00:44 file1
-rw-----  1 johotaro ugrad    239914  1 25 00:44 file2
$ █
```

決まったオプションでしか使わないコマンドや名前の長いコマンドなどを alias で別名にすることで、使いやすくなります。ただし、iTerm 上で設定したエイリアスは、1 度 iTerm を閉じると消えてしまいます。

より高度な技術を身につけたい人は、シェルとは何か、また、.bashrc や.zshrc とはどのようなファイルかを学ぶと、ターミナルを開くたびにこのエイリアスをロードするようにできるでしょう。このように、いくつかのヒントを元に、自分の環境をカスタマイズしていくと、少しずつ上達していきます。たとえば、ホームディレクトリにある.bashrc というファイルにコマンドを書き加えておくと、そのコマンドは iTerm でウインドウを開く度に実行されます、といったことから、自分で環境をカスタマイズするヒントを得られるようになるでしょう。

以下に、便利な alias の例を示します。

```
alias ll='ls -l'
alias lla='ls -al'
alias lat='ls -tal'
alias lt='ls -tl'
alias la='ls -A'
alias l='ls -CF'
alias u='cd ..'
alias uu='cd ../../'
alias uuu='cd ../../../../'
```

2.16.2 nkf コマンド

文字コードと改行コードを変換するコマンドです。Windows などと相互にファイルのやりとりをすると、時々 UNIX から持っていったテキストファイルが読めないことがあります。または自分で作った Web ページが文字化けしてしまって正しく表示されなかったりすることがあります。そういうときにテキストファイルの文字コード・改行コードを変換してくれるのが nkf コマンドです。

書式は **nkf** オプション 変換前ファイル名 > 出力先ファイル名^{*22}となります。オプションは、出力したい文字コードによって-e (日本語 EUC)、-j (JIS コード)、-s (シフト JIS コード)、-w (UTF-8) のどれかから選び、改行コードも-Lu (UNIX)、-Lw (Windows)、-Lm (Mac OS 9 以前の Macintosh) の中から選びます。以下は、Windows で作成したテキストファイルを UTF-8 に変換する例です。

^{*23}

```
$ ls<␣
file1
$ locale<␣
LANG="ja_JP.UTF-8"
LC_COLLATE="ja_JP.UTF-8"
```

^{*22} この2つのファイル名を同じにしまうとそのファイルが消えてしまうので注意してください。

^{*23} 岡本かの子著: 岡本かの子全集5 ちくま文庫 筑摩書房 1993年「快走」<http://www.aozora.gr.jp/cards/000076/card50618.html>

```
LC_CTYPE="ja_JP.UTF-8"
LC_MESSAGES="ja_JP.UTF-8"
LC_MONETARY="ja_JP.UTF-8"
LC_NUMERIC="ja_JP.UTF-8"
LC_TIME="ja_JP.UTF-8"
LC_ALL=

$ cat file1↵
±^⌘«オ   ç† Á©¯   ε½ e⌘'⌘d$   Áey ©ᵛ çh¨   ā   ϕ   ò   ^ ]   B
u       A       v
Qe⌘ w³⌘⌘   ^açãUè   ÁāA ,   Á⌘ āêç   εé⌘ à†©⌘   ^Á½ B⌘⌘A⌘⌘¾ϕ⌘   ^ç
u⌘½   Á½ çA   éᵀN^⌘α   éñ┘l·   à´   A†ϕ¯⌘   ´   ñív
Q⌘^±´¾   tç v⌘¾⌘⌘⌘   è
u» ê⌘⌘ Āà¨   0⌘ xϕ±⌘   Á½ çv
Q⌘⌘$ç Á

$ nkf -w file1 > file2↵
$ ls↵
file1 file2

$ cat file2↵
このとき後から追っかけて来た父親は草原の中に立って遥かに堤防の上を白い塊が飛ぶのを望んだ。
「あれだ、あれだ」
父親は指さしながら後を振り返って、ずっと後れて駈けて来る妻をもどかしがった。妻は、はあはあ言いながら
「あなたったら、まるで青年のように走るんですもの、追いつけやしませんわ」
妻のこの言葉に夫は得意になり
「それにしてもお前の遅いことったら」
妻は息をついで
「それでも一生懸命だもんで、家からここまで一度も休まずに駈けて来たんですからね」
「俺達は案外まだ若いんだね」
「おほほほほほほほほほ」
「あはははははははははは」
二人は月光の下を寒風を切って走ったことが近來にない喜びだった。
二人は娘のことも忘れて、声を立てて笑い合った。
$ █
```

基本的に UNIX では自動的に文字コードは判別してくれるので、普段あまり用は無いかもしれませんが、他のシステムとファイルのやりとりをする場合は注意が必要です。

2.16.3 xclock, xcalc コマンド

それぞれ時計^{*24}，電卓です.

2.17 コマンドリスト

今紹介したものを含め、UNIX にはたくさんのコマンドがあります。それらを表 2.8 にまとめました。各コマンドの詳細は、man コマンドを用いるなどして調べてください。

*²⁴ 時々狂っている時があるかもしれませんがその時は管理者まで連絡をお願いします。

表 2.8: コマンドリスト

コマンドの機能	コマンド名
(ファイルシステム関連) ディレクトリを移動します。 ファイルのパーミッションを変更します。 ファイルのコピーをします。 ディスク使用量を表示します。 ファイルのリンクをします。 ディレクトリにあるファイル名を表示します。 ディレクトリを作成します。 ディレクトリを削除します。 ファイルの移動、ファイル名の変更をします。 ファイルを削除します。 カレントディレクトリを表示します。 ファイルの圧縮・展開をします	cd chmod cp du ln ls mkdir rmdir mv rm pwd tar
(一般的な作業コマンド) テキストファイルの内容を表示します。 日付・時刻を表示します。 カレンダーを表示します。 他の計算機でコマンドを実行します。 ネットワーク上のホスト間でファイルをコピーします。 プログラムの実行時間を計測します。 ウィンドウに表示されている画像をファイルに保存します。 グラフを書きます。 図を書きます。 絵を書きます。 画像形式の変換や、拡大縮小などの編集をします。	cat, more, lv date cal ssh scp time xwd gnuplot tgif xpaint, gimp convert, sips
(「フィルタ」として使われるもの) ファイル内の、指定した文字列を含む行を抜き出します。 ファイルの内容を、先頭から指定した行数だけ表示します。 ファイルの内容を、末尾から指定した行数だけ表示します。 ファイルの内容を、行単位でソートして表示します。	grep, egrep head tail sort
(プロセスに関するもの) プロセスにシグナルを送ります。 計算機で動いているプロセスの情報を表示します。	kill ps, top, pstree
(プリント関連のコマンド) 印刷をします。 プリンタへの印刷の要求を取り消します。	lp cancel

表 2.8: コマンドリスト

コマンドの機能	コマンド名
プリンタへ登録されているジョブの確認をします.	lpstat
(その他のコマンド) その計算機にログインしている人を表示します. 自分のユーザ名を表示します. 指定したユーザのユーザ ID やグループ ID などを表示します. 指定したユーザのグループ ID を表示します. その計算機で誰が何をしているかを調べます. ファイルやディレクトリの所属グループを変更します. システムのアーキテクチャを表示します.	who, finger whoami id groups w, last chgrp arch

第 3 章

Emacs

本章では UNIX や Linux における多機能エディタ **Emacs** (Editor MACroS の略称) の説明をします。マニュアルは公式の物は GNU Emacs Manuals Online^{*1}があり、非公式で日本語翻訳^{*2}が 20.6 対応と少し古いですがが参考になるかもしれません。エディタ (*editor*) とはファイルを編集するツールです。画面上にファイル内容を表示し、その内容を編集することができます。プログラムのソースコードやテキスト文書等を書く場合にはエディタもしくはエディタに似た機能を持ったソフトウェアが必要となります。

Emacs はそのマニュアル (The GNU Emacs Manual) 内で Emacs 自身を”*the extensible, customizable, self-documenting, real-time display editor*” (拡張、カスタマイズ可能で、自己説明的でリアルタイム表示を行うエディタ) と説明しています。この説明どおりに Emacs はプログラムの記述をサポートするように拡張できる、など拡張性に非常に富んでいます^{*3}。

ここでは、Emacs の基礎知識と基本的な操作を解説しています。

3.1 Emacs 基礎知識

本節では Emacs について知っておくべき、また Emacs についてわからない事があった時のヘルプの使い方などについて記述します。Emacs を使用する際や、本章を読み進める際にもしもわからない事があった場合、この節を読みに戻って来てください。

3.1.1 入力表記

Emacs は歴史的経緯により入力するキーの表記が独特な表記となっています。初心者の方にはわかりにくいでしょうが、Emacs のドキュメントや資料は初心者向けを除いた物の大半が以下のような表記になっているためこの章はこの表記で統一されています。

Meta キー

Meta キーについて少し解説が必要かもしれません。*Meta* キーというのは Emacs が作られて間もない頃に使われていた MIT のシンボリック製 LISP マシン用キーボード^{*4}にあった特別な修飾キーです。昔にあった

^{*1} <http://www.gnu.org/software/emacs/manual/>

^{*2} <http://www.geocities.co.jp/SiliconValley-Bay/9285/EMACS-JA/emacs.html>

^{*3} 事実、この章はカスタマイズされた Emacs で書かれています

^{*4} スペースデカットキーボードとも知られています

表 3.1: Emacs のキー表記

表記	キー操作
C-c	ctrl を押しながら c を押す
M-x	<i>Meta</i> キー, 代替キーとして ⌘ (option キー) を押しながら x を押す
ESC	Esc (<i>Escape</i> キー) を押す. C-] でも同じ
ESC x	Esc を押した後 x を押す (押しながらではありません)
SPC	Space
TAB	→ (<i>Tab</i> キー) を押す. C-i でも同じ
RET	↵ (<i>Return</i> キー) を押す. C-m でも同じ
DEL	⌫ (<i>Delete</i> キー) を押す. ↵ の上にある ⌫ のことです

コンピューターやキーボードの名残とってください. 今では *Meta* キーの代わりとして, **Alt**や**ESC**などのキーが使われています. 例えば**M-x**は**ESC x**で置き換えられるなど, **ESC**は *Meta* キーの代わりになる事を覚えておくとよいでしょう.

3.1.2 コマンドの中断

始める前に, 何かがあった時のためにコマンドの中断方法を覚えておきましょう. 間違ったコマンドを選択してしまったりすると, Emacs が自分の思い通りに動いてくれなくなることがあります. なんか変だぞと思ったら, まずは**C-g**を押してみましょう. これはコマンドの中断を行う操作で, 大抵の場合はこれでなんとかなります. 1回押してもまだだめなときは, 何回か押してみてください.

困ったときは **C-g**.

3.1.3 ヘルプ

C-h ?とタイプしてみましょう. Emacs が持っているヘルプ関連のコマンドの全てが, ヘルプバッファ*5が出てきて表示されます. (**C-g**で抜けます). 数多くのヘルプがありますが, 一部を除いてほとんどが英語で出力されます. そのうちで, ヘルプコマンドのヘルプはすでに説明しました. これ以外に特に役立つものを説明します.

モード **C-h m**. このコマンドは現在の主モードの簡単な解説とそのバッファで設定されているキーバインドの解説がヘルプバッファに表示されます.

キー割り付け **C-h b**. キーに割り当てられているコマンドの一覧がヘルプバッファに表示されます.

コマンドを見つける **C-h a**. ここで, 文字列を入力すれば, その文字列を含むコマンド全ての内容がヘルプバッファに表示されます.

キーに割り当てられているコマンドの表示 **C-h c**. ここで知りたいキーを入力すると, それに割り当てられているコマンドが表示されます.

*5 ヘルプ情報を表示するバッファ

3.1.4 チュートリアル

Emacs を初めて使う人は、Emacs の独特なキーバインドなどがとても難しい物に見えるでしょう。Emacs には、初心者が実際に Emacs を操作しながら学習するためのチュートリアルが用意されています。Emacs を使おうと思っている人ならばこの章の解説に留まらず、行ってほしいのがこのチュートリアルです。

Emacs を起動して「**M-x** help-with-tutorial」とタイプしてください。Emacs 入門ガイドが表示され、チュートリアルが始まります。画面の指示に従って、自分でキーボードを操作してください。チュートリアルが終わる頃には、Emacs のキー操作の基礎が身についていることでしょう。しかしひと通り行うには時間がかかりますので、時間がある時に余裕を持って行うと良いでしょう。

— **C-SPC** で Spotlight 検索が起動してしまう場合 —

現在、Spotlight 検索はデフォルトで **⌘-SPC** に割り当てられていますが、古い環境設定が引き継がれている場合に **C-SPC** で Spotlight 検索が起動してしまうことがあります。これを無効にするためには以下の操作を行います。

1. Dock から「システム環境設定」をクリックする。
2. 出てきたウィンドウで、「キーボード」を選ぶ。
3. 出てきたウィンドウの「ショートカット」タブを選ぶ。
4. 左側のボックスで「Spotlight」を選ぶ。
5. 右側のボックスの「Spotlight 検索を表示」のチェックを外す。

3.2 実行と終了

3.2.1 実行



図 3.1: Dock 内の Emacs

デスクトップの Dock にある, Emacs のアイコンをクリックすることで, Emacs を起動できます。(図 3.1) この方法で起動すると図 3.2 のような画面が現れます。

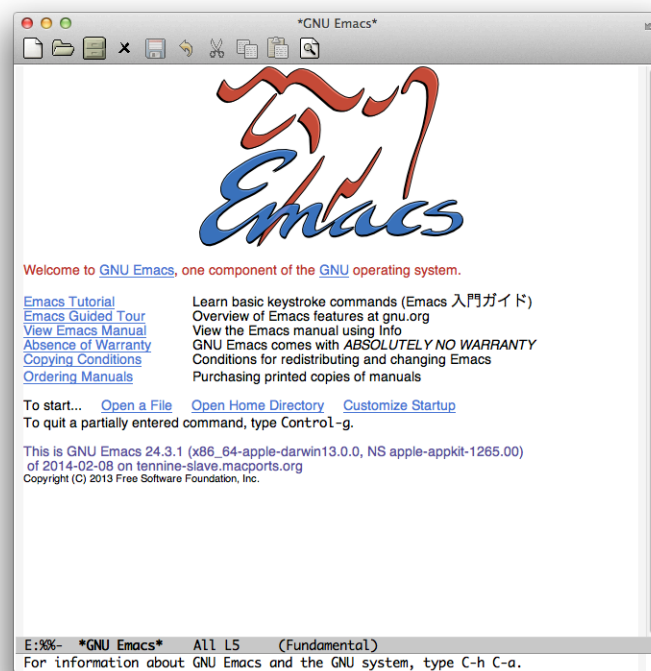


図 3.2: Emacs GUI 画面

また, 下記のようにターミナルで入力すると, ウィンドウを開かずに Emacs を実行できます。

```
emacs -nw
```

ここで渡している-nw ですがこれは-no-window という意味で、ウィンドウを開かずに Emacs を起動するという意味です。この方法で起動すると図 3.3 のような画面が現れます。

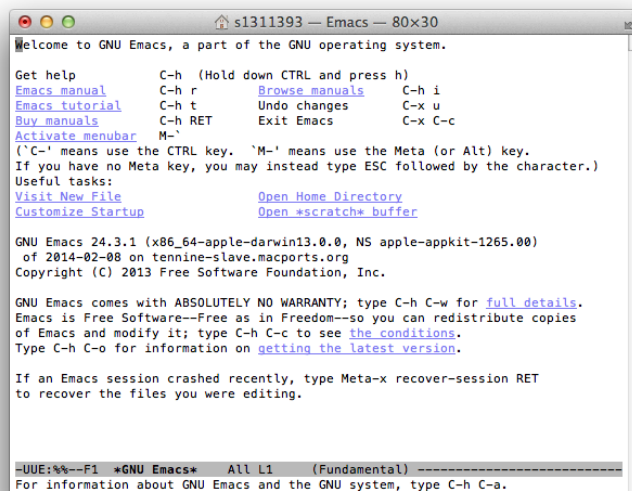


図 3.3: Emacs CUI 画面

起動直後の画面は全然違いますが、実際に操作をするときの方法はどちらとも同じです。なので、画面の説明は図 3.3 を主に行います。まず一番上ですが、これはタイトルバー (*title bar*) と呼ばれており、ウィンドウの名前（ここでは*GNU Emacs*）が表示されています。ウィンドウの右端にある縦長の長方形の部分は、スクロールバー (*scroll bar*) といいます。マウス操作でスクロールさせたいときに使います。下から 2 行目の色が反転している行をモードライン (*mode line*) といい、編集集中の状態（ファイル名など）が表示されています。最下行はエコーエリア (*echo area*) といい、Emacs がメッセージを表示したりします。ちなみに、GUI で起動した場合だとタイトルバーの下にバーがあります、これをツールバー (*tool bar*) といいます。

3.2.2 終了

キーボードから Emacs を終了させる場合には

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

と操作します。3.1.1 項のおさらいになりますがこれは、**ctrl** を押しながら **x** を押し、その後 **ctrl** を押しながら **c** を押すという意味です。

さて、Emacs のウィンドウが消えましたか。もし消えない場合は、エコーエリアを見てください。編集集中で保存していないファイルがあるときには、保存するかどうか確認のメッセージが表示されています。保存する場合には「y」キーを押してください。

3.3 編集

本節では Emacs における文書の編集方法を紹介します。ここでは、基本的なことを紹介します。

3.3.1 文字の入力

Emacs ではキーボードから打った文字はそのまま入力されます。「hello emacs」とタイプすると Emacs には次のように表示されます。

```
hello emacs
```

通常は挿入するようになっており*⁶カーソル位置に既に文字がある場合には上書きされずに挿入されます。

```
at
```

a と t の間にカーソルを合わせて bou と押す

```
about
```

3.3.2 ファイル操作

作成した文書は、ファイルに保存しないと終了時に消えてしまいます*⁷。ここでは読み書き等のファイル操作について説明します。

ファイルを読み込むには `C-x C-f` を使います。 `C-x C-f` と押すとエコーエリアに

```
Find File:~/
```

のように表示され、入力待ちの状態になります。ここでは、通常の編集コマンドのほとんど（カーソル移動、文字の挿入・削除等）が使えます。もし存在しないファイル名を入力した場合にもエラーにはならず、新規にファイルを作成するものとみなされます。

読み込んだファイルを編集後、現在の編集内容を保存するには、 `C-x C-s` を押します。

これまでに示したファイル編集の流れをまとめると、次のようになります。

1. Emacs を起動する
2. `C-x C-f` でファイルを指定する
3. 編集する
4. `C-x C-s` でファイルを保存する
5. `C-x C-c` で Emacs を終了する

「emacs foo.txt」のように編集したいファイル名を付けて Emacs を起動すれば、起動した後に「`C-x C-f` foo.txt」でファイルを読み込んだのと同じになります。

scratch バッファ*⁸のようにファイルと結び付いていないバッファの内容をファイルに保存する時や、編集集中のファイルとは違う名前のファイルに保存したい時は、 `C-x C-w` を使用します。

また、カーソル位置に指定したファイルの内容を挿入するコマンド `C-x i` もあります。

*⁶ `M-x`, `overwrite-mode` で上書きモードを適用でき、上書きモードのときはモードラインに `Ovwr` と表示されます。もう一度 `M-x` `overwrite-mode` と入力すると上書きモードを解除できます。

*⁷ 逆に言えばファイルに保存さえしなければ、何を書いても終了時に消えてくれます。

*⁸ 試し書きや EmacsLisp の確認などに使うバッファ

これらのコマンド等でファイル名を入力する際には、補完機能 (*completion*) を使うことができます。これはファイル名の最初の一部分を入力した時点で **TAB** を押すと、残りの入力を Emacs が補完して確定してくれる機能です。このとき、ファイル名の候補が複数ある場合は一致している部分だけを補完し、候補のリストを表示します。

Emacs ではファイル名の入力以外にも、**M-x** による関数名の入力など、さまざまな局面で補完機能を提供しています^{*9}。とても便利な機能なのでぜひ利用してください^{*10}。というよりも、タイプするキーの数も入力ミスも減って楽になるので、作業効率のためにも積極的に使ってみてください。

自動セーブ

Emacs では、編集中に計算機がダウンした場合などに備え、定期的にファイル名の前後に # を付加したバックアップファイルにファイルを保存（自動セーブ）する機能を備えています。実際に編集中のファイル (test.txt) があるディレクトリをみると、次のようになっています。

```
$ ls -a
.      .#test.txt  ..      test.txt
$
```

このファイルはファイルの保存が正常に行われた時点で削除されます。Emacs でファイルを読み込むときに自動セーブしたファイルが残っていれば、次のような警告がエコーエリアに表示されます。

```
Auto save file is newer; consider M-x recover-file
```

この場合、「**M-x** recover-file」を入力して **C-x C-f** の場合と同様にファイルを指定すれば、バックアップファイルからファイルを復元します。直接「**C-x C-f**」でバックアップファイルを指定して読み込むと、Emacs の内部コードで保存されているために読めないことがあります。必ず「**M-x** recover-file」を使いましょう。

3.3.3 カーソルの移動

本節では Emacs 上でのカーソルの動かし方を説明します。カーソルの動かし方、と聞いて矢印キーでいいじゃないか、と思った人はホームポジションの位置から矢印キーに指を動かしてみてください。右手全体を動かさないと届きませんよね。右手を動かさずに移動できたらいいと思いませんか。また、文の最後に行くために文字数の分だけキーを叩いていませんか。キーを二つ押すだけで文頭や文末に移動できたらいいと思いませんか。Emacs ではこれらが実現できます。カーソルの操作方法は表 3.2 の通りになります。

3.3.4 文字・行の削除

入力した文字の消去には、ふたつのコマンドがあります。カーソル位置の右側の文字を消す **C-d** と、カーソル位置の左側の文字を消す **DEL** です。

- **C-d** による文字の消去

```
about
```

^{*9} プログラムのソースファイルを書いているときに、長い変数名の先頭何文字かをタイプしたところで **M-/** を押してみたりしてください

^{*10} UNIX のコマンドラインでも、入力時に補完機能の使えるシェルがあります。みなさんが標準で使っている *bash* の場合、補完するには Emacs と同様に **TAB** を使います。

表 3.2: 移動コマンド

キー	等価なキー	機能
<code>C-n</code>	<code>↓</code>	カーソルを次行に移動 (next-line)
<code>C-p</code>	<code>↑</code>	カーソルを前行に移動 (previous-line)
<code>C-f</code>	<code>→</code>	カーソルを右に移動 (forward-char)
<code>C-b</code>	<code>←</code>	カーソルを左に移動 (backward-char)
<code>C-a</code>	<code>home</code>	カーソルを行頭に移動
<code>C-e</code>	<code>end</code>	カーソルを行末に移動
<code>C-v</code>	<code>pagedown</code>	カーソルを次ページに移動
<code>M-v</code>	<code>pageup</code>	カーソルを前ページに移動
<code>M-<</code>	<code>C-home</code>	カーソルをファイルの先頭に移動
<code>M-></code>	<code>C-end</code>	カーソルをファイルの最後に移動
<code>M-x</code> goto-line	<code>M-g g</code>	カーソルを行番号で指定した行に移動
<code>C-l</code>		カーソルのある行が画面中央にくるよう画面を書き直す

下の o の位置で `C-d` を押す

```
abut
```

- `DEL` による文字の消去

```
about
```

下の b の位置で `DEL` を押す

```
aout
```

Emacs では「改行」も文字として扱われるので、行末で `C-d` を押すか行頭で `DEL` を押せば行を連結することもできます。

1 行消去するには `C-k` を使います。

```
goto fail;
goto fail;
goto fail;
```

二行めの最初の g にカーソルを置き `C-k` を押す

```
goto fail;

goto fail;
```

このようにカーソルがある場所から行末までが削除されました。なお、`C-k` で消去された行は記憶されています。これを `C-y` でペースト (*paste*) する（貼り付ける）ことができます^{*11}。

^{*11} ちなみにですが、この削除とペーストという言い方は Emacs の文化としては一般的ではありません。本当はキル (*Kill*)。とヤंक (*Yank*) と呼びます。

では今度はペーストしてみます。

```
goto fail;

goto fail;
```

最後の行の末尾で改行し **C-y** を押す

```
goto fail;

goto fail;
goto fail;
```

本節のコマンドをまとめると表 3.3 のようになります。

表 3.3: 削除関連コマンド

コマンド	機能
C-d	カーソル位置の文字を削除
DEL	カーソルの左側の文字を削除
C-k	カーソル位置から行末までを削除し、削除リングに記憶
C-y	削除リングの内容をカーソル位置に挿入

3.3.5 コピー，カット，ペースト

文書の一部を消して他の場所に移動させることをカット&ペースト (*cut & paste*) といいます。 **C-k** と **C-y** を組み合わせるとカット&ペーストができることは既に示しました。次に任意の文字列に対するカット&ペーストを説明しましょう。

Emacs では、 **C-SPC** または **C-@** を使ってバッファの好きな場所に印 (マーク (*mark*)) を付けることができます*12。マークは **C-SPC** を押した所のカーソルにつきます。マークは目には見えませんが、Emacs はその場所を覚えています。次にカーソルを適当な位置に移動します。マークとカーソルの間をリージョン (*region*) といいます。例えば次の図で文字 L の上で **C-SPC** を押した後 (下線で印をつけています)，文字 A の所にカーソルを移動したとします。

```
LispAlien
```

この場合、Lisp がリージョンとなります。A は含まれません。リージョンは **C-w** でカットすることができます。カットした文字列は削除リング (*kill ring*) と呼ばれる領域に記憶されており、 **C-y** で任意の場所にペーストすることができます。

これに対して、文書の一部を消さずに他の場所へ写すことをコピー&ペースト (*copy & paste*) といいます。 **M-w** はリージョンを消さずに削除リングに記憶するコマンドです。これも **C-y** でペーストすることができます。

1. **C-SPC** または **C-@** でリージョンの最初または最後をマーク

*12 もしも SPC の意味がわからないなどの場合は 3.1.1 節を読み直してください。

2. リージョンの最後または最初にカーソルを移動
3. 移動の場合は **C-w**，コピーの場合は **M-w** を使って，リージョンを削除リングに記憶
4. 移動またはコピー先にカーソルを移動して， **C-y** でペースト

本節で使用したコマンドをまとめると表 3.4 のようになります。

表 3.4: コピーアンドペーストコマンド

コマンド	機能
C-SPC または C-@	マークをセット
C-w	リージョンをカットし，削除リングに記憶
M-w	リージョンを削除リングに記憶
C-x C-x	カーソル位置とマークを交換

3.3.6 編集の取り消し

Emacs には，誤って文字を削除した場合などに備えて，編集を取り消す機能（表 3.5）があります。これは **C-x u** または **C-_** を使います。何度も押せば，押した回数分だけさかのぼって編集を取り消すことができます。アンドゥ (Undo) や「元に戻す」といえばピンとくる人もいるでしょう。

表 3.5: 困った時のコマンド

コマンド	機能
C-g	コマンドの中断
C-x u または C-_	最後の（直前の）変更を取り消す

3.3.7 検索・置換

検索

検索 (*search*) とは，探したい文字列にカーソルを移動することです。Emacs では一般的にインクリメンタルサーチ (*incremental search*) という方法で文字列を検索します。インクリメンタルサーチでは，検索対象の文字をタイプするたびに，その次点で入力された文字列と一致する文字列のある場所へカーソルが移動します。つまり，入力するたびに一文字ずつ検索していき結果を絞り込んで行く訳です。一般的な検索と違い，検索したい文字列を全部打ち込んでから検索するわけではありません。

下のほうへ検索するには **C-s**，上のほうへ戻りながら検索するには **C-r** を使います。 **C-s** を押すと，エコーエリアに

```
I-search:
```

と表示されます。ここで検索したい文字列をタイプすると，一致する文字列のある場所へカーソルを移動させることができます。検索中 **C-s** を繰り返し押すと，次々に一致する文字列のある場所へカーソルが移動しま

す*¹³。

検索文字列の入力を誤った場合は`DEL`を押すと最後の入力を取り消し、カーソルは直前の位置へ戻ります。

検索を終了するには`RET`を押します*¹⁴また、`C-g`を押して検索を中止した場合には、カーソルは検索を開始する直前にあった位置に戻ります。

文字列を一度に指定して検索したい場合は、`C-s`(`C-r`)を押した直後に`RET`を押します。これによってエコーエリアの表示が

```
I-search:
```

から

```
Search:
```

に変わり、検索したい文字列を一括して入力してから検索を始めるモードになります。文字列をタイプして`RET`を押すと文字列が検索できるようになり、この場合もさらに`C-s`(`C-r`)を押すことによって次の候補に移動することができます。

置換

置換 (*replace*) とは、ある文字列を他の文字列に置き換えることです。`M-%`を押すと、エコーエリアで置換前の文字列と置換後の文字列を尋ねてきます。それぞれ入力するとカーソル位置から検索を始め、検索文字列が見つかった場合そこにカーソルが移動し、入力待ちの状態になります。文字列を置き換える場合は`SPC`または`y`を、置き換えずに次の候補に移動する場合は`n`または`DEL`を押します。置換を終了するには`q`、中止するには`C-g`を押します。全て`y`の場合は`!`を押すと良いでしょう。

本節で使ったコマンドをまとめると表 3.6 のようになります。

表 3.6: 検索・置換コマンド

コマンド	機能
<code>C-s</code>	ファイルの終わりに向かってインクリメンタルサーチ
<code>C-r</code>	ファイルの始めに向かってインクリメンタルサーチ
<code>C-s RET</code>	ファイルの終わりに向かって一括入力型検索
<code>C-r RET</code>	ファイルの始めに向かって一括入力型検索
<code>M-%</code>	対話的に置換
<code>M-x</code> replace-string	一括型置換

3.3.8 編集のまとめ

本節では Emacs で用いる編集の基本的なコマンド等を紹介してきました。具体的には以下を紹介しました。

- 文字の入力、日本語の入力

*¹³ 標準では文字列の検索と置換は大文字と小文字を区別しません。これを区別したい場合は、`C-s`(`C-r`)とした後で`M-c`とすると大文字と小文字を区別するようになります（このときエコーエリアに `case sensitive` と表示されます）。もう一度 `M-c`とすると区別なくなります。

*¹⁴ `RET` がわからない場合 3.1.1 節をみてください

- カーソルの移動
- 文字・行の削除, コピー, カット, ペースト
- 編集の取り消し
- 検索・置換

これらを用いることで不自由なく文書作成はできるはずです。やりたいことが文書を書くことやごく簡単なプログラムを書くだけならばここまでをしっかり覚えれば十分ですが、最近のプログラミング言語や後に出てくる L^AT_EX (第6章参照) のような構造化文書を書くときにはエディタのサポートがほぼ必須とも言えるため、更に特殊な機能を覚えると良いでしょう。このようなちょっと特殊な機能については第3.5節にて紹介していきますので参考にしてください。

3.4 Mew

ここでは、Mew を使ってインターネットメールやネットニュースを読み書きする方法について説明します。

3.4.1 起動してみましょう

Emacs で、「`[M-x] mew`」とするか、

```
emacs -f mew<␣
```

のようにコマンドラインで入力すると、Mew が起動します。

初回起動時には、次のように入力を求められます。

```
Type 'C-uZ' to collect IMAP folders!
```

指示にしたがって `[C-u Z]` と入力しましょう。

また、以下のようにメールを受信するためのパスワードを訊かれるかもしれません。

```
IMAP password (s9912345@viola6.coins.tsukuba.ac.jp):
```

のようにエコーエリアに表示された場合、ログインパスワードを入れてください。

Mew を終了するには、`[q]` を押します。

3.4.2 メールを読む

Mew を起動したときに、あなた宛てに届いていたメールは自動的に取り込まれます。その後届いたメールは、Summary モードで `[i]` を押すと取り込むことができます。

`[SPC]` を押していくだけで、メッセージを順に読んでいくことができます。また、`[DEL]` で上にスクロールさせることができますし、`[RET]` や `[-]` で一行ずつスクロールさせることもできます。その他、表 3.7 に示すようなコマンドを使ってページを操作することができます。

表 3.7: Summary モードでのキー操作

キー	動作
<code>SPC</code>	メッセージを読み進める.
<code>C-u SPC</code>	メッセージを先頭から再表示.
<code>DEL</code>	現在のメッセージを上スクロールさせる.
<code>RET</code>	現在のメッセージを一行下スクロールさせる.
<code>M-RET</code> または <code>-</code>	現在のメッセージを一行上スクロールさせる.
<code>C-p</code> , <code>C-n</code>	上, 下の行へ移動.
<code>p</code> , <code>n</code>	上, 下方向に移動し表示.
<code>C-u p</code> , <code>C-u n</code>	上, 下方向に移動し表示.
<code>j</code>	指定された行のメッセージへ移動.
<code>i</code>	新着メッセージを取り込む.
<code>g</code>	他のフォルダへ移動. (フォルダについては後述.)
<code>y</code>	本文または添付ファイルの保存.
<code>w</code>	メールを書く.
<code>a</code>	メールの返事を書く. (引用なし)
<code>A</code>	メールの返事を書く. (引用あり)
<code>d</code>	<code>D</code> マークをつける. (削除)
<code>o</code>	<code>o</code> マークをつける. (整頓)
<code>x</code>	削除や整頓の実行.
<code>D</code>	指定されたフォルダの内容を削除する.
<code>?</code>	条件に一致するメッセージに * マークをつける.
<code>C-u ?</code>	指定した文字列を含むメッセージに * マークをつける.
<code>m d</code>	* マークを全て D マークに変更する. (削除)
<code>m o</code>	* マークを全て o マークに変更する. (整頓)
<code>Q</code>	Mew の終了.

3.4.3 メールを書く

メールを書くにはいろいろな方法がありますが、簡単な方法は、Summary モードで `w` を押すことです。Write の `w` と覚えましょう。すると、以下のようなバッファが表示されるはずです。

```
To:
Subject:
From: s9912345@coins.tsukuba.ac.jp Fcc: %backup
X-Mailer: Mew version 6.5 on Emacs 24.3 / Mule 6.0 (HANACHIRUSATO)
----
```

これを Draft モードといいます。Draft モードにおいて、“----”より上をヘッダ、下を本文と呼びます。ここで、ヘッダの To: に続けて、送る相手のメールアドレスを正確に入れましょう。少しでも間違えると、絶

対に届きません。(もしくは、まったく違う人に届くかも知れません。)ここで、今までに書いたことのあるアドレスであれば`TAB`を使って補完することができます。次に、Subject: に続けてメールの題名を入れます。できるだけ「名が体を表した」題名をつけるようにしましょう。そうすれば、受け取った方は題名を見るだけで中身を容易に想像することができますし、あとで「あの用事のメールはどれだっけ?」と探すときにも便利です。

さて、ここまで入力し終わったら、カーソルを“---”より下の本文の所へ移動し、本文を書きましょう。メールの本文は、全角で35字くらいで改行するようにしましょう。こうすることで相手も自分も読みやすいですし、メールの文を引用して返信する際に便利です。(返信については後述します。)手動で改行するのが面倒な場合は、`M-q`を使いましょう。自動的に改行してくれます。

また、題名(Subject:)を日本語で書くと、受け取る人の環境によっては読めない場合があります。受け取る人の環境で読めるかどうかかわからない場合は、題名を英数字で書けば問題ありません。メッセージを書き上げたら、`M->`で文末に移動します。ここで、文末の無駄な改行があったら、`DEL`を押してそれを消してください。

ホームディレクトリの下に .signature というファイルを作っておくと、`C-c TAB`でそのファイルを挿入できます。signature は署名という意味で、このファイルには名前とメールアドレスなどを書いておき、文章の最後につけるとよいでしょう。ただし、4行ほどまでにしましょう。あまりに長い署名だと、迷惑がられます。

いよいよあとは送信するだけになったら、`C-c C-c`を押しましょう。すると、

```
Really send this message? (y or n)
```

と出るので、`y`を押すと、メールが送信されます。

うまくいきましたか?

なお、書いていたメールを送信せずに破棄するには、`C-c C-q`を使います。

メールの返事を書く

届いたメールに返事を書くには、そのメールにカーソルを合わせて`a`または`A`を入力します。Answerのaと覚えると良いでしょう。`a`だとメールの内容は何も引用されませんが、`A`だと各行の頭に>がついて^{*15}引用されます。おそらく`A`を利用することの方が多いでしょう。題名には、元のメールのSubjectの頭にRe: がついたものが自動的につけられます。なお、reは「レス」や“response”の略などではありません。「～について」というれっきとした英単語(もとはラテン語)です。

`a`を入力して返事を書いているときに引用したり、複数のメールを引用するには、`C-c C-y`を使います。`C-c C-y`を押すと、そのときのMessageモードのテキストの一部(Emacsのマークがあれば、そのマークとカーソルの間、なければ全体)が引用されます。

引用した文のうち要らない部分は、きちんと削除しましょう。そうしないとメールの量も増えますし、読む方にとっても煩わしく感じるだけです。行ごとの削除は`C-k`ですから、これを使うと良いでしょう。

あとは、普通にメールを送信する際と何も変わりません。

^{*15} この印は、ホームディレクトリの .mew.el というファイルを編集することで違うものにかえられます。

3.4.4 メールを削除する

要らなくなったメールは、そこにカーソルを合わせて **d** を入力して D マークをつけてから **x** を押すことで削除されます。一気に複数のメールを削除する場合は、削除したいメール全てに D マークをつけてから **x** を入力します。

3.4.5 フォルダ

Summary モードで **g** を入力してみましょう。

```
Folder name (%inbox): %
```

と表示されます。ここでフォルダ名を入力すると、そのフォルダの内容が表示されます。**TAB** を押してみましょう。***Mew completions*** バッファが表示され、フォルダ名の候補が並んでいるはずです。「backup」と入力し **RET** を押してみましょう。今までに送ったメールが表示されるはずです。メールを送るとき、ヘッダに Fcc: %backup というフィールドがありましたが、Fcc は Folder carbon copy の略で、%backup フォルダにそのメールのコピーを残すという意味だったのです。

g を入力し、そのまま何も入力せずに **RET** を押すと、Mew を起動したときに表示される %inbox フォルダに移動できます。

メールを整頓する

Mew では、フォルダを使ってメールを整頓できます。%inbox フォルダで、整頓したいメールにカーソルを合わせて **o** を入力してみましょう。

```
Folder name (%from.s9912345):%
```

のように表示されます。Mew は整頓先を推測し、デフォルト値として括弧の中に表示してくれます。ここで整頓先を入力します。もし、このデフォルト値（ここでは %from.s9912345）が希望通りであれば、そのまま **RET** を押すだけです。このとき、そのフォルダがない場合は

```
\%from.s9912345 does not exist. Create it? (y or n)
```

のように聞かれますので、**y** を入力するとフォルダが作られます。

こうして整頓先が決定したメッセージには、番号のとなりに o マークがつきます。**x** を入力すると、実際に整頓が実行されます。複数のメッセージをまとめて整頓することもできます。

フォルダ名から推測するだけでは思いうようなフォルダを推測してくれないことがあります。Mew の整頓先の推測規則は、ホームディレクトリの .mew.el というファイルを編集することで変更できます。以下に例を示します。（Mew の Info より引用。）

```
(setq mew-refile-guess-alist
  '(("To:"
     ("staff@mew.org" . "+net/mew/staff")
     ("staff@iiijlab.net" . "+net/iiijlab/staff"))))
```

条件を指定してメッセージを選択しマークをつける

指定した条件に合うメッセージにマークをつける機能があります。[?]を入力し、条件を入力すると、条件にあうメッセージに * マークがつきます。* マークは、[m d]や[m o]などを使って、まとめて D マークや o マークなどに変更できます。条件の例を以下に示します。

From:に user が含まれるメッセージ: from=user

To:または Cc:に user が含まれるメッセージ: to=user | cc=user

To:または Cc:に user が含まれていて、かつ From:に user が含まれるメッセージ: (to=user | cc=user) & from=user

例えば、[?]を入力して、条件に「from=spam」と入力して * マークをつけ、[m d]を入力すると、From:に spam が含まれるメッセージに D マークがつきます。その後[x]を入力すれば、それらのメッセージが削除されます。

特定のフィールドではなくヘッダ全体に含まれる文字列を指定したい場合は、head=文字列のように指定してください。

指定した文字列を含むメッセージにマークをつけたい場合は、[C-u ?]と入力してください。

3.4.6 ファイルの添付（マルチパート）

メッセージにファイルを添付したいときは、Draft モードで[C-c][C-a]を入力します。すると、一番下に

```
----- attachments -----
      Multipart/Mixed                               3/
1  Text/Plain(guess)                               *Cover.txt
2                                                     .
-----0-1-2-3-4-5-6-7-8-9-----
```

のような行が挿入されます。このとき、本文は attachments より上の部分で、その下は添付領域と呼ばれます。添付領域で[c]を入力して、ファイルをコピーできます。例えば、.emacs を添付すると次のようになります。

```
----- attachments -----
      Multipart/Mixed                               3/
1  Text/Plain(guess)                               *Cover.txt
2  Text/Plain(guess)                               .emacs
3                                                     .
-----0-1-2-3-4-5-6-7-8-9-----
```

削除するときは、カーソルを合わせて[d]と入力すると、

```
Delete .emacs? (y or n)
```

と出るので、[y]を入力し削除します。マルチパートの作成途中でシングルパートに戻したくなったら、[C-b][d]と入力すると

```
Delete entire attachments? (y or n)  n
```


と出てくるので、`y`と入力すると戻す事ができます。少し説明をすると、マルチパートはディレクトリ構造を一時的に作成しており、今行なったのはマルチパートのために一時的に作成された、一番上のディレクトリ（この場合は3/）に戻ってから`d`を押して削除した訳です。

マルチパートを読む

マルチパートのメッセージ、つまり添付ファイルがあるメッセージも、今まで通り`SPC`を押していけば読むことができます。マルチパートのメッセージには日付の左側に M マークが付いていて、マークにカーソルを合わせて`SPC`を押すと、マルチパートの構造が簡素に表示されます。本文や添付ファイルを保存したいときは、カーソルを合わせて`y`を入力してください。

3.4.7 ニュースを読む

`-`で始まる名前のフォルダを使ってニュースの読み書きができます。Summary モードで`g`を入力して、`-`を入力すると、

```
Folder name (%inbox): -
```

と表示されます。ここで、`TAB`を押してみましょう。*Mew completions* バッファに、ニュースグループ名が表示されます。この、fj.mail.reader.mew などがニュースグループです。ここで読みたいニュースグループの名前を打ち込み`RET`を押してください。

```
Range (update):
```

と表示されたら、そのまま`RET`を押すと、記事が取得されます。表示されなかった時や、最新の情報に更新したいときは、`s RET`と入力すると新しい記事を取得できます。`SPC`を押すと、個々の記事が読めます。操作はメールの場合と同じです。もし`TAB`を押して No matching newsgroup と出た場合はニュースグループ名の取得が必要なので、`C-g`で一度キャンセルしてから`2 Z`と入力してください。

