大型計算機センターでのUNIX入門

南里, 豪志
九州大学大型計算機センター : 助手

https://doi.org/10.15017/1470331

出版情報：九州大学大型計算機センター広報. 31 (2), pp.61-102, 1998-06. 九州大学大型計算機センター
バージョン：
権利関係：

大型計算機センターでの UNIX 入門

南里豪志 *

現在, 九州大学大型計算機センター (以下, 本センター) のほとんどの計算機はUNIX と呼ばれる OS (Operating System) を介して利用できます. UNIX は, 大勢のユーザーに対して同時にサービスを行うことを前提に作られているOS です. そのため, 以前は UNIX というと, 大型計算機やワークステーションのように, PC (パーソナルコンピュータ) より高性能な計算機で利用されることがほとんどでした. しかし最近では, PC の性能の飛躍的な向上もあって, Linux や FreeBSD 等の PC UNIX と呼ばれる UNIX が開発され, 利用されています.

本稿ではこの UNIX について, アクセスの方法, ファイル操作等のコマンド, 文書の編集, マニュアルの参照法等, 基本的な利用法を紹介します. 本稿の内容は, 基本的には本センターの汎用計算機 M1800/20U 上で動作する UXP/M を前提としていますが, 本センターの計算機を利用するために用意された特殊なコマンド以外については, 他のほとんどの UNIX でも利用できます.

なお, 本稿ではキー操作について以下のような表記を用います.

[ESC]	エスケープ・キーを意味します. キーボードによっては, Escape や ESC と表記されている場合もあります.
[↵]	リターン・キーを意味します.
[DEL]	デリート・キーを意味します. キーボードによっては, Del や Delete と表記されている場合もあります.
[BS]	バックスペース・キーを意味します. キーボードによっては, BS や BackSpace や ⇐ と表記されている場合もあります.
[Control]	コントロール・キーを意味します.
[C-]	これは, [Control]キーを押しながら続くキーを押すという操作を表します. [Control] キーは, キーボードによっては, CTR や CTRL と表記されている場合もあります. 例えば, [C-x]は, [Control]キーを押しながら [x]キーを押すという操作になります.
[M-]	これは, [ESC]キーを押した後に続くキーを押すという操作を表します. 例えば, [M-x]は, [ESC]キーを押した後に [x]キーを押すという操作になります.

*九州大学大型計算機センター助手

目次

1 本センターで利用できる UNIX

表 1: 本センターの計算機

ホスト名	OS	IP アドレス	備考
kyu-vpp.cc.kyushu-u.ac.jp	UXP/V	133.5.9.70	スーパーコンピュータ (VPP700/56)
kyu-cc.cc.kyushu-u.ac.jp	UXP/M	133.5.9.1	汎用計算機 (M-1800/20U)
wisdom.cc.kyushu-u.ac.jp	Solaris 2.4	133.5.9.9	ワークステーション (S-4/1000E) (各種アプリケーションの提供)

本センターで UNIX を利用できる計算機のうち主なものを表 1に示します. その他の計算機や, 利用可能な時間については本センターのホームページを参照してください.

このうち本記事では, 主に kyu-cc 上で UXP/M を利用する場合を前提として説明しますが, 記事中のほとんどの操作はどの UNIX でも同じように利用できます.

1.1 本センターの UNIX へのアクセス方法

本センターの UNIX へは, 以下の 3 通りの方法によりアクセスできます.

- インターネットを介したアクセス
- 電話回線を介したアクセス
- 本センター 2F のオープン端末室を利用

1.1.1 インターネット経由

もし、手持ちの計算機がインターネットに接続されていれば telnet 等の端末ソフトを利用することにより, インターネットを介してアクセスができます.

アクセスには telnet, rlogin, rsh等のコマンドを利用してください.

1.1.2 電話回線からのアクセス

本センターの端末サーバ kyu-ts を利用することにより, 電話回線を通じてアクセスすることができます. 電話回線を経由したアクセスの方法については本センターのホームページを参照してください.

1.1.3 X 端末の利用

本センター 2F のオープン端末室には, ユーザーインタフェース用ワークステーション及び X 端末が用意されており, 本センターのユーザーに解放されています.

1.2 問い合わせ先

本センターの計算機利用に関する質問等は, 電話, 電子メールにより受け付けています. 電話番号及び電子メールアドレスは, 「広報」 の裏表紙の見返しに掲載しています.

また, 本センター 2F のプログラム相談員室でプログラムに関する相談や指導を行っています. これらのサービスの詳細な情報も含め, 本センターに関する情報は本センターのホームページに掲載しています. 本センターのホームページは下記の URL から参照できます.

http://www.cc.kyushu-u.ac.jp

1.3 利用上の注意

1.3.1 予算管理

予算管理は各自で行って下さい．もし予算を超過した場合はその時点の翌日から利用資格打切になるので注意して下さい．現在の利用金額はセンターコマンド utfee で表示されます (6節を参照してください)．

1.3.2 パスワード管理

パスワード管理も各自の責任で行って下さい．他人にパスワードが知られてしまった場合，自分のプライバシーが守れないだけでなく，他のユーザにも迷惑をかけることがあります[1] ．このため，他人に自分のパスワードが知られないよう常に気をつける必要があります．また，利用承認書に書かれている初期パスワードは，管理の都合上非常に安易なものを用いています．そのうえ，利用承認書が他人に見られている可能性もあるので，最初にログインした際に，必ずパスワードの設定を変更して下さい．新しいパスワードとなる文字列としては，他人が用意に推測できない文字列を使用して下さい．

パスワードの変更方法は 2.3 節で説明します．

2 UNIX の利用開始から終了まで

UNIX での操作の流れは基本的に以下のようになります．

1. ログイン: ユーザーの認証を行い，利用を開始する．
2. 作業: ファイル操作，ジョブ投入，エディタ利用等
3. ログアウト: 利用を終了する．

この一連の操作をセッションと呼びます．

2.1 ログイン (login)

アクセスが成功すると UNIX は login のためのユーザー認証を始めます．まず，login: という問い合わせに対してユーザー ID を入力します．次に，passwd: という問い合わせに対してパスワードを入力します[2]．このとき，入力したパスワードは表示されません[3]．

これらの入力が正しければ，正規のユーザーと認められ，利用を許可されます．その後，Terminal Type: と表示され，端末の種類を聞かれるので入力してください．(良く分からない場合は ↵ キーを打ってください．) すると kyu-cc% と表示されます．これはプロンプト(prompt)と呼ばれます．この表示は，kyu-cc という計算機で UNIX のセッションが開始されて，ユーザーからのコマンドを受け付けている状態を示しています．

```
UXP/M TELNET (kyu-cc)

login: a79999a
Password:
Fujitsu UXP/M (kyu-cc)
Copyright (c) 1984, 1986, 1987, 1988 AT&T
Copyright (c) 1990, UNIX System Laboratories, Inc.
Copyright (c) 1991, 1992, 1993 FUJITSU LIMITED
All Rights Reserved
Last login: Fri May  3 15:44:08 on kushida.cc.kyu-

Terminal Type:
kyu-cc%
```

[1] 例えばあなたのパスワードが誰かに知られると，その人はあなたの名前を騙って虚偽のメールを書くことができます．

[2] パスワード入力時にパスワードを他人に知られないように注意してください．

[3] パスワード入力中は BS キーや DEL キーを使った文字の入力ができません．キータイプを間違えたと思ったときは，C-u を押して最初から入力してください．

2.2 UNIX のコマンド

ログインして, 以下のようなプロンプトが表示されていれば UNIX のコマンド(command) を実行することができます.

```
kyu-cc%
```

コマンドは, キーボードから入力し, ↵ キーを押すと実行されます. 例えば次の例では現在の日付, 時間を表示するコマンド date が実行されます. UNIX では, アルファベットの大文字と小文字を区別するので, コマンド名を間違えないように注意して下さい.

```
kyu-cc% date
Thu Apr 10 21:23:38 JST 1997
kyu-cc%
```

また, コマンドによっては, 引数や - で始まるオプションを付加することによって, 細かく動作を指定することができます. 上の例では date によって日本標準時が表示されましたが, オプション -u を付加することによりグリニッジ標準時を表示させることができます.

```
kyu-cc% date -u
Thu Apr 10 12:23:45 GMT 1997
kyu-cc%
```

2.3 パスワード

UNIX の特徴の一つとして, マルチユーザー(multi user) を対象とした OS であるということが挙げられます. すなわち, UNIX は同時に複数のユーザーに対してサービスを行うことができます. そのため, 各ユーザーのデータ等を保護するために, Windows95 等の個人ユーザー向け OS と比べて厳格にユーザーの認証を行います. このユーザー認証は, 予め登録されているユーザー ID とパスワードを照合することよって行われます[4].

ユーザー ID 利用承認書中の利用者番号と同一です. ただし, 利用承認書では英文字が全て大文字で表記されていますが, 計算機のユーザー ID は小文字で登録されています. 注意してください. (e.g. H79999A ⟶ h79999a).

パスワード 利用承認書中の初期パスワードという項目に書かれた文字列です.

利用承認書を受け取って最初にログインした時に, 必ずパスワードを変更してください. パスワード設定コマンドは passwd です.

以下にパスワード設定の例を示します. ログインの時と同様, 実際にはパスワードの入力部分は画面に表示されません.

```
kyu-cc% passwd
UX:passwd: 情報: k70043a のパスワードを変更します
旧パスワード:
新パスワード:
もう一度新パスワードを入力して下さい:
kyu-cc%
```

新しいパスワードとしては以下の条件を満たすものを使ってください.

- 2 つ以上の欧字および 1 つ以上の数字または特殊文字を含む.
- 長さが 6 文字以上である.
- 以前のパスワードと 3 文字以上異る文字列である.

条件に合わない場合, その旨表示されて, 再度新しいパスワードを聞いてきます.

[4]銀行で例えると, ユーザー ID が口座番号で, パスワードは暗証番号の代わりになります.

2.4 ログアウト (logout)

セッションを終る時は logout または exit と入力します.
すると, UXP は今回のセッションに関する情報を表示してセッションを終了します.

```
kyu-cc% logout

### Session Information : a79999a@kyu-cc ###
  Login  Time :1996/05/03 17:24:48
  Logout Time :1996/05/03 17:30:52
  Session Time:   364(Sec.)

  User-CPU    :      228703(MicroSec.)
  SystemCPU   :      287087(MicroSec.)
  User-VPU    :           0(MicroSec.)
  SystemVPU   :           0(MicroSec.)
```

3 ファイル

ファイル(file) とは, 計算機上でデータを扱うときの単位の一つです. UNIX には通常の文書ファイルだけでなく, 周辺機器とのインタフェース等, 特殊な目的に利用するファイルがあります. ここでは, UNIX でファイルを操作するために必要な, 構造やいくつかの概念について説明します.

3.1 ファイルの名前

UNIX ではファイルの名前として, アルファベット, 数字, 記号による任意の文字列を許しています[5] . ただし, 記号の中には特殊な意味を持つ物があるため, . (ピリオド), - (ハイフン), + (プラス), _ (アンダースコア) 以外の記号はほとんど使われません[6]. また, - (ハイフン) が先頭になるようなファイル名[7]は, 誤ってコマンドに対するオプションとして扱われるので, 避けたほうが無難です.