```
NNTP case value (default):
```

と表示されたら`RET`を押してください。

```
Collecting newsgroup list... done
```

と表示されればニュースグループの取得は完了です。/Mail/.nntp/@orchid-news % nntp に .mew-folder-alist2 が作成され、その中にニュースグループが格納されているのを確認してください。その後ニュースを読む操作をもう一度最初から行ってください。

記事の保存

ネットニュースの記事は一定期間で削除されてしまうので、興味深い記事があったら保存しておきましょう。Summary モードで、`y`を入力することで保存できます。

記事を投稿する

ネットニュースの記事を投稿するには、ニュースグループの記事の一覧を表示しているときに、**[w]**を入力します。すると、メールを書くときと同じようにバッファが用意され、ヘッダに Newsgroups:フィールドが用意されているはずです。このフィールドに、投稿したいニュースグループ名を入力したら、あとは、メールの場合と同じように、Subject:を入れ、本文を書きます。メールの場合と同様に、

- 本文は全角 35 文字程度で改行する
- 行末に不要な改行は入れない
- 長すぎる署名をつけない

ことに気をつけましょう。ネットニュースの記事はたいへん多くの人が読むものです。記事に住所や電話番号などの個人情報を入れること、特定の人に宛てた内容を書くことはやめましょう。クロスポスト（複数のニュースグループに向けて記事を書く）の場合は、

`Newsgroups: fj.comp.mobile,fj.net.phones.phs`

というように、それぞれのニュースグループをカンマで区切って（スペースは入れません）記述します。同一内容の記事を、クロスポストではなく、個別に投稿することは、マルチポストと呼ばれ、一般には歓迎されません。しかし、カテゴリの異なるニュースグループ（例えば、campus と tsukuba など）に同じ内容の記事を書く場合は、クロスポストでなく、マルチポストにした方が良いでしょう。全てのニュースサーバでその両方のカテゴリが読めるとは限りません。

質問の記事を投稿し、その答えが得られたとしても、「御礼だけの記事」は投稿しないようにしましょう。全世界の人たちは、あなたの御礼記事を読みたくてネットニュースを購読しているのではありません。どうしても御礼がしたくてたまらない場合は、せめてメールにしましょう。

また、複数の人から回答があった場合などは、「サマリ (summary)」といって、問題に対する答えをまとめた記事を投稿すると、情報が集約されていて、あとから読む人たちにも有用で、喜ばれます。

[C-c][C-c]を入力することで投稿できます。

なお、ちゃんと投稿できるかなどをテストするテスト投稿は、ローカル（皆さんの場合は coins.test）で行い、他の所では行わないようにしましょう。また、自分の投稿したニュースの記事が読めるようになるまでには、少し時間がかかる場合があります。

ニュース記事にフォローする

ニュースの記事に返事の記事を書くことを、「フォローする」などと言います。記事を引用せずにフォロー記事を書くときには**[a]**、引用して書くときには**[A]**を入力しましょう。

メールの場合以上に、他の記事の全文引用は避け、不要な部分は削りましょう。ネットニュースの記事は全世界にコピーされていくのですから。

ニュース記事にメールで返事を送る

ニュースの記事を書いた人に対して、メールを出すこともできます。メールの場合と同様に、カーソルを合わせて**[C-u a]**もしくは**[C-u A]**を入力すればメールを送ることができます。

3.4.8 Mew のマニュアル

Mew のマニュアルは Info(**C-h i**) で読むことができます。日本語マニュアル Mew-J もあります。英語マニュアルを読むときは Info 画面で Mew の項目を、日本語マニュアルを読むときは Mew-J を選択してください。もし、日本語マニュアルが文字化けする場合は **C-h i** の前に「**M-x** set-language-environment RET Japanese RET」と入力してください。

3.5 Tips

前章まででは編集などについて解説してきましたが、本章では Emacs のより深い機能や便利なコマンドについて解説します。

3.5.1 info の操作

Emacs の使い方で分からないことがあれば、Emacs の info を活用しましょう。info は texinfo 形式のファイルを参照するためのシステムです。Emacs の使い方は 2 章で紹介した info コマンドを使い、‘info emacs’で見することもできますが、**C-h i** で Emacs 上で見ることができ、便利です。

texinfo と info

ソフトウェアのマニュアルを書き記す際によく使われる形式が texinfo です。texinfo 形式では中に書かれた情報をノードという単位で保持します。さらに、ノード同士を仮想的なリンクで繋ぎ、互いに参照することで、階層的な構造を持たせることが可能です。階層的な構造の例としては、マニュアルの章や節が挙げられます。

Emacs での操作

Emacs で info を利用すると、ノードのリンクを辿ったり、別のノードを参照したりすることができます。これはマニュアルなどを参照する際の、章や節の移動に相当します。表 3.8 ではその方法について例とともに説明します。

表 3.8: info の操作

目的	例	キー
ノードの選択	1.1 節を参照する	Enter
同じ階層の次のノードへ移動	1.1 節から 1.2 節へ移動する	n
同じ階層の前のノードへ移動	1.2 節から 1.1 節へ移動する	p
上の階層のノードへ移動	1.2 節から 1 節の先頭へ移動する	u
info を終了		q

3.5.2 画面の分割

Emacs ではウィンドウをいくつかに分割することができます。分割した領域に別々のバッファを読み込むと別のファイルを参照しながらファイルの編集を行うことができます。画面を分割すれば作業効率を上げる事ができるので覚えると良いでしょう。例として図 3.4 を見てください、このようなことができます。分割のためのコマンドは表 3.9 のようになります。

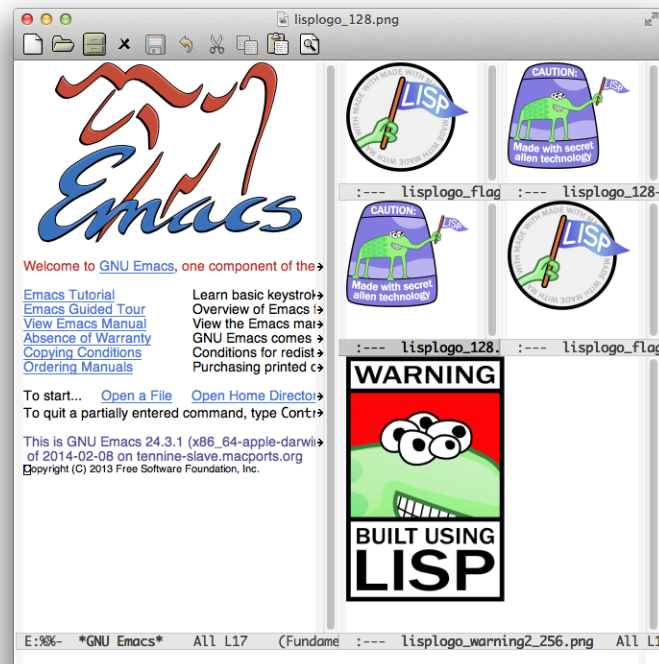


図 3.4: Emacs 分割例

バッファを切り替える時は、`[C-x]` `[C-b]` で現在のバッファリストを表示し、`[C-x b]` と入力した後、作業したいバッファの名前を入力して、バッファを切り替えましょう。メニューバーの Buffers から選ぶ事ができます。

3.5.3 Emacs からのシェル操作

プログラムなどを書いている際にプログラムファイルを保存して、ほかの shell の画面に行きプログラムをコンパイル、実行する、ということすると思いますが、この時の画面遷移が面倒です。もしこれを一つの shell 画面上で実現できたら便利です。これを Emacs では以下のようなコマンドを用いることで実現することができます。

```
M-x shell
```

表 3.9: 画面分割に関するコマンド

コマンド	機能
<code>C-x 0</code>	カーソルのあるウインドウを閉じる
<code>C-x 1</code>	カーソルのあるウインドウ以外を閉じる
<code>C-x 2</code>	ウインドウを水平方向に分割
<code>C-x 3</code>	ウインドウを垂直方向に分割
<code>C-x o</code>	カーソルの次のウインドウに移動
<code>C-x b</code>	ウインドウに割り当てるバッファを切り換える。 どのバッファを表示するかは Emacs が尋ねてくる
<code>C-x C-b</code>	現在作業できるバッファのリストを表示する。必要な所にカーソルを移動して <code>f</code> を押すと、そのバッファがウインドウに表示される
<code>C-x k</code>	ウインドウに表示されているバッファを削除

3.5.4 モード

プログラムを書く時などに便利な機能としてモードというものがあります。モードとは、編集するテキストに応じて Emacs の振舞いを適応させるものです。これは例えば Java というプログラミング言語のコードを書く際に、`java-mode` を使うと Java 言語の構文の強調をしてくれるなど、書くための手助けをしてくれます。モードにはメジャーモード (*major-mode*) とマイナーモード (*minor-mode*) の二つがあります^{*16}。大まかな認識としてはテキストに対しメジャーモードで Emacs をその種類のテキストに特化させ、マイナーモードでユーザにあわせた細かい適応をさせます。詳しい話をすれば、メジャーモードは排他的でありユーザが意図してどれを使うかを選択する事はほとんどありません^{*17}、マイナーモードはメジャーモードと反対に強制的であり原則的には他のマイナーモードに左右されずマイナーモードを使う事ができます、そしてユーザがオン/オフを管理します。

表 3.10 に使用できるモードの一例を示します。

表 3.10: モードの一例

モード名	説明
<code>c-mode</code>	C 言語を扱うメジャーモード
<code>perl-mode</code> または <code>cperl-mode</code>	Perl のためのメジャーモード
<code>java-mode</code>	Java のためのメジャーモード
<code>html-helper-mode</code>	HTML ファイルを作成する際に用いるメジャーモード
<code>whitespace-mode</code>	TAB や Space を可視化するマイナーモード
<code>tex-mode</code>	TeX(第6章参照) のためのメジャーモード

表 3.10 以外にもモードが存在します。自分で調べてインストールしてみるのもいいでしょう。

^{*16} ちなみに編集の節で書いた、上書きモードはマイナーモードに属する物です

^{*17} ファイルに拡張子がついていないか間違った物がついていたという場合などを除き

続いてこのモードをどのように使うかを示します。以下のように入力することでモードをオンにします。

```
M-x [mode-name] ↵
```

副モードの場合だとエコーエリアに「[モード名称] mode enabled(もしくは disabled)」と出ます。これで指定されたモードがオンになります。ですが、Emacs がファイルの拡張子に応じて主モードを選択してくれるのがほとんどなので、手動でモードを有効にする事は副モードを選択する際ぐらいでしょう。

3.5.5 package.el

Emacs24.3 からは Package.el という Linux でのパッケージ管理ツールに類する物が存在します。これを使用すると、コマンドを打つだけで世界中の Emacs ユーザたちが作った拡張機能を使用する事ができるようになります。「`M-x package-install`」を実行した後に、自分の入れたい拡張機能の名前を打ちましょう。もしくは入れられる拡張機能のリストを見て入れたいかもしれません、その場合は「`M-x package-list-packages`」を使ってくださいこのようなコマンドを使い、*auto-complete*^{*18}や *undo-tree*^{*19}といった Emacs を便利にする拡張機能を入れられます。

3.6 自宅で Emacs を使うには

ここでは情報科学類の計算機以外、例えば自宅でも Emacs を使うために主要な OS で Emacs をインストールする方法を記載したいと思います。

3.6.1 Windows

Windows だとそのままでは Emacs が使うライブラリがほぼ無く^{*20}、幾つかの機能を使う時に少し手間がかかります。gnupack 等の環境を導入する方法だとそういった手間は省けるのですが、今回は Emacs を導入したいだけのため公式のバイナリを使う方法を紹介します。

公式の GNU Emacs のページ^{*21}から近くの GNU ミラーを開きましょう、表記が揺れているのですが新しい物は emacs-[バージョン]-[アーキテクチャ]-w64-mingw32.zip というような表記で書かれているので、必要なバージョンとアーキテクチャの物を選択し、ダウンロードしてください。ダウンロードしたファイルを好みの適切な場所に展開して、bin ディレクトリ内の runemacs.exe で起動が出来ます。

この方法だと Emacs 自体だけを入れており、幾つかのライブラリはやはり無いままなので Emacs の機能の一部がライブラリ不足で上手く機能しません。そのような問題に遭遇した場合、個別にライブラリを導入するか gnupack 等の環境を導入する方法を検討してください。

3.6.2 macOS

macOS には標準で Emacs が入っていますが大変古いバージョンの物しか入っていません。最新の安定版を使いたい場合、Emacs for OSX(<https://emacsformacosx.com/>) から最新のパッケージをダウンロード

^{*18} Emacs のための高度な自動補完機能を提供する拡張

^{*19} テキスト編集の取り消し、やり直しを使いやすくする拡張

^{*20} 例えば eww で使う GnuTLS 等はありません

^{*21} <https://www.gnu.org/software/emacs/>

し、開いた中にある Emacs.app を Application フォルダに入れてください。

3.6.3 Linux

Linux では基本的にディストリビューション標準のパッケージマネージャで導入すると良いでしょう。大まかには以下のようなコマンドでインストールできるでしょう。パッケージ名が名前そのままでは無いときもあるので、各ディストリビューション毎に確認をしてください。

— apt 系ディストリビューション —

```
$ apt-get install emacs↵
```

— yum 系ディストリビューション —

```
$ yum install emacs↵
```

パッケージマネージャで導入した後はコマンドラインから起動するか、使用しているデスクトップ環境のランチャー等から起動してください。

第 4 章

ブラウザとメールの使い方

本章では、Web ブラウザを使ってインターネット上のページを閲覧する方法と、メーラーを使って電子メールをやりとりする方法を説明します。Web ブラウザとして Firefox，メーラーとして Thunderbird というソフトウェアの使い方を例にあげて具体的に説明します。これらは Mozilla Foundation が開発や保守を行っているオープンソースのソフトウェアで、Windows や macOS，Linux 系の OS で使用できます。

4.1 Firefox による Web ブラウジング

本節では Firefox を使った Web ブラウジングの基礎について説明します。多数のコンピュータが接続されたネットワークであるインターネットを介して文書を公開・閲覧する仕組みとして、*World Wide Web* (WWW) があります。この WWW で公開されている文書 (Web ページ) を閲覧するためのソフトウェアが、Firefox をはじめとする *Web* ブラウザです。

4.1.1 起動する

Dock から、図 4.1 に示す Firefox のアイコンをクリックすることで起動できます。図 4.2 は起動時の例です。



図 4.1: Dock 内の Firefox のアイコン

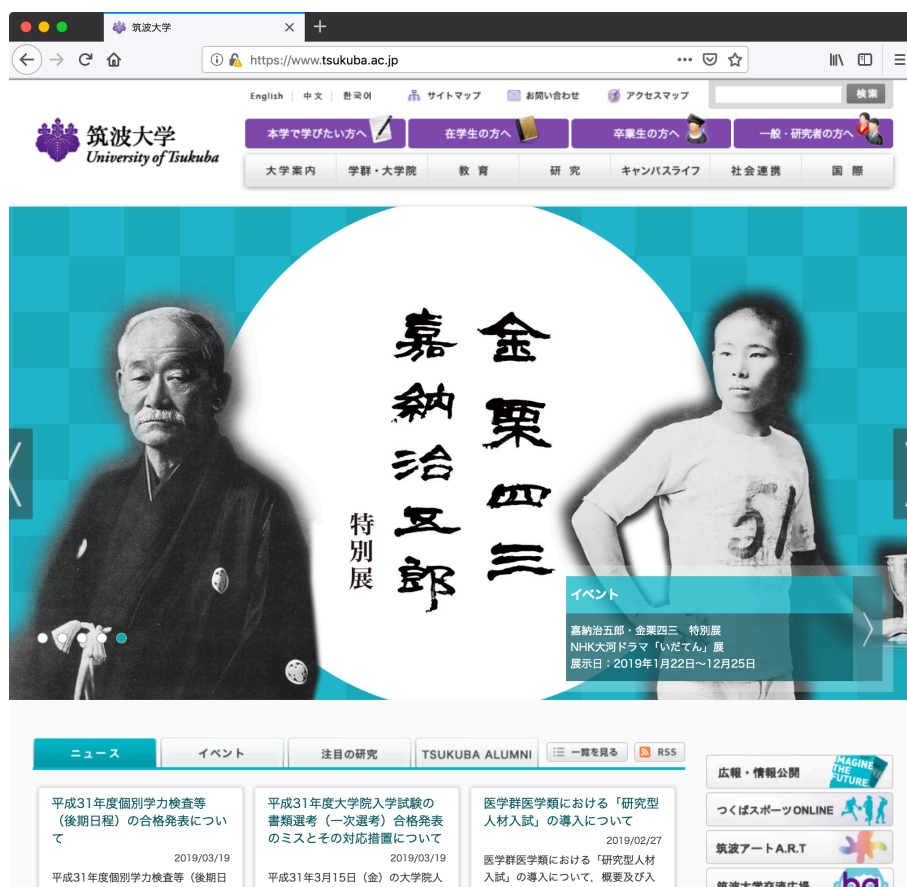


図 4.2: Firefox のウィンドウ

4.1.2 ブラウジングの基本的な操作

本小節では基本的なブラウジングの操作を解説します。

WWW 上の Web ページの場所は URL と呼ばれる、いわば地番の役割を持つ文字列で一意に定められます。例えば筑波大学の Web サイトの URL は `http://www.tsukuba.ac.jp` となります。URL がわかれば Web ブラウザを使ってその場所のコンテンツを得ることができます。また、ある Web ページに記述された別の Web ページの URL をたどることで、新しい Web ページへ移動できます。本節では、これらの 2 つの方法で Web ページを閲覧する方法を、順を追って解説します。

URL を直接指定する方法

自分が見たいページに移動するには、URL を指定する必要があります。Firefox のウィンドウ上部にあるテキストボックス*1 (図 4.2 の上部にある横長の枠) に次の手順で URL を入力することで、目的のページを閲覧できます。

1. テキストボックスへマウスカーソルを移動してクリックし、 (Delete) キーなどで現在の URL を

*1 文字が入力できる場所のこと。

消去する

2. 新しい URL をキーボードから入力する
3. 最後に`return`を押す

リンクをクリックして移動する

Web ページ内に書かれている URL をクリックすると、他の Web ページへ移動します。これを繰り返すことで、Web ページのネットワークを次々にアクセスできます。

次の Web ページへのリンクとなっている部分へマウスカーソルを移動すると、カーソルの形が矢印から指差しの形に変形します。ここでクリックすると、次の情報へと移動できます。

4.1.3 Firefox の基本的な機能

ブラウジングを行う際はウィンドウ上部にある図 4.3 に示すツールバーを操作することで、Web ページを行き来したり、特定のページをブックマークに登録するといった操作が行えます。ここではそのような基本的な操作を説明します。

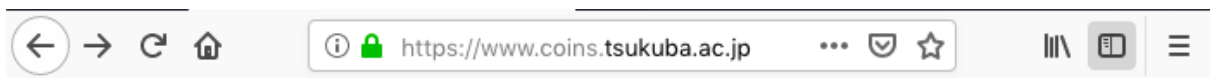


図 4.3: Firefox のウィンドウ上部にあるツールバー

戻る/進む

ツールバー（図 4.3）の左側にある`⏮`と`⏭`のふたつのボタンを使うことで、Firefox を起動してから訪れたページを順番に行き来できます。`⏮`をクリックすることで、ひとつ前にアクセスしたページに戻ることができます。また、繰り返し押すことでそれより前にアクセスしたページへ戻ることもできます。一方前のページに戻ったあとに`⏭`をクリックすることで、新しいページへ進むことができます。

再読込

URL が含まれているテキストボックス（図 4.3）のアドレスバーの右にある`🔄`を押すことで、現在見ているページのデータを再表示します。また、`shift`を押しながらこのボタンをクリックすると、強制的にデータを再取得して、再表示します。

ホーム

ツールバー（図 4.3）の右側にある`🏠`を押すことで、設定されているホームページへ移動できます（4.1.6 節参照）。ここで言う「ホームページ」とは、Firefox 起動時に表示されるページのことを意味します*2。

*2 「Web ページ」の意味で「ホームページ」という言葉が使われる場合がありますが、ここでは異なります。

ブックマーク登録

図 4.3 の☆を押すことで、図 4.4 のように現在開いているページをブックマークに登録できます。ブックマークとは、本にしおりを狭むように、よく訪れるページを記録しておく機能です。ちなみに、既に登録してあるブックマークは、一番上に並んでいるメニューの「ブックマーク」からアクセスできます。また、ブックマークされたページの☆を押すとブックマークを削除するなどといったことができます。



図 4.4: ブックマークの編集

4.1.4 ブラウザのショートカット

1.6.3 節ではキーボードのショートカットを説明しましたが、ブラウザでもショートカットが使えます。表 4.1 にはそのショートカット一覧を載せておきます。⌘+←と記載されている場合は「⌘（コマンド）キーを押しながら←を押す」ことを意味します。

表 4.1: ブラウザのショートカットキーのまとめ

動作	キー
戻る/次へ	+ /
ページ内の文字列検索	+
再読み込み	+ または
印刷	+
新しく Firefox のウィンドウを開く	+
開いている Firefox のウィンドウを閉じる	+ +
Firefox を終了	+
ブックマークに登録する	+
ブックマークを表示する	+
読み込み中止	
新規タブ	+
タブを閉じる	+
次のタブに進む	+ +
前のタブに進む	+ +

4.1.5 タブ

Firefox では、図 4.5 に示すように、タブと呼ばれるページの表示領域を 1 つのウィンドウの中で切り替えて使うことができます。新たなタブを作るにはメニューバーの「ファイル」≫「新規タブ」を選択するか、図 4.5 内の をクリックします。この状態で、URL を入力して他のページにジャンプすると、新しく作ったタブにページが表示されます。また、リンクをクリックした時に、リンク先のページを別のタブに表示させたい時は、マウスカーソルがリンクの上にある状態で右クリックし、「リンクを新規タブで開く」をクリックします。タブを閉じたい時は、閉じたいタブの上部にあるでっばりの上で、右クリックし「タブを閉じる」をクリックします。また、タブの右端についている をクリックしてもタブを閉じることができます。

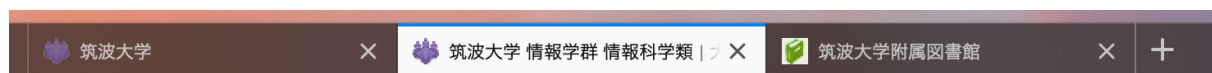


図 4.5: Firefox のタブ

4.1.6 各種設定

ここでは、メニューバーの「Firefox」≫「環境設定」で行える設定について簡単に解説します。

4.1.7 ホームページの設定

一番初めに表示されるページはユーザの好みに応じて別のページに変更することが可能です。ホームページを設定するには、**ホーム**をクリックすることによって行えます。

表示されるページを変更するには「ホームページ」の項目にある、「ホームページと新規ウィンドウ」の項目に「カスタム URL...」を指定し、下のテキストボックスに URL を入力します。あるいは**現在のタブグループを使用**をクリックすることで、その時ウィンドウで開いている全てのタブの Web ページを、ホームページとして設定できます。

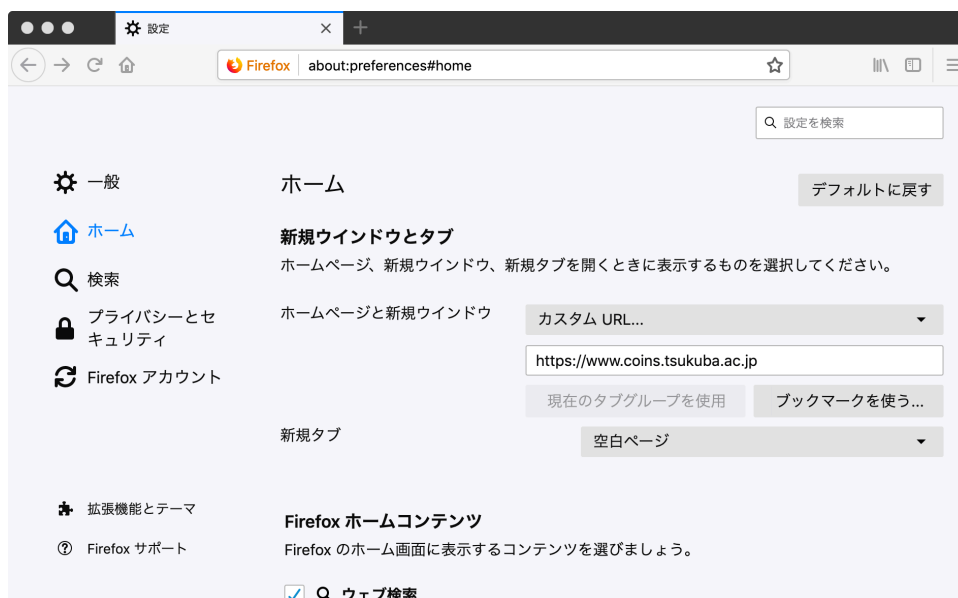


図 4.6: ホームページの設定

4.1.8 プライバシーの設定

プライバシーの設定（図 4.7）を編集すれば、Firefox に入力した履歴の記憶に関する設定を変更できます。また、**履歴を消去...**をクリックして現れる図 4.8 のような画面から、現状で記憶されている履歴を消すことも可能です。

消去する履歴を選択することで Cookie を個別に消去することができます。Cookie（クッキー）とは、訪れた Web サイトによって作成され、Web ブラウザを通じて訪問者のコンピュータに個人のデータを保存する仕組みです。これによってアカウント情報を要求するサイトを利用するときなどに、わずらわしい入力をその都度せずに済みますが、Cookie はユーザーがどのような Web サイトを利用しているのかを収集する目的で利用されることがあります。それを避ける目的で Cookie を削除するという選択もあります。



図 4.7: プライバシーの設定



図 4.8: 履歴の消去

4.1.9 セキュリティの設定

セキュリティの設定は図 4.9 のようになっています。初期設定では

- 危険な詐欺コンテンツをブロックする
- 危険なファイルのダウンロードをブロックする

- 不要な危険ソフトウェアを警告する

の合計3つにチェックが入っているはずですが、ただ、このチェックに関してはFirefoxの初期状態なので、Firefoxが更新される度に変更されてしまうかもしれません。このチェックは自分のプライバシーを守るという観点からも、慣れないうちはあまりいじらないほうが良いでしょう。



図 4.9: セキュリティの設定

4.1.10 文字コードの設定

Web ページは日本語や英語に限らずさまざまな言語で書かれており、さらにさまざまな方法で文字を数字の列へ対応付けて表現されています。この、文字と数値の対応付けを文字コードと呼びます。Web ページのコンテンツを Firefox が誤った文字コードとして解釈してしまうと、読めない文字の羅列が表示されてしまいます。これを文字化けといいます。

Firefox は文字コードを自動的に判別しますが、場合によってはこの判別に失敗します。このような場合、そのページを正しく表示させるためには、図 4.10 のようにその Web ページの文字コードの設定を手動で行う必要があります。

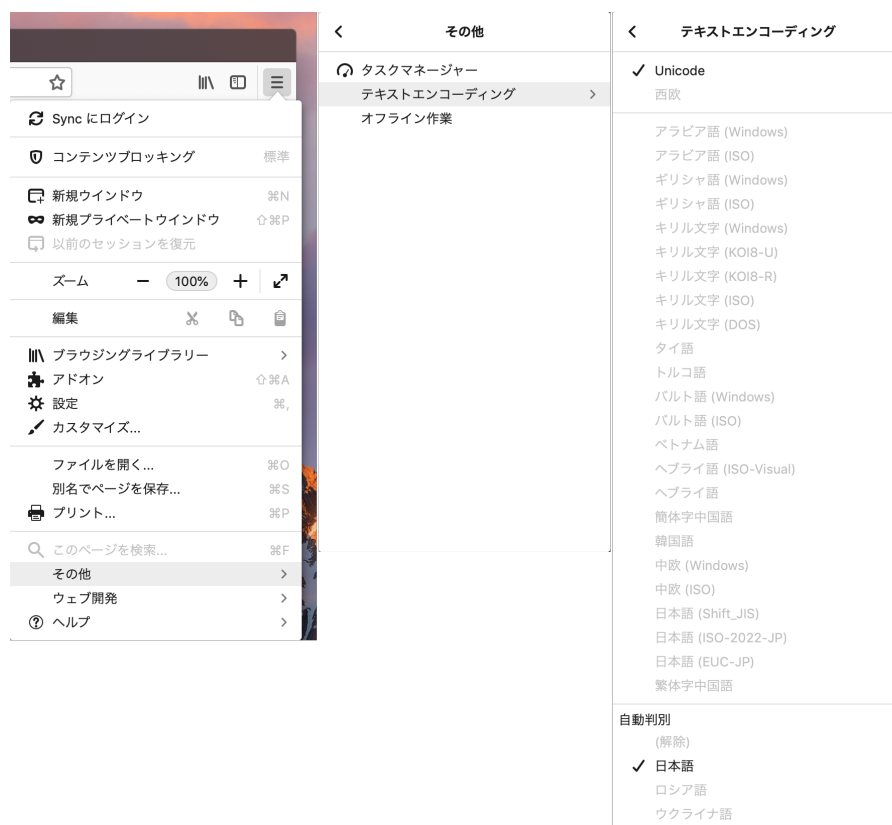


図 4.10: 文字コードの手動設定

設定を行うためには、ブラウザ右上のメニューから「その他」＞「テキストエンコーディング」をクリックします。通常はこの一覧の中の「日本語」を選んでおくことで自動判別を行います。

4.2 Thunderbird による電子メールの利用方法

インターネットを介して誰かとコミュニケーションをとる方法のひとつとして電子メール（メール）を使ったテキストコミュニケーションがあります。メールを送受信するためには、*Thunderbird* や Microsoft Windows Mail といったメーラと呼ばれるソフトウェアを使います。本節ではメールを送受信する仕組みを簡単に述べ、Thunderbird を用いてメールを送受信する方法を説明します。

4.2.1 メールを送受信する仕組み

メーラはメールサーバと通信をしてメールを送受信します。メールサーバとは自分宛のメールを一時的に預かったり、他の人宛のメールの配送をする特殊なコンピュータです。メーラはこのメールサーバに自分宛のメールがないか問い合わせ、預かってもらっている場合は今使っているコンピュータに持ってきたり、逆に指定した宛先に送ってもらうというを行います。ちなみに多くの場合はひとつのコンピュータがメールを受信するため機能と送信するための機能をまとめて担っています。

そこでメールを扱うためには、メーラに対して

- 自分宛のメールを預かるメールサーバの情報
- メールを送信するためのメールサーバの情報
- メールアドレスやパスワードなど、認証に必要な情報

を適切に設定する必要があります。

4.2.2 COINS のメールを利用するための設定

本小節ではメールに設定する必要がある情報を列挙し、iMac にインストールされている Thunderbird の設定方法を解説します。

COINS のメールを利用するにあたって表 4.2 にある情報が必要になります。本節では Thunderbird の設定方法を説明しますが、携帯電話や個人のパソコンのメールで設定する場合も、この情報を設定することで送受信できるようになります。

表 4.2: COINS のメール送受信のための設定

メールアドレス	あなたのアカウント名@coins.tsukuba.ac.jp
受信サーバの種類	IMAP または POP (IMAP 推奨)
受信サーバのアドレス	viola6.coins.tsukuba.ac.jp
送信サーバのアドレス	viola6.coins.tsukuba.ac.jp
ユーザ名	あなたのアカウント名
パスワード	あなたのパスワード

まず Thunderbird を起動します。Dock から図 4.11 にあるような Thunderbird のアイコンをクリックしてください。



図 4.11: Dock にある Thunderbird のアイコン

初回の起動では必ず図 4.12 のようなメールアカウント設定のウィンドウが出てきます*3。

*3 2 回目以降で設定する場合には、メニューバーの「ファイル」>「新規作成」>「新規のメールアカウントを取得」を選択することで行います。

The image shows a window titled "新しいメールアドレスを使いたい方" (For those who want to use a new email address). It contains a text input field with "Mitsuo NAKAJIMA" and a "検索" (Search) button. Below this is explanatory text about how Thunderbird uses the input to find email addresses. There are two checkboxes: "gandi.net" and "Hover.com". At the bottom, there are two buttons: "メールアカウントを設定する" (Set up email account) and "このウィンドウを閉じて後で設定する" (Close this window and set up later).

新しいメールアドレスを使いたい方

Mitsuo NAKAJIMA 検索

いくつかのプロバイダとの提携により、Thunderbird が新しいメールアカウントを提供します。利用可能なメールアドレスを検索するため、ご氏名または好きな言葉を上のテキストボックスに入力してください (半角英数字のみ)。

☐ gandi.net ☐ Hover.com

検索語句は、利用可能なメールアドレスを見つける目的で Mozilla (プライバシーポリシー) とサードパーティのメールプロバイダ に送信されます。

メールアカウントを設定する このウィンドウを閉じて後で設定する

図 4.12: 初回起動時の設定ウィンドウ

ここで「メールアカウントを設定する」をクリックし、図 4.13 に示すようにメールアドレスやパスワードの入力を行います。入力が終わったら「続ける」を押します。

あなたの名前

メールに差出人として記載されるあなたの名前を入力してください。メールは世界中に送信される可能性があるため、名前は漢字ではなくローマ字で書くことが好まれます。

メールアドレス

あなたの COINS のアカウント + “@coins.tsukuba.ac.jp” を入力してください (例: s9912345@coins.tsukuba.ac.jp)

パスワード

あなたの COINS のパスワードを入力してください。「パスワードを記憶する」のチェックを外してください。



あなたの名前: M.NAKAJIMA 受信者に表示される名前です。

メールアドレス: s1499999@coins.tsukuba.a

パスワード:

☐ パスワードを記憶する

新規アカウントを取得 キャンセル 続ける

図 4.13: メールアドレスとパスワードの入力

次に図 4.14 のようなウィンドウで受信サーバと送信サーバについて設定を行う必要があります。表 4.2 にあるように、「受信サーバ」には“viola6.coins.tsukuba.ac.jp”を、「送信サーバ」には“viola6.coins.tsukuba.ac.jp”を入力します。なお表にもあるように受信サーバの種類として POP か IMAP が選択できますが、“IMAP”を選ぶと良いでしょう。ここでは、受信サーバの設定に IMAP を使い、接続の保護に SSL/TLS を、受信サーバの接続の保護*4に SSL/TLS を設定して設定を再テストした例をのせています。入力が終わったら「再テスト」をクリックしましょう。「再テスト」をクリックして問題が起こらなかった場合は、アカウントが正しく設定されたため「完了」をクリックして、アカウントの設定を完了します。（正しく設定できていない場合は、次へ進むことができません。設定を見なおしてください。）

*4 相手が本物であることを確認します。また、通信経路を暗号化します。



あなたの名前: M.NAKAJIMA 受信者に表示される名前です

メールアドレス: s1499999@coins.tsukuba.ac 既存のメールアドレスです

パスワード:
☐ パスワードを記憶する

⚠ Thunderbird はあなたのアカウント設定を見つけられませんでした。

	サーバーのホスト名	ポート番号	SSL	認証方式	
受信サーバー:	IMAP	viola6.coins.tsukuba.ac.jp	993	SSL/TLS	自動検出
送信サーバー:	SMTP	viola6.coins.tsukuba.ac.jp	465	SSL/TLS	自動検出
ユーザー名: 受信サーバー:	s1499999		送信サーバー:	s1499999	

詳細設定 キャンセル 再テスト 完了

図 4.14: メールサーバの設定

以上でアカウントの設定が完了し、COINS のメールを利用できるようになりました。次の小節以降では Thunderbird を使ってメールの送受信を行います。

Thunderbird がフリーズする場合

Thunderbird は macOS のアプリケーション「連絡先.app」内のデータにアクセスする際、フリーズすることがあります。その場合は次に示す手順の設定を行ってください。

1. 「システム環境設定」を開き、「セキュリティとプライバシー」メニューを選択します。
2. 「プライバシー」タブ中の「連絡先」を選択します。
3. 図 4.15 のように、Thunderbird に連絡先へのアクセスを許可するチェックマークを外して下さい。

これで Thunderbird が連絡先にアクセスすることができなくなり、フリーズを回避することができます。

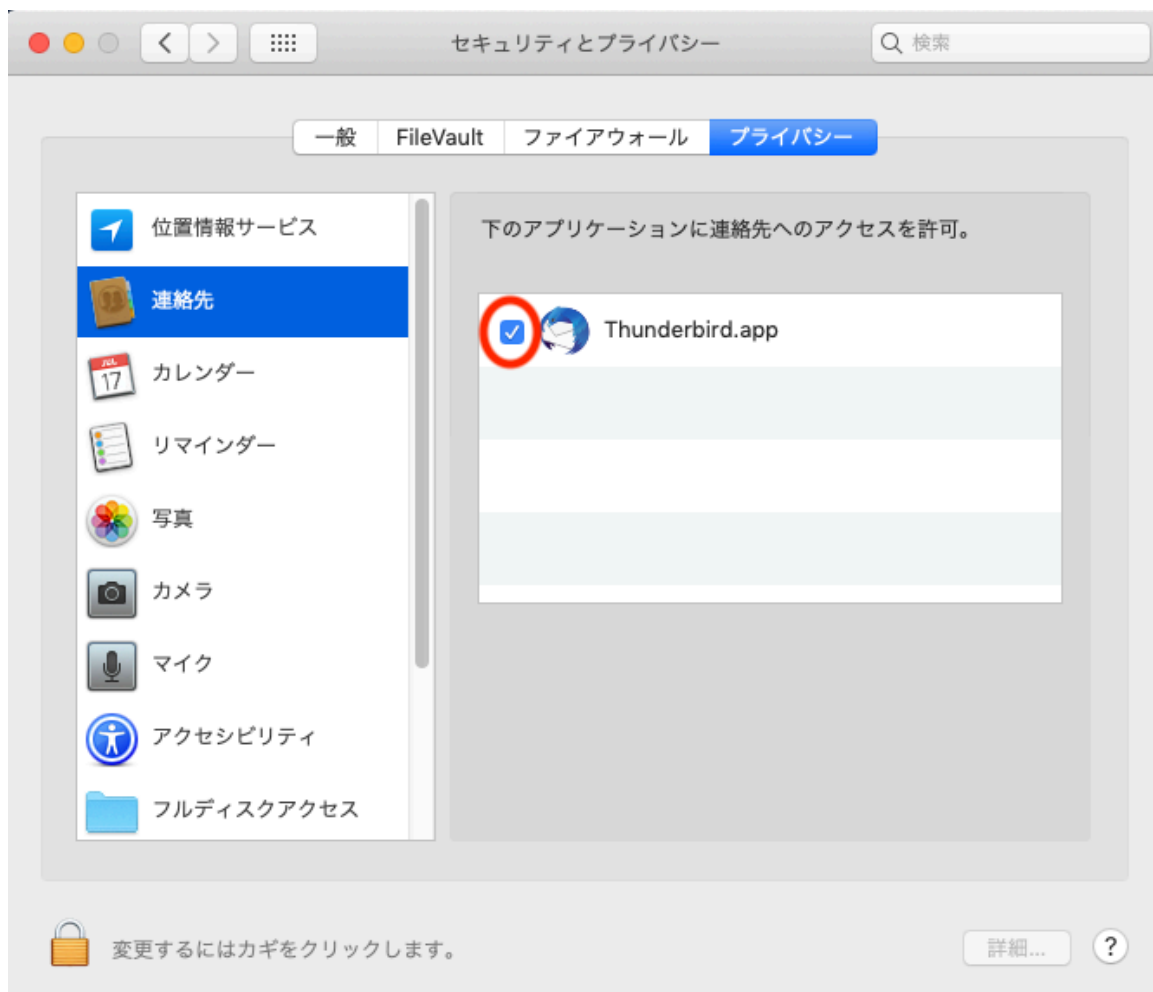


図 4.15: 連絡先へのアクセスを無効にする

4.2.3 メールを受信

Thunderbird ではデフォルトで「起動時に新着メッセージがないか」、「10 分毎に新着メッセージがないか」をメールサーバに確認するため、基本的には自動的にメールの受信を行ってくれます。もし、手動

で受信を行いたい場合、Thunderbird のウィンドウの左上にある「受信」、あるいはメニューバーの「ファイル」>「新着メッセージを受信」を選択してください。その場ですぐメールサーバにアクセスし、新しいメールの有無を確認することができます。

4.2.4 メール作成と送信

Thunderbird ではメールを HTML 形式で作成するか、プレーンテキスト形式で作成するかを選択することができます。デフォルトでは HTML 形式が選択されており、「ツール」>「アカウント設定」>「編集とアドレス入力」とたどっていき、「HTML 形式でメッセージを編集する」のチェックボックスを外すことでプレーンテキスト形式でメールを作成できるようになります。

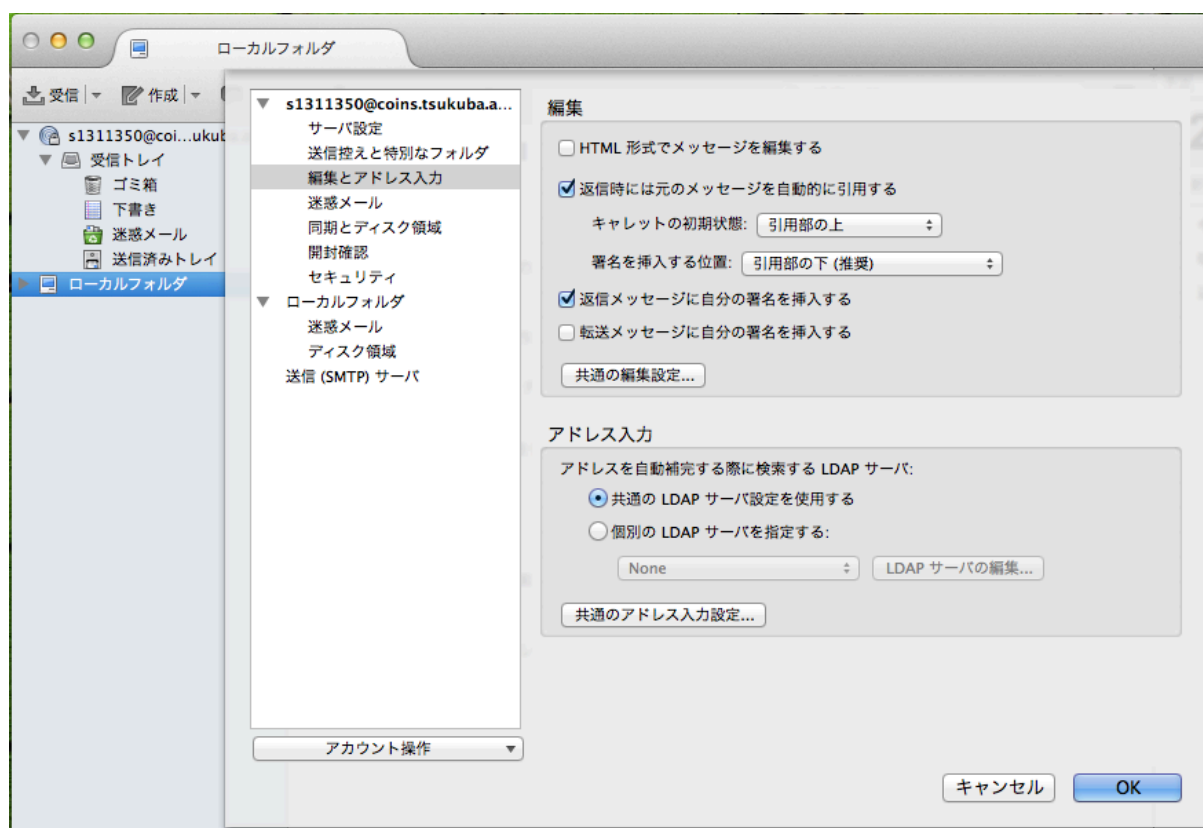


図 4.16: HTML・プレーンテキスト形式の選択

メールを作成するにはウィンドウ上部の「作成」またはメニューバーの「ファイル」>「新規作成」から行います。これらを選択すると図 4.17 にあるようなウィンドウが表示されます。



図 4.17: 新しいメールの作成

メールには宛先・件名・内容が必要ですから、宛先の枠に相手のメールアドレス、件名の枠にメールの題目^{*5}、そして大きな枠に本文を入力します。

また、一緒にファイルを送る場合は図 4.17 の上部にある「添付」をクリックし、添付ファイルを選択するウィンドウから添付したいファイルを選び、「OK」を押してください。この操作により、元のメールのウィンドウの右上に「添付」という項目が増え、一緒に送られる添付ファイルの一覧を確認できます。

最後にウィンドウ上部の「送信」を押すことでメールが送信されます。一方、すぐ送らず、一時的に保存しておきたい場合は「保存」を押すことで、Thunderbird の「下書き」のフォルダに保存され、あとから編集や送信ができるようになります。

メールの振り分け

Thunderbird では受信したメールを複数のフォルダに分別して管理できます。そしてこれを自動的に実行管理を支援するためにフォルダへの自動振り分け機能が備わっています。これをメッセージフィルタと呼びます。メッセージフィルタはユーザが決めたルールに従ってメールをフォルダに振り分けます。たとえば、COINS から来たメールなら COINS フォルダに、件名に「課題」という文字が含まれていたら task フォルダにといったような記述が可能です。新しくフォルダを作るには、左側のメニューの「ローカルフォルダ」を右クリックして新規フォルダを選択し、図 4.18 のように名称を設定します。

^{*5} 件名は本文を端的に表す一言にしましょう。例えば「こんにちは。XX 学類 N 年の XX です」といった件名は悪い例です。

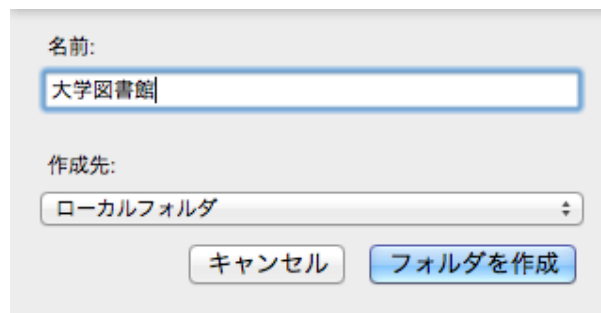


図 4.18: 新しいフォルダを作成するウィンドウ

では実際にルールを書いてメッセージのフィルタリングを行ってみましょう。メニューバーの「ツール」>「メッセージフィルタ」を選択、クリックしてください。すると図 4.19 のような画面が表示されます。ここで「新規」をクリックすることで新しいルール、振り分け規則を追加できます。

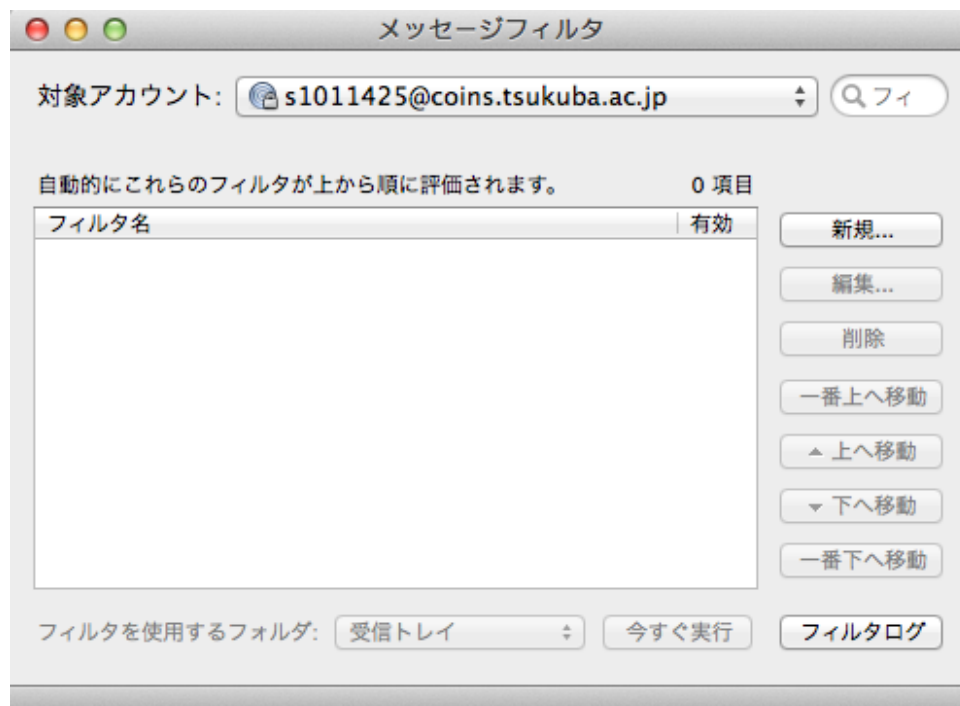


図 4.19: メッセージフィルタの新規作成

ルールの設定は図 4.20 のウィンドウで行います。ルールは「適用のタイミング」と「条件」、「動作」からなります。それぞれさまざまな設定が可能で、柔軟な振り分けを行うことができます。この例では「差出人のメールアドレスに tulips.tsukuba.ac.jp という文字列が含まれていたら“大学図書館”というフォルダへ移動させる」というルールを設定しています*6。

*6 大学図書館からのメールは、ほとんどが???@tulips.tsukuba.ac.jp というメールアドレスから送信されます

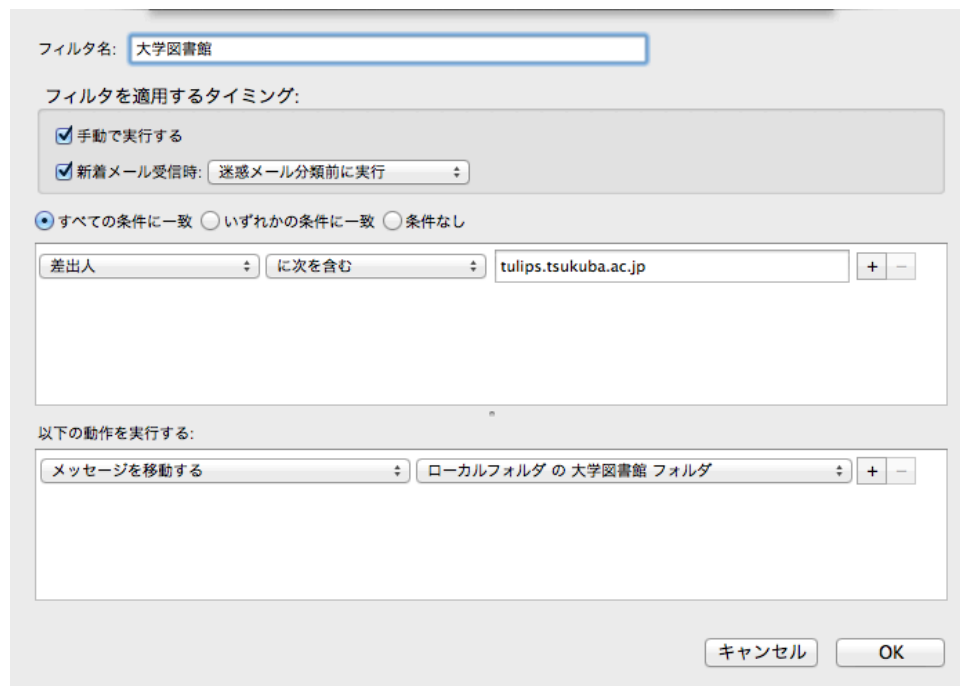


図 4.20: フィルタの条件の設定

4.2.5 ニュースの使い方

メーラとしての利用方法以外にも Thunderbird はネットニュース購読の為のツールとしても可能です。ここではネットニュース利用のための設定について解説します。

ニュースの設定

ニュースグループに参加するためにまずアカウントを追加しましょう。メニューバーの [ファイル] メニュー内の [新規作成] の中から [他のサービスのアカウント] を選択してください。すると図 4.21 にあるようなウィンドウが表示されます。この項目では「ニュースグループアカウント」を選択します。以降の画面ではそれぞれ表 4.3 のような画面が表示情報の入力求められます。



図 4.21: ニュースグループの設定

表 4.3: ニュースグループ参加の設定

項目	内容
あなたの名前	差出人として記載される名前
メールアドレス	あなたのメールアドレス
ニュース (NNTP) サーバ	news.coins.tsukuba.ac.jp
アカウント名	ニュースサーバを識別するための名前 (COINS など)
メール送信サーバのアドレス	viola6.coins.tsukuba.ac.jp
ユーザ名	COINS のアカウント名

ニュースの読み方

以上により設定したニュースグループ内のニュースを読むにはまず、メインウィンドウにある「購読するニュースグループを管理する」から購読したいニュースグループを選択する必要があります。これをクリックすると図 4.22 の画面が現れるので購読したいニュースグループをクリックしてチェックを入れてから「OK」

ボタンを押します。

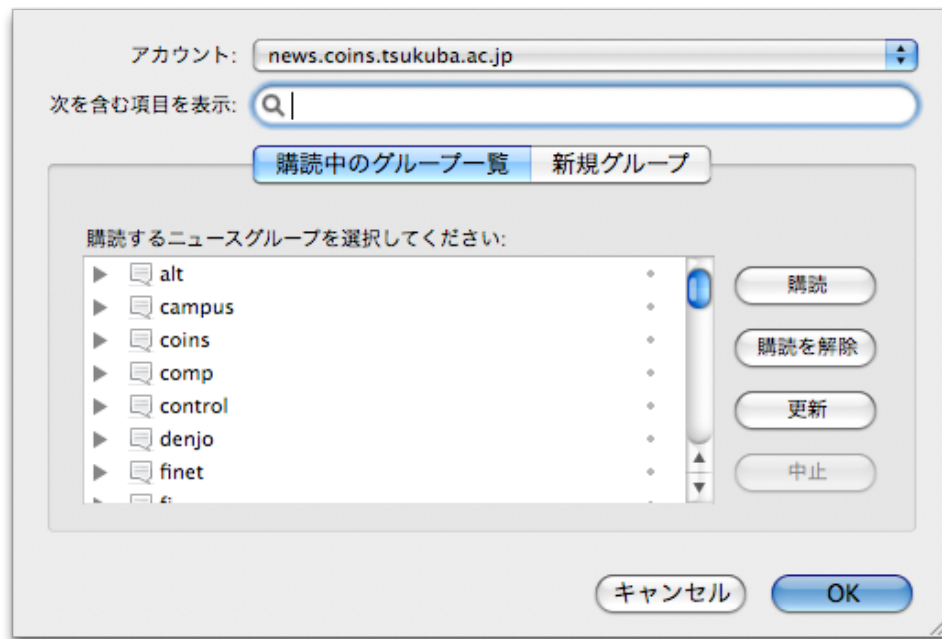


図 4.22: 購読するニュースグループの選択

これにより図 4.23 にあるように購読しているニュースグループに投稿されたニュースの一覧が表示されるので、この中から読みたいニュースをクリックして選択して読みます。

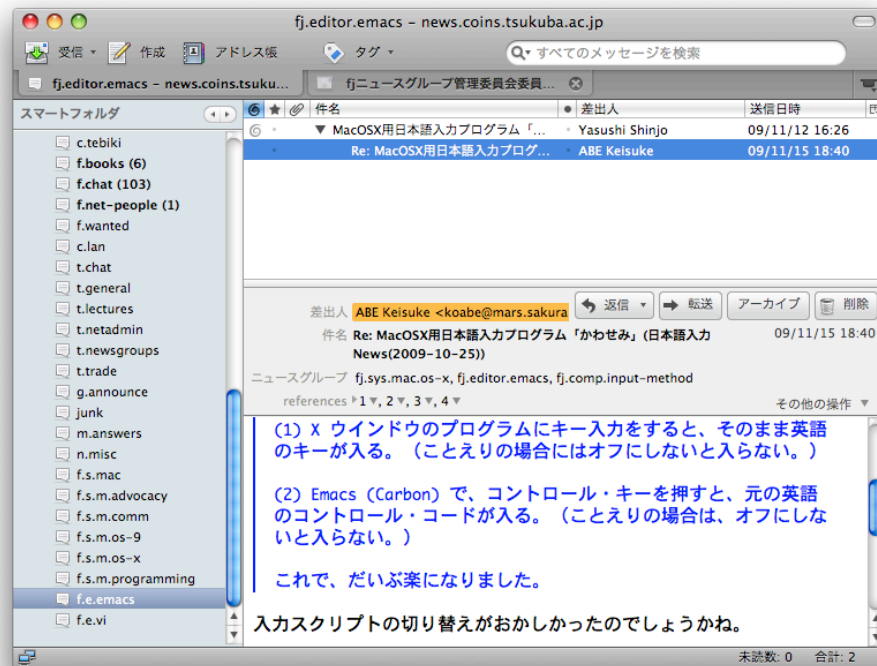


図 4.23: ニュースの購読

4.3 RainLoop

COINS メールを利用する方法として、RainLoop (<https://www.coins.tsukuba.ac.jp/webmail>) を利用する方法があります。RainLoop を用いることで、ブラウザ上で COINS メールを利用できます。ログインの際に必要なログイン名とパスワードは、COINS の計算機にログインする際に用いるものと同じです。



図 4.24: RainLoop ログイン画面

RainLoop では、メールの確認や送受信を行える他、RainLoop の表示設定、メールに記載する署名の設定などを行うこともできます。署名の設定をしておくで初対面の相手にメールを送る場合などに役に立つため、行っておくと良いでしょう。署名を設定するには、次のようにします。

1. <https://www.coins.tsukuba.ac.jp/webmail/> にログインする
2. 右上のプルダウンメニューから「設定」設定を選択
3. 「全般」タブより「表示名」を選択
4. 「表示名」と署名を入力する
5. **更新** ボタンを押す

図 4.25 に、一般的な署名の例を示します。これを参考に、自分の署名を設定してください。

メールの表示名を更新しますか？

メールアドレス **s999999@coins.tsukuba.ac.jp**

表示名

[返信先](#) [Bcc](#)

✓ 返信の引用テキストの前に署名を挿入する

筑波大学情報学群情報科学類
情報 太郎
s999999@coins.tsukuba.ac.jp

更新

図 4.25: 一般的な署名の設定例

第 5 章

Windows と Linux の使い方

これまでの章では macOS を主な対象として説明がなされてきましたが、COINS にはほかにも Windows 及び Linux が用意されています。この章ではこれらの OS について、その使い方を解説します。

まず、Windows の利用について解説していきます。

5.1 サインインとサインアウト

iMac と同様に DELL 機で Windows を利用する際にも、最初に自分のログイン名とパスワードを入力する必要があります。このことを Windows では一般的にログインではなく「サインイン」と言います。またログイン名は Windows では「ユーザー名」と呼ばれています。Windows 起動後にユーザー名とパスワードを入力する欄が見当たらない場合、一度画面をクリックすると入力欄のある画面に移動します。

5.1.1 ユーザー名とパスワードについて

Windows にサインインする際に使用されるユーザー名は、あなたの COINS のアカウントのユーザ名 (iMac を使用する際のもの) と同一です (例: s9912345)。また、macOS と Windows でパスワードは同期されますのでどちらか一方を変更すれば、双方のログインパスワードが変更されることに注意してください。

Windows 環境を初めて利用する際の注意点

3C113 室, 3C205 室, 3C206 室の Windows 環境に COINS アカウントを作成後初めてサインインする場合 (新入学の情報科学類生や情報科学類の科目を履修する他学類生等) には、必ず筑波大学統一認証システムのパスワード変更のページでパスワード変更を行ってからサインインしてください。このパスワード変更を行わないと Windows 環境へサインインできません。

5.1.2 サインイン後の画面

Windows にサインインすると、「デスクトップ画面 (図 5.1)」が表示されます。この画面は Windows でファイルに関する操作をはじめとする各種操作の拠点となるもので、多くのアプリケーションはこの画面で操作することになります。

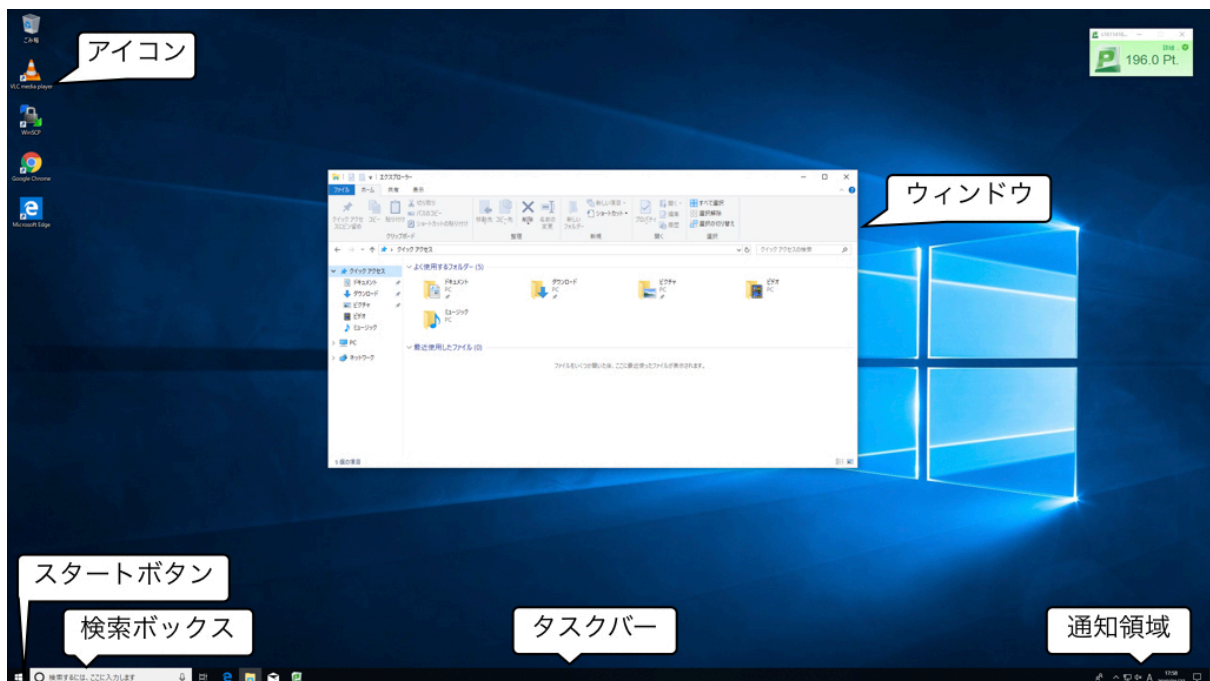


図 5.1: サインイン後の画面

タスクバー

デスクトップ画面では、初期状態で画面の一番下に横長の棒状の領域があります。これを「タスクバー」と呼びます。タスクバーには、現在画面上に表示されているウインドウに対応する、アイコン付きのボタンが表示されます。このボタンをクリックすると、そのウインドウをすぐにアクティブ化することが可能です（図 5.2）。



図 5.2: タスクバーによるウインドウの切り替え

また、表示中のウインドウのタイトルバーに存在する最小化ボタンをクリックすると、そのウインドウは「最小化」されてタスクバー内に格納されます。タスクバーのボタンをクリックすると元に戻ります。

タスクバーの左端には Windows マークが入った「スタートボタン」があり、クリックすると後述するスタートメニューが開きます。このスタートボタンを右クリックすることでアプリケーションを起動したり、シャットダウンを行えるコンテキストメニューを表示できます。

スタートメニュー

スタートメニュースタートメニューでは様々な機能が四角形の「タイル（図 5.3）」で配置され、メールやカレンダーといったアプリケーションを起動したり、天気やニュースといった情報を得ることができます。ま

た、これらのタイルはユーザーの好きなように配置でき、タイルをクリックした状態でドラッグすることで配置を変更できます。スタート画面に不要なタイルを表示させたくない場合は、表示させたくないタイルを右クリックし、コンテキストメニューの「スタートからピン留めを外す」をクリックします。

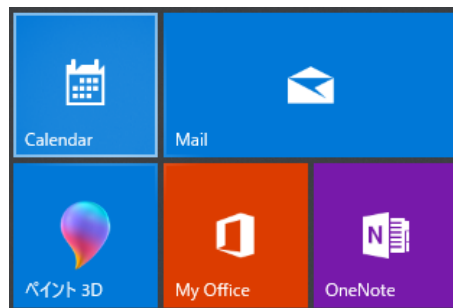


図 5.3: タイル

スタートメニューの左側にはインストールされている全てのアプリケーションの一覧がアルファベット順に表示されます。ここで表示されているアプリケーションは、対応したアイコンをクリックすることで起動できます。

5.1.3 サインアウトとシャットダウン

Windows を利用中に席を外す際には「サインアウト」またはシャットダウンを行ってください。この操作を行わなかった場合、他の利用者に自分のシステムを操作される可能性がありますので危険です。

サインアウトはスタートボタンを押してスタートメニューを表示させ、アカウントのアイコンをクリックしてサインアウトを選ぶことで行えます（図 5.4）。

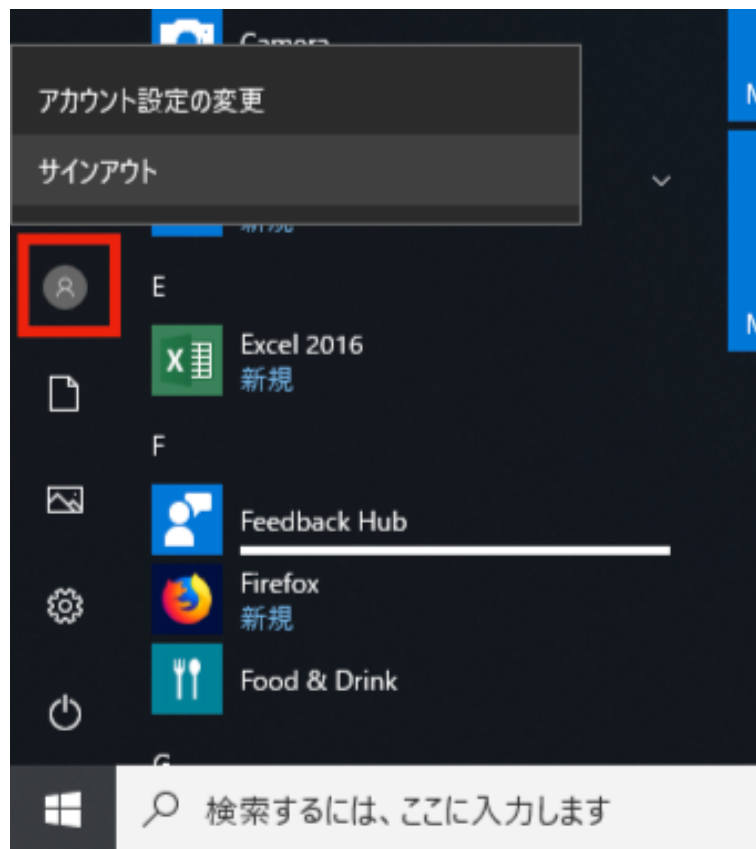


図 5.4: サインアウト

シャットダウンはスタートボタンを押してスタートメニューを表示させ、電源マークをクリックしてシャットダウンを選ぶことで行なえます（図 5.5）。

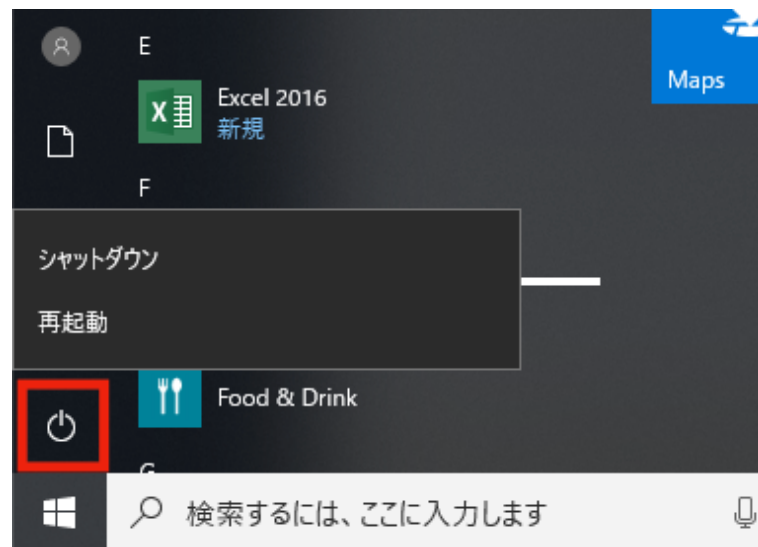


図 5.5: シャットダウン

5.2 ウィンドウの基本操作

Windows でソフトウェアを利用する際は、ディスプレイに表示されるグラフィカルユーザーインターフェース（GUI）で操作することが多くなります。

5.2.1 デスクトップ画面とウィンドウ

デスクトップ画面ではメニューやボタン、アイコンなどをマウスでクリックする等してファイルの作成をはじめとする様々な操作を行うことができます。また、ほとんどの操作はキーボードでも行うことができます*1。ボタンやメニューなどのインターフェースは「ウィンドウ」という領域の上に貼り付けられる形で表示されます（図 5.6）。

*1 様々なショートカットキーが用意されているので調べてみてください。作業時間の短縮につながります。

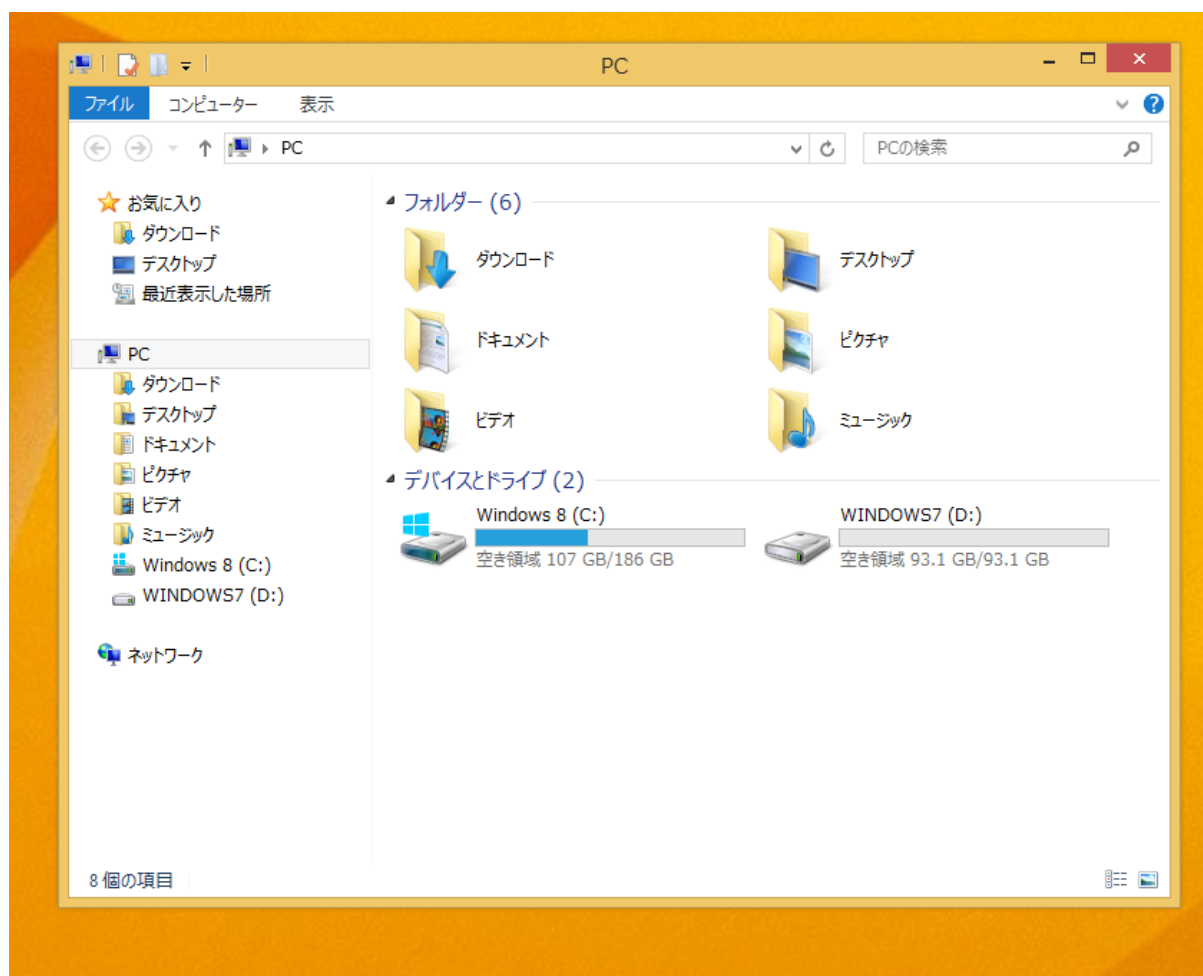


図 5.6: ウィンドウの例

「ウインドウ」とは、Windows のインターフェースの最も基本的な構成要素となるものです。複数のソフトウェアを同時に起動した場合、それらのソフトウェアはそれぞれいくつかのウインドウとして表示されます*2。

例えば、大きな作業机がありその上に書類をたくさん置いた場合、まず今から使う書類を最も上に配置するでしょう。Windows の画面では、複数のソフトウェアを起動してウインドウがいくつか表示されている場合、ユーザーは、複数表示されているウインドウのうち、現在操作したいものを最も手前に表示することができます。

Windows では、最も手前に表示されているウインドウを「アクティブウインドウ」と呼びます。ユーザーは好きなウインドウをいつでもアクティブウインドウにすることができ、これをアクティブ化と呼びます。ウインドウをアクティブ化するには、そのウインドウ内の適当な部分（余白など）をクリックします。

*2 1 つのアプリケーションが複数のウインドウを開く場合もあります。

5.2.2 タイトルバー・リボン・ステータスバー

デスクトップ UI 用のソフトウェアの画面構成には色々なものがありますが、ファイル操作を行うアプリケーションであるエクスプローラのアプリケーションには「タイトルバー」、「リボン」、「ステータスバー」などが付いています（図 5.7）。

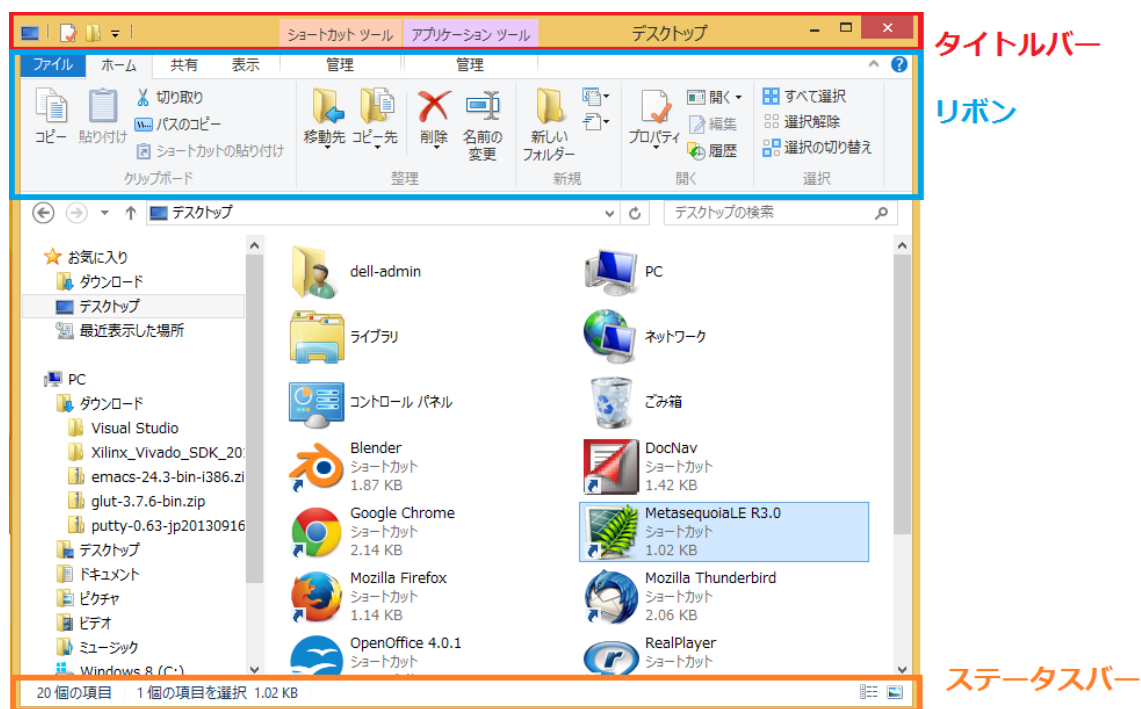


図 5.7: ウィンドウの構成要素

タイトルバー

ウィンドウの上部に貼り付いている領域で、この部分をマウスでドラッグするとウィンドウを移動し、画面上の好きな位置に配置することができます。また、タイトルバーにはそのウィンドウの性質や役割を示す文字列（アプリケーション名など）が表示されています。

またこの部分の右端にはウィンドウの変形や終了を行うためのボタンが用意されています。たとえば「×」ボタンでウィンドウを閉じることができます。これは **[Alt] + [F4]** を同時に押したり、タスクバーのボタンを右クリックして「閉じる」を選択することでも行うことができます。

このタイトルバーにはそのウィンドウで特定の項目が選択されているとき、ツールタイトルがタイトルバーに出現し、これをクリックすることでその項目に対して利用できる操作が後述する「リボン」に表示される「コンテキストタブ」という機能があります。エクスプローラを例にとって説明すると、検索ボックス内をクリックすれば「検索ツール」というツールタイトルと「検索」というコンテキストタブが表示され、検索場所の変更や絞り込み設定を行えます。

また、ファイルの操作を元に戻したり、削除を行ったりといったよく使う操作をすばやく実行することのできる「クイックアクセスツールバー」という機能があります。これは常にタイトルバーの左側にア

アイコン表示され、それぞれクリックすることで対応した操作を即座に行うことができます。アイコン右側の下向き三角のボタンをクリックすることで新しく操作を登録したり、リボン下に表示させることができます。

リボン

タイトルバーのすぐ下に表示されているコマンド一覧の領域です。上側に表示されている、例えばエクスプローラーならば「ファイル」や「ホーム」といったタブをクリックすることで、そのタブの内容に関する機能を下側の領域に表示させることができます。「ファイル」タブをクリックすれば新しくエクスプローラーのウィンドウを開いたり、「ホーム」タブをクリックすればファイルのコピーや貼り付けを行えるボタンが用途ごとにグループ化されて表示されます。

ステータスバー

アプリケーションの現在の状態を表示するための領域です。例えばエクスプローラーならばそのディレクトリに存在するファイル及びフォルダの総数を表す項目数や、選択中の項目数などが表示されます。

5.3 ファイルの操作

5.3.1 デスクトップ UI でのファイル操作

Windows のデスクトップ画面では基本的なファイル操作を、macOS における Finder 同様にマウス操作によるドラッグアンドドロップで視覚的に行えます。Windows でファイル操作を行うための便利なアプリケーションとして、エクスプローラー（図 5.8）があります。

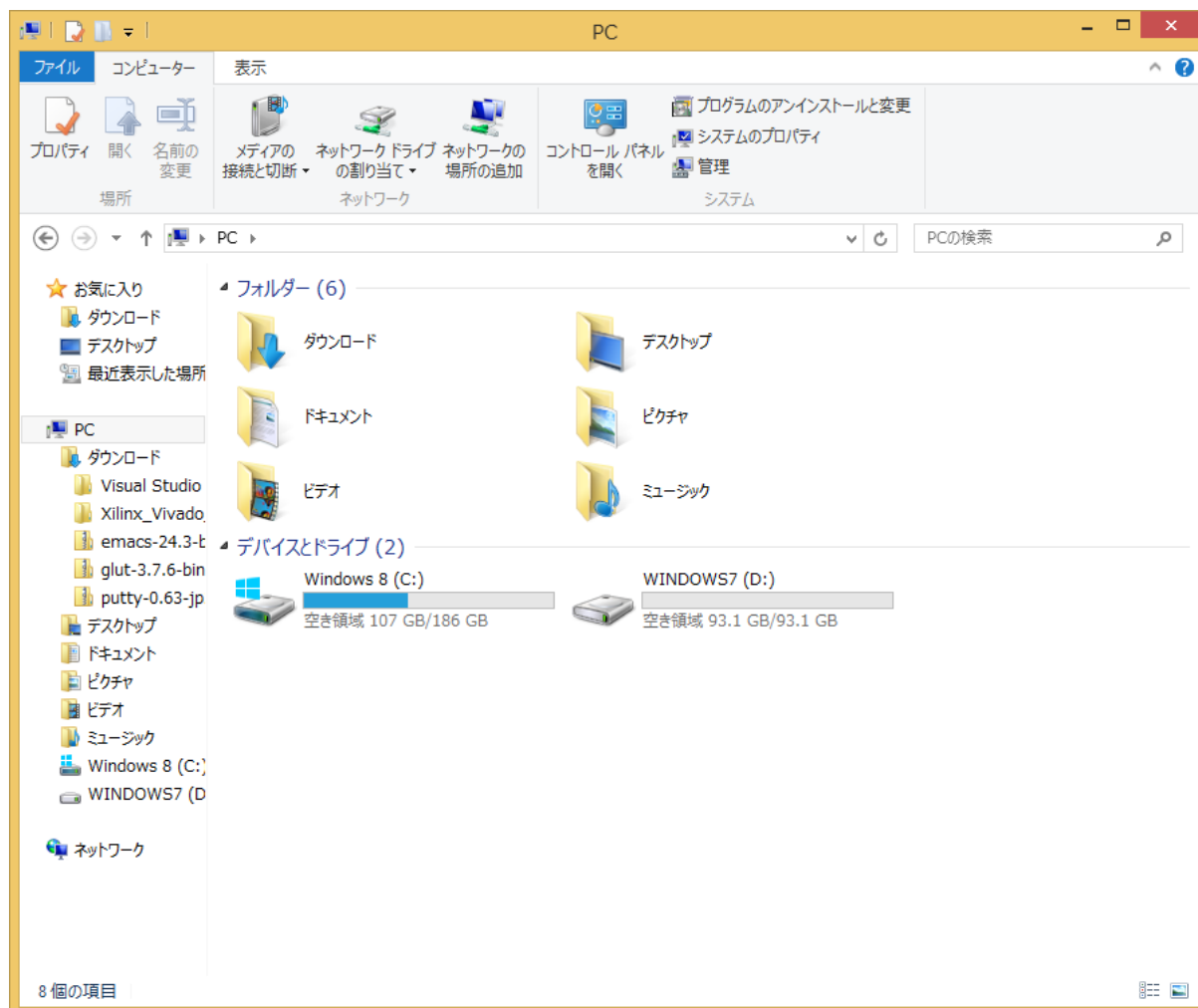


図 5.8: Windows エクスプローラーの画面

エクスプローラーを起動するにはスタートボタンを右クリックして「エクスプローラー」を選択するか、タスクバーにある検索ボックスに explorer と入力すると表示される、「エクスプローラー」をクリックします。エクスプローラーでは、左側の領域（ペインと呼びます）に現在開いているフォルダを中心とした木構造が、右側にそのフォルダ内のファイル一覧が表示されます。

5.3.2 フォルダ構造

Windows も macOS のファイルシステムと同様にファイルを木構造で管理します。macOS におけるディレクトリのことを、Windows では一般的に「フォルダ」と呼びます。つまり、Windows のファイルシステムはフォルダを中心とした木構造となっており、各フォルダにファイルを配置して利用します。

パス記述

Windows において、絶対パス名は以下のように記述します。この例では H ドライブ下の test というフォルダの絶対パス名です。

```
H:¥ test
```

最初のドライブ文字の後には「:」が続きます。また、Windows におけるフォルダ構造の区切り文字には円記号「¥」を用います^{*3}。Windows は基本的にマウス操作が中心なのでパス記述について意識することはありませんが、コマンド入力を行ってファイル进行操作する場合などに使うことになります。

拡張子

ファイルの種類を示す拡張子によって、ファイルを開くためにデフォルトで使用するアプリケーションが決定されます。たとえば拡張子が.docx であれば Microsoft Word が起動する、といった対応が管理されています^{*4}。

実行ファイル

Windows では拡張子が.exe となっているファイルを実行可能ファイル（プログラムファイル）と呼びます。新しいアプリケーションのインストーラや、アプリケーションの実行で利用することが多いファイルの拡張子です。^{*5}

5.3.3 ファイルの移動、 コピー

ファイルをコピーするには、いくつかの方法があります。まず、コピーしたいファイルを右クリックして「コピー」をクリックし、コピー先のフォルダ上を右クリックして「貼り付け」を押すと、そのファイルがコピーされます。同様にフォルダをコピーして貼り付けすると、フォルダ内のファイル全体がコピーされます。

この方法は少し手間がかかるので、もっと簡単な方法を紹介します。コピーしたいフォルダやファイルをドラッグし、**Ctrl**を押しながらコピー先にドロップするだけでコピーが可能です。同様に **Shift**を押しながらドロップすると、コピーではなく移動となります。

ファイルやフォルダを左ボタンではなく右ボタンでドラッグアンドドロップすると、ドロップした地点にメニューが表示されます。ここでコピー、移動、ショートカットの作成を選択することもできます。

5.3.4 ショートカットの作成

Windows では、フォルダの深い位置にあるファイルなどを簡単に呼び出すことができる機能としてショートカットというものがあります（図 5.9）。ショートカットをダブルクリックすると、そのショートカットのリンク先のファイル（ターゲットファイル）をダブルクリックした場合と同等の効果が起こります。

^{*3} 正確には「H:」のあとに「¥」が続く表記であり、これにより H ドライブをルートとしたときの絶対パス名を表します。

^{*4} デフォルトで使用するものとは異なるアプリケーションで開きたい場合は、そのアプリケーションのアイコンにファイルをドロップするなど、明示的に開くアプリケーションを指定する必要があります。

^{*5} ほかに .COM、.BAT、.CMD ファイルなども実行可能ファイルなのでダブルクリックすると起動します。

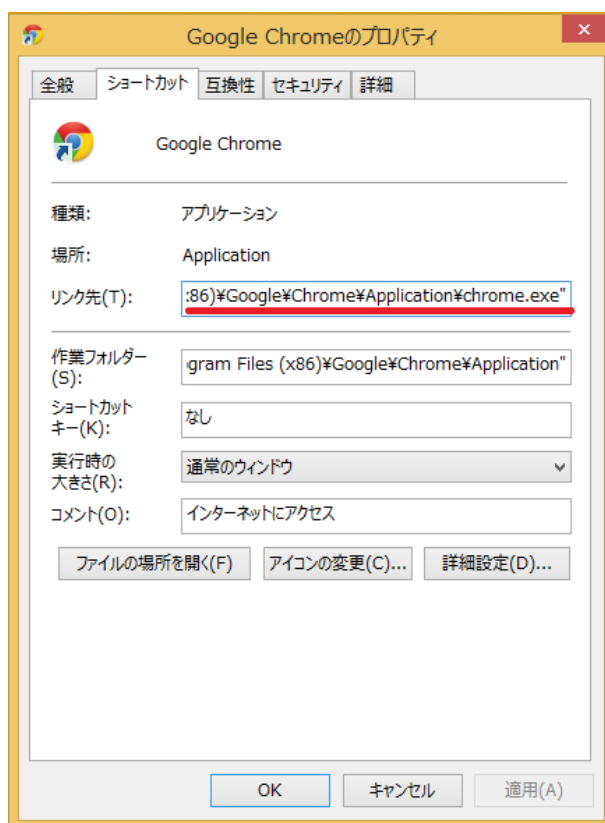


図 5.9: ショートカットのプロパティ画面の例

macOS でも同等のものとしてシンボリックリンクがありますが⁵、Windows のショートカットの場合は実体がない“.LNK”という拡張子のファイルになっています（拡張子は通常表示されません）。Windows でも、ショートカットのターゲットとしてファイルだけではなくフォルダも指定できます⁶。

5.3.5 ファイルの新規作成

一般的に、ファイル名はアプリケーションで編集した内容を最初に保存する際に決定し初めて保存することが多いですが、先に空のファイルを作成しておいて、それをダブルクリックしてからアプリケーションで編集し上書き保存する、という操作方法のほうが便利な場合もあります。Windows では新しいファイルを作成したい場所を右クリックして、メニューの中の「新規作成」から新しく作成したいファイルの種類を選択すると、その種類のファイルが自動的に作成されます。

5.3.6 フォルダ名/ファイル名の変更

すでに存在するファイルやフォルダの名前を変更するには、変更したいアイコンをクリックして[F2]を押すか、右クリックメニューの中から「名前の変更」をクリックします（図 5.10）。

⁵ ショートカットのリンク先としてはフォルダ以外にも、コントロールパネルなどの特殊なオブジェクトが指定できます。

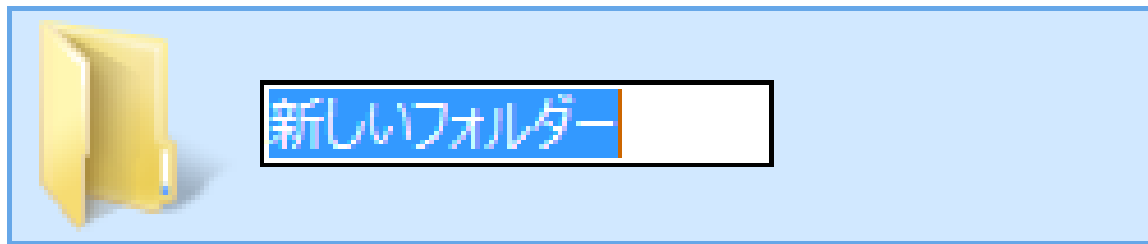


図 5.10: 名前の変更

5.4 プロセス管理

5.4.1 タスクマネージャー

macOS では起動中のプロセス一覧を表示するために `ps` コマンドを、プロセスを強制終了するために `kill` コマンドを使用しますが、Windows ではこれらの操作を「タスクマネージャー」で行うことができます

「タスクマネージャー」も 1 つのアプリケーションです。起動するには、タスクバーを右クリックして「タスクマネージャー」を選択するか、`Ctrl` + `Alt` + `Delete` キーを同時に押し、「タスクマネージャー」をクリックします。Windows10 のタスクマネージャーはデフォルトでは簡易表示されています（図 5.11）。

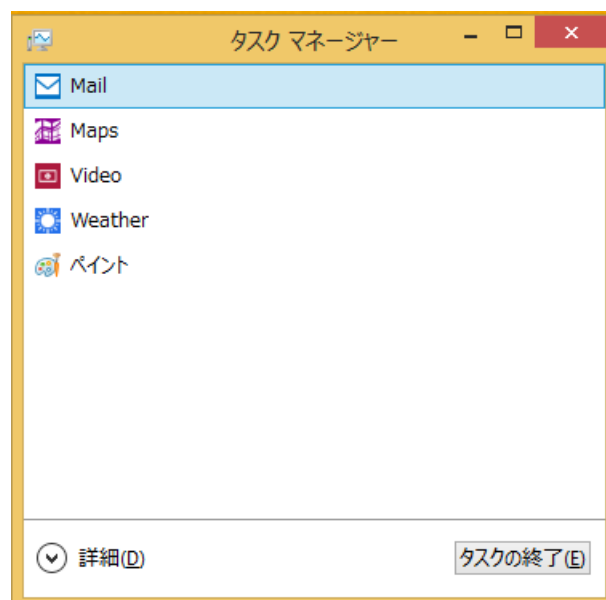


図 5.11: タスクマネージャー（簡易）

ここでは、タスクマネージャーウィンドウの右下に存在する、「詳細」ボタンを押すと表示される詳細画面について解説していきますが、プロセスを終了させるだけならば簡易画面でも行えます。表示されているプロ

セスの一覧から終了したいプロセスを選択し、ウィンドウ内右下の「タスクの終了」ボタンを押せば選択されたプロセスを終了できます。

詳細画面について解説していきます。「プロセス」タブでは Windows 上で起動しているすべてのプロセスが表示されます (図 5.12)。

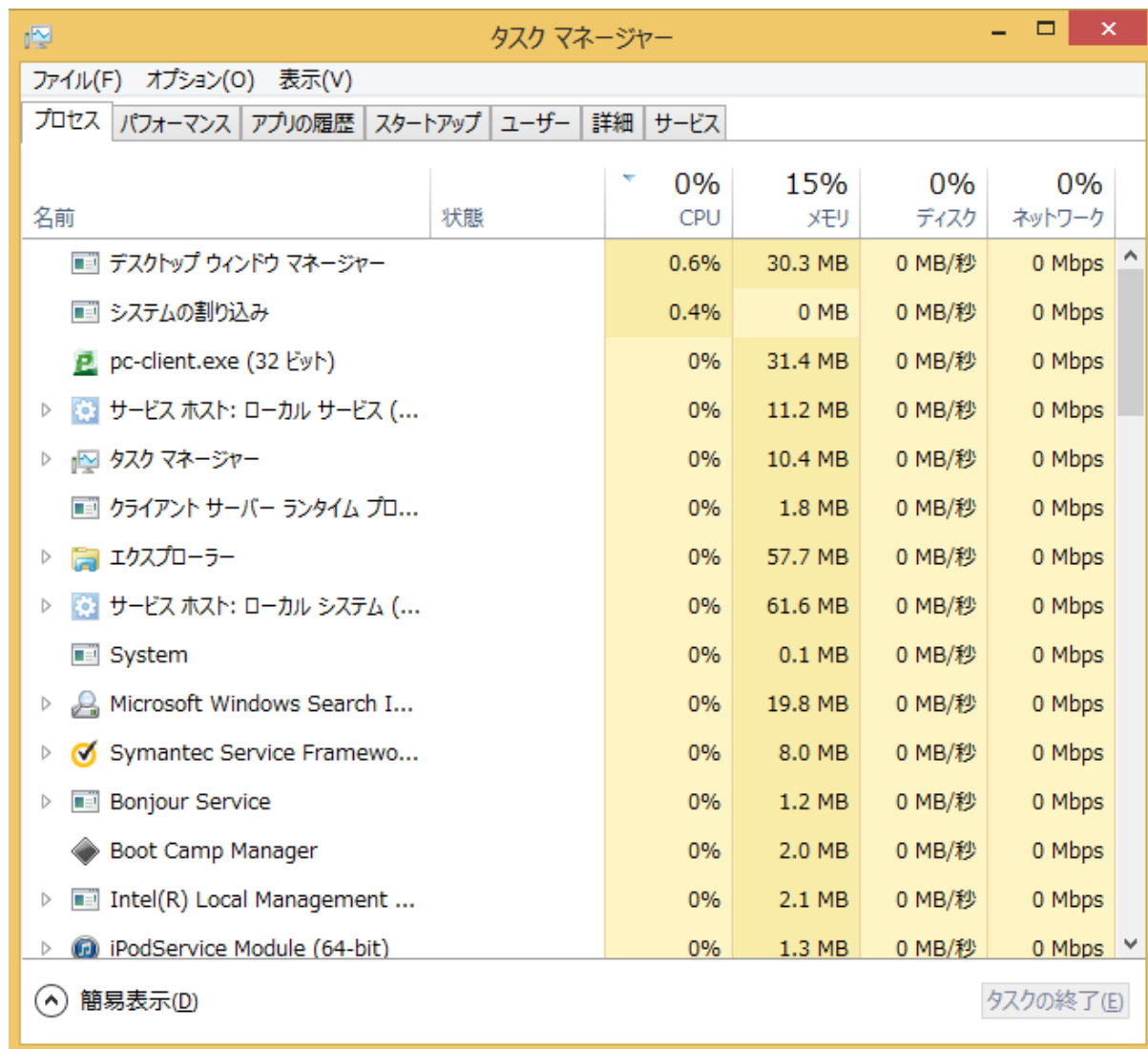


図 5.12: タスクマネージャーのプロセスタブ

また、それぞれのプロセスについて「CPU 使用率」、「メモリ使用率」、「ディスク書き込み量」、「ネットワーク使用量」といった「リソース使用量」が表示されます。この時、プロセスはこれまでウインドウで扱ってきた「アプリ」と画面上に表示されていない「バックグラウンドプロセス」に分けられ、それぞれ「状態」タブでプロセスの応答状態が表示されるため、応答していないなどの理由で終了させる際に対象を見つけやすくなっています。表示されているプロセスに対して右クリックを押すことで、直接プロセスの終了や、プロセスの実行状態を詳細に記録した「ダンプファイル」の生成を行うことができます。ここでは詳細を書きませんが、ダ

ンプファイルは解析することで異常終了した原因を発見する際に役立ちます。

詳細画面にはこの他にもタブが存在します。

「パフォーマンス」タブ

総 CPU 使用率等，総合使用状況を表示する。

「アプリの履歴」タブ

利用中のユーザーアカウントのリソース使用量の履歴を表示する。

「ユーザー」タブ

サインイン中のユーザーのリソース使用量を表示する。

「詳細」タブ

PID 等を含むプロセスのより詳細な実行状況を表示する。

「サービス」タブ

各種アプリケーションのアップデートなどユーザーとのやり取りを必要としないプロセスの一覧を表示する。

計算機室の利用においてすべての機能を使いこなす必要はないため，計算機室の利用方法の観点から役立つ「ユーザー」タブを取り上げて説明します。「ユーザー」タブでは他ユーザーのリソース使用量も表示され，他ユーザーの利用によって自分の利用が妨げられるといった場合にはその利用を切断させることができます（図 5.13）。これは一つの計算機を複数人で利用する場合において，非常に有用な機能の一つです。

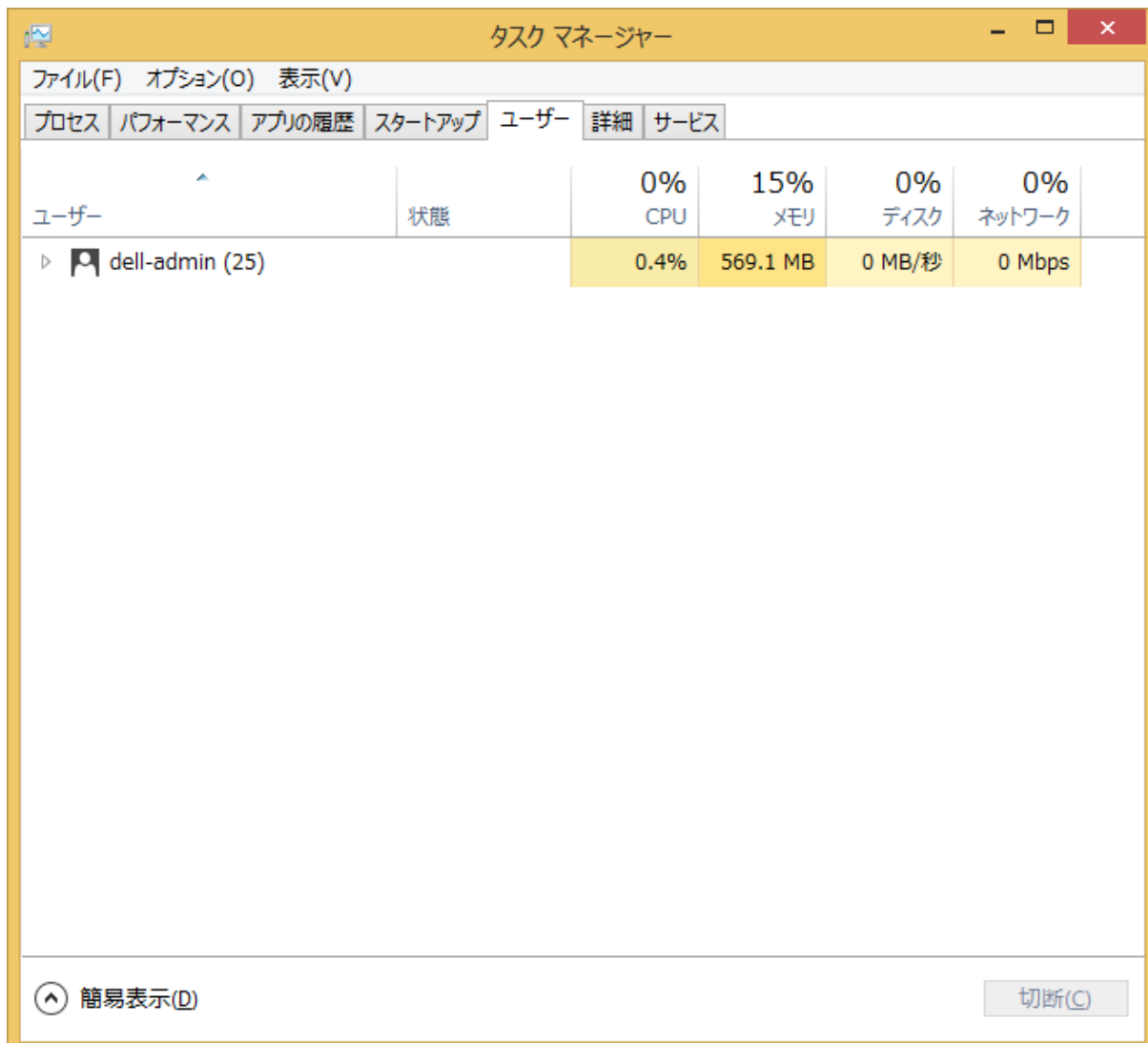


図 5.13: ユーザータブ

5.4.2 プロセスに関する注意事項

Windows でも、macOS のプロセスと同様にプロセス間には親子関係があります。ただしその結合関係はあまり密接ではなく、親プロセスが終了しても子プロセスは終了しません。

Windows で、あるプロセス（親プロセス）が起動した子プロセスや孫プロセスをすべて終了したい場合は、タスクマネージャーでプロセスを右クリックして「プロセスツリーの終了」をクリックしてください。

5.5 ホームディレクトリの利用

5.5.1 ホームディレクトリへのアクセス方法

COINS では macOS で使用する自分のディレクトリ（ホームディレクトリ）に、Windows からアクセスすることができます。これにあたって COINS の Windows マシンでは、5.7.2 項で説明したドライブの 1 つである H ドライブに、このディレクトリを仮想的なドライブ（ネットワークドライブ）としてマウントすることでこれを行っています。ログオンしたら「スタートメニュー」内の「コンピュータ」からエクスプローラーを開いて、H ドライブにアクセスしてみてください。するとあなたが macOS で使っているホームディレクトリの内容が表示されるはずです。H ドライブには実際は COINS のシステムにおけるファイルサーバ上の自分のディレクトリがマウントされており、あたかもハードディスクにアクセスしているかのように、使用することができます。

Windows マシンを使っている時に、ファイルを保存する場合は H ドライブ上のホームディレクトリを使用してください。なお、このとき注意すべき点として macOS と Windows 間での文字コードの違いが挙げられます。それぞれ日本語文字コードが異なっていると、日本語の名前を持ったファイル/ディレクトリは相互にアクセスできない可能性があります。

5.5.2 同時サインインに関する注意

iMac 同様、Windows マシンにおいてもどのマシンを使っても自分用の環境を利用できるように、「ユーザプロファイル」という機能が有効になっています。このプロファイルは、マシンのサインイン時にファイルサーバから読み込まれ、サインアウト時にファイルサーバに変更を書き込みます。この仕組みのため、同時に複数台のマシンを起動したときにプロファイルの変更を行うと書き込みと読み込みの時刻関係によって問題が生じる場合があります。基本的には複数同時にサインインしないよう気をつけてください。

これらのプロファイルは、COINS の環境ではユーザのホームディレクトリ以下の“WinFiles”というディレクトリの下に保存されます。Windows のデスクトップやピクチャ、ダウンロードに保存したファイルなど、ユーザそれぞれのデータについてはログオフ時にこちらのディレクトリに自動的に保存され、次回 Windows マシンにログオンした時に再度利用可能になります。

5.6 Windows からの印刷

Windows 上で作成した Word や Excel ドキュメントを印刷したい場合、直接プリンタを選択して印刷することが可能です。大抵のアプリケーションでは、印刷する際にプリンタを選択するダイアログボックスが表示されます。

印刷画面においてプリンターを選択すると、そのプリンターのオンライン状態や設置されている場所が表示されます。この時、自分の利用している計算機室ではない場所のプリンターを使って印刷すると、その場所まで印刷物を回収に行かねばならず、大変面倒です。印刷する場合は利用するプリンターの場所に注意してください。

表 5.1: 利用可能なプリンター一覧

部屋番号	プリンタ名	種別
3C113	aloe-pr1	モノクロ
3C113	aloe-pr2	モノクロ
3C113	aloe-pr3	モノクロ
3C113	aloe-pr4	モノクロ
3C113	color-c113	カラー
3C205	crocus-pr1	モノクロ
3C205	crocus-pr2	モノクロ
3C205	color-c205	カラー
3C206	dahlia-pr1	モノクロ

プリンタの選択ダイアログボックスで「プロパティ」をクリックすると、いくつかのオプションを設定可能です。割り付け印刷など行いたい場合はここから設定を行ってください。この設定内容は、ユーザごとに保存されます。

5.7 Windows での日本語入力

COINS の Windows で日本語入力を行うには、**Microsoft IME (Input Method Editor)** を使用します。多くのアプリケーションでは初期状態は直接入力モードになっています。キーボードの[半角/全角]キーを押すと、MS-IME が有効になり日本語入力ができるようになります*7。MS-IME での日本語入力は非常に簡単です。基本的にはローマ字入力でひらがなを入力し、変換したい場所で[Space]を押すと漢字変換が行われます。複数の変換候補がある場合は[Space]を何度か押すと次々に変化していきます(図 5.14)。カーソルやマウスを使うことで素早く選択することもできます。変換候補を定めたら[Enter]を押して確定します。変換候補の探索中にカーソルキーと[Shift]を組み合わせるとカーソルを移動することで、変換する語の区切りを変えることもできます。再度[半角/全角]キーを押すと元の直接入力に戻ります。

*7 iMac を利用している場合のキーボードには[半角/全角]キーはありませんが、スペースキーの隣に[ひらがな]キーと[英数]キーがあります。

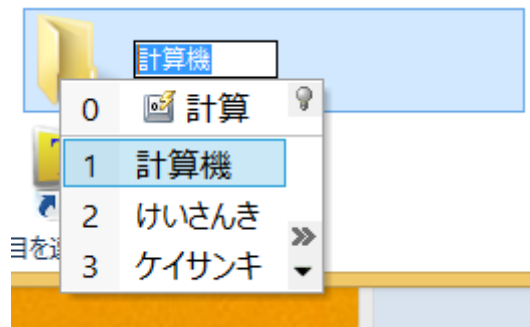


図 5.14: 日本語入力と漢字変換

MS-IME の設定を変更したい場合は、画面右下に「あ」や「A」と表示されているアイコン（図 5.15）を右クリックして「プロパティ」を選択してください。ローマ字入力からかな入力に切り替えるなどの変更が可能です。

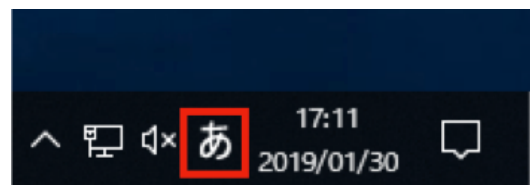


図 5.15: MS-IME

5.8 Windows コマンドの利用

macOS/Linux ではほとんどのファイル操作をコマンドラインから行いますが、Windows では基本的にマウス操作が中心となるためコマンドラインを使う局面があまり多くありません。しかしながら、macOS などと同様の操作をコマンドプロンプト（図 5.16）から行うことが可能であり、慣れてくるとコマンドプロンプトを使った方が速く効率良く作業できる場合もあります。

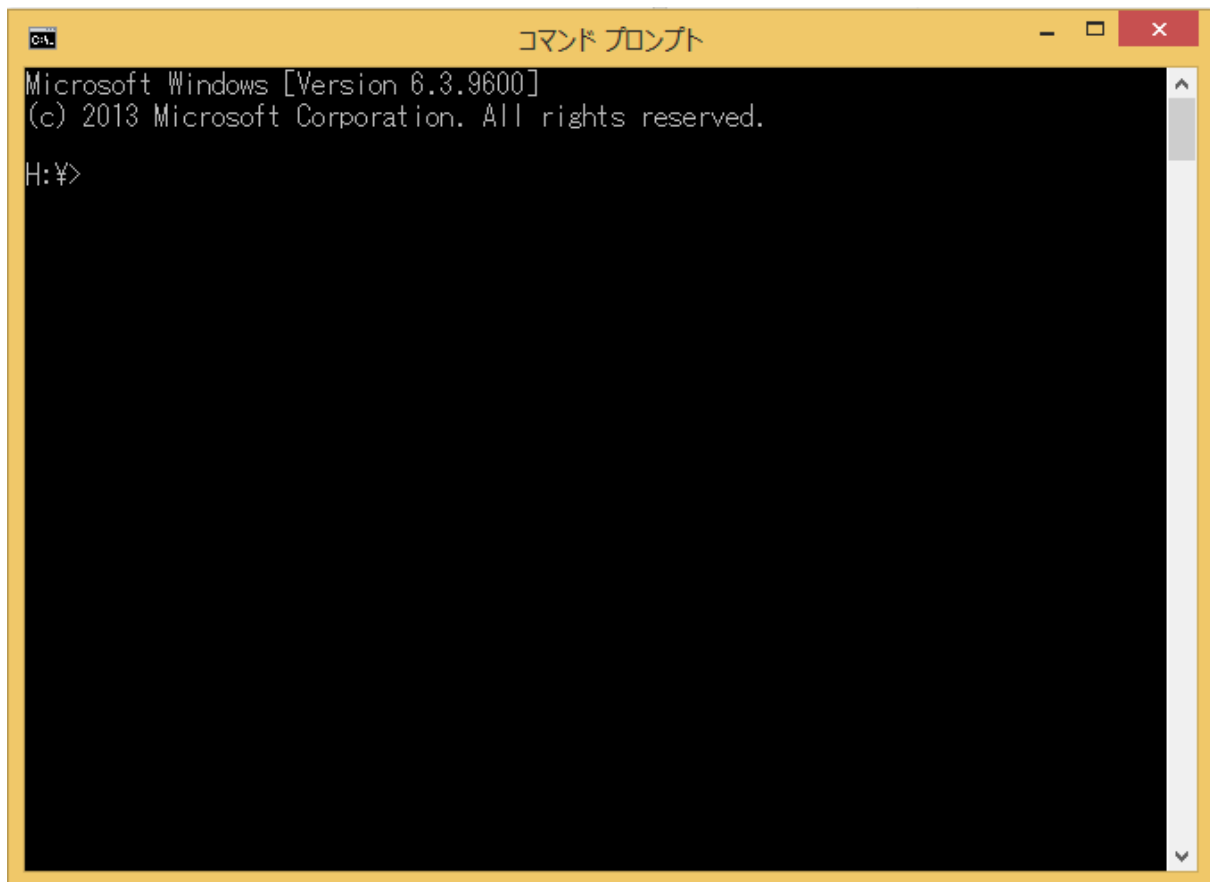


図 5.16: コマンドプロンプトの画面

ここではコマンドプロンプトの使用方法と、Windows における主要なコマンドについて解説します。macOS/Linux のコマンドと Windows のコマンドは違うものが多いですが、ここでは主要なコマンドについて macOS でのコマンドとの対応付けを行いながら解説します。

5.8.1 コマンドプロンプトでよく使われるコマンド

macOS ではコマンド入力を行うためにターミナルエミュレータ（コンソール）を起動しますが、Windows でも同様のツールとして「コマンドプロンプト」というものがあります。起動する際はスタートボタンを右クリックして「コマンドプロンプト」をクリックするか、スタート画面のアプリ一覧から選択してください。起動する際はタスクバーにある検索ボックスに「cmd」と入力し表示される、「コマンドプロンプト」を選択してください。

コマンドプロンプトを起動すると以下のような内容のプロンプトウィンドウが表示されます。