さらに UNIX では, コマンドと同様, ファイル名についてもアルファベットについての大文字, 小文字を区別します. すなわちファイル名 A のファイルとファイル名 a のファイルは別のファイルです. そのため, DOS 等の OS で大文字, 小文字を区別せずに使っていた場合は注意が必要です.

3.2 ファイルの構造

図 1 に, UNIX のファイル構造の例を示します. このように UNIX のファイルは木構造で管理されています.

図中の, 四角で囲まれたファイルは, ファイルを収納するための特別なファイルで, ディレクトリ (directory) と呼ばれます. ディレクトリもファイルの一種ですので, ディレクトリの中に別のディレクトリを収納することができます.

このディレクトリを用いることにより, DOS のディレクトリや, Windows95 のフォルダのように, 種類別にファイルを整理することができます. 例えば, 図 1 において, 仕事に利用するファイルは work ディレクトリの下にまとめられています. これらはさらに, プロジェクトごとに proj1 ～ proj3 という名前のディレクトリに分類されています. このうち proj1 ディレクトリでは, プログラムのファイル (main.f, test1.f 等) が program ディレクトリの下に, 文書ファイル (intro.tex, main.tex 等) が report ディレクトリの下に置かれています.

ファイルの位置

この木構造におけるファイルの位置を表記したものがパス (path) です. パスの表記方法としては以下の二つがあります.

[5]システムによっては日本語の名前をつけることができる場合もありますが, あまり一般的ではありません.
[6]例えば, test(1) や, file* といったファイル名は利用しないでください.
[7]例えば, -test

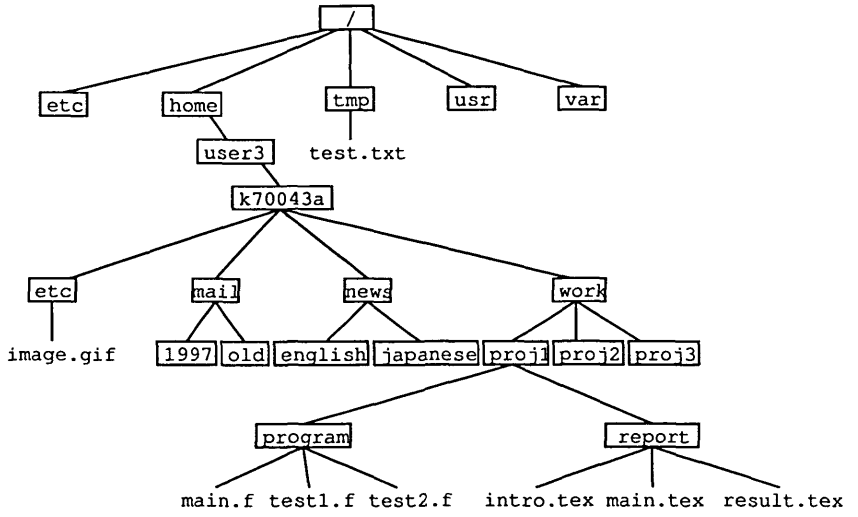

図 1: ファイルの木構造 (例)

絶対パス　この木構造の根となっているルートディレクトリ (root directory) と呼ばれるディレクトリから辿った位置を示す

相対パス　あるファイル (ディレクトリ) からの相対的な位置を示す

例えば図 1 の `test.txt` という名前のファイルの絶対パスは, `/tmp/test.txt` と表記されます. これはルートディレクトリの下の `tmp` ディレクトリの下の `test.txt` という意味です. このように絶対パスは, ルートディレクトリを示す '/' の後に, ルートディレクトリからファイルへ辿る過程で通過したディレクトリを列挙します. ディレクトリとディレクトリ, 及びディレクトリとファイルは / で区切ります[8] .

これに対し, 例えば図 1で, ディレクトリ `program` から見た `test1.f` の相対パスは`test1.f` で, `main.tex` の相対パスは `../report/main.tex` です. このように, 相対パスの表記では最初が / で始まりません. また必要に応じて, 一つ上のディレクトリを示す.. を用います.

UNIX のコマンドにファイル名を指定する場合, 絶対パスと相対パスのどちらを利用することもできます.

カレントディレクトリ

UNIX にログインすると, ユーザーはあるディレクトリを基本的な作業対象とします. この, 作業の中心となるディレクトリのことをカレントディレクトリ (current directory) と呼びます. カレントディレクトリの下のファイルを扱う場合は, パスを指定せずにファイル名だけを指定することができます. また, カレントディレクトリは前述した相対パスの基準となります[9] . 実際には, 必要に応じてカレントディレクトリを変更しながら作業を進めます[10] .

カレントディレクトリはコマンド `pwd` で表示され, `cd` で変更できます (4.1節参照).

ホームディレクトリ

通常, ユーザーにはそれぞれ専用のディレクトリが用意されており, そのディレクトリがログイン時のカレントディレクトリとなります. このディレクトリをホームディレクトリ (home directory) , またはログインディレクトリと呼びます. 例えば kyu-cc におけるユーザー k70043a のホームディレクトリのパスは `/home/usr3/k70043a` です (図 1).

[8] この / は, 一般にスラッシュ (slash) を略して「スラ」と読まれます. 例えば `/tmp/test.txt` は, 「スラ・テンプ・スラ・テスト点テキスト」と読まれます. ただしこれは日本での話で, 外国でどのように読まれるかは知りません.

[9] 実際に UNIX を利用していると, カレントディレクトリはユーザーが現在居る場所のようなイメージがあるので, 「`/tmp` ディレクトリに居る」とか, 「`/etc` ディレクトリに移る」という言い方をすることがあります.

[10] カレントディレクトリを変更 (移動) する最も大きな理由は, パスを入力するのが面倒だからです.

特殊なディレクトリ

特別なディレクトリを表す記号として次のようなものがあります．これらは，パスを指定するときに用います．利用例は 4.1 節を参照してください．

~(チルダ)	ホームディレクトリ
.(ピリオド)	そのディレクトリ自身
..	一つ上のディレクトリ

3.3　ファイルの所有者とアクセス権

UNIX のファイルには，ファイルのアクセス権，ファイルの所有者及び，ファイルのグループという属性があります．

アクセス権には，読み出し権，書き込み権，実行権の 3 種があります．このうち実行権とは，そのファイルをコマンドとして実行する権利です．ただし，ディレクトリの実行権は，そのディレクトリの下に移動する権利を表します．

以下の 3 つのカテゴリのユーザーに対しそれぞれ設定されます．

user	ファイルの所有者であるユーザー
group	ファイルの所属するグループに属するユーザー
other	それ以外のユーザー

UNIX のファイルのアクセス権をこのように細かく設定できるのも，UNIX が複数のユーザーに対して同時にサービスできるマルチユーザー OS であるためです．すなわち，ファイルのアクセス権を正しく設定しなければ，他のユーザーの過失 (もしくは故意) によりファイルが変更されたり消去されたりします．

ファイルの所有者とは，ファイルのアクセス権を変更できるユーザーのことです．通常，あるユーザーのホームディレクトリ[11]，及び作成したディレクトリの所有者はそのユーザーになっています．あるファイルのアクセス権を表示するには，`ls -l` を実行します (4.1節参照)．また，ファイルのアクセス権を変更するには，`chmod` を実行します (4.1節参照)．

さらに，全ユーザーがファイルを作成できるディレクトリとして，`/tmp` があります．このディレクトリは主に一時的なファイルの保存に用いられます．そのため，`/tmp`ディレクトリの下のファイルは，警告無しに消去されることがあります．また，`/tmp` ディレクトリは，バックアップ (テープなど別の記憶装置への退避) も行っていませんので，重要なファイルは必ず自分のホームディレクトリのしたに移しておいてください．

また，ファイルのグループとは，そのファイルが所属するユーザーグループです．UNIX では，複数のユーザーを一つのグループとし，そのグループに対してアクセス権を与えることができます．ただし，そのグループの設定を行えるのは計算機の管理者だけです．`kyu-cc` では，全てのユーザーが `user` グループに所属していますので，他人からのアクセスを制限する場合はグループに所属するユーザーに対しても制限する必要があります．

4　基本的なコマンドの紹介

ここでは，UNIX で用いる基本的なコマンドを紹介します．

4.1　ファイル操作

カレントディレクトリの表示，変更 (移動)

コマンド名	機能
`pwd`	カレントディレクトリの表示[12]
`cd`	カレントディレクトリの変更 (移動) [13]

[11] ディレクトリもファイルの一種ですから，所有者及びアクセス権が設定されます．
[12] Print Working Directory の略です．
[13] Change Directory の略です．

cd コマンドで指定する変更先のディレクトリは相対パスまたは絶対パスで指定します. ただし, 実行するユーザーに対して, ルートディレクトリから変更先のディレクトリに到達するまでの全てのディレクトリに実行権がなければ, カレントディレクトリをそのディレクトリに変更できません. また, cd コマンドに引数を与えなければカレントディレクトリはホームディレクトリとなります.

```
kyu-cc% pwd                          … カレントディレクトリを表示します.
/home/usr3/k70043a
kyu-cc% cd /usr/local/bin            … /usr/local/bin というディレクトリに移動します.
kyu-cc% pwd
/usr/local/bin
kyu-cc% cd .                         … カレントディレクトリに移動します. (= 移動しません. )
kyu-cc% pwd
/usr/local/bin
kyu-cc% cd ..                        … 一つ上のディレクトリに移動します.
kyu-cc% pwd
/usr/local
kyu-cc% cd ~                         … ホームディレクトリに移動します.
kyu-cc% pwd
/home/usr3/k70043a
kyu-cc%
```

ファイル情報の表示

コマンド名	機能
ls	ファイル情報の表示[14]

ls は指定されたディレクトリやファイルに関する情報を表示します. 通常のファイルを指定した場合, そのファイルに関する情報のみが表示されます. ディレクトリを指定した場合, そのディレクトリにあるファイルに関する情報を表示します. また, ls コマンドにファイル名やディレクトリ名を指定しなければ, カレントディレクトリに存在するファイルを表示します.

```
kyu-cc% ls                           … カレントディレクトリに存在するファイルを表示します.
bin  doc  tmp
kyu-cc% ls tmp                       … tmp ディレクトリに存在するファイルを表示します.
test.txt   report.txt
kyu-cc% ls tmp/test.txt              … tmp ディレクトリの下の test.txt というファイルを表示します.
tmp/test.txt
```

ls にオプション -F をつけることにより, ファイルの種類を示す記号をつけて表示されます. 記号の種類としては, 実行可能ファイルであることを示す *, ディレクトリであることを示す / 等があります.

```
kyu-cc% ls -F                        … ファイルの種類を表示するオプションを指定します.
bin/  doc/  tmp/                     … ディレクトリを示す / が付加されて表示されます.
kyu-cc% ls -F tmp
report.txt   test.txt                … 通常のファイルには種類を示す記号は付加されません.
```

ls コマンドは, 基本的に "." で始まるファイルを表示しません. そのようなファイルも表示させたい場合は -a オプションを付加します.

```
kyu-cc% ls -a                        … . (ピリオド) で始まるファイルを表示します.
.    ..   bin  doc  tmp
```

[14] list の略です.