```
Microsoft Windows [Version 6.3.9600] (
c 2013 Microsoft . Corporation All rights . reserved

H:¥ >_
```


ここにコマンドを入力します。デフォルトでは、プロンプト文字列にはカレントディレクトリ（自分が現在いるディレクトリ）の絶対パス名が表示されます。上の例では「H:¥」がこれに当たります。

dir コマンド

指定されたディレクトリ内のファイルやサブディレクトリの一覧を取得します。macOS における ls コマンドと同じような働きをします。引数に何も指定されていない場合は、カレントディレクトリの内容が表示されます。

```
H:¥ test> dir
H:¥ test  のディレクトリ

2014/03/08  13:59    <DIR>          .
2014/03/08  13:59    <DIR>          .
2014/03/08  13:59                0 data.txt
2014/03/08  13:58    <DIR>          hello
```

コマンドプロンプトにてオプションは- もしくは/ を使って行えます。例えば dir のオプション一覧を確認するには、dir /?と入力してヘルプを表示してください。これ以降に説明するほとんどの Windows コマンドでも、引数として"/?"をつけて実行すると簡単なヘルプが表示されます。

cd コマンド

現在の作業フォルダ（カレントディレクトリ）を変更します。絶対パス名または相対パス名を指定できます。相対パス名で指定する場合には macOS と同様に、“.”は現在のフォルダ、“..”はフォルダ構造上で一つ上の階層のフォルダを示します。ディレクトリ名を指定せずに cd と入力した場合には下記の実行例のように現在のフォルダの絶対パス名を表示します。これについては macOS における pwd と同様の働きをします。

```
H:¥ test>cd hello
H:¥ test¥ hello>cd
H:¥ test¥ hello

H:¥ test¥ hello>cd .
H:¥ test>
```

copy コマンド

指定したファイルをコピーします。UNIX の cp コマンドと同等です。この例では「hoge.txt」というファイルをフォルダ構造上で同じ階層のフォルダにコピーします。

```
H:¥ test> copy hoge.txt .
```

1 個のファイルをコピーしました。

複数ファイルについてもワイルドカードにより扱うことができます。ワイルドカードは検索等の際に利用する特殊文字のことで、具体的には「*」や「?」が当たります。「*」は長さ 0 文字以上の任意の文字列に一致し、

「?」は任意の1文字に一致します。copy の対象にワイルドカードを用いることで、例えばテキストファイルのみを同じ階層のフォルダに一括してコピーすることが可能になります。

```
H:¥ test> copy *.txt .
```

2 個のファイルをコピーしました。

この例の、txt 部分を、jpg にすれば、フォルダ内の全ての.jpg ファイルのコピーとなります。

move コマンド

ファイルのコピーではなく移動をします。移動先にはディレクトリ名またはファイル名を指定します。同じディレクトリ内でのファイル名を指定した場合は、ファイル名の変更と同じです。このコマンドは macOS における mv コマンドほぼ同等です。copy コマンドと同じくワイルドカードを用いた複数ファイルの指定が可能です。

```
H:¥ test> move atad.txt hello¥ datadata.txt
```

1 個のファイルを移動しました。

ren コマンド

ファイル名やディレクトリ名の変更を行います。コマンド名は”rename”の省略です。

```
H:¥ test> ren data.txt dat.txt
```

del コマンド

ファイルを削除します。/S オプションをつけるとサブディレクトリも再帰的に削除します。また/F オプションをつけると読み取り専用ファイルについても強制的に削除します。macOS における rm コマンドとほぼ同等の機能を持ちます。/Q オプションをつけることで再帰的に削除する場合に表示されるダイアログを消すことができます。

```
H:¥ test> del hello /S /Q
```

削除したファイル - H:¥ test¥ hello¥ datadata.txt

rmdir コマンド

ディレクトリを削除します。del コマンドと同じく/S オプションをつけるとディレクトリ内のファイルとサブディレクトリをすべて再帰的に削除してしまうので注意してください。

cls コマンド

コマンドプロンプトウィンドウ内に表示されている文字を消去します。コマンドプロンプトの作業フォルダなどの状態は維持されます。

5.9 メール設定

Windows10 上でメールの送受信を行う際には、Thunderbird を利用することができます。macOS を利用する場合と同じ方法で設定ができますので、第 4.2 節を参照してください。

5.10 Linux の利用

COINS の PC では起動時に Linux を選択すると Linux のディストリビューションの 1 つである Ubuntu を利用することができます（図 5.17）。これを用いることで、macOS ではできない Linux 独自の作業などを行うことができます。Linux についても macOS や Windows 同様にユーザ情報として COINS のユーザ名とパスワードを用いてログイン等を行うことができます。

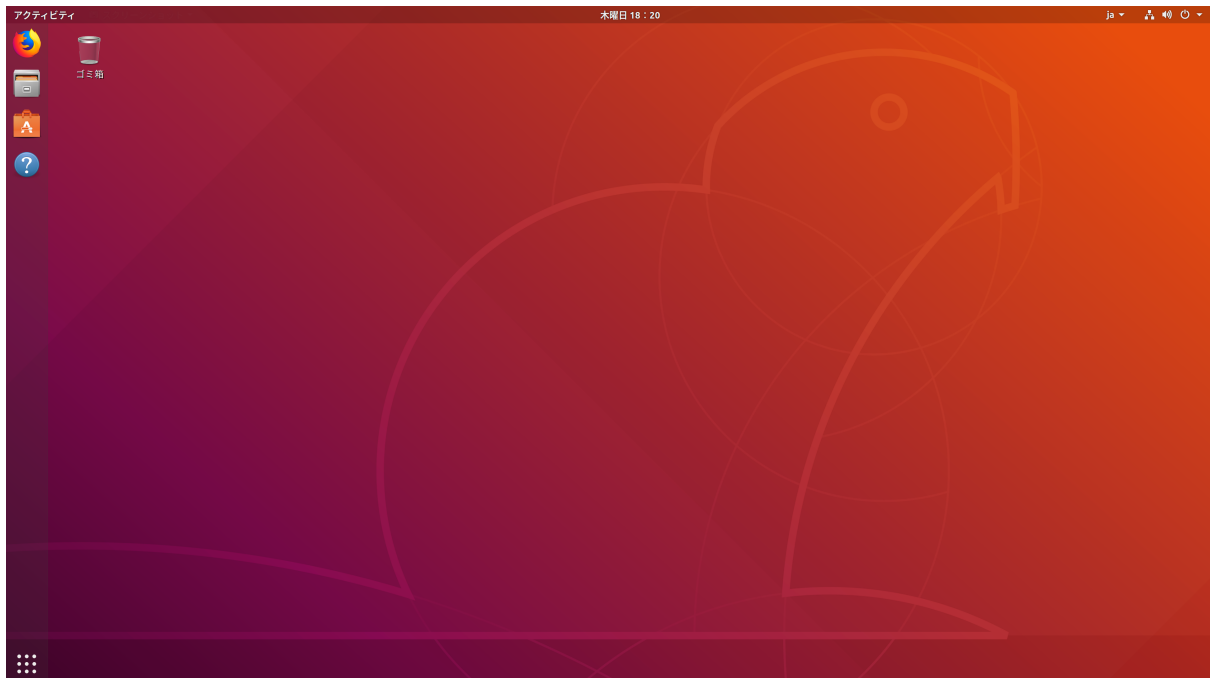


図 5.17: Ubuntu のデスクトップ

Linux においても基本的なコマンドは macOS と同じです。第 2 章で解説したコマンドをターミナルに入力すると同じ動作をします。ただし open コマンドは Linux マシンにはありませんので注意してください。

第 6 章

L^AT_EX

本章では、組版処理ソフトウェアである L^AT_EX について紹介します。この章の目的は簡単なレポートから卒業論文まで書けるような L^AT_EX の記法を身につけることです。

6.1 L^AT_EX の概要

L^AT_EX^{*1}は本やレポート、論文を作成するためのツールです。Microsoft の Word や Just System の一太郎などと同じような用途のソフトウェアです。最初は通常のワープロソフトに比べ、覚えることも多く面倒に感じるかもしれませんが、慣れてしまえば非常に有用なツールです。

L^AT_EX の特徴としては例えば次のようなものが挙げられます。

マークアップ言語である

L^AT_EX はマークアップ言語^{*2}の一つです。この方式によって文書の体裁を論理的に記述できます。

数式や図を綺麗に出力できる

MS Word にも数式エディタが存在しますが、それよりも格段に綺麗な、かつ表現力の高い数式を出力できます。

無償で利用できる

L^AT_EX およびその関連ツールは無償で使うことができます。

複数の OS で動作する

Windows や macOS, Linux 系など様々な OS の上で動作します。

例えばこの手引きも L^AT_EX によって作成されています。

6.1.1 T_EX と L^AT_EX

L^AT_EX を学習するにあたって、同時に T_EX^{*3} という言葉についても学習する必要があります。この二つの意味を正しくおさえておくことは、今後 T_EX や L^AT_EX について学習したり Web などでも情報を集めたりする際に役に立つでしょう。なお、L^AT_EX を利用して pdf ファイルを作成する具体的な方法は 6.2 節から説明しま

^{*1} 「らてふ」や「らてつく」と発音します。どちらで発音しても構いません。その他にも「れいてつく」などと発音することもあります。

^{*2} タイトルや見出しといった文書の構造と、フォントなどといった見た目の情報を文書の中に埋め込むという、文書を記述する方法の一つです。

^{*3} 「てふ」もしくは「てつく」と発音します。

す。まずは実際に L^AT_EX を使ってみたい、という方はここから取り組んでみてください。

T_EX とは

T_EX とは Donald Knuth という人が、1978 年に作成したプログラム言語です。プログラム言語ですから、Java や C 言語と同様に処理系と呼ばれるプログラムがあり、ソースコードから処理系を用いて何らかの成果物を得ます。先ほどの図 6.2 にあるように、T_EX ファイル（ソースコード）は、 ϵ -pT_EX などの T_EX の処理系を用いて DVI ファイルへと変換します。これは Java のソースコードを `javac` といった処理系を用いて、実行ファイルを得る関係と似ています。

T_EX の処理系は Knuth が作ったもの以外にもたくさんあり、日本では主に ϵ -pT_EX や ϵ -upT_EX という処理系が使われています*⁴。

つまり T_EX とは厳密に言うならば、Knuth が作ったプログラム言語かあるいはその処理系を指すということになります。

L^AT_EX とは

T_EX が Knuth が作ったプログラム言語やその処理系を指すことに対して、L^AT_EX とはもともとプログラム言語 T_EX で書かれたプログラムのことです。1980 年に Leslie Lamport という人によって作られました。

L^AT_EX は T_EX をより便利に使えるようにするプログラムで、T_EX の世界ではこれらをマクロパッケージ*⁵と呼びます。Java など汎用言語の世界ではこのようなプログラムのことをライブラリと呼ぶことがあります。が、その関係と似ています。

この T_EX と L^AT_EX の関係を図 6.1 にまとめました*⁶。

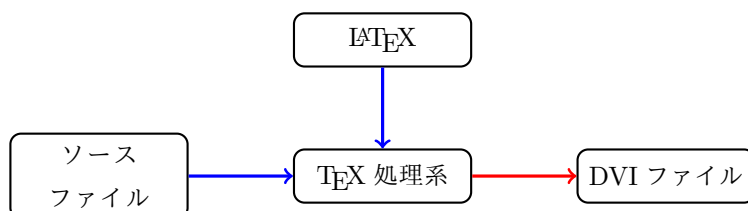


図 6.1: T_EX 処理系とマクロパッケージ L^AT_EX の関係

まず皆さんは Emacs などのテキストエディタを用いて文書を L^AT_EX ファイル（ソースファイル）へ内容を書き込み、それを ϵ -pT_EX など T_EX の処理系に渡します。 ϵ -pT_EX など T_EX の処理系はコンピュータの中から自動的に L^AT_EX を探し出し、これを読み込んで最終的な文書を生成します*⁷。

現在 L^AT_EX は世界中で使われるようになり、少なくとも日本語圏においては多くの文書が L^AT_EX を前提として作成されるようになったので、いつしか“L^AT_EX”が“L^AT_EX パッケージを前提とした T_EX”という意味合いで用いられるようになりました。ですが厳密に言えば L^AT_EX は T_EX で実装されたプログラムです。

*⁴ 有名な処理系については図 6.9 にまとめてあります。

*⁵ あるいは“マクロ”や“パッケージ”と省略することもあります。

*⁶ 正確に言えば、私たちが通常 L^AT_EX を用いる際は L^AT_EX パッケージを T_EX 処理系によって一度コンパイルして得られたフォーマットファイルを用います。

*⁷ 脚注*⁶にあるように、正確には L^AT_EX パッケージをコンパイルして得られるフォーマットファイルを読み込むことになります。また L^AT_EX を読み込まずに T_EX 処理系を使うこともできますが、この章では解説しません。

また、これに伴って、“T_EX”という言葉が“L^AT_EX を使っていない純粋な T_EX”という意味に解釈されることがあります*⁸。ただ、皆さんがマクロパッケージとしての L^AT_EX を話題にすることは少ないでしょうから、この章では L^AT_EX という言葉を、“L^AT_EX パッケージを前提とした T_EX”という意味で用います。

■ L^AT_EX 2.09 と L^AT_EX 2_ε L^AT_EX が T_EX で実装されたマクロパッケージ（プログラム）であるということを前節では述べました。プログラムは“Windows 10”や“macOS 10.14 Mojave”といった OS のようにバージョンを持ちます。L^AT_EX も同様にバージョンがあり、現在は L^AT_EX 2_ε というバージョンの L^AT_EX が使われています。

L^AT_EX 2_ε が登場したのは 1993 年で多くの時間が経過しているのです。現在 L^AT_EX といった場合は L^AT_EX 2_ε のことを指します。この L^AT_EX 2_ε より前の L^AT_EX のことを“L^AT_EX 2.09”と区別します。

6.1.2 ε-pT_EX と dvipdfmx

T_EX ファイルから最終的な PDF ファイルを得るためには、図 6.2 で示したように次の二つのプログラムを用いる必要があります。

ε-pT_EX L^AT_EX ファイルを DVI ファイルへコンパイルするプログラム

dvipdfmx DVI ファイルを PDF ファイルへコンパイルするプログラム

ε-pT_EX

Knuth が作った T_EX を改造して、日本語や縦書きができるようにした pT_EX と、さらに NTS team によって拡張された ε-T_EX をマージした処理系が ε-pT_EX となります。Unicode に対応していない*⁹という欠点がありますが、Web の資料や L^AT_EX に関する日本語書籍の多くが ε-pT_EX を前提としていることが多いので、本章では ε-pT_EX を採用します。

情報科学類の計算機環境にて ε-pT_EX を用いる場合は、macOS のターミナルにて `platex` というコマンドを実行します*¹⁰。すると次のような表示が得られるはずです。

```
$ platex↵
This is e-pTeX, Version 3.1415926-p3.4-110825-2.6 (utf8.euc) (TeX Live 2013)
restricted \write18 enabled.
**
```

このコマンドによって、ε-pT_EX (e-pTeX) が起動します。起動が確認できたら、`ctrl` + `c` でひとまず ε-pT_EX を終了します。

もし pTeX という表示が出た場合、それは ε-pT_EX ではなく pT_EX という別の処理系です。恐らく使用しているコンピュータにインストールされた L^AT_EX の環境が古いのだと思われます。6.24.1 節を参考するなどして最新の L^AT_EX 環境へアップデートしましょう。

*⁸ この問題を解決するために、L^AT_EX などを用いていない純粋な T_EX のことを“plain T_EX”と呼び、処理系の話題と区別します。

*⁹ 文字コードが UTF-8 のファイルを扱うことはできますが、これは内部で `nkf` などを用いて一旦 EUC などの文字コードへ変換しているだけです。

*¹⁰ 情報科学類の計算機にインストールされた他の OS や、大規模計算機には T_EX Live がインストールされていないので、情報科学類の計算機を用いる場合には macOS を使うのがよいでしょう。

dvipdfmx

dvipdfmx は DVI ファイルを PDF ファイルへコンパイルするプログラムです。かつては dvipdfm というプログラムが使われていましたが、それを改造して現在は dvipdfmx が T_EX Live に収録されています。

現在はパッケージの相性という観点から dvipdfm の使用は非推奨ですし、T_EX Live にも同梱されていません^{*11}。

6.2 L^AT_EX を使う上での注意

L^AT_EX の命令に使うバックスラッシュ (`\`) は、情報科学類の iMac の標準設定では円マーク (¥) として入力されてしまいます。バックスラッシュ (`\`) を入力するには、`Option` と ¥ を同時に押します。また、入力の設定を変更することで、¥ を押すと `\` が入力できるようになります。この手引きの 1.5.5 を参照してください。

6.3 L^AT_EX による文書生成の流れ

L^AT_EX は後述する L^AT_EX ファイルというテキストベースのファイルから、DVI ファイルという形式のファイルを経て、PDF ファイルなどの最終的な文書へ変換されます。これは次の図 6.2 のようになります。

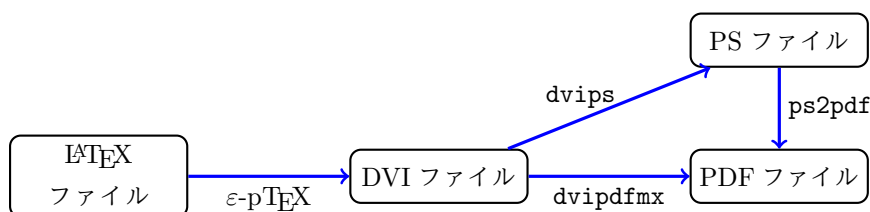


図 6.2: L^AT_EX ファイルから最終的な文書生成の流れ

従って、L^AT_EX ファイルの内容編集は Emacs (第 3 章参照) などといったテキストエディタで行い、それを ε-p T_EX^{*12}などのプログラムを用いて、後述する DVI ファイルへ変換し、さらにそれを dvipdfmx というプログラムにて PDF へと変換することで、最終的な PDF ファイルの文書が得られます。

この変換のことをコンパイルと呼びます。つまり L^AT_EX のソースファイルをコンパイルすることで DVI ファイルが得られ、さらに DVI ファイルをコンパイルすることで PDF ファイルなどが得られます。

まずは各種ファイル形式について説明します。

6.4 L^AT_EX に関連するファイルの形式

前節では“L^AT_EX ファイル”や“DVI ファイル”といった聞き慣れないファイル形式を経て最終的な文書になるという説明をしました。この節では L^AT_EX に関連するファイル形式を列挙して解説します。

^{*11} dvipdfm コマンドを実行すると dvipdfmx が互換モードで起動します。

^{*12} これは「いーびーてつく」もしくは「いーびーてふ」と発音します。

L^AT_EX ファイル

文書の元となるテキストファイルです。これに文書や L^AT_EX の命令を書き込みます。詳しい文法などは後述します。

DVI ファイル

DVI ファイルの DVI とは “**D**evice-**I**ndependent” のことで、文書のレイアウトが表示デバイスに全く依存しない形のバイナリデータです。L^AT_EX ファイルをコンパイルすることで生成されます。この DVI ファイルを直接閲覧・編集するということはあまりなく、別のファイルへ変換するための中間ファイルという役割が一般的です。

6.5 Hello L^AT_EX!

ソースコード 6.1 と同じ内容のテキストファイルを `test.tex` の名前で作成してください。詳細については後述しますので、今はとりあえず無視して転写してください。

Listing 6.1: Hello L^AT_EX!

```
1 \documentclass[a4j]{jsarticle}
2
3 \begin{document}
4
5 % この行はコメントです.
6 % 表示に影響を与えません.
7 \section{Hello \LaTeX!}
8 \LaTeX の文書は次のプログラムでコンパイルします.
9
10 \begin{itemize}
11   \item platex
12   \item dvipdfmx
13 \end{itemize}
14
15 \subsection{コンパイル}
16 以下のようにターミナルでコンパイルします.
17
18 \begin{verbatim}
19 platex -kanji=utf8 test.tex
20 dvipdfmx test.dvi
21 \end{verbatim}
22
23 \end{document}
```

この `test.tex` に対して以下のようにコンパイルを行います^{*13}。

```
$ platex -kanji=utf8 test.tex↵
```

^{*13} このオプション `-kanji=utf8` は L^AT_EX ファイルの文字コードを指定しています。ファイルの文字コードが EUC-JP の場合は `euc`, Shift-JIS の場合は `sjis` と指定します。


```
$ dvipdfmx test<␣>
$ █
```

このコマンドにより、L^AT_EX ファイル 6.1 を ε-pT_EX と dvipdfmx でコンパイルした結果、様々なファイルと共に test.pdf という PDF ファイルが生成され、それは図 6.3 のようになります。

1 Hello L^AT_EX!

L^AT_EX の文書は次のプログラムでコンパイルします。

- platex
- dvipdfmx

1.1 コンパイル

以下のようにターミナルでコンパイルします。

```
platex -kanji=utf8 test.tex
dvipdfmx test.dvi
```

図 6.3: test.pdf

L^AT_EX で文書を作る最初の一步を踏み出しました。もしコンパイルが上手くいかない場合は次の節を参考にしてください。

6.5.1 コンパイルに失敗した時

Java の処理系 (javac) へ文法的に誤ったプログラムを渡したときのように、エラーが発生してコンパイルに失敗することがあります。例えば、次のような状態で T_EX が止ってしまうことがあります。

```
! Undefined control sequence.
1.362 \LATEX
?
```

このような状態になった場合は、ひとまず と入力してから (return) キーで T_EX を終了します。

```
! Undefined control sequence.
1.362 \LATEX
? x<␣>
```

T_EX のエラーは!から始まります。ですのでこの場合は! `Undefined control sequence.`がエラーということになります。また次の行にある1.362 `\LATEX`とは、“362 行目の`\LATEX`の付近でエラーが発生した”ということを示しています。

L^AT_EX のエラーコードはたくさんあります。もし皆さんがエラーと遭遇してしまった場合は T_EX Wiki^{*14}にエラーメッセージの意味が多くまとめられていますので、こちらを参考に間違いを修正するとよいでしょう。

6.6 L^AT_EX のコマンドと環境

前節ではサンプルを基に L^AT_EX ファイルを実際にコンパイルして PDF ファイルを作成しました。本節では実際に L^AT_EX ファイルを記述するための記法を紹介します。

6.6.1 コマンド

ソースコード 6.1 には`\` (バックスラッシュ) から始まる、例えば`\documentclass`や`\section`といったものがあります。これらのことを L^AT_EX では次のように言います。

- コマンド
- マクロ
- コントロールシーケンス (Control Sequence, CS)
- 制御綴せいぎょつづり

このうちどれを使っても間違いではありません。この章では`\`から始まるものを、コマンドという用語で統一します。

このコマンドとは T_EX 処理系へ組版に関する指示を与える命令のことです。コマンドの意味を理解することにより、L^AT_EX で綺麗な文書を作ることができるようになります。

特殊文字

L^AT_EX では一部の ASCII 文字を特殊文字として扱います。それらの文字を下に示します。

特殊文字の一覧

`# $ % & - { } \ ^ ~`

これらの特殊文字は、そのまま書いても表示されず、命令として解釈されます。ただし、これらと同じ形をしている全角の文字は、特殊文字としては扱われません。特殊文字を文章に入れた場合は、表 6.1 に示す方法でエスケープする必要があります。

^{*14} <https://texwiki.texjp.org/?TeX%20%E3%81%AE%E3%82%A8%E3%83%A9%E3%83%BC%E3%83%A1%E3%83%83%E3%82%BB%E3%83%BC%E3%82%B8>

表 6.1: 特殊文字のエスケープ

文字	コマンド	文字	コマンド	文字	コマンド
#	\#	\$	\\$	%	\%
&	\&	-	_	{	\{
}	\}	\	\textbackslash	^	\textasciicircum
~	\textasciitilde				

記号

T_EX や L^AT_EX には、記号を出力するコマンドがいくつかあらかじめ用意されています。一部を表 6.2 に示します。

表 6.2: 記号の一例

コマンド	出力結果	コマンド	出力結果	コマンド	出力結果
\dag	†	\ddag	‡	\copyright	©
\pounds	£	\oe	œ	\OE	Œ
\ae	æ	\AE	Æ	\aa	å
\AA	Å	\o	ø	\O	Ø
?‘	¿	!‘	¡	\$\sim\$	~
‘‘	“	’’	”	--	—
\LaTeX	L ^A T _E X	\TeX	T _E X	---	---
\textasciicircum	^	\textbar		\textless	<
\textgreater	>	\textregistered	®	\texttrademark	™
\textvisiblespace	␣	\textbackslash	\	\textasciitilde	~
\S	§	\P	¶		

引数

L^AT_EX のコマンドには引数といって、そのコマンドに何か情報を渡す必要がある場合があります。このコマンドに渡される情報を引数と呼び、`[]` や `{ }` で囲んでコマンドの後に記述します。このうち、`[]` を使う引数を特にオプションと言い、こちらは書いても書かなくてもよいです。

例えば `\documentclass` コマンドは `a4j` と `jsarticle` という二つの引数を受け取っていて、そのうちオプション `a4j` は省略可能です。

6.6.2 環境

L^AT_EX の“環境”について解説します^{*15}。

^{*15} 環境という概念は純粋な T_EX にはなく、L^AT_EX によって提供されているものです。

環境とは命令の作用する範囲のことであり、`\begin`と`\end`という二つのコマンドで囲むことにより指定します。例えばソースコード 6.1 の 3 行目と 23 行目を見ると、次のようになっています。

```
\begin{document}
....
\end{document}
```

`{}`で囲われた部分は、その環境の名前を表わしています。そして`\begin`は始まりを、`\end`は終わりを表わします。

例えば`\begin{document}`となっていたら、“document 環境の始まり”という意味になり、`\end{document}`は同様に“document 環境の終わり”を意味します。

6.7 クラスファイルの指定

L^AT_EX ではまず、どのような種類の文書であるのかを明示する必要があります。それを行うのが`\documentclass`コマンドです。

ソースコード 6.1 で提示したサンプルを見ると、最初の行が次のようになっています。

```
\documentclass[a4j]{jsarticle}
```

このコマンドにおいて、オプションに紙のサイズや文字の大きさなどを、引数に具体的なドキュメントクラスを指定します。

6.7.1 ドキュメントクラス

日本語の文書では奥村晴彦氏が開発した `jsarticle` や `jsbook` が主に使われています。この二つの使い分けを説明します。

表 6.3: クラスファイルの一覧

クラスファイル	意味
<code>jsarticle</code>	一般的な文書, レポート
<code>jsbook</code>	書籍

授業のレポートなど多くの場合は `jsarticle` を用いれば大丈夫です。

6.7.2 `jsarticle` のオプション

オプションを指定することで、ドキュメントクラスに対して文書の形式等を指定することができます。ソースコード 6.1 では`a4j`というオプションが指定してありました。`a4j`は“A4 用紙の大きさに文書を作成する”という意味です。

以下に `jsarticle` のオプションを記してありますので参考にしてください。

用紙サイズ

`a4paper` 余白の広い A4 サイズ

`b5paper` 余白の広い B5 サイズ

`a4j` 余白の狭い A4 サイズ

`b5j` 余白の狭い B5 サイズ

文字サイズ

オプションで文字サイズの指定がなければ、10pt が本文に適用されます。

- 10pt
- 11pt
- 12pt

段組

新聞のような文書の段組を指定することができます。

`onecolumn` 段組なし

`twocolumn` 2 段組

ページの体裁

両面印刷を前提としたとき、奇数ページと偶数ページでページの体裁を変えるかどうかを決めることができます。

`oneside` 全て同じ体裁

`twoside` ページの体裁は偶数とページと奇数ページとで異なる

組版の確認

`draft` 行の右端がはみでたときに印を付ける

`final` 最終版. 何も印はつかない

この他のオプションについて知りたい方は `jsarticle` などが収録されている `jsclasses` パッケージ^{*16}のドキュメントを参照してください。

6.8 本文

文書の記述を始めるにあたり、ドキュメントクラスを記述しました。次に記述するのは `document` 環境です。 `document` 環境は、どこが文書の本文なのかを明示します。つまり、文書に出力されるのは `document` 環境内のみで、`\begin{document}` の前や `\end{document}` 以降に文章を記述しても無視されます。

6.9 表題と著者

論文やレポートを書く上で表紙は大事です。これらの文書では自分の名前や学籍番号などを記述することが求められます。表紙ではタイトル、名前等を書きますので、本節ではそれらの書き方を紹介します。

表題 `\title{タイトル}` のようにして、文書の表題を記述します。

著者 `\author{山田{ 太郎}}` のように文書の著者を記述します。複数著者がいる場合は、`\and` コマンドで区切ります。

日付 `\date{2011/3/11}` のように、文書にかかわる日付（作成日、提出日など）を記述します。`\date{}` とすると日付は入力されず空となり、`\date{today}` とすると作成日の日付が自動的に入力されます。

^{*16} <http://www.ctan.org/pkg/jsclasses>

ここまでで表題や著者の書き方を紹介しました。しかしこれらだけでは表題ページは生成されません。そこで使用するのが`\maketitle` コマンドです。これを使用することで、タイトル等が整形された形で生成されます。

例えばまとめると以下のソースコード 6.2 のようになります。

Listing 6.2: 表題ページの例