また, ls コマンドは, 基本的にファイル名だけしか表示しません. しかし -l オプションを付加することにより, ファイル名の他に各ファイルについて詳細な情報を表示します.

```
kyu-cc% ls -l                          … カレントディレクトリの下のファイルについて詳細な情報を表示します.
総ブロック数 40
drwx------   2 k70043a  user        512 May 10 12:19 bin
drwx------   2 k70043a  user       1024 Oct  1  1991 doc
drwxr-xr-x   2 k70043a  user        512 Apr  3 13:08 tmp
kyu-cc% ls -al tmp                     … tmp ディレクトリの下のファイルについて詳細な情報を表示します.
総ブロック数 40
drwxr-xr-x   2 k70043a  user        512 Apr 11 13:45 .
drwxr-xr-x   5 k70043a  user        512 Apr 11 11:43 ..
-rw-r--r--   1 k70043a  user        325 Apr  3 13:08 report.txt
-rw-r--r--   1 k70043a  user        794 Mar 20 10:02 test.txt
kyu-cc%
```

オプション -l をつけたときに表示される情報は以下のようになっています.

drwxr-xr-x	2	k70043a	user	512	Apr 3 13:08	tmp
ファイルの種類とアクセス権	リンク数	ファイルの所有者	ファイルの所属するグループ	ファイルの大きさ	そのファイルが最後に修正された日付	ファイル名

このうち, ファイルの種類とアクセス権は, 下のように4つのフィールドに区切ると, 第 0 フィールドがファイルの種類, 第 1, 2, 3 フィールドはそれぞれファイルの持ち主, 同じグループ, それ以外のユーザーに対するアクセス権を表しています.

フィールド	0	1	2	3
文字列	d	rwx	rwx	rwx

ファイルの種類は以下のように表されます[15] .

d　ディレクトリ
-　通常ファイル

また, アクセス権はそれぞれ 3 桁の文字列で表示されます.

r　読み取り可能
w　書き込み可能
x　実行可能
-　指定されたアクセス権は与えられていない.

例えば全てのアクセス権が与えられている場合, rwx と表記され, 書き込み権と実行権が無い場合は r-- と表記されます.

先の例の tmp はディレクトリで, ファイルの持ち主に全アクセス権があり, 同グループ, 及びそれ以外のカテゴリのユーザーに対して読み出し権と実行権があります[16]. また, 持ち主は k70043a, グループは user で, サイズは 512 バイト, 4 月 3 日の 13:08 に修正されている, ということが表示されています.

ファイルのアクセス権の変更

コマンド名	機能
chmod	ファイルのアクセス権変更

[15]他にもありますが本稿では扱いません.
[16]すなわち, 全てのユーザーがこのディレクトリの下に移動することができます.

chmodは, 指定されたユーザーのカテゴリ, 及びアクセス権について, ファイルのアクセス権を変更します. ユーザーのカテゴリはu (所有者), g (同じグループ), o(それ以外) で指定し, アクセス権はr(読み出し権), w(書き込み権), x(実行権) で指定します. また, 指定された権利を許可する場合, カテゴリとアクセス権を + で接続し, 指定された権利を禁止する場合, カテゴリとアクセス権を - で接続します.

例えば, ファイル private.txt を他人 (所有者以外の全てのユーザー) に読ませたくない場合, 次のようにします.

```
kyu-cc% ls -l
総ブロック数 0
-rw-r--r--   1 k70043a  user           0 Apr 15 20:04 private.txt
kyu-cc% chmod go-r private.txt       … g(group) と o(other) に対して read を禁止します
kyu-cc% ls -l
総ブロック数 0
-rw-------   1 k70043a  user           0 Apr 15 20:04 private.txt
kyu-cc%
```

ファイルの削除, 複製, 移動及び名前の変更

コマンド名	機能
cp	ファイルの複製[17]
rm	ファイルの削除[18]
mv	ファイルの移動, 名前の変更[19]

cp で, コピー元, コピー先のファイル名は次のように指定します.

cp コピー元ファイル名 コピー先ファイル名またはディレクトリ名

コピー先ファイル名としてディレクトリを指定すると, そのディレクトリの下にコピー元ファイル名と同じ名前でコピーファイルを作成します. また, 次のようにすると複数のファイルを一度にコピーできます.

cp コピー元ファイル名1 コピー元ファイル名2 コピー元ファイル名3 コピー先ディレクトリ名

コピー先に同じ名前のファイルがある場合は上書きされます.

```
kyu-cc% ls -F
report.txt   test.txt   work/
kyu-cc% cp report.txt report1.txt          … ファイル report.txt を新しいファイル report1.txt に
                                             コピーします.
kyu-cc% ls
report.txt   report1.txt  test.txt   work/
kyu-cc% cp report.txt report1.txt work     … ファイル report.txt と report1.txt を
                                             ディレクトリ work にコピーします.
kyu-cc% ls
report.txt   report1.txt  test.txt   work/
kyu-cc% ls work
report.txt   report1.txt
```

rm では, ファイル名を空白で区切って列挙することにより複数のファイルを消す事ができます.

[17] copy の略です.
[18] move の略です.
[19] remove の略です.

```
kyu-cc% cd work
kyu-cc% rm report.txt report1.txt     … work の下のファイル report.txt, report1.txt を削除します.
kyu-cc% ls
kyu-cc%
```

mv で, 移動元, 移動先のファイル名は次のように指定します.

mv 移動元ファイル名 移動先ファイル名またはディレクトリ名

mv は, ディレクトリ間の移動の他に, 名前の変更にも用いられます. 移動先がディレクトリの場合, そのディレクトリの下に, 名前を変更せずに移動します. 移動先がディレクトリではなく, 通常のファイルで移動元のファイル名とは異る場合, 名前が変更されます. また, 移動先として指定した名前のファイルが既に存在する場合は上書きされます. cp と同様に複数のファイルを一度に移動することもできます.

```
kyu-cc% cd ..
kyu-cc% ls
test.txt     work/
kyu-cc% mv test.txt work              … ファイル text.txt をディレクトリ work に移動します.
kyu-cc% ls
work/
kyu-cc% ls work
test.txt
kyu-cc%
```

注意

cp, mv によって上書きされたり, rm によって削除されたファイルは基本的に元に戻すことが出来ません. そこで, 既存のファイルを上書きしたり, ファイルの削除を行う前に確認を求めるオプションとして, -i があります[20].

```
kyu-cc% rm -i work/test.txt
rm: work/test.txt を消去しますか (yes/no)? no
kyu-cc% ls work
test.txt
kyu-cc%
```

また, これらのコマンドによってあるディレクトリにファイルを追加したり削除したりする場合, コマンドを実行するユーザーはそのディレクトリに対してあらかじめ書き込み権を持っていなければなりません.

ディレクトリの作成, 削除, 複製

コマンド名	機能
mkdir	ディレクトリの作成
rmdir	空のディレクトリの削除
rm -r	ディレクトリ全体の削除
cp -r	ディレクトリ全体の複製
mv	ディレクトリ全体の移動

ディレクトリ全体に対する操作とは, そのディレクトリだけでなく, ディレクトリの下にあるファイル全部に対する操作です. 例えば図 1 の work ディレクトリ全体を削除すると, work の下のディレクトリ proj1, proj2, proj3 や, さらに下のディレクトリ program, report, 及びその下のファイル main.f, test1.f, test2.f, intro.tex, main.tex, result.tex まで全部削除されます. このように, ディレクトリの削除や複製には -r オプションを用います. これに対してディレクトリの移動にはオプションは必要ありません.

[20] 5.4.3節で紹介する alias を使うと, cp, mv, rm を実行するときに必ず -iオプションを付けるように設定できます.

```
kyu-cc% mkdir work2                    … カレントディレクトリの下にディレクトリ work2 を作成します.
kyu-cc% ls
work    work2
kyu-cc% cp -r work work2               … ディレクトリ work 全体をディレクトリ work2 の下にコピーします.
kyu-cc% ls
work    work2
kyu-cc% ls work2
work
kyu-cc% rm -r work2/work               … work2 の下のディレクトリ work 全体を削除します.
kyu-cc% ls work2
kyu-cc% mv work work2                  … ディレクトリ work 全体をディレクトリ work2 の下に移動します.
kyu-cc% ls
work2
kyu-cc% ls work2
work
kyu-cc% rm -r work
kyu-cc% ls
kyu-cc%
```

4.2 テキストファイルに対する操作

テキストファイル (text file) とは, 文書やプログラムのように, 文字を記録しているファイルです.[21] . ここではテキストファイルの表示や検索等を行うためのコマンドを紹介します.

テキストファイルの内容表示

コマンド名	機能
less	ファイルの内容をページごとに出力
more	ファイルの内容をページごとに出力
cat	ファイルの内容を出力
head	ファイルの内容を先頭から指定行数分出力
tail	ファイルの内容を末尾から指定行数分出力

これらのコマンドは, ファイルの内容を標準出力 (5.4.5節参照. ここでは画面) に出力します.

less は, 1 画面表示するごとに出力を停止し, 画面底部に : を出力してユーザーからのコマンド入力を待ちます. この状態へのコマンドとして次のようなものがあります.

less のコマンド

キー入力	機能
[↵]	1 行分上にスクロールし, 次の 1 行を出力
[Space]	1 画面分上にスクロールし, 次の 1 画面を出力
[↵]	1 行分上にスクロールし, 次の 1 行を出力
[C-p]	1 行分下にスクロールし, 前の 1 行を出力
[b]	1 画面分下にスクロールし, 前の 1 画面を出力
[q]	less の終了

これに対し more は, less の機能のうち, 下方向へのスクロールができません. また, 日本語テキストが入ったファイルは, 文字コードによっては読めない場合があります.

cat は, ファイルの内容をそのまま画面に出力するため, 大きなファイルを画面に出力する場合には不適です. むしろ, 後述するリダイレクションやパイプ機能と組み合わせて, 複数のファイルの結合などの, ファイル操作によく利用されます.

[21] これに対して, 人間に直接読めない形で記録してあるファイルをバイナリファイル(binary file) と呼びます.

head, tail に対する行数の指定は - に続けて行います. 例えばファイルの先頭 20 行を出力したい場合, head -20 とします.

ファイル内容の検索及び整列

コマンド名	機能
grep	指定された文字列を含む行を出力する
sort	ファイルを行ごとに整列する

例えば, test.txt というファイルの中で, shell という文字列を含む行を表示する場合, 次のようにします[22].

```
kyu-cc% grep shell test.txt
test.txt:inferior shell that communicates directly with the terminal.  Emacs
test.txt:waits until you exit the subshell.  (The way to do that is probably
test.txt:with 'C-d' or 'exit', but it depends on which shell you use.)  The only
test.txt:way on these systems to get back to the shell from which Emacs was run
test.txt:   Suspending also fails if you run Emacs under a shell that doesn't
test.txt:non-'nil' value to force 'C-z' to start an inferior shell.  (One might
kyu-cc%
```

grep の類似コマンドとして, egrep, fgrep があります. 詳しい使い方はオンラインマニュアル (7 節) を参照してください.

5　シェル: UNIX のユーザーインタフェース

プロンプトを出してユーザーからのコマンドを受け付け, それを UNIX に渡すユーザーインタフェースの役割を果たしているプログラムはシェル(shell)と呼ばれます. ここでは, 最も一般的な C シェル (csh) と呼ばれるシェルの操作法を説明します. この他にも様々なシェルを利用できますが, 本稿で説明する操作はほとんどのシェルで利用できます. 他のシェルを利用する方法については 5.3節で説明します .

5.1　シェルの設定

コマンド名	機能
set	シェル変数の表示, 設定
setenv	環境変数の設定
printenv	環境変数の表示

コマンドの中にはシェルの設定によってその動作を変えるものがあります. このシェルの設定は, シェル変数と, 環境変数によって行います[23].

頻繁に利用される変数としては以下のものがあります.

[22] shell という単語を含む行だけでなく, subshell のように shell が一部に入っているような単語を含む行も表示されます.
[23] 基本的にシェル変数はシェルの動作に影響を与えるのに対し, 環境変数はアプリケーションの動作に影響を与えます.

シェル変数	機能
path	実行するファイルを探すディレクトリのリスト
term	端末の名前 kterm, xterm, vt100 等

環境変数	機能
DISPLAY	X Window の表示画面 (8 節参照)
LANG	表示に利用する言語 japan (日本語) または C (英語)
PAGER	オンラインマニュアルを見るときに利用するコマンド more, less

5.1.1 シェル変数と環境変数の表示

現在のシェル変数の表示にはコマンド set を, 環境変数の表示には printenv を利用します. 以下は筆者の設定です[24].

```
kyu-cc% set
argv    ()
cwd     /usr/local/emacs/info
filec
history 20
home    /home/user3/k70043a
ignoreeof
path    (/usr/sbin /usr/lbin /usr/ccs/bin /usr/bin
/usr/uxp /usr/ucb /usr/local/bin /usr/local/bin/X11 .)
savehist        30
shell   /bin/csh
status  0
term    xterm
```

```
kyu-cc% printenv
HOME=/home/user3/k70043a
PATH=/usr/sbin:/usr/lbin:/usr/ccs/bin:/usr/bin:/usr/uxp:
/usr/ucb:/usr/local/bin:/usr/local/bin/X11:.
LOGNAME=k70043a
HZ=100
TZ=JST-9
TERM=xterm
SHELL=/usr/bin/csh
MAIL=/var/mail/k70043a
PWD=/home/user3/k70043a
```

5.1.2 シェル変数と環境変数の変更

シェル変数の変更にはコマンド set を, 環境変数の変更にはコマンド setenv を利用します. 次の例の set では, コマンドを探索するディレクトリのリストであるシェル変数 pathの現在の値 $path を用いて path の先頭に /usr/ucbを追加しています. また, setenv では X Window の表示画面を指定する環境変数DISPLAY に IP アドレス 133.5.7.84 の端末の画面を設定しています. X Window については 8 節で説明します.

[24]ユーザーによって項目数や値が違います.

```
kyu-cc% set path=(/usr/ucb/ $path)
kyu-cc% setenv DISPLAY 133.5.7.84:0.0
```

シェル変数を無効にする場合はコマンド unset を，環境変数を無効にする場合はコマンド unsetenv を利用します．

```
kyu-cc% unsetenv DISPLAY
kyu-cc% unset term
```

シェル変数 path

UNIX で利用するコマンドのほとんどは，実行ファイルと呼ばれるファイルで実現されています．そのため，本来はその実行ファイルの位置を絶対パスや相対パスで指定する必要があります．しかし，シェル変数 path に列挙されているディレクトリの下にある実行ファイルについては，ファイル名を指定するだけでシェルが自動的に探して実行します．例えば，4.1節の pwd や 4.1 節の ls は，それぞれ /bin/pwd, /bin/ls に実行ファイルがあります．これらのコマンドを，パスを指定せずに実行できたのは，シェル変数 path に /bin というディレクトリが含まれているからです．

シェルは，シェル変数 path に含まれているディレクトリを左から順に探索します．そのため，同じ名前の実行ファイルが，path に記述された複数のディレクトリにあった場合，それらのディレクトリのうち最も左に記述されているものの下にある実行ファイルが実行されます[25]．

5.2　設定の保存

よく利用する設定は，ホームディレクトリの下にある以下のファイルに記述しておくと，次のログインから自動的に反映されます．

- .cshrc

  csh の立ち上げのたびに実行される．

- .login

  login 時に一回だけ実行される．(.cshrc のあと)

もしこれらのファイルが無い場合は，9節で紹介するようなエディタを用いて新規に作成します．これらのファイルのサンプルを以下に紹介します．内容については，sh, cshのオンラインマニュアル (7節) を参考にしてください．

[25] 環境変数 PATH は，シェル変数 pathと連動しています．すなわち，片方を変更した場合，もう片方も同じように変更されます．このような変数として他に，TERM と term, PWD と pwd 等があります．

.cshrc

```
# define the number of previous commands to be remembered
set history = 20

# define the search path for command execution
set path = (/usr/local/bin /usr/bin /usr/ccs/bin\
            /usr/uxp /usr/ucb /usr/bin/X11\
            /usr/graphman/bin /usr/cgms/bin\
            ~/local/bin .)

# define the alias of history command
alias h history
alias ls ls -aCF

# never make core file
limit coredumpsize 0
```

.login

```
# set default file protection masking bits for new files.
# "022" means "no write permission for other users"
umask 022
# set the shell ignores end-of-file from terminal
set ignoreeof
# set the MAIL path
setenv MAIL /usr/mail/${LOGNAME}
# NQS environment check
if ( $?ENVIRONMENT != 0 ) then
  exit
endif
# terminal-dependent stuff
# EDITOR is some editor, not necessary full-screen
if ($TERM="") then
    echo -n "Terminal Type: "
    set newterm='line'
    if ($newterm == "") set newterm=vt100
    setenv TERM $newterm
    unset newterm
endif
setenv EDITOR /usr/local/bin/mule -nw
setenv LANG   japan
setenv PAGER less

stty erase '^H' kill '^U' intr '^C' echoe tostop
```

5.3 他のシェルを利用する

現在, csh 以外で良く知られているシェルとしては, 以下のようなものがあります.

名前	パス	ヒストリ	ジョブ管理	コマンド行編集	alias
sh	/usr/bin/sh	×	×	×	×
jsh	/usr/bin/jsh	×	○	×	×
csh	/usr/bin/csh	○	○	×	○
bash	/usr/local/bin/bash	○	○	○	○
ksh	/usr/bin/ksh	○	○	○	○
tcsh	/usr/local/bin/tcsh	○	○	○	○

パスとはそのシェルの実行ファイルのパスです[26]．ヒストリとは，それまでのコマンド実行の履歴を記憶し，再利用できる機能です(5.4.2節)．コマンド行編集とは，途中まで書いたコマンドやオプションを書き直すことができる機能です．alias とは，コマンドとして別名を利用することができる機能です(5.4.3節)．

sh, jsh, ksh, csh, bash については，オンラインマニュアル(7節)が用意されています．

シェルを実行すると，新しいシェルのプロンプトが表示され，元のシェルに変わってユーザーからの入力を受け付けるようになります．このシェルの実行を終了する場合はexit コマンドを実行します．

tcsh

最近では csh の機能を拡張した tcsh を使うユーザーが増えています．

tcsh は，強力なコマンド編集機能を提供しています．これは，コマンド入力中にコマンド行を編集する機能です．主なコマンド編集機能に次のようなものがあります．

C-f	一文字前進
C-b	一文字後退
C-a	コマンド行の先頭へ移動
C-e	コマンド行の末尾へ移動
C-u	コマンド行をカット
C-k	コマンド行のカーソル位置より末尾までカット
C-y	直前にカットされた内容をペースト
C-p	ヒストリ上の一つ前のコマンドを表示
C-n	ヒストリ上の一つ後のコマンドを表示
TAB	ファイル名やコマンド名の補完

tcsh を利用するには，csh から /usr/local/bin/tcsh というコマンドを実行する方法と，ログイン時に実行されるシェルを /usr/local/bin/tcsh に変更する方法があります．csh から実行した場合，tcsh を終了するためには，exit を実行します．

kyu-cc でログイン時に実行されるシェルを /usr/local/bin/tcsh に変更する場合，6 章で説明するエディタを利用して，ホームディレクトリの下に .utmsrcというファイルを作成し，次の内容を記述します．

.utmsrc

```
SHSEP:ON
SHELL:/usr/local/bin/tcsh
```

ここで SHSEP はログアウト時に使用状況を表示するか否かを指定します．

5.4　シェル (csh) の便利な機能

ここでは csh の便利な機能を簡単に紹介します．詳細は csh のオンラインマニュアル(7 節)に記述されているので，そちらを参照してください．

5.4.1　ファイル名の入力補助

コマンドに指定するファイル名が長かったり，ファイル数が多い場合に，キータイプの労力を減らす意味で有用な機能として，ファイル名の置換と補完があります[27]．

ファイル名の置換

ファイル名の一部に * がある場合，* の部分に適当な文字列を入れると一致するようなファイル名をアルファベット順に並べたリストに置換されます．これは，類似した名前を持つ複数のファイルに対して一括して操作を行う時に用います．

[26] シェルもアプリケーションなので，UNIX の実行ファイルとして実行します．
[27] これらは非常に便利ですが，キータイプの訓練の妨げになるという欠点も持っています．

```
kyu-cc% ls
proj1.txt  proj2-old.txt  tmp1.c     tmp1.txt   tmp2.c     tmp2.txt
kyu-cc% mkdir proj
kyu-cc% mv proj*.txt proj            … ファイル proj1.txt, proj2-old.txt をディレクトリ proj に移動し
ます.
kyu-cc% ls
proj      tmp1.c    tmp1.txt  tmp2.c    tmp2.txt
kyu-cc% ls proj
proj1.txt  proj2-old.txt
kyu-cc% mkdir C
kyu-cc% mv *.c C                     … ファイル tmp1.c, tmp2.c をディレクトリ C に移動します.
kyu-cc% ls
C         proj      tmp1.txt  tmp2.txt
kyu-cc% rm tmp*                      … ファイル tmp1.txt, tmp2.txt を削除します.
kyu-cc% ls
C         proj
```

例えば上の例で

```
kyu-cc% mv proj*.txt proj
```

というコマンドは, 実際には

```
kyu-cc% mv proj1.txt proj2-old.txt proj
```

というコマンドに置換され, 実行されます.

ファイル名の補完

シェル変数 filec を set することにより, ファイル名を補完する機能を利用できます. これは, ファイル名を途中まで入力し, ESC を押すことにより, 一致するファイル名で残りの文字を補完する機能です. そのディレクトリに同じ文字列で始まるファイル名があると, 一致する部分までを表示します.