```
1 \documentclass[a4j]{jsarticle}
2
3 \title{日本語入力SKKについて}
4 \author{\text{情報 雅彦} \and Yukiyoshi KAGAKU}
5 \date\today
6
7 \begin{document}
8
9 \maketitle
10
11 日本語入力SKKとは……
12
13 \end{document}
```

これをコンパイルすると次の図 6.4 となります。

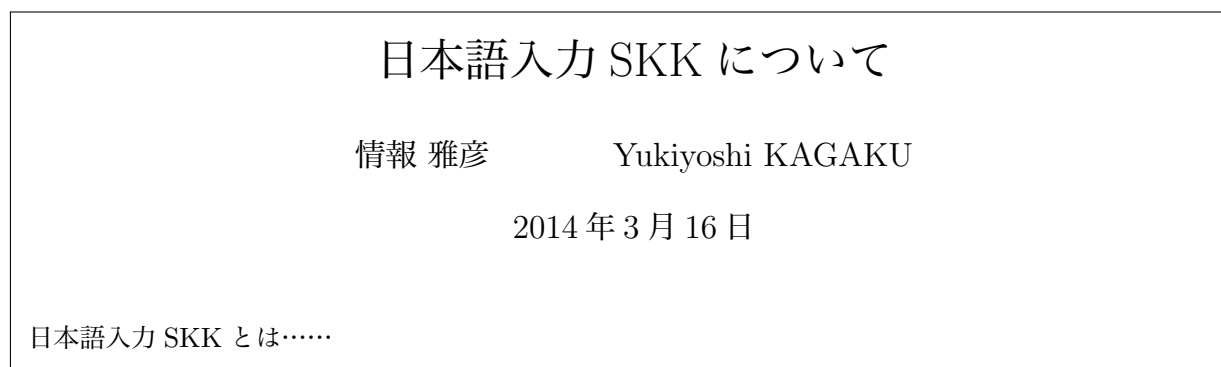


図 6.4: 表題ページを出力する

6.10 見出し

よい文章を書くためには文章の中の章立てを作ることが重要です。新聞などを見ると、まず大きな文字で書かれた見出しを用いてその後の本文が一体何について論じているのかを一言で述べ、本文では端的にこのことについて論じたいという目標を明らかにします。見出しという文章の目標があることで、何について意識して読めばよいのかが明瞭になり、読者にとって文章がより理解しやすくなります。

L^AT_EX における見出しの書き方を紹介します。見出しの中で最も大きいのは`\part` (部) になりますが、大きいものから始めなければいけないということはありません^{*17}。レポートなどの場合は`\section` (節) から

^{*17} ただし、例えば`\chapter`の後に`\subsection`を使うなど、途中の見出しを飛ばすのはよくありません。

始めればいいでしょう。表 6.4 は見出しの表です。

表 6.4: 見出しの種類

見出し	意味
<code>\part</code>	部
<code>\chapter</code> ^{*18}	章
<code>\section</code>	節
<code>\subsection</code>	小節
<code>\subsubsection</code>	小々節
<code>\paragraph</code>	段落
<code>\subparagraph</code>	小段落

以下のソースコード 6.3 は見出しの例です。

^{*18} `\chapter` コマンドは `jsarticle` クラスには含まれていません。 `jsbook` に含まれています。

Listing 6.3: 見出しの例

```
1 \part{地球}
2 地球は太陽系の惑星の一つで太陽から三番目に近く、生物が生存している星である。
3
4 \chapter{アジア}
5 アジアはヨーロッパを除いたユーラシア大陸全般を指す。
6
7 \section{日本}
8 日本は日本列島を主な領土とする東アジアの国家である。
9
10 \subsection{茨城}
11 日本の県の一つで関東地方の北東に位置し、東は太平洋に面する。
12
13 \subsubsection{つくば}
14 茨城県南部に位置し、筑波研究学園都市はつくば市全域を区域とする。
15
16 \paragraph{天王台}
17 つくば市の東部に位置し、地域内のほぼすべてが筑波大学や研究施設である。
18
19 \subparagraph{1-1-1}
20 筑波大学天王台キャンパスがある。
```

これは次の図 6.5 のようになります。

第 I 部

地球

地球は太陽系の惑星の一つで太陽から三番目に近く、生物が生存している星である。

第 1 章

アジア

アジアはヨーロッパを除いたユーラシア大陸全般を指す。

1.1 日本

日本は日本列島を主な領土とする東アジアの国家である。

1.1.1 茨城

日本の県の一つで関東地方の北東に位置し、東は太平洋に面する。

つくば

茨城県南部に位置し、筑波研究学園都市はつくば市全域を区域とする。

■天王台 つくば市の東部に位置し、地域内のほぼすべてが筑波大学や研究施設である。

1-1-1 筑波大学天王台キャンパスがある。

図 6.5: 見出しのコンパイル結果

この例では説明のため各見出しが一度しか用いられていませんが、本来このような文章の構造は避けるべきです。

6.11 書体・文字サイズ

見出し (6.10 節) のコマンドを用いて文書を章立し文章の構造を作ることが重要であるように、一部だけ書体を変更するなどして、文章の中で特に注目して欲しい部分を明示的にするというのも文章を良くするうえで重要です。

6.11.1 書体の変更

和文書体

和文（ひらがな・カタカナ・漢字）に関する書体を変更する場合には次のコマンドを用います。

`\textgt` 引数の文字列をゴシック体にします。

`\textmc` 引数の文字列を明朝体にします。

`\textbf` 引数の文字列を太字にします。

和文ではゴシック体にすることで文章の中で強調を表わすことが多いです。ただ、強調をする場合は`\emph`コマンドを用いる方が意味も分かりやすく欧文にも使えるので便利です。

`\textbf` コマンドでボールドにしたり、あるいは`\underline` コマンドで下線を引いたりといった方法による強調は、ページにおける黒色の割合が増えて密度が高く見えてしまうなどの理由から、安易な多用はお勧めしません。

欧文書体

主に欧文の書体を変更します。

`\textbf`

引数の文字列をボールド (**Bold**) にします。基本的には見出しや表題の文字列に使います。

`\textit`

引数の文字列をイタリック (*Italic*) にします。強調する時に使います*¹⁹。

`\texttt`

引数の文字列をタイプライタ (Typewriter) にします。ユーザーの入力部分などに用います。

`\textsc`

引数の文字列をスモールキャピタル (SMALL CAPITAL) にします。

前にも述べましたが`\textbf`の多用は、ページに文字が詰っている印象を与えるので使用はほどほどにしましょう。

6.11.2 文字サイズ

文字サイズを変更する際は次のコマンドを使います。

*¹⁹ 強調する際には`\texttt`コマンドよりも、`\emph`コマンドの方が直感的でかつ和文にも使えます。

表 6.5: 文字サイズを変更するコマンド

コマンド	結果
<code>\tiny</code>	tiny
<code>\scriptsize</code>	scriptsize
<code>\footnotesize</code>	footnotesize
<code>\small</code>	small
<code>\normalsize</code>	normalsize
<code>\large</code>	large
<code>\Large</code>	Large
<code>\LARGE</code>	LARGE
<code>\huge</code>	huge

注意すべきは、これらのコマンドは引数を取ってその文字列の大きさを変更するというわけではないことです。例えば、`\large{large text}`とするのは誤りです。これらのコマンドはグループという、`{ }`で囲んだ部分や環境の中全体に作用します。つまり、`{\large large text}`などとするのが正しい使い方です。

ただ、これらのコマンドを用いて安易に文字の大きさを変更するべきではありません。多くの場合は見出し(6.10 節)のコマンドなどによって適切な文字サイズになります。本文の文字サイズを無闇に変更すると、本文の高さが各行で異なってしまって見た目の印象がよくありません。文字サイズの変更は例えば表(6.15 節)の中など、本文ではない部分について行うのがよいでしょう。

6.12 改行・改ページ

L^AT_EX においてはただ改行しただけでは改行されません。これは改行や改ページといったことは T_EX の処理系などが適切と判断した場所で行われるため、L^AT_EX のソースコードを編集する私たちが気にすることではないという考えによります。

次の例を見てください。

```
1 ソースコードでは、この直後に改行するものの、
2 改行されません。
```

結果は次のようになります。

ソースコードでは、この直後に改行するものの、改行されません。

一見直感とは異なり不自由に思うかもしれませんが、これにより、一つの段落が非常に長くなってしまったとしても、ソースコードでは複数の行に分けて記述することができます。従って、テキストエディタを横にスクロールする必要がなくなりより快適な編集ができます。

ただ T_EX の処理系による改行や改ページの判断が必ずしも適切とは限りません。止むを得ず、編集者が改行や改ページを行わなければならない局面があるので、ここでは改行や改ページの方法を説明します。

6.12.1 改行

改行するためにはどうしたらいいかというと\\を使います。

```
1 この直後に改行する。  \\
2 改行されます。
```

結果は次のようになります。

この直後に改行する。
改行されます。

このように改行されますが、この方法は先にも説明した通り止むを得ない場合にのみ使うべきです。普段の改行については T_EX の処理系に任せるべきです。

6.12.2 改ページ

次に改ページの方法を紹介します。改ページは`\newpage`コマンドや`\clearpage`コマンドで行うことができます。この二つのコマンドは二段組など段組の中で用いられた場合の挙動が異なります。

`\newpage` 次の段がある場合は次の段へ行き、ない場合は改ページを行います

`\clearpage` 段組に関わらず改ページを行います

ここでは実行例を示しませんので各自でやってみてください。

6.13 箇条書き

箇条書きの方法を紹介します。L^AT_EX の箇条書きは3つの種類があります。

6.13.1 順序なし箇条書き

`itemize` 環境は順序なしの箇条書きを作ります。

```
1 \begin{itemize}
2   \item 鍋
3   \item フライパン
4 \end{itemize}
```

結果は次のようになります。

- 鍋
- フライパン

6.13.2 順序あり箇条書き

`enumerate` 環境は順序ありの箇条書きとなります。

```
1 \begin{enumerate}
2   \item フライパンに油を敷く
3   \item 卵を入れる
4 \end{enumerate}
```

結果は次のようになります。

1. フライパンに油を敷く
2. 卵を入れる

6.13.3 定義リスト

定義リストとは単語とその意味を列挙する際に用います。L^AT_EX では `description` 環境を用いて次のように書きます。

```
1 \begin{description}
2   \item[つくば]
3     筑波大学があります。
4
5   \item[土浦]
6     つくば国際大学があります。
7 \end{description}
```

このようになります。

つくば 筑波大学があります。
土浦 つくば国際大学があります。

6.14 図

これまでは文字列を扱う記述について紹介してきましたが、本節ではレポートでは必須の画像の取り扱い方法を紹介します。

6.14.1 graphicx パッケージ

画像を挿入するにあたり、`\documentclass` コマンドと `\begin{document}` の間^{*20}に `\usepackage[dvipdfmx]{graphicx}` と記述し、`graphicx` パッケージ^{*21}を読み込んでいます。`graphicx` パッケージは画像を L^AT_EX で使うために必要なパッケージです。このパッケージは画像の他にも、図形などに関する様々な機能を提供しますが本章では解説しません。詳細を知りたい方は脚注^{*21}の URL を参照してください。

この `graphicx` パッケージによって、画像を表示するための `\includegraphics` コマンドが使えるようになります。

6.14.2 バウンディングボックス

L^AT_EX で画像を用いる前に、バウンディングボックスという用語について理解する必要があります。L^AT_EX におけるバウンディングボックスとは画像の大きさを示すものです。画像の大きさとは縦と横の長さ（距離）になります。これがなければ、L^AT_EX は画像の大きさが分からず画像を文書に埋め込めません。

残念なことに T_EX は PNG や JPEG, PDF といったバイナリファイルから画像のバウンディングボックスを取得することができません^{*22}。そこで PNG や JPEG, PDF といったバイナリファイルの画像を文書で用いる場合、次の2つの方法のうちいずれかを用いてバウンディングボックスを T_EX へ渡す必要があります。

- `\includegraphics` コマンドの `bb` オプションを用いる
- `extractbb` というプログラムで生成する

bb オプションは非推奨

L^AT_EX について Web の資料などを調べたことがある方は、`bb` オプションを用いて `\includegraphics` コマンドへ直接バウンディングボックスを指定する方法を知っているかもしれません。

ですが、この方法は画像の“物理的な長さ”を手動で計測する必要があります。例えば皆さんが画像の情報としてよく用いるピクセルという単位は、実は物理的な長さではなく、画素が何個という数を表す単位です。

ある物理的な長さに、どれだけの画素が詰め込まれているかという密度を表す量を解像度とよび、解像度を表す単位としては *DPI* (dots per inch: 1 インチあたりの画素数) などが存在します。

`bb` オプションを用いる場合、ピクセルといった数の情報と、解像度という画像の密度を表す情報を用いて物理的な長さを計算する必要があります。この作業は大変な労力がかつ人間の計算ミスもありえます。従って、次に説明する `extractbb` というプログラムによって算出する方が、手作業によるミスもなく簡単でよいとされています。

^{*20} このことをブリアンブルといいます。

^{*21} <http://ctan.org/pkg/graphicx>

^{*22} 反対に、テキストファイルで記述されている EPS であれば、ファイルの先頭に書かれているバウンディングボックスの情報を読み込んで使うことができます。

extractbb によるバウンディングボックスの生成

PNG, JPEG の画像や PDF を貼る場合は, `extractbb` コマンドを用いて次のようにすることでバウンディングボックス情報ファイルが生成されます. 例えば `example.png` のバウンディングボックスを生成する際は, ターミナルで次のようにします.

```
$ extractbb example.png↵  
$ █
```

すると, 画像と同じディレクトリに `example.xbb` というファイルが生成されます. このファイルにバウンディングボックスが記述されています. `\includegraphics` コマンドを用いると, 画像の名前と同じ `xbb` ファイルを検索して, `xbb` ファイルがあれば利用するので, PNG, JPEG の画像や PDF を用いる際はあらかじめ `extractbb` コマンドを使ってバウンディングボックスを生成しておきましょう.

6.14.3 画像の表示

`\includegraphics` コマンドを用いた例を以下のソースコード 6.4 に示します.

Listing 6.4: 画像の挿入

```
1 \documentclass[a4j]{jsarticle}  
2 \usepackage[dvipdfmx]{graphicx}  
3 \usepackage{float}  
4  
5 \begin{document}  
6  
7 \begin{figure}[H]  
8   \centering  
9   \includegraphics[width=5cm]{example-image-a.png}  
10  \caption{画像の挿入}  
11  \label{fig:testpicture}  
12 \end{figure}  
13  
14 \end{document}
```

実行結果は図 6.6 のようになります.

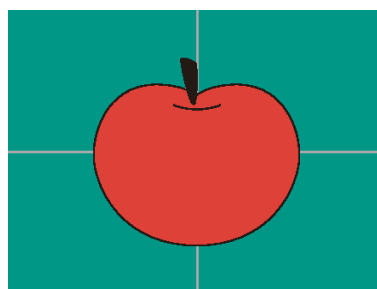


図 6.6: 画像の挿入

float パッケージ

まず 3 行目にて, float パッケージを読み込んでいます. このパッケージは画像を配置を直感的にするものですので, 導入をお勧めします.

figure 環境

figure 環境は環境内にあるものが図であることを示す環境です. `\begin{figure}[H]` の `[H]` の部分は表示位置に関する設定を表わしています. `H` は float パッケージによって提供されるものですが, L^AT_EX の標準では次のような指定ができます.

表 6.6: figure 環境などで使える位置指定子

位置指定子	意味
<code>h</code>	環境が使用された位置に出力します
<code>t</code>	環境が現れるページの上端に出力します
<code>b</code>	環境が現れるページの下端に出力します
<code>p</code>	表や図からなるページを作成しそこに出力します

これらは `[htbp]` といった具合に併記することもでき, そうした場合は書かれた順に優先度を持ちます. 例えば `[htbp]` の場合はまず, “`h`” に従い環境が使用された位置に出力しようとします. しかしそれが何らかの理由によってできない場合, T_EX 処理系は次に “`t`” に従い, ページの上端に表示しようとします. というように, 最終的に可能なものを用います.

ただ, 多くの場合は float パッケージを用いて `H` のみを指定すればよいので, これらを覚える必要はあまりないでしょう.

\centering コマンド

これはその名の通り, `\centering` コマンド以降の文章や画像などを中央寄せにします. このコマンドは現在の環境, あるいはグループの中で終了します.

\includegraphics コマンド

また 9 行目では `[width=5cm]` にて, 画像の大きさを 5cm と指定しています. そして, `example-image-a.png` という画像を読み込んでいます^{*23}. 実際に使うときは, L^AT_EX ファイルからの相対パス名で画像を指定します.

\caption コマンド

そのあとに来る `\caption` コマンドは画像に対する簡単な説明 (キャプション) を書きます. 一般的に図のキャプションは図の下に記述します. このコマンドは省略することができます.

^{*23} あらかじめ画像 `example-image-a.png` を用意する必要があります.

\label コマンド

\label コマンドは、画像に本文などから参照するためのラベルを付加します。このコマンドは省略することができます。このラベルの使用目的、方法は後述の参照（6.17 節）にて紹介します。

6.15 表

L^AT_EX においては `tabular` 環境を用いて表を作成し、その表の配置やキャプションを `table` 環境やその中で指定します。表は図とは異なり、プリアンブルには何も追記する必要はありません^{*24}。

実際に利用する際には以下のソースコード 6.5 のように記述します。

Listing 6.5: 表の例

```

1 \begin{table}[H]
2   \caption{25マス計算（掛け算）}
3   \label{table:tableexample}
4
5   \centering
6   \begin{tabular}{|r||c|c|c|c|c|}
7   \hline
8   掛け算 & 2 & 3 & 5 & 7 & 1 \\ \hline
9   1 & 2 & 3 & 5 & 7 & 1 \\ \hline
10  4 & 8 & 12 & 20 & 28 & 4 \\ \hline
11  9 & 18 & 27 & 45 & 63 & 9 \\ \hline
12  3 & 6 & 9 & 15 & 21 & 3 \\ \hline
13  8 & 16 & 24 & 40 & 56 & 8 \\ \hline
14 \end{tabular}
15 \end{table}

```

結果は次の表 6.7 のようになります。

表 6.7: 25 マス計算（掛け算）

掛け算	2	3	5	7	1
1	2	3	5	7	1
4	8	12	20	28	4
9	18	27	45	63	9
3	6	9	15	21	3
8	16	24	40	56	8

図と同様に `\begin{table}` の直後に `[H]` があります。 `figure` 環境と同様に表 6.6 にある表示位置を指定することもできます。ですが、特に理由がない限り `[H]` を用いれば問題ありません。

`table` 環境の中には `figure` 環境と同様にキャプション（`\caption` コマンド）、ラベル（`\label` コマンド）を書くこともできます。

^{*24} ただし `figure` 環境のように表示位置として `H` を用いる場合は `float` パッケージが必要となります。

表は図とは異なり，一般的にキャプションを表の上に記述します．図のソースコード 6.4 と表のソースコード 6.5 を見比べてもらえばわかりますが，それぞれのキャプションの位置はソース上のその位置と同じです．つまり，これらを逆にすることも可能ということですが，一般的にはこの順番を用います．

6.15.1 tabular 環境

`tabular` 環境は表に特有な，まるで数学の行列式のような構造を作る環境です．この `tabular` 環境は L^AT_EX の内部でとても複雑な処理をしています．そのお陰で，私たち一般の L^AT_EX 利用者は容易に表を作ることができます．

セル内の文字位置

`\begin{tabular}` の直後に，`|r||c|c|c|c|c|` という記述がありますが，これは表の線と内容の配置を示しています．各アルファベット (`r` や `c`) は各列の文字の配置を示し，一行につき一つしか指定できません．それぞれの意味は以下のようになります．

表 6.8: セル内の位置指定子

位置指定子	意味
<code>l</code>	文字列を左寄せにします
<code>c</code>	文字列を中央寄せにします
<code>r</code>	文字列を右寄せにします

縦線

また，各々の間に書かれている `|` は表の列間の線を表しており，これは書いた本数分だけ線が引かれます．ここでの例では，一列目と二列目の間には二本の線が引かれていますが，それ以外の場所是一本です．

セルと横線

続いて `tabular` 環境の中身を見てみます．各列は `&` によって分けられます．また，各列は `\\` によって行の終わりを表現します．そして `\hline` コマンドは水平線を引くためのものです．例のように二つを連続で記述することにより二重線を記述できます．

複数の列をまたぐセル，複数の行をまたぐセル

また，表を利用する際には二つ以上のセルにまたがる表を作成したい，ということがあります．このような書き方は以下のソースコード 6.6 のようにすることで可能になります．

Listing 6.6: 複数セルにまたがる表

```

1 \begin{table}[H]
2   \caption{複数セルにまたがる表}
3   \label{table:multicolumn}
4
5   \centering
```

```

6 \begin{tabular}{|l|c|r|r|}
7   \hline
8   \multicolumn{4}{|c|}{メンバ} \\ \hline \hline
9   主戦力 & Aさん & Bさん & Cさん \\ \hline \hline
10  & Dさん & Eさん & Fさん \\ \hline
11  副戦力 & Gさん & Fさん & Gさん \\ \hline \hline
12  & Hさん & Iさん & Jさん \\ \hline
13  補欠 & \multicolumn{3}{|c|}{新人} \\ \hline
14 \end{tabular}
15 \end{table}

```

実行結果は表 6.9 のようになります。

表 6.9: 複数セルにまたがる表

メンバ			
主戦力	A さん	B さん	C さん
	D さん	E さん	F さん
副戦力	G さん	F さん	G さん
	H さん	I さん	J さん
補欠	新人		

ここで用いているのは`\multicolumn`コマンドです。これは直後の引数で記述した数の列をまとめて一つの列とし、その次の引数で当該セル内の配置を指定し、最後の引数にセル内に入るコンテンツを記述します。

ここまでは複数の列にまたがるセルが対象でしたが、次に見るのは複数の行にまたがるセルです。これは`\cline`コマンドを用いることで表現できます。表の2行目、3行目を見てみると2列目と3列目に上下を分けるような線が入っています。これは`\cline`の直後の`{}`によって範囲を指定できます。2、3行目と同様に4、5行目も以下のように記述することで、3列目と4列目に上下を分けるような線が引かれています。これを用いることで、上下を分けるような線の引かれていない場所を結合セルといて取り扱うことができます。

6.16 脚注

脚注とは本文に書くほどのことではない補足を書くために用いるものです。

`\footnote` コマンド

L^AT_EX では`\footnote` コマンドを用いて、次のように行うことができます。

```
\TeX\footnote{Knuthが作ったプログラム言語。}は優れた組版能力があり……
```

実行すると次の図 6.7 のようになります^{*25}。

^{*25} 実行結果では脚注番号が“a”になっています。これはサンプルの都合によるもので、通常は算用数字が振られます。

T_EX^aは優れた組版能力があり……

^a Knuth が作ったプログラム言語.

図 6.7: 脚注の例

`\footnotemark`, `footnotetext` コマンド

`\footnotemark` コマンドと `\footnotetext` コマンドを用いることで、脚注番号と脚注本文を別々に書くことができます。次のようにします。

```
1 \TeX\footnotemark は優れた組版能力があり……
2
3 \footnotetext{Knuthが作ったプログラム言語. }
```

`\footnotemark` が脚注番号を出力し、`\footnotetext` が脚注本文を構築します。上記の例の結果は図 6.7 と同じです。これは脚注本文がとても長くなってしまって、`\footnote` で書くとソースコードが読み難くなってしまうのを防ぐ効果があります。

6.17 参照

論文やレポートにおいて画像や表を挿入した場合、参照を付けなければならないことがあります。ここでは L^AT_EX の便利な点である図表の参照方法について述べます。図表の番号を自分で付け、それへの参照を自分でつけた番号をもとに行うといった作業が L^AT_EX では必要ありません。L^AT_EX は参照に必要な番号を自動で割り振ってくれるのです。

参照の仕方ですが、`\ref` コマンドと `figure` 環境 (6.14 節) などで紹介された `\label` コマンドを用いることで実現できます。6.14 節で用いた図と、6.15 節で用いた表を例としますと以下のように書くことができます。

図 `\ref{fig:testpicture}` と表 `\ref{table:tableexample}`

次のようになります^{*26}。

図 6.6 と表 6.7

このように `\ref` コマンドの中に、`\label` コマンドでラベルとして記述した文字列を入力することで参照できます。注意していただきたいのは、別の図表に同じラベルを付けると後に出てきたラベルの方が優先されてしまいます。ラベルは重複させてはいけません。そこで、重複を防ぐためにも図のラベルは `fig:` から、表のラベルは `table:` から始めるなどという慣習があります。

^{*26} この例では参照に PDF のリンクが貼られています。これは `hyperref` パッケージの機能を使って行なっているので、これを読み込んでいない場合は参照の番号が表示されるだけでリンクは貼られません。`hyperref` パッケージを用いたリンクの貼り方は <https://texwiki.texjp.org/?hyperref> をご覧ください。

6.17.1 見出しの参照

`\label` コマンドは図や表以外にも、`\section` コマンドといった見出しや脚注にも付けることができます。例えばこの節には次のようにラベルが付けられています。

```
1 \section{参照}
2 \label{sec:reference}
```

次のように参照します。

```
1 第\ref{sec:reference}節
```

どのようになるのかは、本章にたくさんある参照を見ると分かります。

6.17.2 脚注の参照

`\footnote` コマンド内で `\label` コマンドを用いれば、脚注を参照することも可能です。次のようにします。

```
1 \LaTeX\footnote{\label{fn:latex}Lamportが作ったマクロです。} ..... \\\
2 ..... \\\
3 \LaTeX については脚注\ref{fn:latex}を参照してください。
```

次のようになります。

L^AT_EX^a

.....

L^AT_EX については脚注 *a* を参照してください。

^a Lamport が作ったマクロです

6.18 数式

さて、ここまでのところで画像や表を用いたレポートや論文は書けるようになりました。しかし、コンピュータサイエンスの分野にいる以上は数式とは切っても切れない縁があります。そこで、文書内の数式をきれいに書く技術が必要になってきます。MS Word などにも数式エディタが存在するように、もちろん L^AT_EX にも数式を書くための方法が用意されています。例えば、以下のような式を記述することができます。

$$\int \frac{1}{x^2-1} dx = \frac{1}{2} \log \left| \frac{x-1}{x+1} \right| + C$$

通常の数式エディタでは指数をうまく表示できなかつたり、複雑になると全体の形が崩れたりしてしまうことが多々ありますが、L^AT_EX は世の中のほとんどの数式に対応しています。なぜなら世の中の数学者が論文を書く際には多くの場合 L^AT_EX を使い、もし既存の L^AT_EX にはない新しい数学的表現を導入したならば、それ

は L^AT_EX の機能で実装されます。従って、L^AT_EX は世の中にある多くの数学的な表現に対応していると言えます。

また、この節では L^AT_EX による数式の表現力を示すために、やや専門的な数式が例として書かれています。この節はあくまで L^AT_EX の数式能力に関する解説なので、例にある数式の意味が分からなくとも問題はありません。

6.18.1 amsmath パッケージを利用して数式を書く

L^AT_EX には数式を扱うためのさまざまなコマンドや環境が提供されていますが、現在数式を扱う論文はほとんどが amsmath パッケージ^{*27} を用いていると言われています。つまり業界のデファクトスタンダードということで、本節でも amsmath を前提とした数式の書き方を説明します。

amsmath パッケージは graphicx パッケージ (6.14 節) と同じく、`\usepackage` コマンドによって、`\usepackage{amsmath}` という記述で読み込むことができます。

実際に amsmath パッケージを用いて数式を書いてみましょう。次のソースコード 6.7 をコンパイルしてみてください。

Listing 6.7: Hello amsmath!

```
1 \documentclass[a4j]{jsarticle}
2 \usepackage{amsmath}
3
4 \begin{document}
5
6 \begin{align}
7 \left(
8   \begin{array}{cc}
9     2 & -1 \\
10    -3 & 4
11   \end{array}
12 \right)
13 \left(
14   \begin{array}{c}
15     1 \\
16     2
17   \end{array}
18 \right) = \left(
19   \begin{array}{c}
20     0 \\
21     5
22   \end{array}
23 \right)
24 \end{align}
25
26 \end{document}
```

^{*27} <http://www.ctan.org/pkg/amsmath>

次の式 (6.1) のようになります。

$$\begin{pmatrix} 2 & -1 \\ -3 & 4 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 5 \end{pmatrix} \quad (6.1)$$

amsmath パッケージをより詳しく知りたい方へ

ここで利用した amsmath パッケージは $\mathcal{A}\mathcal{M}\mathcal{S}$ -L^AT_EX の一部です。 $\mathcal{A}\mathcal{M}\mathcal{S}$ -L^AT_EX とは American Mathematical Society^aによって開発された、L^AT_EX 数式機能を強化するマクロパッケージ amsmath と、American Mathematical Society の組版に合わせた amscs というクラスファイル、さらに数式に用いる記号類のフォントをまとめたものをいいます。このクラスファイルとは、jsarticle (6.7 節) などと同じく文書のレイアウトを定義するものです。これは American Mathematical Society へ論文などを投稿するときに利用されています。

^a <http://www.ams.org/home/page>

さて、肝心の書き方ですが数式を記述する方法が大きく分けてディスプレイ数式とインライン数式の 2 つが存在します。

6.18.2 ディスプレイ数式

ディスプレイ数式とは、数式を本文とは別の行にして表示する方法です。ディスプレイ数式のために、次のようなコマンドや環境が提供されています^{*28}。

- align 環境
- align*環境
- \[コマンドと\]コマンド

これらの違いを順に説明します。

align 環境

ソースコード 6.7 に用いられていた環境です。この align 環境は番号付きディスプレイ数式を提供します。気付いた方もいるでしょうが、ソースコード 6.7 を実行すると “(6.1)” のような番号が式の右端に自動で付与されます。これが番号付きということの意味です。

また、この番号は複数の行にわたる数式を記述した場合も、自動で各行に振られます。次のソースコードを見てください。

Listing 6.8: 複数行にわたる数式

```
1 \begin{align}
2 \quad \overrightarrow{\text{dd}} \text{ \&= } \overrightarrow{\text{rd}} + \left( \vec{\text{d}} \cdot \vec{\text{n}} \right) \vec{\text{n}} \quad \text{\label{eq:vectordd1}} \end{align}
```

^{*28} equation 環境や equarray 環境, \$\$ など、この他にも L^AT_EX が提供する数式用の環境がありますが、 $\mathcal{A}\mathcal{M}\mathcal{S}$ -L^AT_EX で提供される align 環境などを使う方が良いとされています。

```

3   &= \vec{d} - \left(\vec{d} \cdot \vec{n}\right)\vec{n} \label{eq:vectordd2}
4 \end{align}

```

次のようになります^{*29}。

$$\vec{dd} = \vec{rd} + (\vec{d} \cdot \vec{n}) \vec{n} \quad (6.2)$$

$$= \vec{d} - (\vec{d} \cdot \vec{n}) \vec{n} \quad (6.3)$$

数式内で改行をしたい場合は、`\`で改行場所を指示する必要があります。これは数式の改行位置を T_EX の処理系が判断できないからです。

また、式 (6.2) と式 (6.3) は=の位置で整列しています。これは揃えたい場所に`&`を置くことで実現できます。

ソースコード 6.8 を見て気付いた方もいるかもしれませんが、数式に対してもラベルを用いることができます。つまり、他の場所からの参照 (6.17 節) が可能ということです。書き方は通常の参照と同様に`\ref`コマンドを用います。

■一部の数式にだけ番号を振る 最初の例では両方の数式に数式番号が振られています。しかし一つの数式が複数の行にわたっており、数式番号をその全てには付けたくないということが考えられます。そのような場合は`\nonumber`コマンドを用いて、番号の付与を抑制することができます。次のようにします。

```

1 \begin{align}
2 \left(\lambda r.r\right)\left(\lambda x.\lambda y.x\sim y\right) & \rightarrow_{\eta} \left(\lambda r.r\right)\left(\lambda x.x\right) \\
3 \left(\lambda r.r\right)\left(\lambda x.\lambda y.x\right) & \rightarrow_{\beta} \lambda x.x \\
4 & \nonumber \\
5 \end{align}

```

結果はこのようになります。

$$\begin{aligned} (\lambda r.r)(\lambda x.\lambda y.x y) &\rightarrow_{\eta} (\lambda r.r)(\lambda x.x) \\ &\rightarrow_{\beta} \lambda x.x \end{aligned} \quad (6.4)$$

このようにある行には番号を付けず、ある行には付けるということが可能です。

align*環境

こちらは `align` 環境とは異なり、番号なしディスプレイ数式を提供します。`align` 環境では原則全ての行に番号が振られていましたが、`align*`環境は数式に番号を振りません。従って、番号がないので`\label`を用いて参照を行うこともできません。番号が振られない以外に `align` 環境と違いはありません。次の例を見てください。

```

1 \begin{align*}
2 \tau ::= \alpha, \mid, \text{int}, \mid, \text{bool}, \mid, \tau_1 \rightarrow \tau_2 \mid \\
3 \sigma ::= \tau, \mid, \forall \alpha. \sigma \\
4 \end{align*}

```

^{*29} この例では番号が (6.2) から始まっていますが、これは先ほどの式 (6.1) の番号を 1 としているためです。

実行すると次のようになります。

$$\begin{aligned}\tau &::= \alpha \mid \textit{int} \mid \textit{bool} \mid \tau_1 \rightarrow \tau_2 \\ \sigma &::= \tau \mid \forall \alpha. \sigma\end{aligned}$$

このように、行に番号が振られていません。

\[コマンドと\] コマンド

\[コマンドと\]コマンドで囲まれた部分は一行のみのディスプレイ数式となります。align 環境や align*環境のように\\を用いて複数行の数式を書くことはできません。次のようになります。

```
1 \[
2   Fun\left(x, \underbrace{\text{Let}\%
3     \left(f, \overbrace{\text{Fun}\left(y, x\right)}^{\forall \alpha_2. \alpha_2 \rightarrow \alpha_1}
4       \rightarrow \alpha_1\right), f\right)}_{\forall \alpha_2. \alpha_2 \rightarrow \alpha_1}
5     : \forall \alpha_2. \alpha_1 \rightarrow (\alpha_2 \rightarrow \alpha_1)
6 \]
```

実行すると次のようになります。

$$Fun \left(x, \underbrace{Let \left(f, \overbrace{Fun(y, x)}^{\forall \alpha_2. \alpha_2 \rightarrow \alpha_1} \right)}_{\forall \alpha_2. \alpha_2 \rightarrow \alpha_1} \right) : \forall \alpha_2. \alpha_1 \rightarrow (\alpha_2 \rightarrow \alpha_1)$$

さて、これまでの例で L^AT_EX はとても複雑な数式を表現することができるということが分かったのではないのでしょうか。

6.18.3 インライン数式

先に述べたディスプレイ数式 (6.18.2 節) は数式と本文を完全に分離していました。ですが、レポートなどでは文章の中に数式を挿入したいということがあります。そのような時には\$を用いて本文に数式を埋め込むことができます。次に例を示します。

離散フーリエ変換

X_k ($k = 0, 1, \dots, N-1$) は級数

$X_k = \sum_{n=0}^{N-1} x_n \mathrm{e}^{-i \frac{2\pi}{N} kn}$ となり、

この計算量は $\mathcal{O}(N^2)$ になる。

次のようになります。

離散フーリエ変換 X_k ($k = 0, 1, \dots, N-1$) は級数 $X_k = \sum_{n=0}^{N-1} x_n e^{-i \frac{2\pi}{N} kn}$ となり、この計算量は $\mathcal{O}(N^2)$ になる。

このように\$で囲まれた部分が数式として解釈されます。

6.19 ソースコード

情報科学類の皆さんは課題のレポートや論文などでソースコードを出力したいということがあるでしょう。L^AT_EX では `listings` パッケージ^{*30}という伝統的なマクロパッケージによって、この本で用いられているようなソースコードの出力を簡単に行えます。

6.19.1 listings パッケージの読み込み

`graphicx` パッケージ (6.14 節) と同様に、`\usepackage` コマンドを用いてプリアンブルに `\usepackage{listings}` と記述します。

6.19.2 設定

`listings` パッケージは大変高性能なパッケージなのですが、初期設定ではあまり美しい出力が得られません。設定は次の二つのコマンドで設定できます。

`\lstdefinestyle`

新しいスタイルを定義する命令です。

`\lstset`

全てのソースコードに対して設定を適用します。

スタイルの定義

例えば一つのレポートに Java のソースコードと C 言語のソースコードが混在する、という局面を考えるとします。そして、Java のソースコードの設定と C 言語のソースコードの設定は別々にしたいという時に、“スタイル” という設定を編集者が定義し、ソースコードに応じて柔軟に見た目を切り替えることができます。

`\lstdefinestyle` は次のように使います。

```
1 \lstdefinestyle{java}{
2   language=java,
3   morekeywords={lambda}
4 }
5
6 \lstdefinestyle{c}{
7   language=c,
8   numbers=left
9 }
```

このようにして、新しいスタイル `java` と `c` を作ることができます。

^{*30} <http://www.ctan.org/pkg/listings>

全てのソースコードに関する設定

統一したい設定については`\lstset`コマンドにて設定します。例えばこの手引きでは次のような設定を行っています^{*31}。

Listing 6.9: 手引きの `listings` パッケージの設定

```

1 \definecolor{solarized@base03}{HTML}{002B36}
2 \definecolor{solarized@base02}{HTML}{073642}
3 \definecolor{solarized@base01}{HTML}{586e75}
4 \definecolor{solarized@base00}{HTML}{657b83}
5 \definecolor{solarized@base0}{HTML}{839496}
6 \definecolor{solarized@base1}{HTML}{93a1a1}
7 \definecolor{solarized@base2}{HTML}{EEE8D5}
8 \definecolor{solarized@base3}{HTML}{FDF6E3}
9 \definecolor{solarized@yellow}{HTML}{B58900}
10 \definecolor{solarized@orange}{HTML}{CB4B16}
11 \definecolor{solarized@red}{HTML}{DC322F}
12 \definecolor{solarized@magenta}{HTML}{D33682}
13 \definecolor{solarized@violet}{HTML}{6C71C4}
14 \definecolor{solarized@blue}{HTML}{268BD2}
15 \definecolor{solarized@cyan}{HTML}{2AA198}
16 \definecolor{solarized@green}{HTML}{859900}
17
18 \lstset{
19   basicstyle=\small\ttfamily\color{solarized@base00},
20   rulesepcolor=\color{solarized@base03},
21   numberstyle=\scriptsize\color{solarized@base01},
22   keywordstyle=\color{solarized@blue},
23   stringstyle=\color{solarized@cyan}\ttfamily,
24   commentstyle=\color{solarized@base01},
25   emphstyle=\color{solarized@red},
26   backgroundcolor=\color{solarized@base3},
27   sensitive=true,
28   breaklines=true,
29   breakatwhitespace=true,
30   framerule=0pt,
31   frame=1
32   showstringspaces=false,
33   tabsize=2,
34   basewidth={0.57em, 0.52em},
35 }
```

設定できる項目は`\lstdefinestyle`コマンドと`\lstset`コマンドの間に違いはありません。設定は膨大にあるので、詳細は脚注*30にあるドキュメントを読んでいただくしかないのですが、その中でも特に重要なものを書き出しておきます。

^{*31} 手引きでは `color` パッケージを用いて文字や背景の色を設定しています。 <http://www.ctan.org/pkg/color>

language	のを取ります。
ソース中に書いてあるものが、何言語なのかを記述します*32.	• none • left • right
style	
<code>\lstdefinestyle</code> にて定義した設定を指定します.	firstnumber
basicstyle	行番号の初期値です. これに数値を設定するとその番号から始まるのですが, <code>auto</code> または何も指定しないと, 前回の番号の次の値から始まります.
keywordstyle	
キーワードの文字スタイルを決めます.	frame
breaklines	枠に関する設定です. 引数はいろいろあるのですが, とりあえず <code>tbrl</code> を入れておけば, 四角で囲われます.
<code>breaklines=true</code> とすると, 自動改行が有効になります. 見た目上の行と実際の行は区別されます. 自動改行がないと <code>listings</code> の枠からはみ出る可能性があるので, 有効にする方が良いでしょう.	tabsize
numbers	タブの幅がスペース何個分かを数値で与えます.
行番号の位置に関する設定です. 値は次のも	xleftmargin, xrightmargin
	左右のマージンを決めます.

6.19.3 listings パッケージの使い方

ようやく `listings` パッケージを使うための準備が整いました. 使い方は次の3種類があります.

- L^AT_EX ソース中に直接埋め込む (`\lstlisting` 環境)
- ファイルから読み込む (`\lstinputlisting` コマンド)
- 本文中に挿入する (`\lstinline` コマンド)

順に解説します.

`\lstlisting` 環境

ソースコードを L^AT_EX ファイルに直接埋め込む際は `\lstlisting` 環境を用います. 次のようにします.

```

1 \begin{lstlisting}[language=java, numbers=left]
2 public class HelloWorld {
3     public static void main (String[] args) {
4         System.out.println("Hello World !!");
5     }
6 }
7 \end{lstlisting}
```

設定 6.9 を用いて実行すると次のようになります.

*32 記述の仕方が独特なので注意が必要です. 詳しくは `listings` パッケージのドキュメント (脚注*30) を参照してください.

```

1 public class HelloWorld {
2     public static void main (String[] args) {
3         System.out.println("Hello World!!");
4     }
5 }

```

このように行番号が自動で振られ、文字列や予約語の色が自動で変更されています。

`\lstinputlisting` コマンド

`\lstinputlisting` コマンドはファイルを指定してそのファイルをソースコードとして読み込み、整形を行って表示します。

次のようにします。

```
\lstinputlisting[style=java]{src/example.java}
```

まず、`[style=java]`にて、`\lstdefinestyle` コマンド (6.19.2 節) にて定義した設定を呼び出し、それに基づいて `src/example.java` を読み込み表示します。

`\lstinline` コマンド

これは本文の中にプログラムを埋め込む際に利用します。次のようになります。

```

1 この時、変数\lstinline[language=java]|x|の型は
2 \lstinline[language=java]|int|となる。

```

次のようになります。

この時、変数`x`の型は`int`となる。

情報科学類の学生は綺麗なソースコードを出力するために、秘伝のタレとなった `listings` の設定を持っている方が少なくありません。他の人の書いたレポートなどのソースコードがとても美しく出力されていたら、その人に声をかけて設定を覚えてもらおうのもよいでしょう。

6.20 書いた通りに出力する

`\verb` コマンドや `verbatim` 環境を利用するとテキストを書いた通りに出力することができます。これらを利用した場合、特殊文字をエスケープせずに使用することができます。特殊文字については 6.6.1 の特殊文字の項を参照してください。

6.20.1 出力したいテキストが複数行の場合

複数行にわたるテキストを書いた通りに出力したい場合、`verbatim` 環境を利用します。

```
1 \begin{verbatim}
```

```

2 verbatim環境を使うと、
3 書いた通りに出力することができます!#^^#
4 \end{verbatim}

```

結果は次のようになります。

verbatim 環境を使うと、
書いた通りに出力することができます!#^^#

6.20.2 出力したいテキストが1行の場合

1行のテキストを書いた通りに出力したい場合、`\verb`コマンドを利用します。以下の例のように、書いた通りに出力したい文字列を*以外の1文字で挟みます。

```
1 \verb"書いた通りに出力したい文字列を*以外の1文字で挟む"
```

結果は次のようになります。

書いた通りに出力したい文字列を*以外の1文字で挟む

6.21 BibT_EX を用いた参考文献

ここまででほとんどのレポートは問題なく書けるようになりました。しかしまだ論文（主に卒論）は書けません。ほとんどの卒論は他の文献を読まずには成り立ちません。つまり、何らかの参考文献が存在します。そのような論文はほぼ必ず参考文献を掲載します。ここでは参考文献を載せるための方法を紹介します。

6.21.1 BibT_EX を用いない

まず BibT_EX^{*33}を用いる前に、BibT_EX が自動生成しているプログラムについて説明します。ソースコード 6.10 を例に見ていきましょう。

今回は L^AT_EX の第一人者である奥村晴彦氏の著書、“[改訂第6版] L^AT_EX 2_ε 美文書作成入門”を例にします。

Listing 6.10: nobib.tex

```

1 \documentclass[a4j]{jsarticle}
2 \begin{document}
3 奥村先生の本\cite{奥村_黒木201310}を参考とした。
4 \begin{thebibliography}{10}
5   \bibitem{奥村_黒木 201310}%
6     奥村晴彦, 黒木裕介. [改訂第版6] ㄒTEX 2ε美文書作成入門. 技術評論社, 改訂第6,10 2013.
7 \end{thebibliography}
8 \end{document}

```

^{*33} 「びぶてふ」もしくは「びぶてっく」と読みます。 <http://ctan.org/pkg/bibtex>

まずこれをコンパイルしてみます。

```
$ platex -kanji=utf-8 nobib.tex↵
$ platex -kanji=utf-8 nobib.tex↵
$ dvi2pdf nobib.dvi↵
$ █
```

2回 `platex` コマンドを実行しているのは、1度目に後述の`\cite`による参照情報の更新を行い、2度目に最終的な出力ファイルを生成しているからです。

以下のような PDF ファイルが得られると思います。

図 6.8: nobib.pdf

奥村先生の本 [1] を参考とした。

参考文献

- [1] [奥村晴彦, 黒木裕介. \[改訂第 6 版\] L^AT_EX 2_ε 美文書作成入門. 技術評論社, 改訂第 6,10 2013.](#)

thebibliography 環境

ソースコード 6.10 の 4 行目から 6 行目は `thebibliography` 環境となっており、ここで文献情報を書いていきます。`\begin{thebibliography}{LONGEST-LABEL}~\end{thebibliography}`の間に`\bibitem[LABEL]{KEY} ENTRY`を並べていきます。

LABEL

参考文献のラベル（ソースコード 6.10 の赤文字部分）を指定します。このオプションは省略可能で、省略した場合には [1], [2], [3], のように 1 から連番となります。

もちろん数字以外にも英数字や空白、日本語などが指定できます。

KEY

文献を参照するための目印（ソースコード 6.10 の緑文字部分）を記述します。これを後述の`\cite`コマンドにより参照します。他の目印と重複しないようにしてください。

ENTRY

文献の内容（ソースコード 6.10 の青文字部分）を記述します。ここには任意の L^AT_EX コードが記述できます。

LONGEST-LABEL

LABEL の中で一番長い文字と同じ数だけ数字を入力します（ソースコード 6.10 の黄文字部分）。

たとえば LABEL を一切変更せず参考文献の数が 2 桁だった場合、22 とか 00, 72 など適当な 2 桁の数字を入力しておきます。2 桁の数字ならなんでも構いません。

また LABEL を変更し、`< changed label >` というラベル（`\bibitem[< changed label >]{なんとか}.....`）が最長だった場合、文字数は 15（記号が 2 文

字，ローマ字が12文字，空白が1文字）なので **123456789012345** など任意の15桁の数字を入力します。

\cite コマンド

\cite コマンドは文献の参照に用います。ソースコード 6.10 では“**奥村_黒木 201310**”という指定をしています。

\label コマンドと \ref コマンドによる参照方法（6.17 節）とは少し違うということがお分かりいただけたでしょう。参照する際の命令が違うとはいえ，基本的にはその性質は変わりません。いちいち参照するたびに数字を書き換える必要もありません。

6.21.2 BibT_EX を用いる

参考文献を掲載するには thebibliography 環境を用いる方法もありますが，L^AT_EX にまかせている処理はラベルに番号を振っている程度です。そこで文献内容などのレイアウトを一括して自動でおこなってくれるのが BibT_EX です。

BibT_EX は参考文献に関する書誌情報と呼ばれる，その参考文献の著者やタイトルなどという情報を決められたフォーマットで記述されたファイルを読み込みます。そして BibT_EX はその書誌情報を処理して L^AT_EX で扱える形にし埋め込んでくれるソフトウェア（処理系）です。これにより，参考文献のデータベースを単一の L^AT_EX ファイルのみならず，他のファイルとも共有できます。

6.21.3 書誌情報の入手

参考にした論文や資料の書誌情報は多くの場合 Web で入手できるので，皆さんが自力で作成する必要はありません。BibT_EX の書誌情報を提供している Web サイトは，T_EX Wiki の BibT_EX 関連ツール^{*34}にて紹介されています。また，CiNii^{*35}など論文検索サービスや学会の Web サイトでは論文の BibT_EX 書誌情報を公開している場合が多いです。

先程と同様，“[改訂第6版] L^AT_EX 2_ε 美文書作成入門”の BibT_EX 書誌情報^{*36}を例にします。BibT_EX の書誌情報は次のようなテキストファイルとなっています。

Listing 6.11: bibunsho.bib

```
1 @BOOK奥村{黒木_201310',
2 title改訂第版={ [6] ε美文書作成入門LaTeX2},
3 author奥村={ 晴彦 and 黒木裕介 },
4 publisher技術評論社={},
5 year={2013},
6 month={10},
7 edition改訂第={6},
8 isbn={9784774160450},
```

^{*34} <https://texwiki.texjp.org/?BibTeX%E9%96%A2%E9%80%A3%E3%83%84%E3%83%BC%E3%83%AB>

^{*35} 日本の論文検索サービス。 <http://ci.nii.ac.jp/>

^{*36} この書誌情報は Lead2Amazon (<http://lead.to/amazon/jp/>) という 書誌情報検索サービスから取得したものを一部改変したものです。


```

9 url={http://amazon.co.jp/o/ASIN/4774160458/},
10 price¥={ 3,360},
11 totalpages={432},
12 timestamp={2014.03.13},
13 }

```

このファイルを“bibunsho.bib”という名前で保存します。

6.21.4 L^AT_EX ファイル側の書式

BibT_EX の書誌情報を利用する L^AT_EX ファイルを用意する必要があります。次のソースコード 6.12 のようにします。

Listing 6.12: bibunsho.tex

```

1 \documentclass[a4j]{jsarticle}
2
3 \bibliographystyle{junsrt}
4
5 \begin{document}奥村先生の本
6
7 \cite{奥村黒木を参考とした. {_201310'}}
8
9 \bibliography{bibunsho}
10 \end{document}

```

このソースコード 6.12 には、`\bibliographystyle` コマンドと `\cite` コマンド、さらに `\bibliography` コマンドの3つが利用されています。これらについて解説します。

`\bibliographystyle` コマンド

これは参考文献の表示方法を設定するコマンドです。指定できるスタイル^{*37}とその意味を表 6.10 にまとめました。

表 6.10: 使用できるスタイルの例

スタイル	意味
jplain	最も標準的なスタイル
jalpha	文献の番号が著者名と出版年をあわせたものになる
jabbrv	できるだけ短かくする
junsrt	本文で引用した順に並べる

^{*37} BibT_EX のスタイルファイルはプログラム言語 T_EX ではなく、特殊なプログラム言語にて記述してあります。普通はスタイルファイルの実装について知る必要はありませんが、興味のある方は http://tug.ctan.org/info/bibtex/tamethebeast/ttb_en.pdf をご覧ください。

\cite コマンド

ソースコード 6.12 では“奥村_黒木 201310”という指定をしています。これは BIB_TE_X 書誌情報 bibunsho.bib (6.11) の1行目で指定している文献名です。

\bibliography コマンド

最後に \bibliography コマンドを用いて参考文献の一覧を出力します。ソースコード 6.12 にて、\bibliography コマンドに bibunsho という文字列を渡しています。これは読み込む BIB_TE_X 書誌情報 ファイルを示しています。ただし、書誌情報ファイルの拡張子は除きます。今回の例では bibunsho.bib というファイル名ですので、拡張子を取り除き bibunsho という文字列を与えます。

6.21.5 BIB_TE_X を用いた L^AT_EX ファイルのコンパイル

コンパイルする時の手順が増えます。BIB_TE_X を用いたコンパイルのやり方は以下のように行います。以下の例では先に述べた bibunsho.tex をコンパイルしています。

```
$ platex -kanji=utf8 bibunsho.tex↵
$ pbibtex -kanji=utf8 bibunsho↵
$ platex -kanji=utf8 bibunsho.tex↵
$ dvipdfmx bibunsho↵
$ █
```

pbibtex という、BIB_TE_X の処理系を日本語へ対応させたプログラムを用います。

thebibliography 環境を用いた場合と同じように、参照情報の更新と出力ファイル生成のために platex を2回実行します。

コンパイルが無事に終わると次のようになるはずです。

奥村先生の本 [1] を参考とした。

また巻末には次のような参考文献の一覧が出力されています。

参考文献

[1] 奥村晴彦, 黒木裕介. [改訂第6版] L^AT_EX 2_ε 美文書作成入門. 技術評論社, 改訂第6,10 2013.

6.22 文書の余白

文書の余白とは文書の上下左右の隙間のことです。ここでは文書の余白を調整する方法を紹介します。

さまざまな方法がありますが、geometry パッケージ^{*38}を用いる方法が最も簡単でかつ安全です。これは以下のように使用します。

^{*38} <http://www.ctan.org/pkg/geometry>

```
1 \usepackage{geometry}
2 \geometry{left=25mm, right=25mm, top=30mm, bottom=30mm}
```

左右の余白を 25mm にし、上下の余白を 30mm に設定しています。

この他にも `geometry` パッケージは特定のページだけ余白を設定するなど便利な機能が多数あるので、脚注*38 の URL にあるドキュメントを読むとよいでしょう。

6.23 目次の生成

目次は見出しをまとめたもののことで主に各章節への参照として用いられます。本文を書く際にはほとんど気にすることなく、目次に必要な情報は、`\section` などの見出し (6.10 節) から自動で取得されるので、皆さんは目次を表示するコマンドを記述するだけです。

目次を表示したいところに、`\tableofcontents` コマンドを使用すれば目次が表示されます。

6.24 発展的な T_EX の話題

この節では他の節に比べてやや発展的な T_EX の話題について述べます。

6.24.1 L^AT_EX をインストールする

皆さんが自身のコンピュータで L^AT_EX の編集をしたいことがあるでしょうから、この節では macOS, Windows, Linux 系 OS について L^AT_EX のインストール方法を紹介します。いずれも T_EX Live と呼ばれる、ありとあらゆる L^AT_EX の周辺ツールを同梱したものをインストールするだけです。

Windows の場合

<http://www.tug.org/texlive/acquire-netinstall.html> から `install-tl.exe` をダウンロードし、EXE ファイルを実行します。

macOS の場合

<http://tug.org/mactex/> から `MacTeX.pkg` をダウンロードし、`pkg` ファイルを実行します。

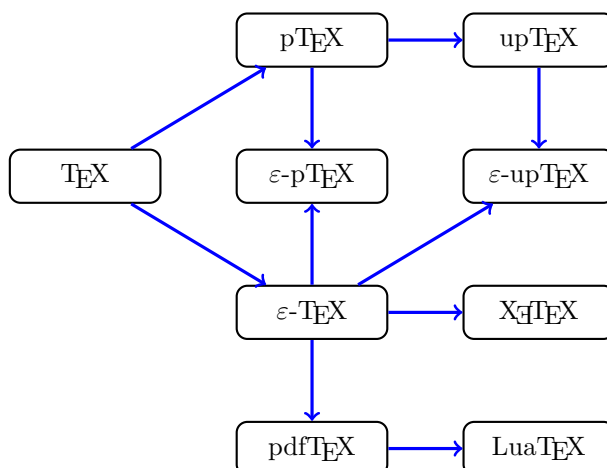
Linux の場合

パッケージマネージャーで `texlive` と検索し、ヒットしたものを調べてインストールします。例えば Ubuntu では `sudo apt-get install texlive-full` とします。

T_EX Live は後の節 (例えば 6.14 節など) にて解説される `graphicx` パッケージといった各種のマクロパッケージ (後述) を全て含みます。この章はいくつかのマクロパッケージを紹介しますが、紹介するものは全て T_EX Live に収録されているものです。従って、T_EX Live のインストール後に皆さんが何かパッケージをインストールする必要はありません。

6.24.2 ϵ -pT_EX 以外の T_EX 処理系

T_EX 処理系を図 6.9 にまとめました。図中の矢印は派生を示して、例えば ϵ -pT_EX は pT_EX と ϵ -T_EX の派生であるという意味です。興味のある方はこれらについて Web などで調べてみると良いでしょう。

図 6.9: T_EX 処理系の一部

ε-upT_EX

ε-pT_EX の派生として日本でよく利用されている ε-upT_EX^{*39} という処理系があります。これは内部の文字コードを Unicode へ対応させ「高（はしごたか）」といった特殊な文字や韓国語などを扱えるようにしてあります。

pdfT_EX

T_EX の派生で特に有名な pdfT_EX^{*40} は、DVI ファイルを介さず直接 PDF を生成します。DVI ファイルを通らないので `dvipdfmx` を必要とせず、PDF の強力な表現力を利用できることから欧文圏では高い人気があります。ただ pdfT_EX は日本語に関する組版処理が ε-pT_EX と比べて劣るなどの理由から、日本ではあまり用いられていません。

注意すべきは、海外のフォーラムなどで L^AT_EX について論じられている場合、この pdfT_EX を前提としている可能性が極めて高いということです。するとソースコードをコピーしたにも関わらず、ε-pT_EX では正しく動作しないということがありえます。

X_ƒT_EX

X_ƒT_EX はコンピュータにインストールされたフォントを直接使うことで、面倒な設定をすることなく、フォントの高度な情報を利用することができます。日本語のフォントを想像するとよく分からないかもしれませんが、例えばアラビア語といった複雑な文字を美しく表現したい場合、T_EX の処理系を改造するなどして対応するのは大変なことです。そこで X_ƒT_EX はフォント情報を読み込むことで、処理系を改造することなく複雑な文字をも容易に扱えます。

^{*39} 「いーゆーピーてっく」もしくは「いーゆーピーてふ」と発音します。 <http://sourceforge.jp/projects/eptex/>

^{*40} <http://ctan.org/pkg/pdftex>

LuaT_EX

最近では pdfT_EX の後継として LuaT_EX^{*41}の開発が盛んに行われています。LuaT_EX は L^AT_EX ファイルの中にプログラム言語 Lua^{*42}を記述できるというもので、Lua を用いて複雑な組版処理を簡潔に実装できるとされています。また、X_YT_EX のようにフォント情報を読み込むといった機能にも対応しています。

日本では LuaT_EX-ja^{*43}というプロジェクトがあり、ε-pT_EX に匹敵する日本語組版能力を LuaT_EX に実装しようと熱心な開発が行なわれています。

6.24.3 ConT_EXt

L^AT_EX 以外にも純粋な T_EX を強化するためのマクロパッケージはいくつか存在します。最近では海外を中心に、ConT_EXt^{*44}という L^AT_EX 以外のマクロパッケージを用いた文書も増えつつあります。L^AT_EX とは違い、次のような文法となっています。

Listing 6.13: ConT_EXt による文書の例^{*45}

```

1 \setuphead[title][style={\ss\bfd},
2   before={\beginngroup},
3   after={John Doe, the author\smallskip%
4     \currentdate\bigskip\endgroup}]
5
6 \starttext
7
8 \title{\CONTEXT}
9
10 \section{Text}
11 \CONTEXT\ is a document preparation system for the
12 \TEX\ typesetting program. It offers programmable
13 desktop publishing features and extensive
14 facilities for automating most aspects of
15 typesetting and desktop publishing, including
16 numbering and cross-referencing (for example to
17 equation \in[eqn:famous-emc]), tables and figures,
18 page layout, bibliographies, and much more.
19
20 It was originally written around 1990 by Hans
21 Hagen. It could be an alternative or complement
22 to \LATEX.
23
24 \section{Maths}
25 With \CONTEXT\ we could write maths. Equations
26 can be automatically numbered.
27
```

^{*41} <http://ctan.org/pkg/luatex>

^{*42} 高速な動作や組み込みの容易さという特徴を持つプログラム言語。

^{*43} <http://sourceforge.jp/projects/luatex-ja/>

^{*44} 「こんでくすと」と発音します。 <http://wiki.contextgarden.net/>

```

28 \placeformula[eqn:famous-emc]
29 \startformula
30     E = mc^2
31 \stopformula
32 with
33 \placeformula[eqn:def-m]
34 \startformula
35     m = \frac{m_0}{\sqrt{1-\frac{v^2}{c^2}}}
36 \stopformula
37
38 \stoptext

```

6.24.4 TikZ による図

TikZ^{*46}とはPGFと呼ばれるT_EX用の図形描画を目的としたマクロパッケージのフロントエンドです。TikZを用いることで、さまざまな図をテキスト形式で記述することができます。本章の図6.2や図6.1もTikZにより生成されています。

例えば図6.2は次のようなソースコードで作られます。

```

1 \begin{tikzpicture}[scale=1.5,
2   block/.style = {rectangle, draw=black, thick, text width=6em, align=center,
3     rounded corners, minimum height=2em},
4   textblock/.style = {rectangle, text width=6em, align=center, minimum height=2em}]
5
6 \draw (-1, -1) node[block] (A) ソースファイル{};
7 \draw (2, -1) node[block] (B) {ファイルDVI};
8
9 \draw[->, very thick, blue] (A) -- (B);
10
11 \draw (0.5, -1.2) node (Bd) {\epTeX};
12
13 \draw (5, -1) node[block] (C) {ファイルPDF};
14 \draw (5, 0.2) node[block] (D) {ファイルPS};
15
16 \draw[->, very thick, blue] (B) -- (C);
17 \draw (3.5, -1.2) node (Cd) {\dvipdfmx};
18
19 \draw[->, very thick, blue] (B) -- (D);
20 \draw (3.25, -0.3) node (Dd) {\texttt{dvips}};
21
22 \draw[->, very thick, blue] (D) -- (C);
23 \draw (5.45, -0.4) node (Dd) {\texttt{ps2pdf}};
24 \end{tikzpicture}

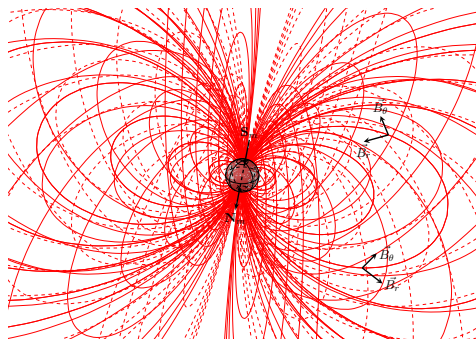
```

TikZを説明するのはとても大変で紙面の都合もあるので、この節ではTikZによって作られた美しい図を

^{*45} http://en.wikipedia.org/wiki/ConTeXt#Example_of_code

^{*46} 「ていっくず」と発音します。 <http://www.ctan.org/pkg/pgf>

T_EXample.net^{*47}よりいくつか拝借してきました。興味のある方はぜひ脚注*47へアクセスして、素晴らしい図のソースコードを眺めてみましょう。



Schematic Earth dipolar magnetic field. The field lines placed in the page plane are drawn as thick lines, those back with dashed lines and the field lines in front of the page with thin lines.

図 6.10: <http://www.texample.net/tikz/examples/dipolar-magnetic-field/>

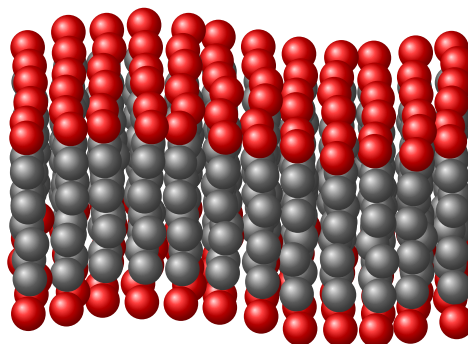


図 6.12: <http://www.texample.net/tikz/examples/membrane-surface/>

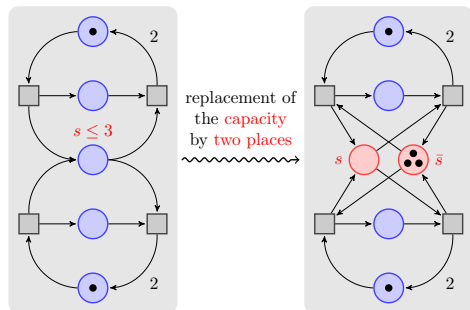


図 6.11: <http://www.texample.net/tikz/examples/nodetutorial/>

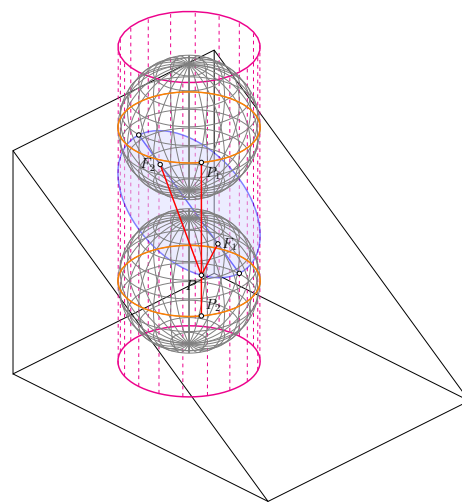


図 6.13: <http://www.texample.net/tikz/examples/dandelion-spheres/>

TikZ はこのような高度な図を Illustrator といった高価なソフトウェアを用いずに、あくまでテキストベースで作ることができます。

また T_EX - L^AT_EX Stack Exchange^{*48}という Web サイトにて、天下一科学系図コンテスト^{*49}というスレッドが立ち、TikZ に限らずさまざまなソフトウェアを用いて作られた、素晴らしい科学系の図が投票により順位付けされています。上記の図や T_EXample.net を見て感動した方はこちらを見るのもよいでしょう。

^{*47} <http://www.texample.net/tikz/examples/>

^{*48} <http://tex.stackexchange.com/>

^{*49} 原題は “Nice scientific pictures show off” です。 <http://tex.stackexchange.com/questions/158668/nice-scientific-pictures-show-off>

参考文献

- [1] 奥村晴彦, 黒木裕介. [改訂第 6 版] L^AT_EX 2_ε 美文書作成入門. 技術評論社, 改訂第 6, 10 2013.
- [2] ページエンタープライゼズ. L^AT_EX 2_ε マクロ&クラス プログラミング基礎解説. 技術評論社, 8 2002.
- [3] T_EX Wiki. <http://oku.edu.mie-u.ac.jp/~okumura/texwiki/>. Accessed: 2014-03-14.
- [4] T_EX - L^AT_EX Stack Exchange. <http://tex.stackexchange.com/>. Accessed: 2014-03-14.
- [5] マクロツイーター. <http://d.hatena.ne.jp/zrbabbler/>. Accessed: 2014-03-14.
- [6] LuaT_EX-ja. <http://sourceforge.jp/projects/luatex-ja/>. Accessed: 2014-03-14.
- [7] T_EX by Topic. <http://www.ctan.org/pkg/texbytopic>. Accessed: 2014-03-14.
- [8] Tame the BeaST. http://tug.ctan.org/info/bibtex/tamethebeast/ttb_en.pdf. Accessed: 2014-03-14.

第 7 章

言語処理系

この章では COINS の計算機環境上での Java, C, C++, FORTRAN, MATLAB によるプログラムの開発方法や、実行方法を説明します。

7.1 言語処理系とは

コンピュータが動作を行うためには、CPU が解釈できる機械語の形式でのプログラムが必要になります。しかし機械語は人間には解釈しづらいため、一般的には C など人間が読みやすいプログラミング言語を使ってプログラムを書きます。

この言語を解釈して直接 CPU などの処理系が解釈することのできる言語に変換するためのプログラムがあります。機械語、あるいはそれに準ずる形式の実行ファイルを出力するタイプの処理系をコンパイラ (Compiler) といい、ソースファイルを読み出したり人間の入力を解釈してその都度命令を実行するタイプのをインタプリタ (Interpreter) といいます。コンパイラは、ソースファイル (Source File) を入力して実行ファイル (Execution File) を生成します。このとき、ソースファイル中のプログラムのことをソースコード (Source Code)、コンパイラを使って変換を行うことをコンパイル (Compile) と言います。インタプリタもソースコードを機械語に変換して実行しますが、普通は機械語のファイルを出力しないところがコンパイラと異なります。

7.2 Java コンパイラ

Java は、オブジェクト指向言語 (Object Oriented Language) です。オブジェクト指向言語とは、プログラムをオブジェクトという機能の単位で構成しようとするプログラミングスタイルのための言語です。

Java でプログラムを書くということは、オブジェクトを作成する際のひな形となる、クラスの定義を記述するということです。クラスの定義には、そのオブジェクトがどのようなデータを持っているかという定義と、どのような手続き (メソッド) を持っているかという定義が含まれます。Java のプログラムを実行すると、プログラマが記述したクラス定義に従ってオブジェクトが生成され (インスタンス化)、生成されたオブジェクト間でのやりとりを通じてプログラムの実行が行われます。

また、Java のプログラムは高いポータビリティを持っており、同じプログラムを macOS, Windows, Linux いずれでも実行することができます*1。特に、通常の Java のプログラム以外に、アプレットと呼ばれる Web

*1 近年では Android も Java が機能するプラットフォームとして有名です。

ページなどに埋め込み可能なプログラムを作ることができます。これについては、この節の後半で説明をしていきます。

7.2.1 Java プログラムのコンパイルと実行

Java のソースファイルのファイル名は、ファイル中で宣言されているクラスの名前に、“.java”の拡張子をつけたファイルにしなければなりません。例えば、MyTest というクラスの定義を含むソースファイルは、MyTest.java というファイル名でなくてはなりません。

まず、サンプルとして “Hello, Java World!” と表示するためのプログラムを紹介します。このプログラムは、以下の通りとなります。

Listing 7.1: MyTest.java

```
1 public class MyTest {
2     public static void main(String args[]) {
3         System.out.println("Hello, Java world!");
4     }
5 }
```

このプログラムをコンパイルするためには、javac コマンドを用いて以下のようにコマンドを入力します。

```
$ javac MyTest.java↵
```

これにより、MyTest.class という名前のクラスファイル (*Class File*) が生成されます。クラスファイルとは、ソースファイルをコンパイルした結果のバイトコードが書かれています。バイトコードは、Java の実行環境が解釈して実行します。この実行環境のことを Java Runtime Environment (JRE) と言います。

Java の実行環境を使ってプログラムを実行させるためには、java コマンドを使用します。引数には、main 関数の入っているクラスのクラス名を指定します。引数に指定するのは、あくまでもクラス名のみで、クラスファイルの名前ではないことに注意します。

MyTest クラスを実行させる場合には次のようにします。

```
$ java MyTest ↵
Hello, Java world!
$ █
```

7.2.2 Java アプレットのコンパイルと実行

アプレットを使用したプログラムでは、通常の Java プログラムと比べてプログラムの作り方が変わってきます。また、Web ページ上に埋め込むので、HTML ファイルも必要になってきます。

ここでは、サンプルとして、アプレットのウィンドウを開いて “Hello, Java World” と表示するプログラムを説明します。サンプルの Java のコードは以下の通りです。

Listing 7.2: MyTestApl.java

```
1 import java.applet.Applet;
2 import java.awt.Graphics;
3
4 public class MyTestApl extends Applet {
```

```
5 public void init() {
6     resize(150,30);
7 }
8 public void paint(Graphics g) {
9     g.drawString("Hello, Java world!", 50, 25);
10 }
11 }
```

これをコンパイルするには、先ほどと同じようにして javac コマンドを使用します。

```
$ javac MyTestApl.java↵
```

コンパイルしたアプレットのプログラムを呼び出すための HTML ファイルを用意します。今回は、以下のファイルを使用します。

Listing 7.3: MyTestApl.html

```
1 <html>
2   <head>
3     <title>MyTestApl Applet</title>
4   </head>
5   <body>
6     MyTestApl Applet
7     <hr>
8     <applet code="MyTestApl.class" width=150 height=30></applet>
9     <hr>
10    <a href="MyTestApl.java">Source</a>
11  </body>
12 </html>
```

これを実行するには、appletviewer コマンドを使用します。また、このプログラムを実行すると図 7.1 のようになります。

```
$ appletviewer MyApplet.html↵
```



図 7.1: Java アプレットの実行画面

7.3 C

C によるプログラミングの方法を説明します。デバッグ方法を解説します。ここでは、C プログラムのより高度なコンパイル方法を紹介します。

7.3.1 C コンパイラ

C コンパイラ (*C Compiler*) は C で書かれたソースコードをもとにして実行ファイルを出力します。以下、C コンパイラを用いてソースファイルをコンパイルし、実行する手順を説明します。

C のソースファイルには、“.c” という拡張子をつけます。例えば、sample.c というファイル名とします。ソースファイルは、Emacs や vi などのテキストエディタで作成してください。

まず、ターミナルに “Hello, C world!” という文字列を出力する簡単なソースコードを示します。ソースファイルのファイル名は、test.c としてください。左側に書かれている数字は行数を示していますが、説明のために便宜上つけてあるだけで、実際のソースコードには入力する必要はありません。

Listing 7.4: test.c

```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello, C world!\n");
6     return 0;
7 }
```

利用しているターミナルによっては、ソースコードにあらわれるバックスラッシュ (\) が円記号になる場合もありますが、これは同じ ASCII コード (0x5C) が割り当てられていることによって起こります。^{*2}強制的にバックスラッシュを入力するには、+を同時に押します。

このソースファイルをコンパイルするためには、gcc というコマンドを使ってコンパイルを行います。gcc とは、GNU Compiler Collection (gcc) を実行するためのコマンドです。^{*3}gcc は Free Software Foundation という組織がオープンソースで提供しているコンパイラで、様々なコンピュータ上で動作します。gcc を使ってコンパイルをするために、以下のように test.c が入っているディレクトリでコマンドを実行してください。

```
$ gcc test.c
```

コンパイルを実行すると、a.out という名前の実行ファイルが生成されます。生成された実行ファイルを実行するには、以下のようにします。

```
$ ./a.out
Hello, C world!
$
```

実行すると、“Hello, C world!” という文字列がターミナルに表示されます。

^{*2} Shift JIS, EUC-JP などがこの問題を抱えています。UTF-*では円記号とバックスラッシュは別のコードになっています。

^{*3} macOS 環境下では実際には GNU Compiler Collection ではなく、LLVM が起動する

gcc コマンドでオプションを指定せずにコンパイルを行うと、前述のように a.out という名前の実行ファイルが出力されます。これを任意の名前に変えるためには、gcc の “-o” オプションを使用します。例えば、hello という実行ファイルを出力したい場合、以下のようにします。

```
$ gcc -o hello test.c
```

7.3.2 ヘッドファイル, ライブラリ

先ほどのプログラムでは、プログラム内で文字列を出力するために printf 関数が使われています。この関数を用いるために、printf 関数が宣言されている stdio.h という名前のファイルをインクルードしています。このようなファイルをヘッドファイル (*Header File*) と言います。

また、関数の宣言に対応する関数の実体は、ライブラリファイル (*Library File*) と呼ばれるファイルの中に存在します。ライブラリファイルは、コンパイル時に結合する必要がある、このことをリンク (*Link*) すると言います。標準的なライブラリファイルは自動的にコンパイル時にリンクされますが、そうではないライブラリは明示的に指定する必要があります。これについては、7.3.10 節で詳しく説明をします。

どの関数がどのインクルードファイルに宣言されているかを調べるためには、man コマンドを使ってマニュアルで調べます。調べるときには、シェルから “man (関数名)” と実行します。また、このマニュアルの中では、関数の使い方や関数の引数、オプション、注意などが載っていますので、関数を使うときにはこのマニュアルを見るようにしましょう。

例えば、sqrt 関数の man を開くと、マニュアルに以下のように載っています。書式の部分では、関数の戻り値、引数などが書いてあります。説明の部分では、この関数の使い方などが書いてあります。バグの部分では、関数を使う際の注意が書いてあるので、注意が必要です。

— sqrt 関数の man —

```
SQRT(3)                BSD Library Functions Manual                SQRT(3)

NAME
    sqrt -- square root function

SYNOPSIS
    #include <math.h>

    double
    sqrt(double x);

    long double
    sqrtl(long double x);

    float
    sqrtf(float x);

DESCRIPTION
    The sqrt() function compute the non-negative square root of x.

SPECIAL VALUES
    sqrt(-0) returns -0.

    sqrt(x) returns a NaN and generates a domain error for x < 0.

SEE ALSO
    math(3)

STANDARDS
    The sqrt() function conforms to ISO/IEC 9899:1999(E).

BSD                December 11,                2006    BSD
```

7.3.3 参考になる資料

- man gcc
- ハーバート・シルト著，柏原 正三 監修「独習 C」（翔泳社）
- B.W. カーニハン/D.M. リッチー著 石田晴久訳「プログラミング言語 C（第 2 版）ANSI 規格準拠」

(共立出版)

7.3.4 デバッグ

7.3.5 printfを使った簡単なデバッグ

ここでは、C プログラムのデバッグについて説明します。デバッグ (*Debug*) とは、プログラムの不具合 (バグ) を修正することです。簡単なデバッグの方法としては、ソースコード中にデバッグ情報の出力を埋め込む方法があります。

簡単にできるデバッグの方法として、printf を使ってプログラム中で使用している変数などの情報を出力する方法があります。これは単にプログラム中で printf 関数を使って変数の中身を表示するだけですが、簡単にできます。しかし、この方法だとデバッグ情報を出力する部分がプログラム中に埋め込まれてしまい、デバッグが終わった後にデバッグ情報の出力部分をコメントアウトする手間がかかります。

そこで、#ifdef 文を用いてデバッグの時とそうでない時で、プログラムのコードを切替えられるようにします。これにより、デバッグの時だけ、デバッグ情報を出力することが可能になります。

次のプログラムは、#ifdef 文を使ってプログラムを切替えられるようにしたものです。

Listing 7.5: test_debug.c

```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello, C world!\n");
6 #ifdef DEBUG
7     printf("This is debug.\n");
8 #endif
9     return 0;
10 }
```

test_debug.c は test.c の 6 行目～8 行目に#ifdef 文を使ったプログラムを追加したものです。“#ifdef (マクロ名) ~ #endif” にはさまれた部分は、(マクロ名) が定義されているとコンパイルの対象になります。もし、マクロが定義されていない場合、この部分はコンパイル時に無視されます。つまり、コンパイル時にマクロを定義することで、デバッグ用のプログラムを生成することができます。

マクロを定義する方法には、#define をソースファイル中に定義する方法とコンパイル時にコンパイラのオプションで指定する方法があります。このうち、コンパイル時にマクロを指定するには、コンパイルオプションに“-D(マクロ名)”と指定します。

```
$ gcc -o test_debug test_debug.c -DDEBUG↵
$ ./test_debug↵
Hello, C world!
This is debug.
$ gcc -o test_debug test_debug.c↵
$ ./test_debug↵
```

```
Hello, C world!
```

7.3.6 デバッガの利用

デバッガ (*Debugger*) というデバッグをサポートするためのプログラムを使うことで、効率的にデバッグを行うことができます。ここでは、上記2つの方法をそれぞれ説明します。

`printf` を使ったデバッグの方法について説明しましたが、この方法では、デバッグしたい箇所に全部のデバッグ情報を書かなければなりません。また、短いプログラムの場合、ソースコードを見ているだけでも簡単にバグを発見することができますが、ソースプログラムが長く、複雑になると、バグを見つけることが難しくなってきます。

そこで、デバッガというデバッグをサポートするツールを使います。一般的なデバッガでは、プログラムを途中で止めて変数の中身を見たり、1行ごとにプログラムを実行したりすることができます。これにより、非常に効率良くデバッグを行うことができます。例えば、課題のプログラムなどでバグがどこにあるかわからなくなったら、デバッガを使ってデバッグすることで、バグを見つけることができるかもしれません。

ここでは、次の簡単な剰余計算を行うプログラムをデバッグするとします。デバッガには、`gdb` (GNU デバッガ) を使用します。

Listing 7.6: test_gdb.c

```
1 #include <stdio.h>
2
3 int mymod(int i, int j) {
4     while(i >= j)
5         i -= j;
6     return i;
7 }
8
9 int main() {
10     int x = 20, y = 3;
11     printf("%d (mod %d) = %d\n", x, y, mymod(x, y));
12     return 0;
13 }
```

デバッガを利用するためには、コンパイル時にデバッグに必要な情報をプログラムに埋め込んでおく必要があります。そのためには、`gcc` に “-g” オプションをつけてソースコードをコンパイルします。

```
$ gcc -o test_gdb test_gdb.c -g
```

このようにして実行ファイルを生成すると `gdb` を利用してデバッグを行うことができます。

次に、`gdb` コマンドを使って、`gdb` を起動させます。引数には、実行するファイルのファイル名を指定します。

```
$ gdb test_gdb
```


起動すると、以下のように表示されます。ここから、コマンドでデバグを操作します。

gdb の起動

```
$ gdb test_gdb↵
GNU gdb 6.3.50-20050815 (Apple version gdb-1346) (Fri Sep 18 20:40:51
UTC 2009)
Copyright 2004 Free Software Foundation, Inc. (省略)

(gdb) █
```

デバグの使い方はいろいろとあるのですが、ここでは、プログラムを特定の場所で止めるブレークポイントの機能を使って、途中でプログラムを止めてデバグする方法を例にして説明をしていきます。

ブレークポイントを設定するためには、break コマンドを使います。break コマンドの引数には、関数名や行数を指定します。このとき、複数のファイルで構成されているプログラムにブレークポイントを設定するためには、ファイル名を指定する必要があります。ファイル名を指定するには、“break (ソースファイル名):(関数名 or 行数)” というように、引数にファイル名を含めて記述します。

```
(gdb) break mymod↵
Breakpoint 1 at 0x100000e96: file test_gdb.c, line 4.
```

ブレークポイントを設定したら、run コマンドを使ってプログラムを実行します。このとき、デバグするプログラムに実行するための引数を付けたい場合は、run コマンドの引数にその引数を入力します。

```
(gdb) run↵
Starting program: /home/gikan/tebiki/tebiki01/compiler/test_gdb

Breakpoint 1, mymod (i=20, j=3) at test_gdb.c:4
4      while(i >= j)
```

実行したプログラムは、ブレークポイントを設定した場所に到達すると、その場所でプログラムが一時停止し、gdb でコマンドを入力できるようになります。

この状態で、変数の中身を見るためには、print コマンドを使用します。例えば、変数 i の中身を表示するためには、以下のようにします。

```
(gdb) print i↵
$1 = 20
```

また、この状態で変数の中身を書き換えることもできます。その場合には、“(変数名) = (値)” と入力します。

一時停止している状態で 1 行処理を進ませたい場合には、next コマンドを使います。これにより、1 行ごとに処理を実行させていくことができます。また、next など、何度も同じコマンドを入力しなければならない

とき、毎回同じコマンドを入力するのは非常に面倒です。そのため、gdbには前の操作をもう一度繰り返す機能がついています。これを行うためには、何も入力されていない状態で`↵`を押します。

一時停止している状態では、さらに他の場所にブレークポイントを設定することができます。また、設定したブレークポイントを削除するためには、`clear` コマンドを使います。clear コマンドの引数には、break コマンドと同じようにして、削除するブレークポイントの関数名や行数を指定します。

最後に、プログラムの実行を再開する場合には、`continue` コマンドを使用します。これにより、プログラムの実行を再開します。また、デバッガを終了するには、`quit` コマンドを使用します。

これ以外にも、条件付きのブレークポイントの設定やバックトレースの表示など、大変便利な機能があります。とくに、セグメンテーションフォルトをはじめとするランタイムエラーの追跡に、gdb はたいへん役に立つでしょう。詳細は、`man` コマンドや gdb 起動中の `help` コマンド、インターネット上の情報などを参考にしてください。

7.3.7 分割コンパイル

大きなプログラムは、いくつものソースファイルに分割して作成します。その場合、コンパイラを使ってソースファイルごとにオブジェクトファイルを作成し、最後にそれをリンクします。このことを分割コンパイルと言います。

まず、オブジェクトファイルを作成するために、gcc で “-c” オプションをつけてソースファイルをコンパイルします。次に、このままでは実行できないため、リンカを使ってプログラムをリンクします。これには、gcc の引数にオブジェクトファイルを指定して実行します。

例えば、次のような2つのソースファイルがあるとします。

Listing 7.7: test_main.c

```
1 extern void myfunc();
2 int main() {
3     myfunc("Hello, \u00a0make\u00a0world.\n");
4     return 0;
5 }
```

Listing 7.8: test_main.c

```
1 #include <stdio.h>
2 #include <string.h>
3
4 void myfunc(char *s) {
5     printf("(%d)\u00a0%s", (int)strlen(s), s);
6 }
```

これらのファイルをコンパイルしてオブジェクトファイルを作成するには、以下のようにします。

```
$ gcc -c test_main.c↵
$ gcc -c test_lib.c↵
```

次に、生成されたオブジェクトファイルをリンクします。リンクするためには、以下のようにします（-o オプションは、プログラムの実行ファイル名を決めるためのもので、リンク自体には関係ありません）。これに

より、test というプログラムが作成され、普通のプログラムと同じようにして実行することができます。

```
$ gcc -o test test_main.o test_lib.o↵
$ ./test↵
(19) Hello, make world.
```

7.3.8 make を使ったコンパイル

大きなプログラムを分割コンパイルする場合、毎回全てのソースコードをコンパイルしていたら非常に時間がかかってしまいます。そこで、*make* というプログラムを使って、前回コンパイルしたときから編集されたファイルのみを再コンパイルするようにします。

make コマンドでは、*Makefile* というソースコードをコンパイルする手順などを記した設定ファイルを作成します。例えば、先ほどの分割コンパイルの例のプログラムでは、以下のように Makefile を記述します。

```
Makefile

CC = gcc
SRC = test_main.c test_lib.c
OBJ = test_main.o test_lib.o
TARGET = test

$(TARGET): $(OBJ)
    $(CC) -o $(TARGET) $(OBJ)
```

make コマンドを使うためには、Makefile があるディレクトリで、make コマンドを実行します。これにより、編集されたファイルのみを自動的に選んで、必要なアクションを実行します。

```
$ make↵
gcc -o test test_main.o test_lib.o
$ ./test↵
(19) Hello, make world.
```

7.3.9 最適化

最適化 (*Optimization*) とは、コンパイラが効率のよい実行ファイルを生成する機能です。この機能を使うことで、プログラムの実行速度が速くなったり、実行ファイルのサイズが小さくなったりします。

最適化を行うためには、コンパイル時のオプションに “-O” に続けて数字のパラメータを与えます。“-O1”、“-O2”、“-O3” の順に最適化の度合いが強くなっていきます。“-O2” のオプションをつけてコンパイルするためには、以下のようにします。

```
$ gcc -O2 -o test test.c↵
```

7.3.10 ライブラリ

ライブラリ (*Library*) とは、ある機能を提供するための関数群をまとめたものです。この関数群をファイルにまとめたものをライブラリファイルと言います。例えば、算術演算のライブラリであれば、libm.a というライブラリファイルを使用します。

ライブラリに入っている関数を使うためには、まず、その関数に必要なヘッダファイルをインクルードします。しかし、これだけだと関数の宣言のみで関数の中身がありません。そのため、コンパイル時にライブラリファイルを一緒にリンクします。

次のようなプログラムをコンパイルするとします。このプログラムは、math.h をインクルードしており、プログラム中で算術演算のライブラリに入っている sqrt 関数を使用しています。

Listing 7.9: test_math.c

```
1 #include <stdio.h>
2 #include <math.h>
3 int main() {
4     double x = 2.0;
5     double y = sqrt(x);
6     printf("%f\n",y);
7     return 0;
8 }
```

このプログラムをコンパイルするときには、“libm.a” のライブラリファイルが必要とします。コンパイル時にライブラリファイルを指定するためには、“-l” の後にライブラリファイルの “lib” と “.a” を取り除いた部分を記述します。例えば、“libm.a” というライブラリファイルであれば、“-lm” とします。以下に実際にコンパイルするときのコマンドを示します。

```
$ gcc test_math.c -lm↵
```

標準のライブラリは、/lib か /usr/lib にあるという設定になっているので、gcc はこの 2 か所以外は探しません。これら以外にライブラリがある場合は、ライブラリの絶対パス名を書くか、-L オプションを用いて、ライブラリを検索するディレクトリを指定する必要があります。

7.4 C++ コンパイラ

ここでは、C++ コンパイラの使い方について説明をします。

7.4.1 C++ プログラムのコンパイルと実行

C++ のソースファイルの拡張子は、“.c++”、“.cc”、“.cpp”、“.c” とします。ただし、C のソースファイルと紛らわしいので “.c” をつけるのは、やめた方が良いでしょう。

C の場合は、コンパイラに gcc を使っていましたが、C++ の場合には g++ を使います。C と C++ のコンパイル方法の違いは、gcc を使うか g++ を使うかということだけで、オプションなどはほとんど変わりません。

例えば、次の test.cpp をコンパイルするとします。

Listing 7.10: test.cpp

```
1 #include <iostream>
2
3 int main() {
4     std::cout << "Hello, C++ world!" << std::endl;
5     return 0;
6 }
```

このソースファイルをコンパイルするには、以下のようにします。

```
$ g++ test.cpp
$ ./a.out
Hello, C++ world
```

これにより、a.out という名前の実行ファイルが生成されます。

C コンパイラと同様にして、“-o” オプションをつけることで、生成される実行ファイル名を指定することもできます。また、C の部分で説明をした gdb でのデバッグも、C と同様に行うことができます。

7.4.2 参考になる資料

- man g++
- ハーバート・シルト著 神林靖監修「独習 C++」(翔泳社)
- B. ストラウストラップ著 長尾高弘訳「プログラミング言語 C++ (第3版)」(アスキー・アジソンウェスレイシリーズ)
- M. A. エリス, B. ストラウストラップ著 足立剛徳, 小山裕司訳「注解 C++ リファレンスマニュアル」(トッパン)

7.5 FORTRAN コンパイラ

ここでは、FORTRAN コンパイラの使い方について説明をします。

7.5.1 FORTRAN95 プログラムのコンパイルと実行

FORTRAN のコンパイラとして、GNU FORTRAN95 コンパイラ (gfortran) を利用します。GNU FORTRAN コンパイラは、Free Software Foundation が無料で配布しています。ここでは、このコンパイラを使って説明をしていきます。

FORTRAN 言語のソースファイルの拡張子は、“f” あるいは “.F” とします。例えば、以下の test.f をコンパイルするとします。

Listing 7.11: test.f