5.4.2 ヒストリ機能

コマンド名	機能
history	現在までのコマンドの履歴を表示する.

ヒストリ機能とは以前に実行したコマンドの履歴を保存し, 再利用する機能です. シェルがコマンドの履歴をいくつまで記憶するかはシェル変数 historyで指定します.

```
kyu-cc% set history=10                  … 直前に実行した履歴を 10 個記憶します.
kyu-cc% history
    34  14:43   cd ~/tmp
    35  15:08   mkdir work
    36  15:08   touch work/test.txt
    37  15:09   rm -i work/test.txt
    38  15:22   ls
    39  15:22   \rm -rf work
    40  15:22   which pwd
    41  15:22   which ls
    42  15:22   which \ls
    43  15:36   history
kyu-cc% !l                              … 1 で始まるコマンドのうち最後に実行したコマンドを実行します.
ls
...
kyu-cc% !38                             … 履歴中の 38 番のコマンドを実行します.
ls
...
```

以前実行したコマンドを呼び出すには, 先頭に ! をつけます. この後に数字 n が指定されると, n 番めに実行されたコマンドを, その他の文字列の場合は, 先頭がその文字列である最近のコマンドを実行します

5.4.3　コマンドに別名をつける

コマンド名	機能
alias	コマンドに別名をつける

よく使うコマンド, 長いコマンド, いちいち覚えていられないコマンドを別の名前で登録しておくことができます. 例えば, rm や mv 等のコマンドで誤ってファイルを消してしまわないように, ファイルを消す前に確認を求めるオプション -i は, 以下のようにすれば付け忘れることがありません.

```
kyu-cc(2)% alias cp cp -i
kyu-cc(3)% alias mv mv -i
kyu-cc(3)% alias rm rm -i
```

また, 以下のように設定すると, DOS で使っていたコマンドを仮想的に利用できるようになります.

```
kyu-cc% alias dir 'ls -l'
kyu-cc% alias copy cp
kyu-cc% alias rename mv
kyu-cc% alias delete rm
 ...
```

5.4.4　一時的なディレクトリの移動

コマンド名	機能
pushd	カレントディレクトリを一時的に待避して別のディレクトリに移動する
popd	待避されていたディレクトリに戻る
dirs	退避されているディレクトリを表示する

カレントディレクトリを一時的に変更することができます. ちょっと別のディレクトリに移って作業をするという時に便利です.

```
kyu-cc% pushd /tmp                    … カレントディレクトリを待避して /tmp ディレクトリに移動します.
/tmp ~
kyu-cc% pushd /usr/local              … カレントディレクトリを待避して /usr/local ディレクトリに移動しま
す.
/usr/local /tmp ~
kyu-cc% pwd
/usr/local
kyu-cc% dirs                          … カレントディレクトリは /usr/local で, ディレクトリ /tmp, ~ が保
存されています.
/usr/local /tmp ~
kyu-cc% popd                          … カレントディレクトリを /tmp に戻します.
/tmp ~
kyu-cc% popd                          … カレントディレクトリを ~ に戻します.
~
```

5.4.5 リダイレクション とパイプ

記号	機能
>	> の左側のコマンドによる出力を, 画面に表示する代りに右側のファイルに保存する
>>	>> の左側のコマンドによる出力を, 画面に表示する代りに右側のファイルに追加する
<	> の左側のコマンドに対する入力として, キーボードの代りに右側のファイルの内容を利用する
|	| の左側のコマンドによる出力を, 画面に出力する変わりに右側のコマンドへの入力として (キーボードの代りに) 利用する

リダイレクション (redirection) とは, 本来画面に表示される文字列をファイルに出力したり, キーボードから入力される文字列をファイルから入力するための機能です.

例えば下の例では, まず, ls で本来画面に表示されるはずのファイルリストを, /tmp/dirs というファイルに格納します. 次に, sort という整列コマンドの入力として, キーボードの変わりに /tmp/dirs を利用し, さらにその出力を画面の代りに /tmp/rdirsというファイルに格納します.

```
kyu-cc% ls > /tmp/dirs
kyu-cc% sort -r < /tmp/dirs > /tmp/rdirs
```

また, パイプ (pipe) もリダイレクションに良く似た概念です. こちらは, 2つのコマンドの出力と入力を連結する時に使います.

例えば下の例では, ls で本来画面に表示されるはずのファイルリストを, sort の入力として (キーボードからの入力の代りに) 渡しています. これによって, 上の例で利用した /tmp/dirs という一時的なファイルが不要になります.

```
kyu-cc% ls | sort -r
```

周辺機器を扱うためのファイル

UNIX では, 周辺機器とのインターフェイスもファイルを介して行います. リダイレクションやパイプは, この特殊なファイルを用いて実現しています. 例えば, キーボードに対応するファイルを読み出すことによってキーボードからの入力が行われ, ある画面に対応するファイルに書き込むことによってその画面への出力が行われま

す. このような周辺機器に対応するファイルをデバイスファイルと呼び, 文書ファイルなどの通常のファイルと区別します.

UNIX におけるデバイスファイルの中でも特に重要なものが標準入力ファイル(standard input file) と標準出力ファイル (standard output file) です. 標準入力ファイルは通常キーボードに対応しており, コマンドに対する入力として読み出されます. これに対してコマンドの実行結果が表示される画面に対応するデバイスファイルが標準出力ファイルです. また, エラーを出力するためのデバイスは標準出力とは別の, 標準エラー出力ファイル(standard error)になっています.

リダイレクションは, その名の示す通り, 標準出力ファイルへ向いていた出力を通常のファイルにリダイレクト(redirect) したり, 標準入力ファイルからの入力を通常のファイルにリダイレクトしたりする機能です. また, パイプは, 一方のコマンドの標準出力ファイルを, もう一方のコマンドの標準入力ファイルにつなげる機能です.

5.4.6 ジョブ管理

キー入力	機能
[&]	バックグラウンドでコマンドを実行
	`kyu-cc% a.out &`
	`[1] 14420`
	`kyu-cc%`
`stop`	バックグラウンドジョブの中断
	`kyu-cc% stop %1`
`fg`	中断されたジョブをフォアグラウンドで再開
	`kyu-cc% fg`
[C-z]	フォアグラウンドジョブの中断
[C-c]	フォアグラウンドジョブの強制終了
`jobs`	実行中のジョブの表示
`bg`	中断されたジョブをバックグラウンドで再開
	`kyu-cc% bg %1`
`kill`	ジョブの強制終了
	`kyu-cc% kill %1`

シェルで実行されたコマンドはジョブ(job)として扱われます. 通常のジョブは実行されると終了までシェルのプロンプトが帰ってきません. このような状態のジョブをフォアグラウンド(foreground)のジョブと呼びます. これに対し, シェルとジョブを並行して実行させることもできます. このような状態のジョブをバックグラウンド (background)のジョブと呼びます. コマンドの最後に & を付けて実行することにより, バックグラウンドでジョブを起動することができます.

バックグラウンドで処理されているジョブには番号が付けられます. この番号は %に続けて指定することにより, 上記のコマンドで利用できます.

バックグラウンドジョブの機能は, 時間のかかる処理を実行する場合や, 8節で説明する複数のウィンドウで処理を行う場合に利用します.

また, フォアグラウンドで動作しているジョブは [C-z] により中断する事ができます. これは例えば, `less` でファイルを見ているときに`ls` コマンドでファイルをちょっと見てみたい, という様な場合に利用します.

本センターでバックグラウンドジョブを利用する場合は以下のことに注意してください.

バックグラウンドジョブは本センターのバッチジョブとは違います

バックグラウンドでジョブを起動しても通常の対話型処理として課金されます.

ログアウトの前に必ず全てのバックグラウンドジョブを終了させてください

バックグラウンドで起動したジョブは, ログアウトした後も計算機に残るので, 課金され続けます. ログアウトの前に必ず `jobs` コマンドを実行し, ジョブが残っていないか確認してください.

6 本センターのコマンド

ここでは, 本センターの計算機を利用するために特別に用意されたコマンドを紹介します. ここで紹介するコマンドは他の UNIX では利用できないか, 利用方法が若干異なります.

6.1 ワークステーションへの登録

コマンド	機能
touroku	ワークステーションへの登録

本センターにはライブラリサーバやプリンタサーバ, ユーザーインタフェース等の目的で以下のようなワークステーションが用意されています.

ホスト名	IP アドレス	目的
qvisa	133.5.8.32	AVS の利用
medics	133.5.8.33	カラー PS プリンタの利用, 及びメディア変換
qgas-o1	133.5.8.34	AVS の利用
qgas-o2	133.5.8.35	AVS の利用
gws-o1	133.5.8.231	α-Flow, Masphyc の利用
gws-o2	133.5.8.232	α-Flow, Masphyc の利用
vhsun	133.5.250.46	ユーザーインタフェース (本センター内でのみ利用可能), MENTAT II
vhsgi	133.5.250.45	可視化サーバ (本センター内でのみ利用可能), AVS の利用
qapls	133.5.8.40	Unix 版 SAS の利用

これらのワークステーションを利用する場合, UXP/M で touroku を実行します. 引数を指定しなければ簡単なマニュアルが出力されるので, マニュアルにしたがって再度コマンドを入力します.

すべてのワークステーションに登録したい場合は, touroku all を実行します.

6.2 課金情報の表示

コマンド	機能
utfee	課金情報の表示

utfee を利用することにより, 以下のようにそのユーザーの課金情報を見ることができます.

```
kyu-cc% utfee
18:05:21 1936 I Utfee start.
***** User Information *****
  * ユーザ名    : k70043a
  * 有効期限    : 19990331
  * 予算額      :  1000000
  * UXP 利用額  :    51529
  * MSP 利用額  :       40
18:05:21 1937 I Utfee end.
kyu-cc%
```

6.3 プリンタへの出力

コマンド	機能
lp	プリンタへの出力
lp -dps	プリンタへのテキストファイルの出力
lp -dps -Tps	プリンタへのポストスクリプトファイルの出力
colorps -d npsf	カラープリンタへの出力
PS	
lp -c	NLP への出力

lp コマンドは, 指定したプリンタに対して出力を行います. 本センターの UXP/M から直接出力できるプリンタは, 400 dpi のポストスクリプトプリンタ, カラープリンタ及び NLP です[28].