```
1 PROGRAM TEST
2 REAL X,Y
3 X = 3.0
4 Y = SQRT(X)
```

```
5 WRITE(*,*) Y
6 END
```

各行の先頭にはインデントを入れてください。このプログラムをコンパイルするには次のようにします。

```
$ gfortran test.f↵
```

これにより、a.out という名前の実行ファイルが生成されます。また、C コンパイラや C++ コンパイラと同様に “-o” オプションをつけることで、生成される実行ファイル名を指定することもできます。

7.5.2 参考になる資料

- man gfortran
- 中田育男著「基礎 FORTRAN」(岩波書店)

7.6 MATLAB

MATLAB とは MathWorks 社が開発している数値解析ソフトウェアです。C や C++、Java といったプログラミング言語では実装が困難な行列計算やベクトル演算、グラフ化等を簡単に行うことができ、数値計算やネットワーク*4から、制御工学や電気工学といった分野にも非常に有用な機能を多数備えています。また、MATLAB Engine というライブラリを導入して、C や C++ から MATLAB の機能を利用する、といったことも可能になります。MATLAB は Windows、macOS、Linux だけでなく iOS や Android でも利用できます。すべての OS での画面について操作を説明することは困難なため、ここでは Windows における画面を用いて解説します。

7.6.1 起動

インストールされたプログラムの一覧から MATLAB を選択することで、ウィンドウが「コマンドウィンドウ」、「ワークスペース」等に分割された GUI で MATLAB が起動します。

*4 ここでのネットワークとはグラフ理論におけるネットワークです。

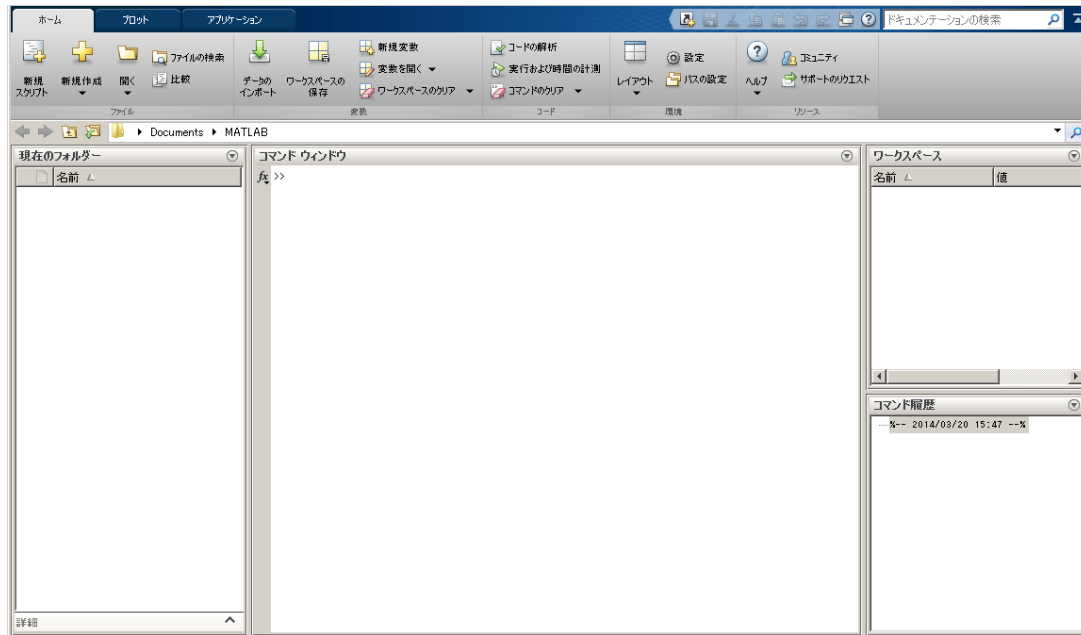


図 7.2: MATLAB

MATLAB はターミナルから利用することもできますが，GUI を用いて利用した方が配列や変数の状態を確認できたり，「*Simulink*」といった MathWork 社が提供するソフトウェアと連携しやすくなりますので，ここでは GUI で操作しているものとして解説していきます．

7.6.2 計算の実行

MATLAB で変数や配列の初期化及び代入を行ったり，それを用いた数値計算を行う場合は「コマンドウィンドウ」に記述します．変数の初期化を行う場合は，以下のように記述します．

```
a=10
```

このように記述し，を押すことで入力した内容が実行されます．この場合の実行結果は以下の通りです．これはコマンドウィンドウに表示されます．

```
a =
```

```
10
```

MATLAB では基本的に「int」や「float」といった変数型を指定する必要はありません．上記の a に整数ではなく小数点以下を含む実数を代入してもエラーは起きません．配列の場合は列区切りに「,」を，行区切りに「;」を用います．例えば，サイズ 2×2 の行列の初期化は以下のように記述します．また，記述した数式の末尾に「;」を加えると，実行後に実行結果がコマンドウィンドウに表示されません．サイズが大きい行列の計算を行う場合等，コマンドウィンドウが流れてしまうことを考えて「;」を加えたほうが良い場合もあります．

```
A=[10,10;20,20];
```

「;」を加えて実行結果を表示しなかった場合でも変数や配列の内容を確認する方法はあります。例えば先ほど初期化を実行した「A」の内容を確認したい場合、次のように記述すれば確認できます。

```
A←
```

この場合の実行結果は以下の通りです。

```
A =

    10    10
    20    20
```

また、すべての変数と配列の内容は「ワークスペース」でも確認できます。ワークスペース内にはこれまでに利用した変数や配列名が表示され、確認したい対象の名前をダブルクリックするとコマンドウィンドウが分割されて対象の内容が表示されます。

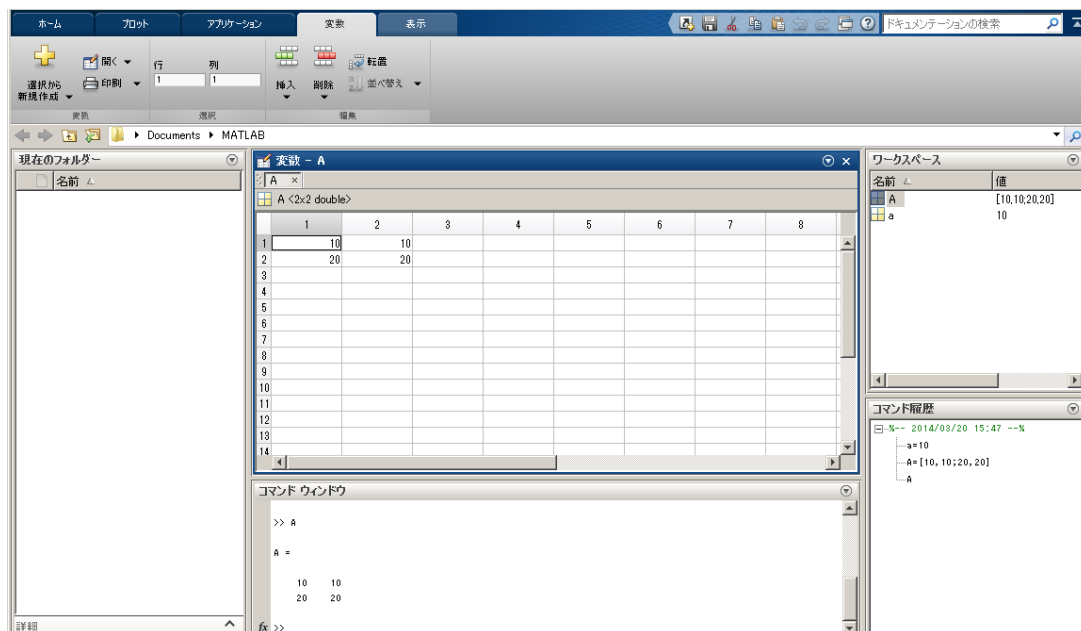


図 7.3: 変数内容の参照

節の始めでも説明したとおり、MATLAB は行列計算を簡単に行うことができます。たとえば行列の積は以下のようにして計算できます。

```
A*[1;2]←
```

この場合の実行結果は以下の通りです。

```
ans =

    30
    60
```

C や C++ で行列の積を計算する実装を考えた場合、for 文を用いて各要素についてひとつずつ計算する、といった方法がまず頭に浮かぶかもしれませんが、MATLAB では「*」を用いるだけで積をもとめることがで

きます。この実行結果で注意して欲しい点がひとつあります。この計算では計算結果を別の変数に代入していないため、実行結果のこれまで「a」や「A」にあたる部分には「ans」と表示されています。この「ans」は変数として残らず、ワークスペースにも残らないため、この後に ans の内容を確認することはできません。計算結果を後で確認したい場合には以下のように変数の代入式にする必要があります。

```
Answer=A*[1;2];↵
```

このほかにも逆行列や行列式の計算などが MATLAB では簡単に行うことができます。複雑な行列計算が行われるグラフ理論を学ぶ場合等に MATLAB を用いれば、学習の手助けになるかもしれません。

7.6.3 関数の実装

MATLAB ではよく利用する計算の一連の流れを関数として保存し、コマンドウィンドウから呼び出すことができます。関数を記述する場合には、まずコマンドウィンドウへ以下のように記述し、MATLAB が扱うことのできる.m ファイルを作成します。

```
edit filename.m↵
```

これを実行すると新たにエディタ画面が起動します。関数を記述する際には先頭に関数名及び引数と戻り値を示す記述と、末尾に関数の終わりを示す記述が必要となります。例えば引数の値を2倍にして返す関数ならば以下のように記述します。

filename.m

```
function[value]=twice(Parameter)
value=2*Parameter;
endfunction
```

1 行目の value が引数を表し、twice が関数名、Parameter が戻り値を示しています。引数や戻り値は複数指定することが可能で、その場合は変数名を「,」で区切ります。3 行目は関数の記述の終わりを示しています。

関数を記述し、ファイルを保存するとコマンドウィンドウで関数を呼び出すことができますようになります。例えば以下のように記述します。

```
x=twice(a)↵
```

実行結果は以下の通りです。

```
x =
```

```
20
```

また、これまで一行ずつ記述を行ってききましたが、複数の記述を一度に実行する方法もあります。まず、関数の記述の際に作成したような.m ファイルを作成し、エディタに目的である複数の記述を行います。ここでは「filename2.m」というファイルに、以下のこれまでにを行った行列の初期化と計算を記述します。

filename2.m

```
a=10;  
A=[10,10;20,20];  
A*[1;2]  
Answer=A*[1;2];
```

1 行目の記述はコメントです。MATLAB ではコメント行は先頭に「%」を記述します。記述後にファイルを保存し、コマンドウィンドウでファイル名「filename2.m」を記述し、実行すると記述された内容が一行ずつ実行されます。

7.6.4 その他のコマンドと関連リンク

ここでは最も初歩的な行列演算を取り上げて解説しましたが、MATLAB では行列演算だけでなくさまざまな計算に利用できる関数が備えられています。MATLAB のその他のコマンドについては MathWorks 社公式サイト [のドキュメンテーションセンター内の MATLAB 項目 \(https://jp.mathworks.com/help/matlab/functionlist.html\)](https://jp.mathworks.com/help/matlab/functionlist.html) から学ぶことができます。また、冒頭で紹介した MATLAB engine についても同項目内にページが存在します。(<https://jp.mathworks.com/help/matlab/calling-matlab-engine-from-cpp-programs.html>) 個人の計算機環境で MATLAB を用いて数値計算を行いたい場合、MATLAB と互換性を持った数値解析ソフトウェアの「*GNU Octave*」(<https://www.gnu.org/software/octave/>) が役に立ちます。GUI のデザインは MATLAB とは異なりますが、こちらはフリーウェアであり、C の標準ライブラリに含まれる関数の多くが実装されているため C に慣れた人はこちらのほうが使いやすいかもしれません。ただし、MATLAB で実装されている関数の全てが Octave にも実装されている訳ではないという点に注意してください。

第 8 章

VMware Fusion

—— ディスク容量に関する注意 ——

本章で説明する操作の一部は、大量のディスク容量を要求します。各個人に割り当てられたディスクの容量の割り当て（quota）が不足した場合、メールが受信できなくなったりログインできなくなったりする場合があります。このような問題を回避するために、10.11 節を参考に quota の上限の引き上げの申請を行ってください。

本章では仮想計算機の概要と、仮想化ソフトウェアのひとつである *VMware Fusion* の使い方を説明します。使い方の具体的な例として、macOS 上に仮想計算機を作成し、Linux ディストリビューションのひとつである CentOS をインストールします。

8.1 仮想計算機の概要

仮想計算機（Virtual Machine）とは、普段私たちが利用している計算機の上で、あたかも複数の計算機が動作しているかのような環境を作ることができる仕組みです。仮想計算機を利用することで、例えば計算機室の iMac の上で、図 8.1 のようにあたかも macOS のアプリケーションのひとつであるかのように Windows や Linux といった他の OS を起動して、利用できるようになります。

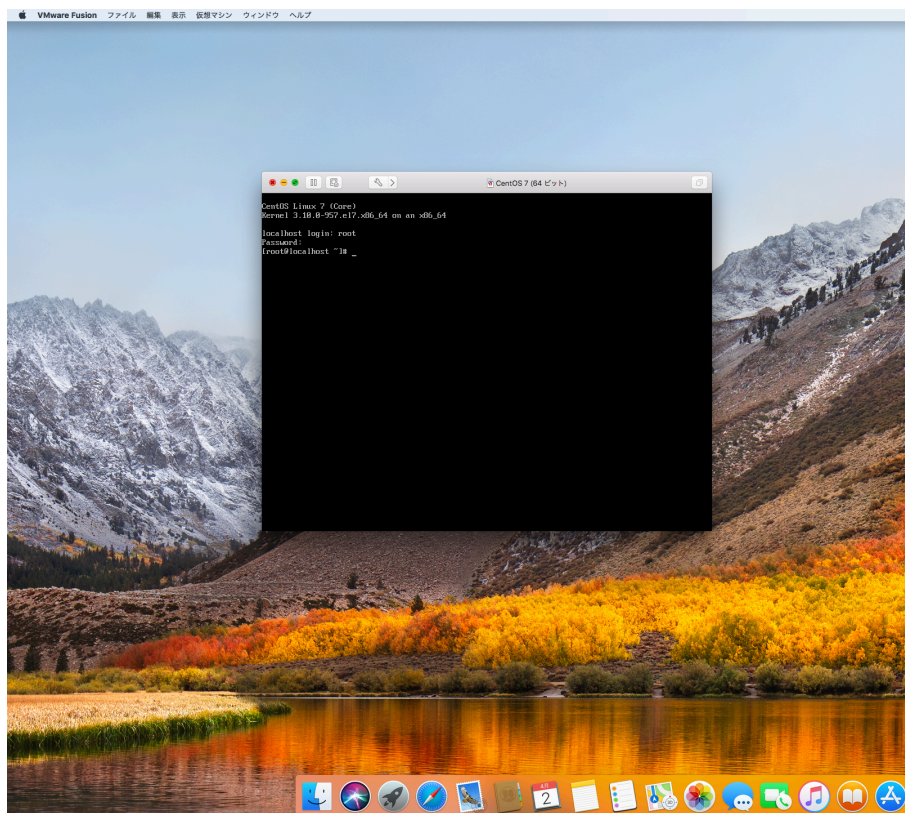


図 8.1: 仮想計算機の使用例

このとき、実計算機のことをホストマシン（Host Machine）、その上で動作している仮想計算機のことをゲストマシン（Guest Machine）といいます。同様に、ホストマシンの上で動作している OS のことをホスト OS（Host OS）、ゲストマシンの上で動作している OS のことをゲスト OS（Guest OS）といいます。

仮想計算機を用いることで、次のようなことができるようになります。

- macOS が動作している計算機の上で仮想計算機を作成し、そこに Windows や Linux 系 OS をインストールして利用できます。
- 仮想計算機には、自由に OS をインストールして利用できます。Windows や Linux の他にも、FreeBSD や OpenSolaris, Plan 9 のような OS を、ライセンスの範囲内で自由に動作させることができます。
- ゲスト OS の管理者権限を持つことができます。ゲスト OS 上でなら、OS の設定を変更したり、管理者でないと実行できない作業をしたり、カーネルのデバッグなどの特殊な操作ができます。

8.2 仮想計算機に OS をインストールする手順

本節では *VMware Fusion* を用いて仮想計算機に OS をインストールする手順を説明します。例として Linux のディストリビューションのひとつである CentOS をインストールします。仮想計算機を作る手順は次からなります。

1. インストールする OS のディスクイメージを用意する

2. ゲストマシンを作る
3. OS をインストールする

本節ではこれに従って説明します。

8.2.1 インストールする OS のディスクイメージを用意する

ゲストマシン上にインストールするための OS のインストールディスクか、そのディスクイメージを用意する必要があります。情報科学類の学生ならば、学類から Windows のディスクイメージとライセンスキーを取得できます。あるいは、Linux などの自由に使える OS のディスクイメージをインターネット経由でダウンロードしてくることもできます。

ディスクイメージを手に入れたら、自分のホームディレクトリに保存しておきましょう。例えば本節で説明に用いる CentOS のディスクイメージは、<http://www.centos.org/download/> から入手できます。

8.2.2 ゲストマシンを作る

初めて仮想計算機を使うときは、まずゲストマシンを作成します。VMware Fusion を図 8.2 に示すアイコンをクリックして起動すると、図 8.3 のような画面が表示されます。



図 8.2: VMware Fusion のアイコン



図 8.3: VMware Fusion の初期画面

ゲストマシンを新しく作成するときは、図 8.3 から「ディスクまたはイメージからインストール」を選択します。ディスクイメージを選択する画面（図 8.4）に移るので、インストールしたい OS のディスクイメージを指定し、「続ける」をクリックします。



図 8.4: CentOS のインストール画面

次に、図 8.5 のように、仮想マシンのブートで使用するファームウェア設定を尋ねられます。入力が終わったら「続ける」をクリックします。



図 8.5: ゲスト OS のアカウントとパスワードの設定

最後に 8.6 へ移ります。ここで「終了」をクリックすることで、図 8.7 のように仮想マシンを保存する場所を指定して終了します。また「設定のカスタマイズ」をクリックすることで、図 8.8 に示す画面へ移り、メモリのサイズやディスク容量などの設定ができます。



図 8.6: 仮想マシン作成の完了



図 8.7: 仮想マシンの保存先の指定



図 8.8: 仮想マシンの詳細な設定

基本的には、このアシスタントの指示通りに設定していけば、問題ありません。設定項目は OS 毎に違いますので、ここでは特に気をつけるべき点を示します。慣れないうちは、これに従ってください。

- 基本的には、初期設定のままで問題ありません。
- CPU の個数は特別な理由が無ければ 1 とします。
- メモリの割り当ては 1024MB 程度を上限に、基本的にはアシスタントの提案に従いましょう。
- 仮想ハードディスクは、欲張らずに必要な最低限の量を割り当てましょう。学生は、使用できる実ディスク領域に制限が設けられています。これを越えて割り当てることはできません^{*1}。
- ネットワークの種類は、**インターネット共有**を選択します。

以上で仮想マシンの作成が完了します。この仮想マシンを起動すると、ゲスト OS のインストールが開始されます。

OS をインストールする

作成した仮想マシンを起動すると OS のインストールが開始します。VMware Fusion を起動した直後に現れる図 8.9 の画面から、作成した仮想マシンをクリックすることで起動できます。なお OS のインストールは自動的に行われます。

^{*1} この制限を quota といいます。特に必要な時は、担当の技術職員の方に申請すると制限を変更してもらうことができます。

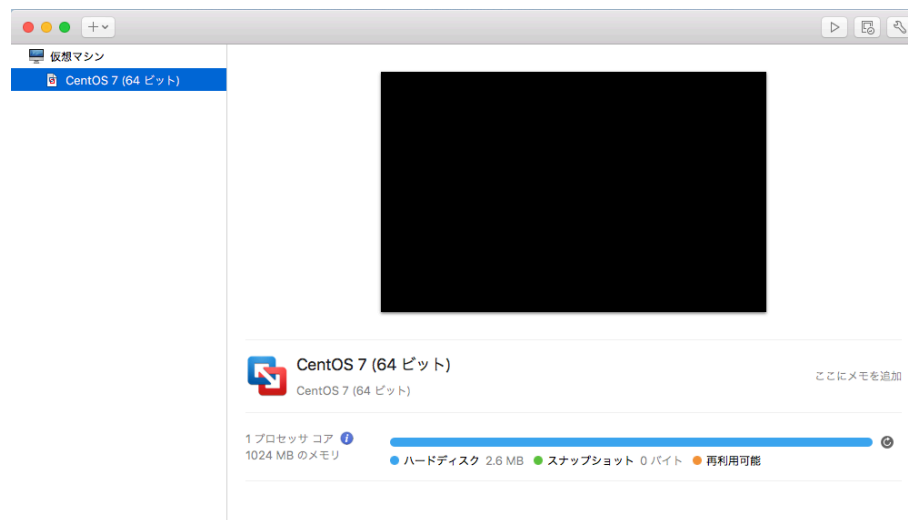


図 8.9: 仮想マシンの起動

以上で仮想マシンが作成され、ゲスト OS がインストールされました。

8.3 ゲスト OS を利用する方法

本節ではインストールしたゲスト OS を利用する方法を説明します。起動と終了、フルスクリーンモード、スナップショットなど、VMware Fusion を使う上で必要な機能を取り上げます。

8.3.1 ゲストマシンの起動と終了

作成したゲストマシンを利用するには、起動と終了の仕方を知る必要があります。仮想計算機はソフトウェア的に再現された計算機ですから、使い始める前には仮想的に電源を入れ、使い終わったら仮想的に電源を切る必要があります。

VMware Fusion を起動すると図 8.10（再掲）のような画面が表示されるので、左の一覧から起動したい仮想マシンを選択すれば、ゲストマシンの電源が入って、ゲスト OS が起動します。

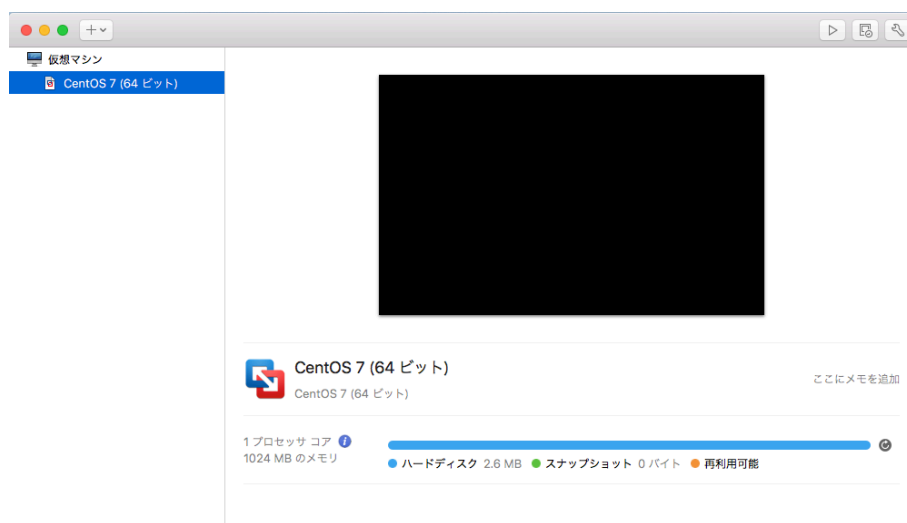


図 8.10: 仮想マシンの起動

一方ゲストマシンの電源を切る場合は、普段実計算機を利用しているときと同じように、Windows ならスタートメニューから終了を選べば、Linux なら `shutdown -h now` と打てば、ゲスト OS が終了し、ゲストマシンの電源も自動的に切れます。もしゲスト OS がフリーズするなどして通常の手段で終了できなくなったときは、アプリケーションメニューから **仮想マシン** > **シャットダウン** をクリックすれば、強制的にゲストマシンの電源を切ることができます*2。

8.3.2 フルスクリーンモード

画面表示をフルスクリーンモードにすることで、ゲスト OS に画面を占有させることができます。フルスクリーンモードにするには、メニューから **表示** > **フルスクリーン** を選択します。フルスクリーンモードを解除するには、**ctrl** + **⌘** を同時に押します。

8.3.3 スナップショット

仮想計算機の便利な機能のひとつとして、ゲストマシンのある瞬間の状態を保存して、あとでその時の状態を復元できるという仕組みがあります。この時、保存した状態のことをスナップショット (Snapshot) と呼びます。

定期的なスナップショットをとっておけば、たとえば操作ミスや悪意のあるソフトウェアを実行するなどの原因でゲスト OS が起動しなくなってしまったような場合でも、前回スナップショットをとった時の状態まで簡単に巻き戻すことができます。

スナップショットをとるには、メニューから **仮想マシン** > **スナップショット** > **スナップショットをとる** を選択します。その後表示されるウインドウにスナップショットの名前と説明を記入して **OK** を押すと、スナップショットの作成が始まります。この作業にはすこし時間がかかります。

*2 シャットダウンボタンを押すと、実マシンで電源ボタンを押した時と同じ動作をします。仮想ハードディスク上のデータが破損する可能性があるため、なるべく各 OS の正規の終了手段をとるようにしましょう。

過去にとったスナップショットの状態へ仮想マシンを復元するには、メニューから**仮想マシン** > **スナップショット** > **スナップショットに復元**を選択すると、スナップショットの復元が始まります。復元が完了すると、ゲストマシンはスナップショットをとった時と同じ状態に戻ります。このとき、スナップショットをとってからあとに加えた変更や、保存したファイルは消えてしまうので注意しましょう。

8.4 VMware Fusion のより高度な機能

本節では VMware Fusion を使って仮想マシンを扱う上で便利な機能を紹介します。ホスト OS とゲスト OS の間でファイルをやりとりするための共有フォルダ機能やホスト OS の CD/DVD ドライブにアクセスする機能、ゲスト OS をデバッグする方法について取り上げます。

8.4.1 共有フォルダを用いてファイルをやりとりする

ホスト OS にあるファイルをゲストマシンにコピーしたり、あるいはその逆を行いたい場合、共有フォルダを使います。

メニューから**仮想マシン** > **共有** > **共有 設定...**をクリックし、**共有フォルダ**の項目を開きます。図 8.11 の画面から、**+**を押して、ホスト OS とゲスト OS の間で共有したいフォルダを選択することで、設定が完了します。なお共有フォルダに設定したファイルは、ゲスト OS 内ではデスクトップに表示されます。共有フォルダの利用には VMware Tools が必要です。**仮想マシン** > **VMware Tools のインストール**をクリックしてインストーラを仮想マシンに接続し、インストールを行ってください。



図 8.11: 共有フォルダの設定

8.4.2 ゲスト OS からホストマシンの CD/DVD ドライブを利用する

ゲストマシンからホストマシンに入った CD-ROM などメディアにアクセスできます。メニューから**仮想マシン** > **CD/DVD**をクリックし、画面に従って設定すれば、利用できるようになります。

8.4.3 VMware を利用したゲスト OS のデバッグ

VMware を利用すると、ゲストとして動作している OS そのもの（カーネル）にデバッガを接続してデバッグができるようになります。ここでは Linux マシン上でゲストの Linux カーネルをデバッグする場合を例に、その方法を簡単に紹介します。

vmlinux の準備

まず最初に、デバッグシンボルが含まれた Linux カーネルの実行ファイルを準備する必要があります。ディストリビューションに付属のカーネルを利用している場合は、適宜用意しておいてください。自分でカーネルをコンパイルする場合は、コンパイルオプションとして、

```
Kernel hacking -> Compile the kernel with debug info
```

の項目を有効にします。コンパイルしたあと、カーネルのソースディレクトリ直下にある `vmlinux` が目的のファイルです。`vmlinux` ができ上がったら、ホストマシン上の適当なディレクトリにコピーしておきます。

VMware の設定

つぎに、VMware の設定を変更します。まずデバッグを行いたいゲストマシンが保存されたディレクトリを開きます。そして”ゲストマシン名.vmx”というファイルを開いて、次の一行を追記します。

```
debugStub.listen.guest32 = "TRUE"
```

これで、次にゲストマシンを起動した時から、デバッガからの接続を受け入れるようになります。起動後に `netstat` などで、8832 番のポートが開かれていることを確認します。

デバッガでの接続

それでは実際にゲストマシンへのアタッチを行います。

gdb でのアタッチ

```
$ gdb↵  
(gdb) file /path/to/your/vmlinux↵  
(gdb) target remote localhost:8832↵
```

成功するとゲスト OS の実行が停止し、アタッチが完了します。適当にブレークポイントなどを仕掛けて `continue` してみましょう。デバッグの仕方は、通常のプログラムに対して行うのとはほぼ同じです。

8.5 困ったときは

8.5.1 フルスクリーンから戻れない

`ctrl` + `⌘` を同時に押します。

8.5.2 ゲスト OS に Ctrl-Alt-Del の入力を求められた

メニューから **仮想マシン** > **Ctrl-Alt-Del の送信** をクリックします。このほかにも、ホスト OS が横取りしてしまつてゲスト上で入力できないようなキーの組み合わせをいくつか選択できます。

第 9 章

ネットワーク

9.1 学類計算機資源の利用

COINS では実習に使う iMac を，ネットワークを経由して自宅等の外部の計算機からアクセスできるように設定しています．本章では，この接続方法について説明します．

9.1.1 利用可能な計算機

COINS では学生がリモートからログインして利用可能なマシンとして以下のようなものが用意されています．

表 9.1: 利用可能な計算機一覧

マシン名	種類	場所
aloe1 ～ aloe96	iMac	3C113
crocus1 ～ crocus40	iMac	3C205
viola1 ～ viola14	ネットワークサーバ	https サーバ (viola5), メール送信サーバ (viola6) など
www	WWW サーバー	情報科学類公式 HP など

これらのマシンは COINS ネットワークの外部からでも，たとえば自宅からでも利用することができます．外部からアクセスする際には，たとえば aloe15 というマシン名の iMac マシン名は “aloe15.coins.tsukuba.ac.jp” というホスト名^{*1}を用いる必要があります．

9.1.2 学外からのリモートログインに関する注意

COINS システムは学外からは公開鍵認証でのみアクセス出来ます．そのため，接続を試みる前に自身の公開鍵をホームディレクトリ下の .ssh ディレクトリ内 authorized_keys ファイルに登録しておく必要があります．登録の際は公開鍵を USB メモリに入れて持ち運ぶか，メールで自分宛てに送ると良いでしょう．初めて接続するホストの際には必ずホストのフィンガープリントを COINS システムの主な SSH サーバの公開鍵の

^{*1} 完全修飾ドメイン名 (FQDN, Full Qualified Domain Name) とも言います．

ハッシュ値^{*2}と同じであるかチェックして、そのホストが真正な物であるかの確認を行ってください。このような確認を行わなかった場合、Man-in-the-Middle 攻撃等を受けている事に気づかず接続してしまい攻撃者が自由にアカウントにアクセスできてしまう可能性があります。

詳しい手順などの情報は COINS の Wiki を参照してください。 <https://www.coins.tsukuba.ac.jp/ce/pukiwiki.php?SSH%A5%A2%A5%AF%A5%BB%A5%B9>

9.1.3 学類計算機へのリモートログイン

iMac マシンを使う際にはログインするということが必要でした。このときは直接マシンの前に座って、そのマシンのキーボードからユーザ名やパスワードを入力しました。

一方で、目の前にないコンピュータにネットワーク経由でログインするということが可能です。これをリモートログインといいます。

リモートログインにおいても通常のログイン方法と同様に、ユーザ名とパスワード等で自分がユーザで間違いない事を接続先のコンピュータに伝える必要があります。しかしこれらの機密情報が、ネットワーク上にそのままの形^{*3}でながれてしまうと、盗聴された時にパスワード等の認証情報が盗まれてしまう可能性があります。

そのためこれを防ぎ、リモートログインを行う際のセキュリティを確保するために *SSH (Secure SHell)* というプロトコルが使用されています。これを用いることで暗号化された通信路が利用でき、その上で安全に機密情報をやりとりできます。

iMac はこの SSH を用いて COINS ネットワークの内部や外部からリモートログインして操作することができます。ここではそれぞれ Windows, macOS/Linux から COINS のマシンにログインする手順を解説します。

Windows からの SSH 接続

Windows には標準で SSH クライアント仮想端末ソフトウェアが入っていないので、自宅の Windows マシンから SSH 接続する場合にはこれをインストールする必要があります。

ここでは COINS の Windows マシンに標準でインストールされている *Tera Term*^{*4} というソフトウェアを例にして Windows での SSH 接続の方法について説明します。このほかにもフリーウェアとして、PuTTY^{*5}、Poderosa^{*6}などの優れたソフトウェアがありますので、ぜひ試してみてください。

Tera Term の起動

Tera Term を起動するには、「チャームバー」の「検索」から「Tera Term」を検索して実行してください。すると図 9.1 のようなウィンドウが表示されます。

この「Tera Term: 新しい接続」ウィンドウで SSH 接続のための設定を行います。これが表示されていない場合は、**ファイル** > **新しい接続 (N)** を選択してください。

^{*2} <https://www.coins.tsukuba.ac.jp/ce/pukiwiki.php?coins%A5%B7%A5%B9%A5%C6%A5%E0%A4%CE%BC%E7%A4%CASH%A5%B5%A1%BC%A5%D0%A4%CE%B8%F8%B3%AB%B8%B0%A4%CE%A5%CF%A5%C3%A5%B7%A5%E5%C3%CD>

^{*3} クリアテキストといいます。

^{*4} <http://sourceforge.jp/projects/ttssh2/>

^{*5} PuTTY: A Free Telnet/SSH Client, <http://www.chiark.greenend.org.uk/~sgtatham/putty/>

^{*6} <http://ja.poderosa.org/>

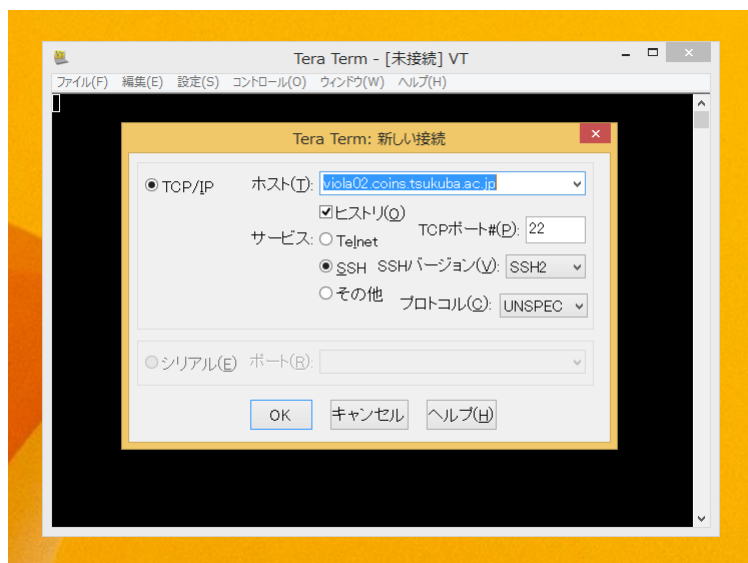


図 9.1: Tera Term のウィンドウ

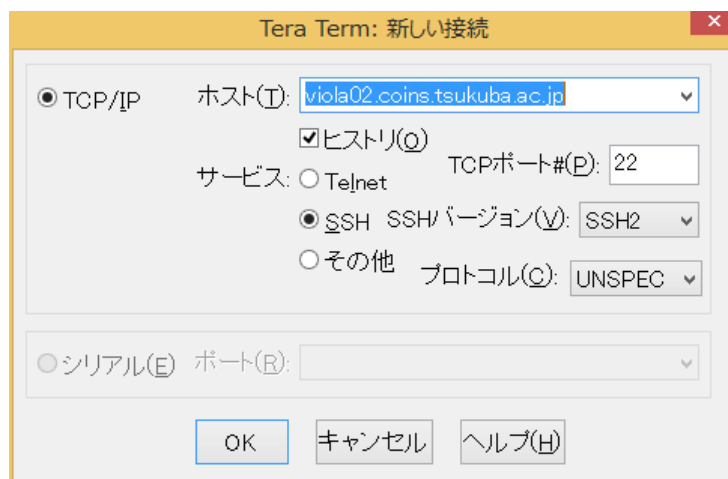


図 9.2: 新規接続ウィンドウ

COINS のマシンに接続するために、[ホスト] には接続先の計算機のホスト名または IP アドレスを入力します。また [サービス] として「SSH」、[TCP ポート] として「22」、[SSH バージョン] として「SSH2」をそれぞれ選択・入力してください（図 9.2）。

[OK] ボタンを押すと、初めて接続するホストであるなどの場合、図 9.3 のようなウィンドウが表示されます。ここでは「本当にこのホストに接続してよいのか」と「次からこの警告を表示しないように、このホストの情報を記録するか」をユーザに尋ねています。

前者は、ここで接続しようとしている“viola01.coins.tsukuba.ac.jp”という名前のホストが本当に接続したい COINS のマシンかどうか、ウィンドウ中央の「サーバ側のホスト鍵指紋」を見て確認することが求められ

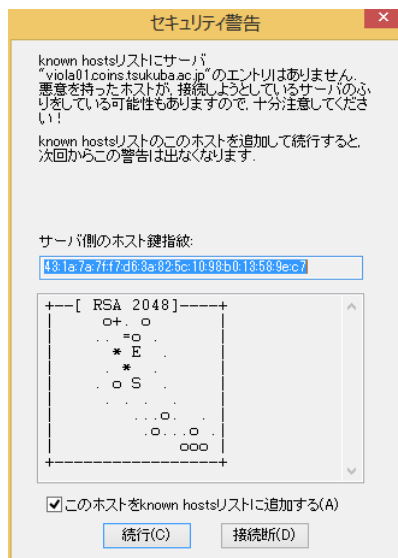


図 9.3: セキュリティ警告

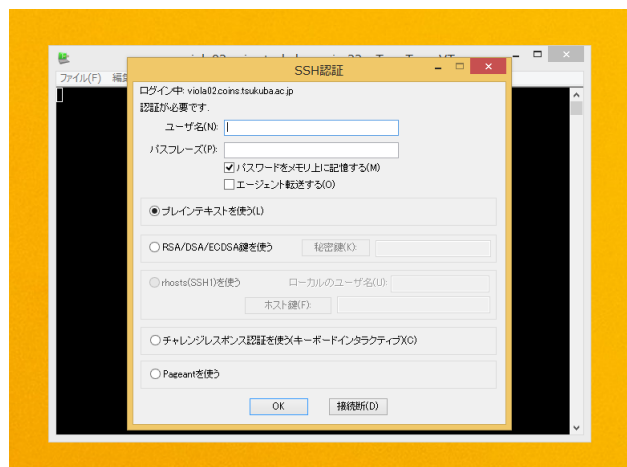


図 9.4: 認証情報の入力

ています。中間者攻撃^{*7}を受けた場合、「接続したいホスト名」と「実際に接続していくホスト」が異なる場合があります。この時、誤ってこの偽のホストにパスワードなどを送信してしまうと、不正にアカウント情報を利用されるおそれがあります。これを防ぐためにホスト鍵指紋（フィンガープリント）を見て、本当にそのホストが指定したホストであるかを確認します。aloe グループなどの iMac のフィンガープリントは、そのホストに対して直接以下のコマンドを実行することで簡単に調べることができます^{*8}。一度確認してみるとよいでしょう。

実行例

```
crocus10:~ s99991399$ ssh-keygen -l -f /etc/ssh_host_rsa_key.pub
2048 8a:cb:69:57:b0:fa:fc:70:e7:55:2e:b4:06:3d:f9:63 (RSA)
```

後者は、このフィンガープリントを本物であるとして記録するかどうかを判断します。記録する場合、このホスト名とフィンガープリントの対応が known_hosts リストに保存され、この組み合わせについてユーザに問い合わせを今後行いません。これにより、次の接続の時からこの「セキュリティ警告」のウィンドウが表示されなくなります^{*9}。逆に先ほどあげた「偽のホスト」についてこれを行ってしまうと、次から確認することができなくなるため、間違いに気づきにくいという欠点もあります。[このホストを known_hosts リストに追加する] チェックボックスのチェックは慎重に行ってください。それぞれ確認/設定を終えたら、**続行** ボタンをクリックしてください。

この次の画面（図 9.4）で、実際に接続に必要な認証情報の入力を行います。ここでは [ユーザ名] に COINS のユーザ名、[パズフレーズ] にパスワードを入力します。パスワードによる認証ではなく、公開鍵認証を利用

^{*7} 暗号通信を盗聴したり介入したりする手法の一つ

^{*8} 2014 年 3 月時点での実行例につき、変更される可能性があります。

^{*9} 例外として、すでに保存している組み合わせに反する場合は再度表示されます。これは、たとえばマシンが変わった場合や中間者攻撃が行われた場合に起こります。

している場合は [RSA/DSA 鍵を使う] ラジオボタンをクリックして [秘密鍵] を指定してください。入力し終えたら [OK] ボタンをクリックすることで、この情報を用いて接続を行います。

以上の手続きにより COINS のマシンにリモートログインすることができます。一度ログインすると、実際にそのマシンの前に座って端末を立ち上げて使用するのと同様に作業することができます。また COINS では、ユーザのファイルや設定などの情報は全てのコンピュータで同期されているため、基本的にどの iMac にログインしても同様の作業ができます。

Tera Term でリモートログインできるようになったら、環境を好みに合わせて変更すると良いでしょう。端末の背景色や文字色などの変更から、文字コードや改行コードなど（図 9.5）の変更までさまざまにカスタマイズすることができます。



図 9.5: Tera Term の設定

macOS/Linux からの SSH 接続

Windows とは異なり macOS と Linux には標準で SSH クライアントソフトウェアが入っています。これはターミナルからコマンドとして使うことができます。本小節ではこのコマンドを使ったログイン方法について解説します。

それぞれ macOS では iTerm や X11 など、Linux ではターミナルを開き、以下のように入力します。ここではユーザ名として "s9912345" を用いていますが、適宜自分のユーザ名に読み替えてください。詳細は 2.14.1 を参照してください。

実行例

```
[coins-tarou@mypc ~]$ ssh crocus15.coins.tsukuba.ac.jp -l s9912345 -o FingerprintHash=md5
The authenticity of host 'crocus15.coins.tsukuba.ac.jp (130.158.86.213)' can't be
established.
RSA key fingerprint is MD5:ea:40:99:be:63:f3:c7:16:b5:3f:fa:3a:19:05:b6:14.
Are you sure you want to continue connecting (yes/no)?
password:パスワードをここに入力してください
Last login: Thu Feb 16 04:15:00 2014 from (接続元のアドレス)
[s9912345@crocus15 ~]$
```

ユーザ名とパスワードは COINS で使っているものを入力します。ここでは-l オプションをつけることでユーザ名を指定してログインしています。COINS 内部のマシンから同じく COINS 内部のマシンにリモートログインする場合は不要な場合があります。

9.1.4 大規模計算について

学類の計算機資源を用いて大規模計算を行う場合は、「情報科学類教育用計算機を利用した大規模計算に関する規定」*¹⁰を確認した上で規定に従った利用を心がけてください。

9.2 無線 LAN サービス

COINS では、計算機室内に無線 LAN アクセスポイントを設置しています。所定の設定を行うことで、情報科学類の学生は持ち込んだノートパソコンを利用して、インターネットや COINS ネットワークにアクセスできます。本節では利用方法を説明します。

9.2.1 無線 LAN の利用可能な場所

情報科学類の COINS ネットワークにアクセスするための無線 LAN は、主に第3エリア A,C 棟など情報科学類の計算機室の周辺で使用可能です。これを用いて第3エリア内において無線 LAN 経由でインターネットにアクセスしたり、COINS 内の計算機にログインしたりすることができます。

9.2.2 必要な機器

無線 LAN を利用するためには、IEEE 802.11a, IEEE 802.11b, IEEE 802.11g, IEEE 802.11n といった通信方式に対応した一般的な無線 LAN アダプタと、それに対応したノートパソコン、オペレーティングシステムが必要です。また、接続にあたって IEEE 802.1X を用いて利用者認証を行っているため、これをサポートしている OS を用いる必要があります。Windows (WindowsXP 以降) や macOS は標準でこれに対応しています。

9.2.3 無線 LAN 利用上の注意

無線 LAN は電波を利用して通信しますので、電波の内容が傍受される可能性があります。そのため、無線 LAN 上の通信ではできるだけ機密情報を流さないように気をつけるとともに、暗号化することをおすすめします。たとえば SSH のポート転送を利用したり、VPN プロトコルを利用するなどといった方法が考えられます。

9.2.4 無線 LAN の利用

無線 LAN の具体的な利用・設定方法については COINS Web ページにある「情報科学類コンピューティング環境*¹¹」内の「その他サービス」より、「無線 LAN 接続」の項目を参照してください。

*¹⁰ 情報科学類コンピューティング環境 (<http://www.coins.tsukuba.ac.jp/ce/>) より確認できます。

*¹¹ <http://www.coins.tsukuba.ac.jp/ce/>

9.2.5 macOS 環境からの無線 LAN の利用方法

上記ホームページ上には Windows 環境での無線 LAN の設定方法しか載っていないため、本節では macOS 環境における無線 LAN の利用方法についての説明を行います。

まず、環境設定の画面(図 9.6)から無線 LAN をセットアップします。環境設定の画面から「ネットワーク」を選択すると図 9.7 が表示されます。

準備ができたなら設定を行います。もし、この画面とは異なり左の欄に Wi-Fi 表示されていなければ、左の欄の下部にある + をクリックすると Wi-Fi を追加することができます。



図 9.6: 環境設定の画面



図 9.7: ネットワークの設定画面

まず、左の欄の Wi-Fi を選択します。右下の「詳細」をクリックします。すると図 9.8 のような画面が表示されます。上部のタブから Wi-Fi を選びます。次に使ったことのあるネットワークのすぐ下にある + をクリックし、新たにネットワークを追加する画面を表示します(図 9.9)。ここでは AP-C113AN に繋げる事とするため、この画面で「ネットワーク名」を”AP-C113AN”とし、「セキュリティ」で”WPA2 エンタープライズ”を設定します(図 9.9) 次にモードを自動のままにし、COINS の認証を使うため COINS のユーザ名とパスワードを入力します(図 9.10)

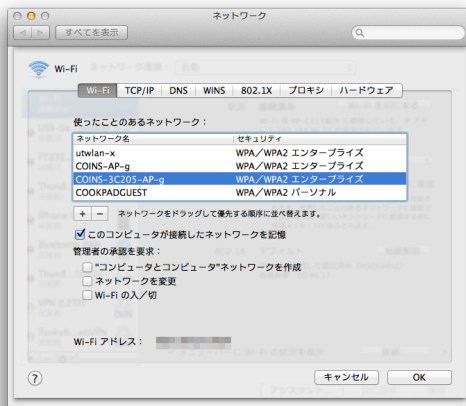


図 9.8: ネットワークの追加

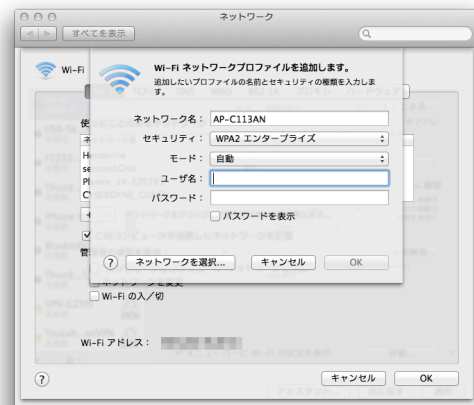


図 9.9: セキュリティを WPA2 エンタープライズに設定

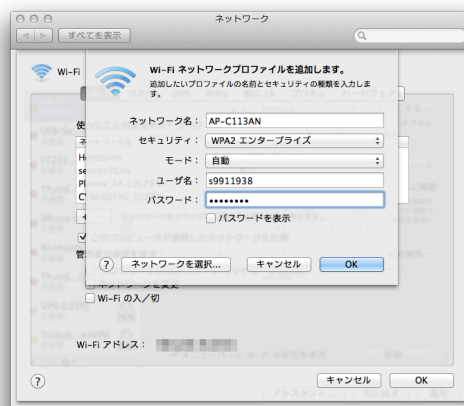


図 9.10: ユーザ名、パスワードの記入

9.3 VPN サービスの利用

COINS では、学外から COINS のネットワークに接続するために VPN サービスを提供しています。VPN とは Virtual Private Network（仮想私設網）の略で、インターネットなどの公衆網に暗号化されたトンネルセッションを張り、仮想的な専用線としてネットワークを利用する方式のことです。

COINS に VPN 接続すると、COINS の LAN 内にコンピュータを直接接続している状態とほとんど同じようにネットワークを利用することができ、学内からしかアクセスできない Web ページにアクセスできるようになります。情報科学類の学生は、学類のアカウントを VPN サービスに利用できることで計算機の環境が整えばすぐにこのサービスを利用できます。

VPN プロトコル（通信の方式）には PPTP や L2TP/IPsec, SoftEther Protocol, SOCKS などがありますが、COINS では L2TP/IPsec による VPN サービスを提供しています。

COINS に VPN 接続すると、COINS のネットワーク内部に直接いるかのようにネットワークを利用することができます。CiNii (<https://ci.nii.ac.jp>) で論文を読むなど、学内からアクセスするとより多くの事ができる Web サイトに自宅から VPN 経由でアクセスできるようになります。

9.3.1 Windows 環境における VPN 接続のための設定

ここでは Windows 環境での VPN アクセスの方法を Windows 10 を例に解説します。

表 9.2: VPN の設定項目

VPN プロバイダー	Windows（ビルトイン）
サーバ名	vpn.coins.tsukuba.ac.jp
接続名	任意の名称
サインイン情報の種類	ユーザー名とパスワード
ユーザ名	COINS アカウントのユーザ名
パスワード	COINS アカウントのパスワード
VPN の種類	事前共有キーを使った L2TP/IPsec
事前共有キー	https://www.coins.tsukuba.ac.jp/ce/ の「VPN 接続」を参照

Windows 10 で VPN 接続をするためにはまず接続先を作成する必要があります。スタートメニューから「設定（歯車のアイコン）」をクリックし、出てきたメニューの中から「ネットワークとインターネット」をクリックします（図 9.11）。続いて出てきた一覧の中から「VPN」をクリックします（図 9.12）。

すると、VPN 接続に必要な情報の入力が必要と求められます（図 9.13）。ここには表 9.2 に示した設定情報を入力してください。また「接続名」にはわかりやすい名前（たとえば「COINS」など）を入力してください。全ての項目の入力が終わったら「作成」をクリックしてください。

VPN 接続

接続先の作成と同様の手順でスタートメニューの「設定（歯車アイコン）」「ネットワークとインターネット」の順にクリックします。そこから先ほど作成した COINS（図 9.14）を選択し、「接続」をクリックすると COINS VPN に接続することができます。うまく接続できない場合はサーバ名やパスワード、事前共有キーなどを見直してください。

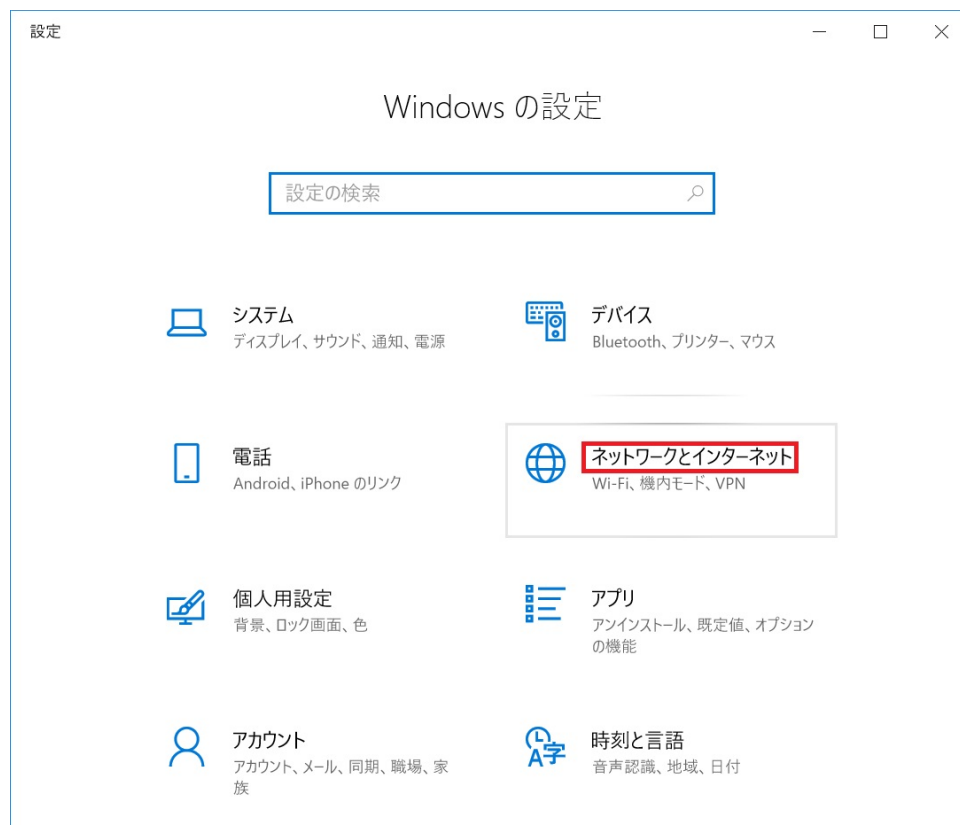


図 9.11: ネットワークと共有センター

どうしても接続できない場合は、`coins-admin@coins.tsukuba.ac.jp` へ問い合わせてください。

接続に成功した場合は、画面の右下に図 9.15 のような通知メッセージが表示されます。VPN に接続中であれば、あなたのコンピュータがまるで COINS のネットワークに直接接続されているのと同等の状態で作業することができます。また COINS VPN サーバとの間での通信は 128bit で暗号化されていますので、COINS のネットワークと自宅の間のやりとりを安全に行うことができます。

VPN 接続のデフォルトゲートウェイ化とその解除

Windows の VPN 接続では、デフォルトの設定では VPN 接続が確立した場合その接続に使用されている仮想的な WAN アダプタのネットワークがデフォルトゲートウェイとなるようにルーティングテーブルが自動的に書き換えられます。

このため COINS VPN に接続した状態でインターネットにアクセスすると、一度 COINS のサーバを経由して外につながるようになります。インターネット上の Web ページにはクライアントの IP アドレスを調べることができるページがありますので、試してみるとよいでしょう（診断くん <http://taruo.net/e/> など）。

一方で VPN 側がデフォルトゲートウェイになっている状態で Web サイトなどを見ようとすると、全てのパケットが筑波大学内のサーバを経由することになってしまいます。このため通信速度が若干低下してしまいます。

必要なときは VPN 側にアクセスし、それ以外の場合は通常の接続を用いる、といったことをしたい場合は

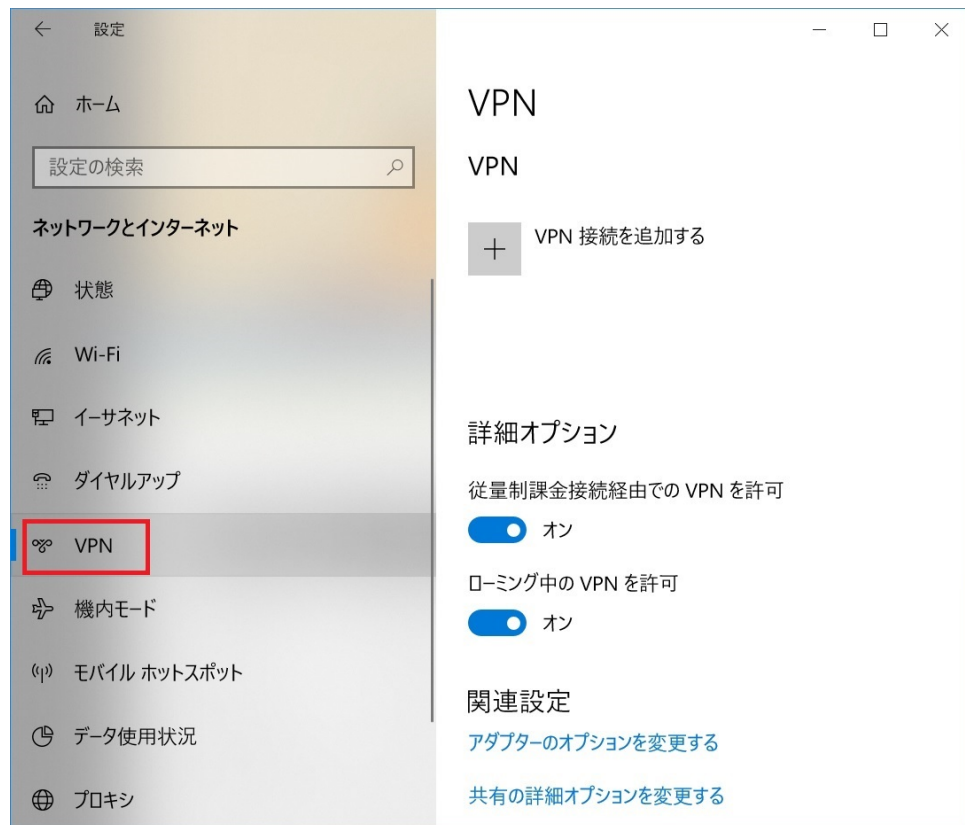


図 9.12: 接続またはネットワークのセットアップ

ルーティングテーブルが書き換わらないように設定する必要があります。ここではその手順を説明します。

先ほどの VPN 接続のリストの下にある「アダプターのオプションを変更する」を選択してください。その中の COINS の VPN 接続の項目を右クリックしてメニューの中から「プロパティ」をクリックしてください。「(接続先の名前)のプロパティ」というウィンドウが表示されたら、「ネットワーク」タブをクリックして、その中の「インターネットプロトコルバージョン 4 (TCP/IPv4)」をダブルクリックします。

新しく現れたウィンドウの「詳細設定」をクリックすると、図 9.16 のようなウィンドウが表示されるので、その中の「リモートネットワークでデフォルトゲートウェイを使う」のチェックボックスをオフにしましょう。これにより次回接続時からデフォルトゲートウェイが書き換わらないようになります。

← 設定

VPN 接続を追加する

VPN プロバイダー

Windows (ビルトイン) ▼

接続名

coins-vpn

サーバー名またはアドレス

vpn.coins.tsukuba.ac.jp

VPNの種類

事前共有キーを使った L2TP/IPsec ▼

事前共有キー

●●●●●●●●●●

サインイン情報の種類

ユーザー名とパスワード ▼

ユーザー名 (オプション)

s9876543

パスワード (オプション)

●●●●●●●●●● 🔍

☒ サインイン情報を保存する

保存 キャンセル

図 9.13: 接続先の設定



図 9.14: VPN 接続情報の入力

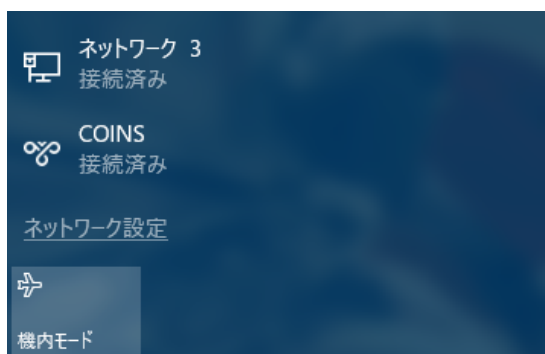


図 9.15: VPN 接続時の表示



図 9.16: デフォルトゲートウェイの設定解除

9.3.2 macOS における VPN 接続

本小節では VPN を macOS で利用する方法を紹介します。

接続方法

まず、ネットワークの設定画面を開きます (図 9.17)。ここに、VPN インターフェースを新しく作成・追加します。欄の下にある **+** を押すと図 9.18 が表示されます。この画面で「インターフェイス」から「VPN」、 「VPN タイプ」から「L2TP over IPsec」を選び、「サービス名」は任意の名前を付け、 **作成** を押します。これで VPN のインターフェイスが作成されました。

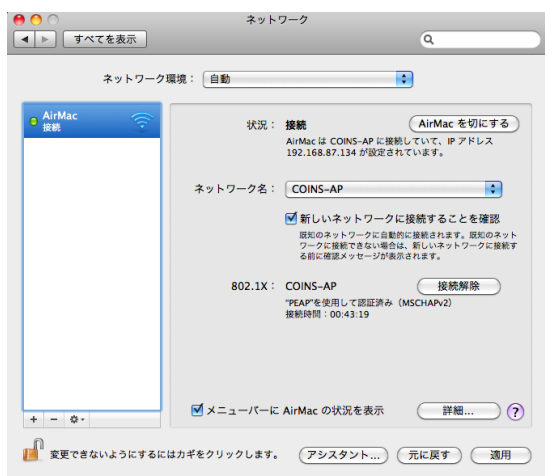


図 9.17: ネットワーク設定画面

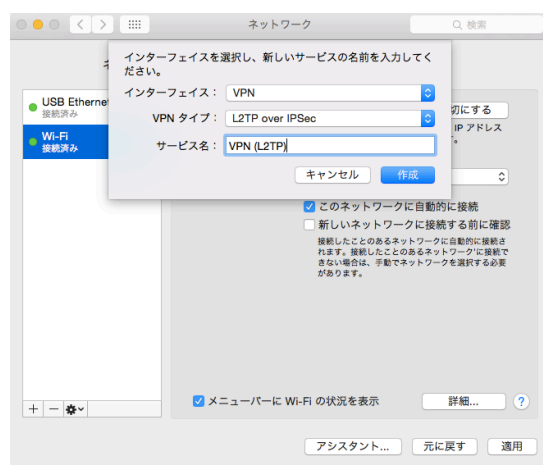


図 9.18: VPN インターフェイスの作成

次に VPN に新しい接続を設定します。先ほど作成したインターフェイスを選択し、右の欄の構成のプルダウンメニューから「構成を追加」を選択します。これを選んだ段階で図 9.19 のポップアップが表示されます。ここに作成する構成の名前を任意につけ (例では coins-vpn としました)、 **作成** を押します。これにより構成のメニューに作成した構成が出現します。この状態で、接続を行うために必要な情報を入力します (図 9.20)。ここで入力するのはサーバアドレス (vpn.coins.tsukuba.ac.jp) とアカウント名、事前共有キーです。

VPN 接続は認証を必要としますので、認証の設定を行います。アカウント名入力欄の下にある **認証設定** をクリックすると認証方法が問われます (図 9.21)。まず、パスワードにチェックを入れ COINS アカウントのパスワードを入力します。続いて、「共有シークレット」を選択し、事前共有キーを入力します。これで接続準備は完了です。

では、つないでみましょう。認証設定の下に **接続** を押して接続を開始してください。接続が完了すると図 9.22 のようになります。また、先ほどの場所にある **接続解除** をクリックすると接続を切ることができます。

うまく接続できない場合

前述の設定だけではうまく設定できない場合があります。データを分割する際の個々のサイズ (MTU; Maximum Transmission Unit) の設定が大きすぎると、うまく接続できないことがあるので、その場合は以下のようなコマンドで MTU の値を小さくしましょう。

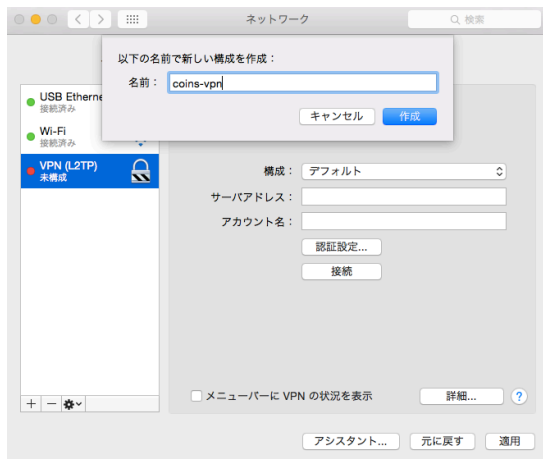


図 9.19: VPN の構成の作成

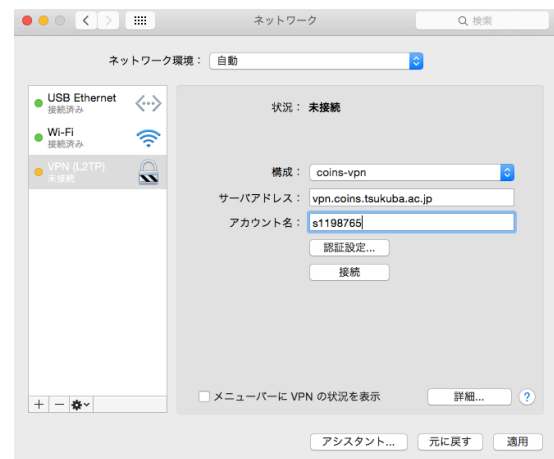


図 9.20: VPN 接続のための情報入力

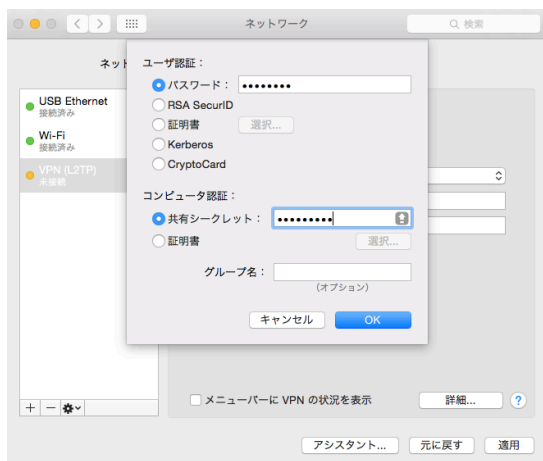


図 9.21: パスワード入力

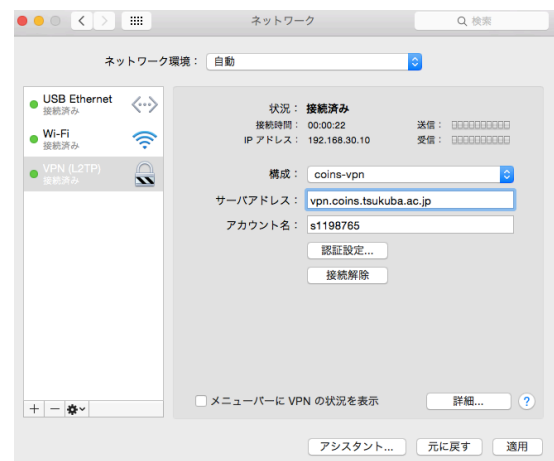


図 9.22: VPN 接続完了

```
$ sudo ifconfig ppp0 mtu 1300↵
```

第 10 章

困ったとき

本章では、情報科学類計算機システムを利用する際にありがちな、困ったときの対処方法を説明します。

10.1 バックスラッシュを入力する

バックスラッシュ（\）を入力するには、**Option**と $\yen$ を同時に押します。

また、入力の設定を変更することで、 $\yen$ を押すと $\backslash$ が入力できるようになります。この手引きの 1.5.5 を参照してください。

10.2 プロセスの強制終了

アプリケーションを使っていると途中で反応しなくなってしまうことがあります。そのまま待っていれば反応が戻ってくることもあるのですが、強制的にアプリケーションを終了しないとどうにもならないこともあります。その場合には、プロセスを強制終了します。

プロセスを強制終了するためには、3 つ方法があります。1 つ目は、 ⌘ + **alt** + **esc**を同時に押して、強制終了するアプリケーションを選択する方法です。キーを押すと図 10.1 のダイアログが出てくるので、強制終了したいアプリケーションを選択します。

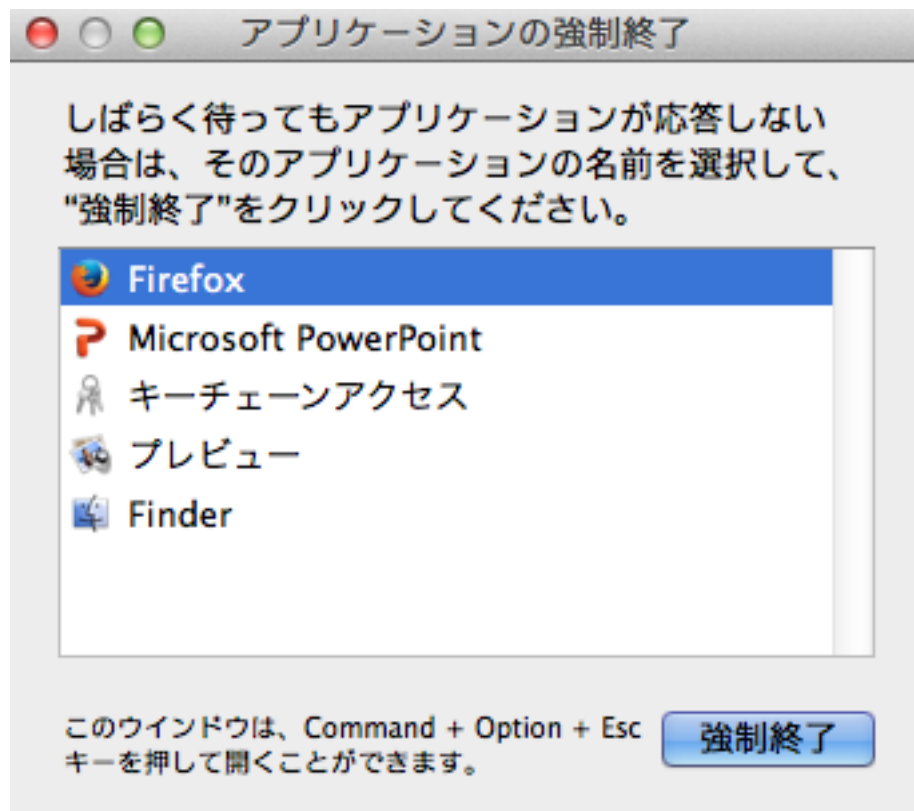


図 10.1: アプリケーションの強制終了画面

2 つ目は、図 10.2 のように macOS の Dock から強制終了したいアプリケーションを右クリックして、メニューから強制終了をクリックします。



図 10.2: Dock からの強制終了

3 つ目は、シェルからプロセスを終了させる方法です。この方法では、まず、ps コマンドを使ってプロセス番号を調べます。次に、プロセスを終了させるための kill コマンドを使って、指定したプロセス番号を kill します。

Firefox を kill する

```
$ ps aux | grep firefox↵
tebiki01  839    0.0  1.8  632144  75172  ??  S    5:36PM   0:15.93
    /Applications/Firefox.app/Contents/MacOS/firefox-bin -psn_0_249917
tebiki01  896    0.0  0.0  2425708    284 s002  R+   5:41PM   0:00.00
    grep firefox
$ kill 839↵
```

10.3 macOS 環境が固まらないようにするために

macOS 環境が固まる事例が増えています。原因としては以下が考えられます。

1. アプリケーションが作成するキャッシュが増えすぎている。
2. ディスク使用量の上限に達している。

1 については、`~/Library/Caches` の下にあるファイルを削除して下さい。2 に関しては、ファイルを削除してディスク使用量を減らして下さい。

なお、固まった場合はすぐには電源を切らずに、まずは技術職員に相談してください。

10.4 プリントアウトができない場合

プリントアウトしようとしてもできない場合があります。理由としては、以下のようなことが考えられます。

- プリンタの用紙がない
- プリンタのトナーがない
- プリンタの印刷上限に達している

プリンタの用紙やトナーがない場合には、coins-trouble@coins.tsukuba.ac.jp まで、プリンタ名を明記して連絡をしてください。また、プリンタの印刷枚数は上限が決まっており、それを超えるとプリントアウトができなくなります。印刷の方法や印刷枚数の確認については、1.8 節を参照してください。

10.5 Web ページ公開について

各ユーザのホームディレクトリにある Web ページのためのスペース（~/public.html）を公開する場合には、ホームディレクトリのパーミッションを少なくとも 711（rwx--x--x）にする必要があります。また、Web ページを公開する場合の URL は、[http://www.coins.tsukuba.ac.jp/~\(ユーザ名\)](http://www.coins.tsukuba.ac.jp/~(ユーザ名)) です。詳しい情報は、情報科学類コンピューティング環境（<http://www.coins.tsukuba.ac.jp/ce/>）を見てください。

10.6 COINS のシステムがダウンしている場合

COINS のシステムにログインしようとしても、どのマシンにもログインすることができない場合、COINS のシステムがダウンしている可能性があります。

COINS のシステムがダウンしている場合、平日の技術職員室（3E108）が空いている時間帯であればそちらへ直接問い合わせてください。ただし、夜間や休日など、技術職員室が空いていない時間帯であれば、coins-trouble@cs.tsukuba.ac.jp までメールで問い合わせてください。

10.7 Firefox・Thunderbird が起動できなくなった場合

Firefox や Thunderbird は、アプリケーションを強制終了した場合、次に起動したときに以下のエラーが出て起動できないことがあります。

すでに別の Firefox を開いています。同時に複数の Firefox を開くことはできません。

この場合、ロックファイルが残ったままになっていて起動できないので、以下のようにしてロックファイルを削除します。

```
$ rm ~/Library/Application\ Support/Firefox/Profiles/*/.parentlock<br>$ rm ~/Library/Thunderbird/Profiles/*/.parentlock<
```

詳しくは、Safari を使って、<http://www.coins.tsukuba.ac.jp/ce/> の「よくある質問」を参照してください。

10.8 COINS 宛のメールを他のメールアドレスに転送したい場合

COINS のメールを普段使っていないと、大事なメールが来たときに見落としてしまう可能性があります。そこで、普段自分が使っている携帯電話やパソコンのメールアドレスに転送するようにしましょう。

転送の設定をするには、ホームディレクトリのすぐ下に `.forward` ファイルを作成して、その中に転送先のメールアドレスを書きます。

例えば、自分が使っているメールアドレスが「s9912345@coins.tsukuba.ac.jp」、転送先のメールアドレスを「hoge@example.com」としたとき、転送先のメールアドレスに転送する場合は、コロンが改行で区切り、

```
.forward  
hoge@example.com
```

と書きます。しかし、この方法だと来たメールが全て書かれたメールアドレスに送られるだけで、元の COINS のメールボックスにはメールが残りません。ですので、ローカルアカウントにも転送するようにしましょう。

```
.forward  
\s9912345, hoge@example.com
```

と書くと元のメールボックスに保存したまま転送することができます。

このように `.forward` ファイルはアドレスを記述するだけの単純な物なのですが、注意しないとメールが届かないことやメールがループしてサーバーに負担がかかるという事が発生します。このような転送の設定をする時は、記述されたアドレスが全て疎通可能な物か、メールがループするような設定になってないか^{*1}に注意を払った上、必ずテストメールを送って正常に動いているかを必ず確認しましょう。

10.9 ディスククォータに関する警告メールが来てしまった

COINS システムではディスクをみんなで共有しているため、ある特定の人だけディスク容量を使ってしまわないようにディスククォータと呼ばれる使用量制限がかかっています。COINS システムでは、一人あたり 5GByte に設定しています。

COINS システムを長く使っていると、ディスククォータの制限に引っかかってしまい以下のような警告メールが来てしまうことがあります。もし、このようなメールが来た場合には 10.9.1 節を参考にして必要のないファイルを削除してください。削除せずに放置しておくと、ログインできなくなることもあるので注意してください。

また、主専攻実験などでディスク使用量が多くなってしまった場合には、担当の先生や技術職員に相談してください。

^{*1} 例えばアドレス A とアドレス B はお互い相手に転送する設定になっていて無限にメールが転送されること。

—— ディスククォータの警告メール（2018 年度以前のもの） ——

タイトル：[coins-quota] 注意：ディスク容量の制限に達しようとしています

s9912345 さん

こんにちは。coins-admin スタッフです。

あなたのホームディレクトリのディスク容量が制限に達しそうになっています。
必要ないファイルを削除して空き容量を確保してください。

現在、あなたのホームディレクトリは 3338 MByte 使用しています。
ディスク容量の制限は 3072 MByte となっており 108 % にあたります。

制限に達して一定以上書きこむと、それ以上書き込みができなくなります。
最悪の場合、ログインできなくなるのでご注意ください。

空き容量の確保の仕方については、こちらのページをご確認ください。

<http://www.coins.tsukuba.ac.jp/ce/quota.html>

もし、どうすれば良いかわからない場合は coins-admin@coins.tsukuba.ac.jp までご連絡ください。

このメールは s9912345 宛に自動送信されています。

お心当たりの無い方は、上記 coins-admin のアドレスまでご連絡いただければ幸いです。

10.9.1 必要のないファイルを調査、削除する

最初にどのフォルダがどれだけの容量を使用しているか調べます。2.8.2 の du コマンドを使用して、各フォルダのディスク使用量を確認しましょう。

d はその後に階層数を指定して、何階層まで表示するかを指定しています*2。ここでは、1 を指定しており、1 階層分のみを表示しています。試しに表示をしてみましょう。コマンドはホームディレクトリで実行します。

各フォルダの容量を調べる

```
cosmos02:~ s0711460$ du -h -d 1 .↵
1.0G    ./backupfiles2010
303M    ./Library
639M    ./Maildir
956K    ./Spotlight-V100
```

*2 一部の計算機は d オプションが使えません。代わりに --max-depth=オプションを使用します。

```
10.0K  ./ssh
718M   ./WinFiles
4.0K   ./Desktop
4.0K   ./Trash
4.0K   ./Xcode
1.4G   ./Downloads
248M   ./development
5.3M   ./Documents
197M   ./public_html
21M    ./netbeans
8.5K   ./netbeans-registration
5.3M   ./eclipse
81M    ./temp
600K   ./matlab
26K    ./lib
37K    ./ssh_watch
45K    ./subversion
5.0K   ./w3m
4.6G   ./
```

この場合だとホームディレクトリ全体で 4.6GByte 使用しています（一番最後の出力）。この中から必要なさそうなファイルを探しましょう。上記の場合だと Downloads ディレクトリが容量を使っているの、この中身を削除すればよさそうです。また、backupfiles2010 という昔のバックアップファイルが残っており、これももう古くなってしまったため必要がなさそうです。これら必要ないファイルは rm コマンドで削除しましょう。

10.10 ブラウザのキャッシュを削除する

ブラウザのキャッシュを削除する方法を説明します。Firefox や Google Chrome などのブラウザのキャッシュが容量をとっている場合があります。その場合、Library フォルダのサイズが大きくなってしまいます。

Firefox のキャッシュを削除する

まずメニューバーの **Firefox** > **環境設定** をクリックし、環境設定を開きます。次に **プライバシーとセキュリティ** パネルを選択し、**Cookie とサイトデータ** セクションの **データを消去...** をクリックすると図 10.3 のようなメニューが表示されます。ここで **ウェブコンテンツのキャッシュ** が選択されていることを確認して **消去** をクリックします。



図 10.3: Firefox のキャッシュの削除

Google Chrome のキャッシュを削除する

メニューバーから **Chrome** > **閲覧履歴を消去** をクリックします。すると図 10.4 のようなメニューが出てくるので、キャッシュが選択されていることを確認し、**閲覧履歴の消去** をクリックします。なお、消去するキャッシュの期間を“すべて”に設定しなければ、指定された期間に作られたキャッシュのみ削除されます。



図 10.4: Chrome のキャッシュの削除

Safari のキャッシュを削除する

まずキャッシュを削除する機能を有効にします。メニューバーから **Safari** > **環境設定** を選択し、図 10.5 のように“詳細”タブ下部の「メニューバーに“開発”メニューを表示」のチェックボックスにチェックします。するとメニューバーから **開発** > **キャッシュを空にする** をクリックできるようになるので、これをクリックすることで、キャッシュを空にできます。

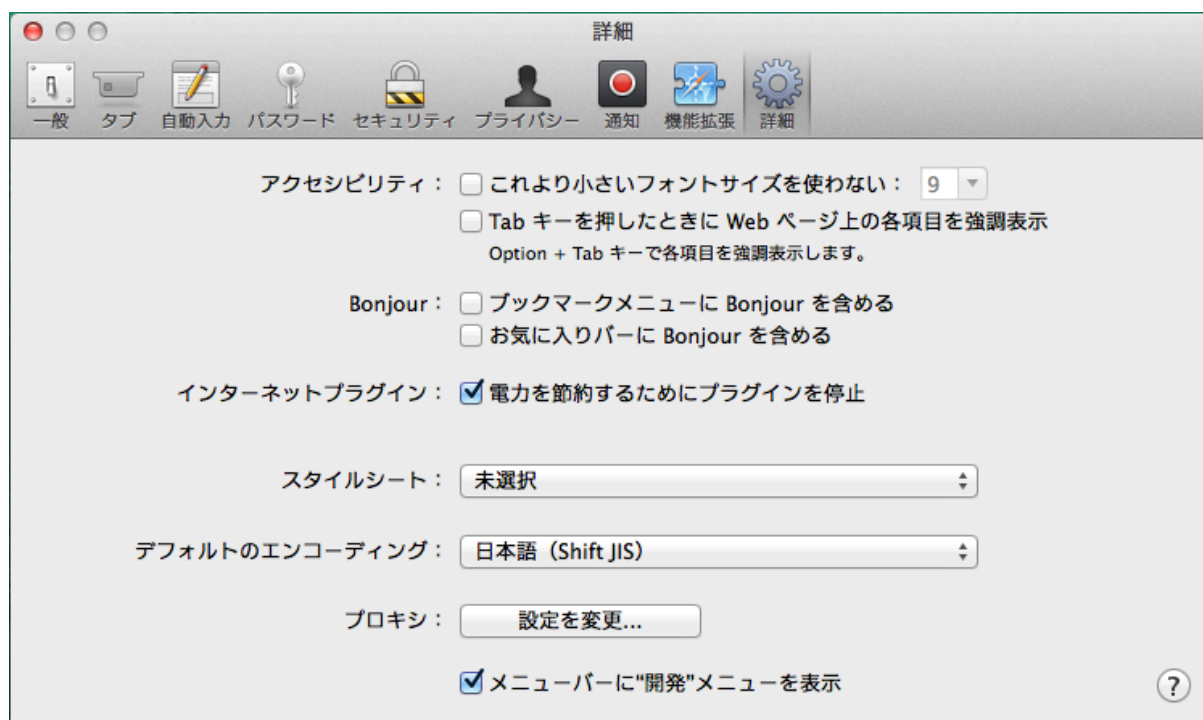


図 10.5: Safari のキャッシュの削除

10.11 quota の上限を大きくしたい

授業や実験などで大きなファイルを扱っている場合、10.9.1 節や 10.10 節などを試しても、ディスク使用量を quota の制限以下に抑えられなくなることがあります。このような場合は 10.13 節を参考に、技術職員に問い合わせてください。

10.12 間違えてファイルを消してしまった

COINS では snapshot 機能ももちいて、システム全体のデータのバックアップを自動的にとっています。このバックアップを用いることで、約 2 週間前までのファイルであれば、復元ができます。バックアップされたファイルは、`/home/.snap/all/`内のディレクトリ（ディレクトリ名はそのディレクトリにバックアップが作成された年月日）に保存されています。

あとは、`cp` コマンドなどをつかって目的のファイルを取り出しましょう。たとえば、ユーザ `s19xxxxx` の 2019 年 3 月 10 日に取られたバックアップは、`/home/.snaps/all/2019_03_10_0000/ugrad/19/s19xxxxx` に保存されています。

```
$ ls -l /home/.snaps/all合計
112
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_09_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_10_0000
```



```
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_11_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_12_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_13_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_14_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_15_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_16_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_17_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_18_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_19_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_20_0000
drwxr-xr-x 15 root root 4096 月 3  2 13:56 2019_03_21_0000
drwxr-xr-x 15 root root 4096 月 3 21 11:32 2019_03_22_0000
```

以下は、ユーザ s19xxxxx のホームディレクトリのバックアップ（2019 年 3 月 10 日現在）を、ls コマンドで参照した例です。

```
$ ls /home/.snaps/all/2019\_03\_10\_0000/ugrad/19/s19xxxxx↵
Maildir _OLD_HOME_2019_ .bashrc
```

10.13 各種問い合わせ先

以上に記した方法で解決ができない場合、授業中であれば TA か教員に尋ねてみましょう。もしくは、coins-trouble@coins.tsukuba.ac.jp 宛に、以下のことを記載してメールを送ってください。

- 利用者名（アカウント名）
- 利用した計算機名
- 症状が発生した時刻
- 具体的なトラブルの内容

あるいは、技術職員室（3E108）まできて技術職員に直接問い合わせてください。また、COINS のシステムに関する一般的な質問につきましては、coins-admin@coins.tsukuba.ac.jp 宛にメールを送ってください。

索引

alias コマンド, 51
appletviewer, 170

break point, 176

C, 171
C++, 179
cat コマンド, 41
cd コマンド, 40
chmod コマンド, 43
compile, 168
compiler, 168
convert コマンド, 46
cp コマンド, 39

debug, 174
du コマンド, 45

Emacs, 56
 info, 74
 Mew, 67
 Package.el, 77
 Undo, 65
 インクリメンタルサーチ, 65
 エコーエリア, 60
 置換, 66
 画面の分割, 75
 キー表記, 57
 検索, 65
 自動セーブ, 62
 スクロールバー, 60
 タイトルバー, 60
 チュートリアル, 58
 ツールバー, 60
 ヘルプ, 57
 ヘルプバッファ, 57
 補完機能, 61
 マイナーモード, 76
 メジャーモード, 76
 モード, 76

Firefox, 79
FORTRAN, 180
Free Software Foundation, 171

gcc, 171
gdb, 175, 197
gfortran, 180
GNU Compiler Collection, 171
GNU Octave, 185
GNU デバッガ, 175

grep コマンド, 42
Guest Machine, 187
Guest OS, 187

head コマンド, 41
Host Machine, 187
Host OS, 187

include, 172
info コマンド, 48
interpreter, 168

Java, 168
javac, 169

kill コマンド, 47

L^AT_EX, 124
L^AT_EX
 \$, 152
 \[, \], 152
 \\, 139
 align 環境, 150
 align*環境, 151
 $\LaTeX$, 150
 amsmath パッケージ, 149
 \and, 132
 \author, 132
 \bibliography, 161
 \bibliographystyle, 160
 BIB_TE_X, 157
 \caption, 143
 \centering, 143
 \chapter, 133
 \cite, 159
 \clearpage, 139
 color パッケージ, 154
 ConT_EXt, 164
 \date, 132
 description 環境, 140
 document 環境, 132
 \documentclass, 131
 dvipdfmx コマンド, 126
 DVI ファイル, 126
 \emph, 137
 enumerate 環境, 140
 ε -p_TE_X, 125
 ε -up_TE_X, 163
 extractbb コマンド, 142
 figure 環境, 143
 float パッケージ, 143

`\footnote`, 146
`\footnotemark`, 146
`\footnotesize`, 137
`\footnotetext`, 146
`geometry` パッケージ, 161
`\geometry`, 161
`graphicx` パッケージ, 141
`\hline`, 145
`\huge`, 137
`\includegraphics`, 143
`itemize` 環境, 139
`jsarticle`, 131
`jsbook`, 131
`\label`, 144, 147
`\LARGE`, 137
`\Large`, 137
`\large`, 137
`LATEX 2ε`, 125
`listings` パッケージ, 153
`\lstdefinestyle`, 153
`\lstineline`, 156
`\lstinputlisting`, 156
`lstlisting` 環境, 155
`\lstset`, 154
`LuaTEX`, 164
`\maketitle`, 132
`\multicolumn`, 145
`\newpage`, 139
`\normalsize`, 137
`\paragraph`, 133
`\part`, 133
`pbibtex` コマンド, 161
`pdfTEX`, 163
`platex` コマンド, 125
`\ref`, 147
`\scriptsize`, 137
`\section`, 133
`\small`, 137
`\subparagraph`, 133
`\subsection`, 133
`\subsubsection`, 133
`table` 環境, 144
`\tableofcontents`, 162
`tabular` 環境, 145
`TEX`, 124
`TEX Live`, 162
`\textbf`, 137
`\textgt`, 137
`\textit`, 137
`\textmc`, 137
`\textsc`, 137
`\texttt`, 137
`thebibliography` 環境, 158
`TikZ`, 165
`\tiny`, 137
`\title`, 132
`\underline`, 137
`\usepackage`, 141
`\verb`, 157
`verbatim` 環境, 156
`XYTEX`, 163
環境, 130
グループ, 137
コマンド, 129
コントロールシーケンス, 129
制御綴, 129

バウンディングボックス, 141
ブリアンブル, 141
マクロ, 129
見出し, 133
`ls` コマンド, 36
`lv` コマンド, 41

`make`, 178
`Makefile`, 178
`man` コマンド, 47
`MATLAB`, 181
`Microsoft IME`, 117
`mkdir` コマンド, 38
`MTU`, 212
`mv` コマンド, 39

`nkf` コマンド, 52

`open` コマンド, 45

`Poderosa`, 200
`printf` 関数, 172
`ps` コマンド, 46
`PuTTY`, 200
`pwd` コマンド, 40

`quota` コマンド, 45

`rm` コマンド, 40
`rmdir` コマンド, 38

`Shibboleth`, 9
`Simulink`, 182
`Snapshot`, 195
`snapshot` 機能, 40
`sort` コマンド, 43
`source code`, 168
`SSH`, 200
`ssh` コマンド, 50
`SSH 接続`, 200

`tail` コマンド, 42
`Tera Term`, 200
`thunderbird`, 87
`TWINS`, 6

`Virtual Machine`, 186
`VMware Fusion`, 187
`VPN`, 207

`World Wide Web`, 79
`WWW`, 79

`xcalc` コマンド, 53
`xclock` コマンド, 53

アクセス権, 36
アプレット, 168

移動プロファイル, 116
インクルード, 172
印刷, 23
インスタンス, 168
インターネット, 79
インタプリタ, 168

ウィンドウ, 106
ウィンドウシステム, 18
`Web ブラウザ`, 79

オブジェクト指向言語, 168

カーネルデバッグ, 197

拡張子, 110

拡張子, 36

仮想計算機, 186

カレントディレクトリ, 35

キーボード, 3

機械語, 168

キャプチャ, 21

共有フォルダ, 196

クッキー, 84

クラスの定義, 168

クラスファイル, 169

ゲスト OS, 187

ゲストマシン, 187

言語処理系, 168

コマンド, 35, 45

コマンドウィンドウ, 182

コマンドのリスト, 53

コマンドプロンプト, 118

ゴミ箱, 12

コンパイラ, 168

コンパイル, 168

最適化, 178

サインアウト, 103

サインイン, 101

システム環境設定, 13

実行ファイル, 168

ショートカット (Windows) , 110

ショートカット (キーボード) , 21

ショートカット (ブラウザ) , 82

処理系, 168

スタートボタン, 102

スタートメニュー, 102

ステータスバー, 108

スナップショット, 195

セキュリティの設定, 85

絶対パス名, 36

相対パス名, 35

ソースコード, 168

ソースファイル, 168

タイトルバー, 107

タイル, 102

タスクマネージャー, 112

タブ, 83

ディレクトリ, 30

ディレクトリデリミタ, 35

デスクトップ, 10

デバッグ, 175

デバッグ, 174

統一認証システム, 6

ドック, 11

ネットニュース, 96

パーミッション, 36

バグ, 174

パス名, 35

パスワード, 6

引数, 35

ディレクトリ構造, 31

ファイルの復元, 40

ファインダー, 12

フォルダ, 109

ブックマーク, 82

プライバシーの設定, 84

フルスクリーンモード, 195

ブレークポイント, 176

プログラミング言語, 168

分割コンパイル, 177

ヘッダファイル, 172

ホームディレクトリ (^) , 36

ホームページの設定, 84

ホスト OS, 187

ホストマシン, 187

マウスの操作, 18

無線 LAN, 204

メールサーバ, 87

メッセージフィルタ, 94

文字コード, 86

文字化け, 86

ライブラリ, 179

ライブラリファイル, 172, 179

リボン, 108

リモートログイン, 200

リモートログイン, 49

リンク, 172

ログアウト, 5

ログイン, 3

ログイン後の画面, 10

ワークスペース, 183

平成 30 年度 情報科学類 計算機運用委員会 構成員

教員
天笠 俊之（委員長）
新城 靖
金澤 健治
三宮 秀次
技術職員
澤村 博道
山形 朝義
山崎 豊

本手引きは平成 30 年度手引を元に改版され，下記の方々の協力により作成されました．

氏名
田地章倫
細谷啓
吉本祐樹
和田朋子

発行日 2019.4

編集 筑波大学情報学群情報科学類計算機運用委員会