7 オンラインマニュアルの利用

コマンド名	機能
man	オンラインマニュアルを見る

UNIX では, man コマンドで UNIX コマンドのマニュアルを画面上で見ることができます. これをオンラインマニュアルと呼びます. コマンドによっては日本語のオンラインマニュアルが用意されているものがあります. 環境変数 LANG を japan にすると, 日本語のオンラインマニュアルがある場合はそちらが表示されます.

```
kyu-cc% setenv LANG japan
```

英語のオンラインマニュアルを読むためには環境変数 LANG を C にします[29].

```
kyu-cc% setenv LANG C
```

また, なにも設定しなければ, マニュアルは more を使って表示されます. しかし, 環境変数 PAGER を less にすると, less の機能を利用できるようになります (more, less については 4.2節を参照してください).

```
kyu-cc% setenv PAGER less
```

オンラインマニュアルの詳しい使い方も, man のオンラインマニュアルに記述してあります.

```
kyu-cc% man man
 man(1)                    (UXP/M extension)                    man(1)
  名前
       man — マニュアル・ページの表示
  形式
       /usr/uxp/man [-c] [-k string ...] [-s section[subsection][,...]]
            [-m mandir[,...]] [-x suffix[,...]] title ...
  機能説明
       man は、 title に対応するすべてのマニュアル・ページを表示します。マ
       ニュアル・ページとは、文法書の記事と使用手引書の文法の記事のことです。
...
```

[28] この他に, UXP/M から直接は使えませんが, 1997 年 1 月に導入された A0 カラープリンタがあります. プリントアウトするためには本センター 2F に来て, ワークステーション vhsgi にログインする必要があります. 利用法については
『まあ, お茶でも飲みながら V – でっかく印刷 –』, センター広報 Vol.30 No.1, pp.41–50, 1997
及び本センターのホームページを参照してください.

[29] 日本語のオンラインマニュアルの中には, 誤訳を含むものや日本語にしたためにかえって分かりにくくなったものがあります. そのため, ほとんどの場合で英語のオンラインマニュアルを読んだ方がマニュアルの正確な意味を知ることができます.

上の例では, コマンド man のマニュアルを表示しています. 形式の項目では, まずコマンド man の絶対パスが書かれ, その後ろに指定可能なオプションと, オペランドが書かれています. [と] で囲まれたオプションについては省略できます. また, [,...] と表記されているオプションは, その直前のオプションの値として, ','(コンマ) で区切って複数指定できることを示しています.

8 X Window による画面操作

Machintosh や Windows95 の普及により, 重なりあった窓やアイコンをマウスで操作するウィンドウシステムが広く受け入れられるようになりました. UNIX でも, **X Window**と呼ばれるシステムでこのような操作を行うことができます.

画面に時計を表示する

例えば IP アドレスが 133.5.8.99 の計算機上のディスプレイに kyu-cc の時計を表示させたい場合は, 以下のようにします.

```
% xhost +133.5.9.1                      … 133.5.9.1 (= kyu-cc) からこの計算機への画像出力を許可します.
% telnet kyu-cc.cc.kyushu-u.ac.jp       … kyu-cc にログインします.
.
.
.
kyu-cc% setenv DISPLAY 133.5.8.99:0.0   … 画像の出力先をこの計算機にします.
kyu-cc% xclock                          … 画面に時計を表示するコマンドです.
```

8.1 X サーバと X クライアント

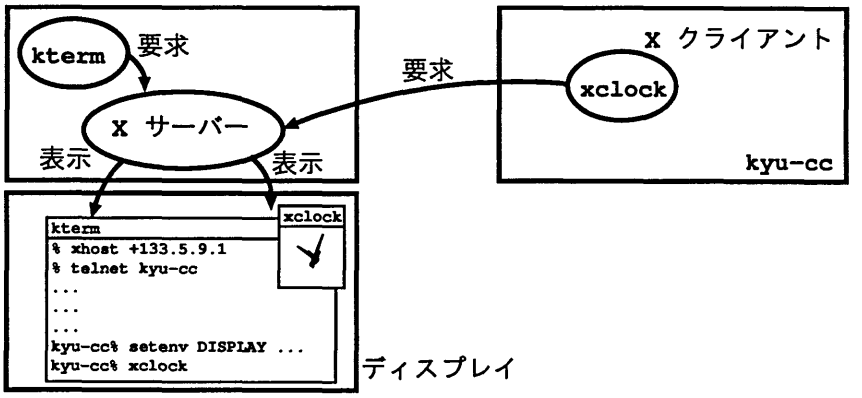

図 2: X サーバと X クライアント

X Window の画面は, その画面に対応する **X** サーバと呼ばれるプロセスによって管理されます (図 2). 通常, この X サーバは画面が表示されているディスプレイが接続されている計算機で実行されています. 画面への出力はこの X サーバへ要求を出すことによって行われます. これに対して X サーバへ要求を出すプロセスは **X** クライアント と呼ばれます. 上の例では, 133.5.8.99 の計算機で X サーバが起動されており, xclock が X クライアントとなって要求を出しています[30] .

X クライアントを実行する時に, どの X サーバに対して出力を要求するかは, 環境変数 DISPLAY で決定されます. 通常, DISPLAY には X サーバの IP アドレスに:0.0 を付加した文字列を設定します. ただし, 最近の X サーバは, ア

[30]最近, Windows95 上で動作する X サーバがいくつか販売されています. まだちょっと高いですが, PC に UNIX を導入するよりも簡単に UNIX の X クライアントを実行できます.

クセスを許可した計算機上の X クライアントからの要求しか受け付けません. このアクセスの許可は, xhost コマンドによって行います.

コマンド名	機能
xhost	X サーバへのアクセスを許可する

上の例では, まず X サーバが動作している計算機で, IP アドレスが 133.5.9.1 の計算機 (= kyu-cc) に対してアクセスを許可した後, telnet で kyu-cc にログインし, DISPLAY 変数を設定してから xclock を実行しています.

8.2 端末エミュレータ

X Window における基本的なユーザーインタフェースを提供するアプリケーションに, 端末エミュレータがあります. 端末エミュレータ上に表示されたシェルのプロンプトに対してコマンドを入力することにより, 計算機を操作することができます.

通常, X Window 上で利用される端末エミュレータは xterm, または kterm です[31] . 本節では kterm について説明します.

kterm の起動

kterm も X クライアントであるため, X サーバ側でアクセスを許可し, kyu-cc で環境変数 DISPLAY を正しく設定しておきます.

コマンド名	機能
kterm	日本語端末エミュレータを起動する

kterm を実行するとウィンドウの外枠が出るので, マウスで適当な所に移動し, マウスの左ボタンをクリックします. するとその枠内に kterm が表示されます.

kterm が実行されると自動的にシェルを起動し, コマンドを受け付けます. これに対して kterm を実行したシェルは, kterm 上のシェルからログアウトするか, kterm を実行したシェルで C-z や C-c を押してkterm を中断しない限りコマンドを入力できません.

```
kyu-cc% kterm
```

そのため, kterm はバックグラウンドで実行する方が便利です.

```
kyu-cc% kterm &
kyu-cc%
```

上の例では kterm をバックグラウンドで起動しています. これにより, kterm 上のシェルと kterm を起動したシェルの双方でコマンドを入力することができます. (バックグラウンドでの実行については 5.4.6節を参照して下さい.)

9 Mule を使った文書編集

本節では Mule と呼ばれるエディタを利用して, UNIX で文章やプログラムを編集する方法を紹介します. Mule とは, Free Software Foundation というグループの人達が開発している GNU Emacs というエディタを多国語に対応するように拡張したものです[32].

Mule では, Control キーと ESC キーを多用します. 冒頭にも書きましたが, 本節ではこれらの操作を以下のように記述します.

[31] kterm は, xterm に日本語機能を追加したものです.

[32] Mule は主に UNIX で利用されているエディタですが, 最近は "Mule for Windows" という, Windows で利用できる Mule も開発されています. 基本的に GNU Emacs や Mule, Mule for Windows はネットワークを介して (例えば ftp://ftp.chiba-u.ac.jp/pub/misc/mule) 無料で手にいれることができます. また, Mule for Windows の CD-ROM を付録としているような本が書店で売られているので探してみてください.

C- Controlキーを押しながら続くキーを押す
例えばC-xは, Controlキーを押しながらxキーを押すという操作です.
M- ESCキーを押した後に続くキーを押す
例えば, M-xは, ESCキーを押した後にxキーを押すという操作です.

9.1 Muleの起動と終了

準備

本センターのX端末やワークステーションからログインしている場合, もしくはXサーバーが動作している計算機からログインしている場合, 環境変数DISPLAYを正しく設定することにより, Muleを新しいウィンドウ(8章, X Window参照)として起動することができます. また, X Windowを利用しない場合は, オプションとして-nwをつけることにより, 新しいウィンドウを起動しなくなります. ただし, Muleをウィンドウとして起動しない場合, 環境変数TERMを正しく設定する必要があります. ほとんどの端末では, TERMをvt100, xterm, ktermのいずれかに設定すればMuleが正常に起動します.

```
kyu-cc% setenv TERM vt100
```

ここでは, 新しいウィンドウとして起動しない方について説明します.

Muleの起動

```
kyu-cc% mule -nw
```

mule -nwの後ろに編集したいファイル名を指定することもできます. ここで存在しないファイルを指定した場合は, 新しく作成されるファイルとして処理されます. (実際はまだ作成されていません)

Muleが起動すると, 最初図3のような画面になり, 起動メッセージが画面に表示されます. このメッセージは, ある程度時間が経つか何か入力された時点で消えます.

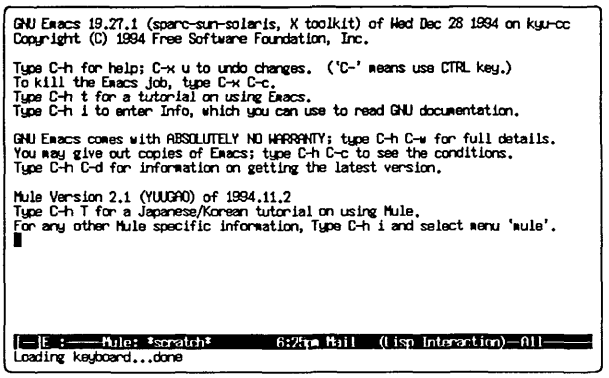

図 3: Muleの起動(起動メッセージ)

Muleの終了

Muleの終了は次のようにして行います.

キー操作	機能
C-x C-c	Muleの終了

ここで, 内容が変更されたにもかかわらず保存動作がされていないバッファが存在すると, そのバッファを保存するかどうかをMuleが問い合わせてくるので, y(yes)かn(no)で答えます. もし, yと入力した場合は, バッファを対応す

るファイルに保存して終了します. n と入力した場合は, 本当に終了してよいか再度問い合わせてくるので, yes か no で答えます. yes の場合は, 強制終了し, 編集した内容は破棄されます. no の場合は, 終了を中止します. Mule のバッファについては後で説明します.

9.2 Mule 画面の説明

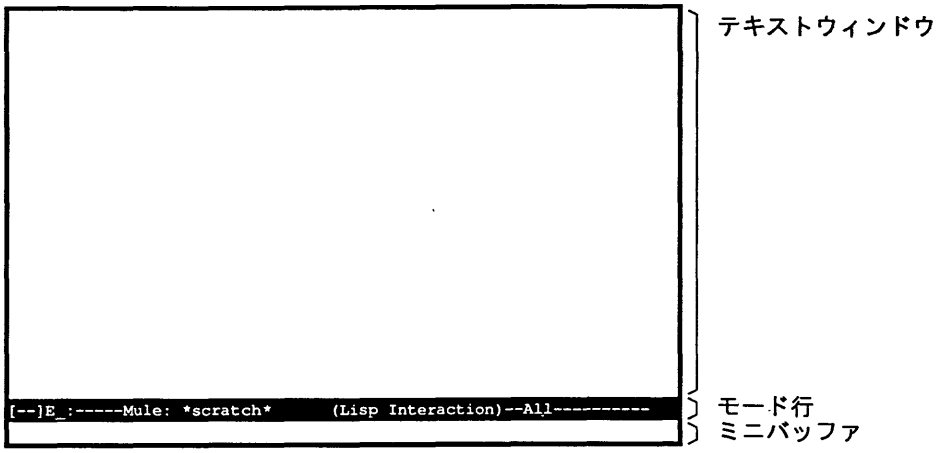

図 4: Mule の画面

Mule の起動画面において, 広い空白の部分をテキストウインドウ と呼びます (図 4). 通常, このテキストウィンドウ上でテキスト編集を行います. テキストウインドウには, 長方形のカーソルが表示されており, キーボードから文字を入力するとこのカーソルの位置に文字が挿入されます.

下方の黒帯は, モード行と呼ばれ, バッファに関する情報等が表示されます. バッファについては後述します.

最下行 (モード行の下) は, ミニバッファと呼ばれる領域で, Mule からのメッセージの表示やコマンドの入力に使用します.

9.3 バッファ

Mule では, ファイルの内容をバッファと呼ばれる一時的な記憶領域に読み込み, そのバッファに対して編集を行ないます (図 5). そのため, バッファへの変更をファイルに反映させるためには保存動作が必要です. また Mule は, 複数のバッファを同時に保持することができます[33]. 複数のファイルを読み込むと, それぞれのファイルについてバッファが用意されます.

バッファの名前

Mule の各バッファには, 元となったファイル名と同じ名前がつけられます. しかし, ファイル名が同じでも異るディレクトリにあるファイルは違うファイルなので, 違うバッファで扱う必要があります. そこで Mule では, 同じファイル名のファイルを複数読み込んだ場合, 2 番目のバッファからは `file<2>` のようにファイル名に番号を付加した名前がバッファ名となります. もちろん, このようなバッファを保存してもファイル名は変更されません.

バッファの操作

キー操作	機能
C-x b	カレントバッファの選択
C-x k	バッファの削除

[33]同時に開くことが出来るバッファの数はメモリの容量に依存します.

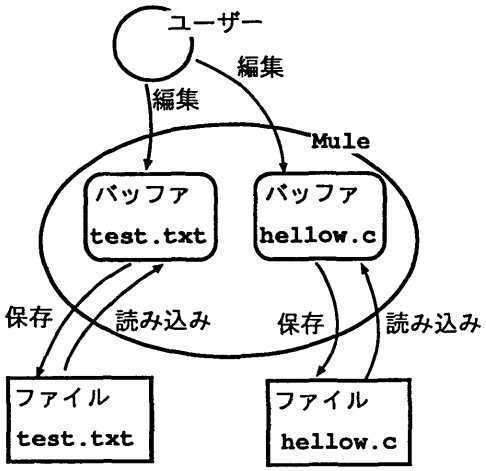

図 5: Mule のバッファ

Mule では, 複数のバッファに対して同時に編集を行うことはできないため, 編集を行うバッファを選択する必要があります. このようなバッファをカレントバッファ と呼びます. 基本的には, カーソルのあるバッファがカレントバッファです.

C-x b と入力すると, バッファ名を聞いてくるので, 選択するバッファ名を入力します.

```
Switch to buffer: (default ...)
```

バッファ名を省略して ↵ だけを打つと, defaultの後に表示されている名前のバッファが選択されます.

C-x k と入力すると, バッファ名を聞いてくるので, 削除するバッファ名を入力します.

```
Kill buffer: (default ...)
```

これも, バッファ名を省略して ↵ だけを打つと,defaultの後に表示されている名前のバッファが削除されます.

編集を行ったにも関わらず保存動作をしていないバッファを削除する場合, 本当に削除していいかどうか確認してくるので, yes か no で答えます.

```
Buffer ... modified; kill anyway? (yes or no) yes
```

保存動作を行わずにバッファを削除しても, 編集内容が反映されないだけで, バッファの元となったファイルが消去されることはありません.

9.4 Mule のファイル操作

ファイルの読み込み

キー操作	機能
C-x C-f	カレントバッファにファイルを読み込む

ファイルを Mule に読み込むには, 前述した, Mule の起動時に編集するファイルを指定する方法と, この, Mule の中で行う方法があります.

C-x C-f を入力すると, ファイル名を問い合わせてきます.

```
Find file: ~/uxp.txt
```

もし, 指定したファイルと同じファイル (名前が同じであるだけでなく, パスが同じであるファイル) を既に読み込んでいる場合, そのバッファをカレントバッファとします. まだ同じファイルを元にしたバッファがなければ, 新しいバッファを用意して指定したファイルを読み込み, そのバッファをカレントバッファとします. (図 6).

図 6: カレントバッファ名 (uxp.txt) の表示

また, 指定したファイルが存在しない場合, そのファイル名をバッファ名とする新しいバッファを用意します. この場合, 実際のファイルはまだ作成されていません. 実際のファイルが作成されるのは, そのバッファに対して保存動作を行った時です.

ファイルの挿入

キー操作	機能
C-x i	カーソルの位置にファイルを挿入する

ファイルの読み込みと同様にファイル名を聞いてくるので挿入するファイル名を入力します.

ファイルへの保存動作

キー操作	機能
C-x C-s	カレントバッファの内容を元となったファイルに書き込む
C-x C-w	カレントバッファの内容を指定されたファイル名のファイルに書き込む

C-x C-wでは, ファイル名を聞いてくるので, 新しく書き込むファイル名を入力します. この場合, 新しいファイル名を元にしてバッファ名も変更されます.

バックアップ機能

Mule は, 誤ってファイルを書き換えてしまったり, Mule が異常終了した場合に備えて, 元の内容を復帰させるための機能が用意されています. これらの機能は何も設定せずに利用できます.

バックアップファイル Mule では, 保存動作を行う際, 編集前のファイルが存在すると, 古いファイルの内容をファイル名に『~』を付けたファイルに保存します. これにより, 間違って編集し, 保存してしまっても, 編集前のファイルがあるので mv でファイル名を変更するだけで元に戻すことができます.

ただし, このバックアップファイルは自動的に消えることはありません. 不要なバックアップファイルはなるべく消去してください.

自動セーブ機能 Mule は, 変更されているのにセーブしていないカレントバッファの内容を, 一定字数の入力があった場合もしくは一定時間が経過する毎に, ファイル名の前後に『#』を付けた名前のファイルとして保存します. これは, Mule が正常に終了しなかった場合にそれまでの編集作業を可能な限り復活させるための機能です. バッファに対して保存動作が実行されると自動的にこの一時ファイルは消去されます.

9.5 Mule によるテキスト編集

テキストウィンドウにカーソルがある状態で, キーボードからテキストを入力することにより, カレントバッファに文字が入力されます. 改行は ↵ キーで行います. 画面上では, 右端まで入力するとカーソルが次の行に移りますが, 右端に \ が表示されるので, 実際は改行されていないことが分かります.

カーソル移動

テキストウインドウ上でのカーソルの移動は以下のキー操作によって行ないます[34]．

キー操作	機能
C-f	1文字左に移動
C-b	1文字右に移動
C-p	1行上に移動
C-n	1行下に移動
C-a	行頭に移動
C-e	行末に移動
C-v	1画面文下に移動
M-v	1画面文上に移動
M-<	バッファの先頭に移動
M->	バッファの最後に移動

カーソルの移動にあわせて画面はスクロールします．

編集範囲の指定

キー操作	機能
C-SPACE	カーソルの位置にマークを付ける

Mule では，テキストの範囲を指定し，その範囲に対して操作を行うことができます．この範囲は リージョン(region) と呼ばれます．リージョンは，任意の場所に設定されたマーク(mark) から，現在カーソルのある位置までの範囲となります．このマークは C-SPACE で付けます．マークを付けるとミニバッファに Mark set と表示され，マークの位置が記憶されます．しかし，このときテキストウィンドウには変化はありません．

記憶されるマークは各バッファに対して 1個所だけです．すなわち，あるバッファでマークをつけると，そのバッファに付けられた以前のマークは利用できません．

テキストの削除

キー操作	機能
DEL	カーソルの直前の文字を削除する
C-d	カーソルの位置の文字を削除する
C-k	カーソルの位置から行末まで削除する 空行の場合は行を削除する
C-w	リージョンを削除する

Mule では，文字単位，行単位，及びリージョンの削除を行うコマンドが用意されています[35]．このうち行単位及びリージョンに対して削除を行った場合，テキストを消去するとともに消去した内容を記憶します．また，C-k を連続して複数回入力した場合，連続して削除された全ての行を記憶します．この記憶した内容を利用することにより，次節で説明するテキストの移動とコピーを行うことができます．

テキストの移動とコピー

キー操作	機能
C-y	記憶した領域をカーソルの位置に挿入する

[34] 計算機によってはカーソルキーを使える場合があります．

[35] BACKSPACE キーは後述するオンラインヘルプという機能に割り当てられているので注意して下さい．もしオンラインヘルプモードに入ったら C-g を数回入力して抜けます．

これにより, テキストの移動 (すなわちカット&ペースト) や, テキストのコピー (すなわちコピー&ペースト) を行うことが出来ます. 記憶された領域は, 新しい領域が記憶されるまで消されることがないため, C-y によってペーストを繰り返すことができます. そのため, テキストのコピーは, 削除した直後に, カーソルを移動せずにC-yで元に戻し, 次に目的の場所に移動して再度 C-y とすればコピーできます.

また, テキストを消去せずにリージョンを記憶することもできます.

キー操作	機能
M-w	リージョンを記憶する

これを利用することにより, テキストを消去せずにコピーを行うことができます.

テキストの検索

次のコマンドで検索することができます.

キー操作	機能
C-s	カーソル以降を前方検索する
C-r	カーソル以前を後方検索する

これらのコマンドを入力すると, (図 7参照) のように表示されるので, 検索文字列を入力します. この際, Mule は検索

図 7: 前方検索の文字列要求

する文字列が入力し終ってから検索を開始するのではなく, 文字が 1 文字入力される都度にそれまで入力された文字列を検索します. 目的の文字列を入力し終った時点で入力した文字列と一致する最初の箇所にカーソルが移動しています.

この状態で, 更に次の該当文字列を検索したい場合は, 更にC-sまたはC-rを押します. この際, 前方検索中にC-rを押して後方検索に変えるといった検索方向の変更も可能です.

カーソルを検索位置に置いて終了したい場合は, ↵ または ESCを押します. カーソルは見つかった文字列の位置で検索が終了し, 検索を開始した位置にマークが付けられます.

また, カーソルの移動コマンド等を入力した場合も検索が終了します.

検索では, 最後に検索した文字列を記憶しています. 同じ文字列で再度検索を開始したい場合は, 検索コマンドを 2 回入力することで検索を再開できます.

複数の画面を同時に表示する

Mule では, テキストウインドウを分割して複数のテキストウインドウを持つことができます (図 8参照). これを マルチプルウィンドウ と呼びます. マルチプルウィンドウを利用する目的としては, 複数のバッファを画面に表示, 編集する他に, 一つのバッファを複数のウインドウで編集することもあげられます. これによって, 大きな文書ファイルやプログラムなどを編集する際に, 他の場所を参照しながら書き進めると行った利用ができます. ウインドウの分割, 消去, 選択には以下のコマンドがあります.

キー操作	機能
C-x o	次のウインドウを選択する
C-x 0	選択されたウインドウを消去する
C-x 1	選択されたウインドウ以外のすべてのウインドウを消去する
C-x 2	ウインドウを上下に 2 分割する
C-x 3	ウインドウを左右に 2 分割する

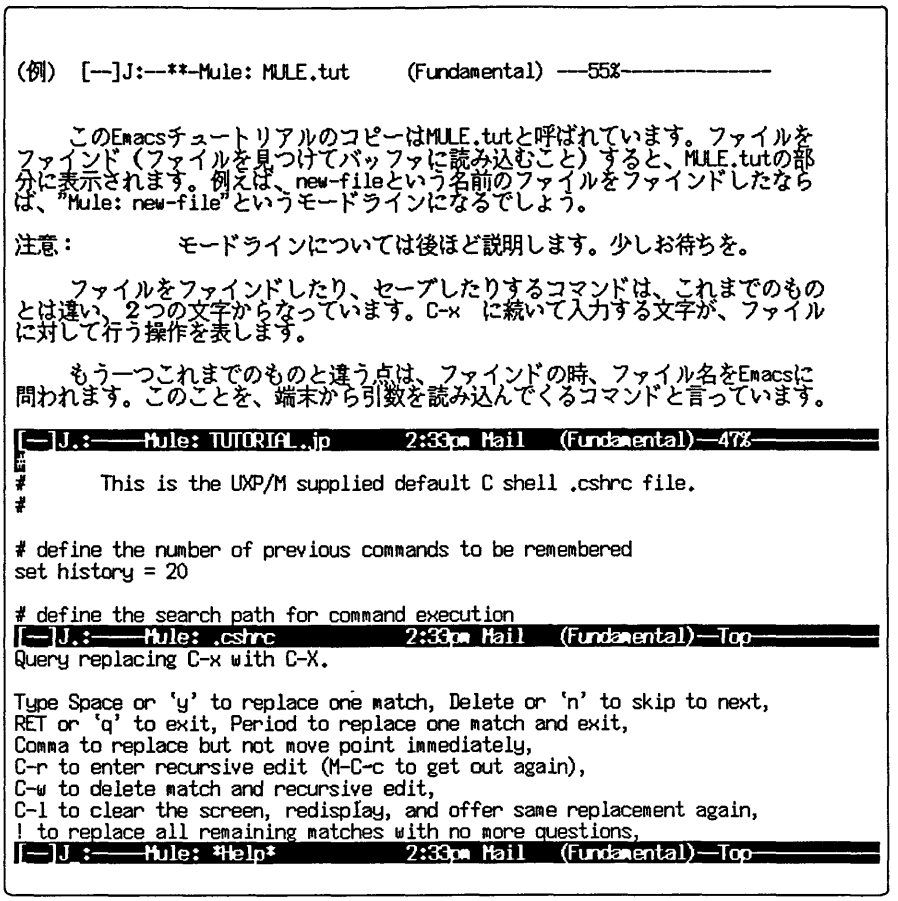

図 8: ウインドウの分割例

複数のウィンドウが開いている場合, カーソルがあるウインドウがコマンドによる編集の対象となります. C-x o により, 次のウインドウを選択します.

9.6 Mule 内のコマンド

Mule には他にも様々な機能が用意されています. これらのほとんどは, 以下のようにして実行されます.

キー操作	機能
M-x コマンド名	ミニバッファでのコマンド入力

このうち, 頻繁に利用するコマンドを紹介します.

指定行への移動

行番号を指定してカーソルを移動したい場合は, 次のようにします.

キー操作	機能
M-x goto-line	指定行への移動

その後, ミニバッファに表示される図10のような問い合わせに対して行番号を入力します.

置換

キー操作	機能
M-x query-replace	対話的に文字列の置換を行なう[36]

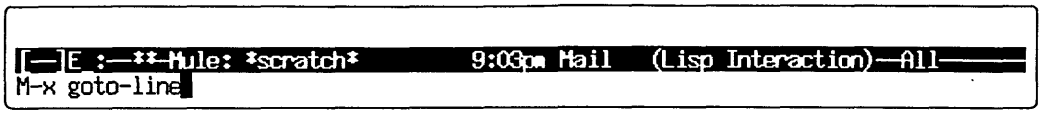

図 9: goto-line コマンド

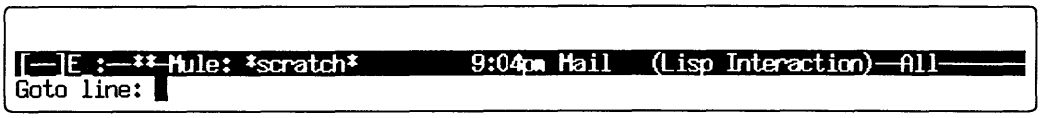

図 10: 行番号の要求

コマンドを入力すると, ミニバッファで置換対象の文字列, 置換後の文字列を順に要求されるので入力します. 例えば, large を huge に変換する場合, まず

```
Query replace: large
```

と, 元の文字列 (*large*) を入力します. 次に変換する新しい文字列を聞いてくるので,

```
Query replace large with: huge
```

のように新しい文字列 (*huge*) を入力します. 入力が終了すると, Mule は置換対象の文字列を前方検索し, 最初に見つかった文字列にカーソルをあわせて, 次のように確認を求めます.

```
Query replacing usr with user: (? for help)
```

ここで, 置換して良ければ SPACE または y を, 置換したくなければ DEL または n を入力します. また, ここで ! (感嘆符) を入力すると, それ以降全ての置換対象となる文字列に対して, 無条件に置換を行います. 置換対象となる文字列が見つからなかった場合, 及び ↵ または q を入力した場合, ミニバッファに Done と表示して置換を終了します.

9.7 Mule による日本語入力

UXP/M では, かな漢字変換サーバ Wnnと, その Mule 用インタフェースである eggによる日本語入力をサポートしています.

egg には, 以下に示すようなモードがあり, 適宜移行していくことにより日本語の入力を行ないます.

(1) 透過モード

Mule 起動時のモード. 日本語入力を行いません. この時, モード行は以下のようになっています.

```
[--]E_:-----Mule: *scratch*          (Lisp Interaction)--All----------
```

(2) ローマ字かなモード

透過モードで C-\ を打つと, モード行が以下のようになります. これをローマ字かなモードと呼びます.

```
[あ]E_:-----Mule: *scratch*          (Lisp Interaction)--All----------
```

もう一度 C-\ を打つと透過モードに戻ります.

キー操作	機能
C-\	透過モードとローマ字かなモードの切替え

このモードでローマ字を入力すると次のフェンスモードに移ります.

(3) フェンス・モード

ローマ字かなモードでローマ字を入力するとカーソルの位置に縦棒（|）２つが現れ, その間に入力をローマ字かな変換したひらがなが表示されます.

```
kyuushuudaigakuoogatakeisankisenta-
```

例えば, 上記のようにキーボードから入力すると以下のように表示されます.

```
|きゅうしゅうだいがくおおがたけいさんきせんたー|
```

この縦棒をフェンスと呼び, この状態をフェンス・モードと呼びます.

フェンス・モードでは, ローマ字と, 以下に示すようなフェンス・モード編集コマンドおよび漢字変換コマンドしか入力できません.

フェンス・モード編集コマンド

キー操作	機能
C-f	カーソルを１文字右に移動
C-b	カーソルを１文字左に移動
C-a	カーソルをフェンスの先頭に移動
C-e	カーソルをフェンスの最後に移動
C-d	カーソルの位置の文字を消去
C-k	カーソルからフェンスの最後までを消去
C-t	フェンス内の文字の転置
C-\	半角文字入力モードへの切替え
C-_	JIS コードによる入力
↵	確定入力
C-g	フェンス内の文字を消去してフェンス・モードを終了
SPACE	漢字変換モードへ移る

(4) 漢字変換モード

フェンス・モードで SPACE と入力すると漢字変換モードに移ります[37] . この時, モード行は,

```
[漢]E_:--**-Mule: *scratch*          (Lisp Interaction)--All----------
```

となり, フェンス中のひらがなが漢字に変換されます.

```
|吸収だ 医学 大型 計算機 センター|
```

漢字変換モードでは一個の半角空白により文節が区切られ, 文節単位に変換が行われます. 変換が正しくない場合, 文節毎に漢字変換コマンドを叩いて再変換します. また, 文節の切り方が適当でない場合, 文節の長さを調節します.

[37]UXP/M ではじめて漢字変換を行う場合, 個人用の設定ファイルを作成するかどうかを質問してくるので, すべてに yes と答えて下さい.

漢字変換

キー操作	機能
SPACE	次候補
C-p	前候補
C-o	カーソルがある位置の文節をのばす
C-i	カーソルがある位置の文節を縮める
↵	確定入力
C-g	漢字変換モードからフェンス・モードに戻る

カーソルがある文節が漢字変換の対象とされるので, 目的の文節までカーソルを移動させる必要があります. カーソルの文節間移動は以下のコマンドで行ないます.

カーソルの文節間移動

キー操作	機能
C-f	次の文節に移動
C-b	前の文節に移動
C-a	最初の文節に移動
C-e	最後の文節に移動

また, 変換候補の一覧をミニバッファに表示し, 選択することもできます (図 11参照).

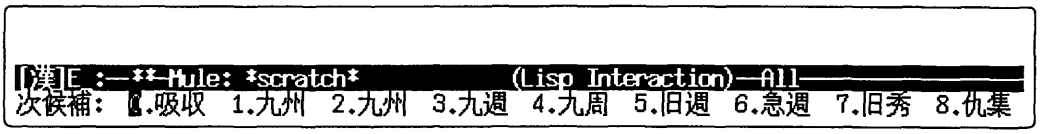

図 11: 変換候補の一覧表示

ここでは以下のコマンドを利用できます.

変換候補一覧での操作

キー操作	機能
M-s	変換候補を一覧表示する
C-n	次の変換候補一覧を表示
C-p	前の変換候補一覧を表示
C-f	変換候補一覧でのカーソルの右移動
C-b	変換候補一覧でのカーソルの左移動
C-a	変換候補一覧でカーソルを先頭に移動
C-e	変換候補一覧でカーソルを最後に移動
↵	カーソルのある変換候補を選択して一覧表示から出る
C-g	変換候補の一覧表示から出る

9.8 Mule マニュアル

チュートリアル

キー操作	機能
C-h t	チュートリアル (英語) を呼び出す
C-h T	チュートリアル (英語以外) を呼び出す

Mule には, 初心者のための練習用チュートリアルが付属しています (図 12). これまで説明してきたような Mule の操作方法を, 実際に端末で試しながら覚えることができます. C-h T を入力すると, ミニバッファに表示する言語を聞かれます. 日本語のチュートリアルを呼び出したい場合は Japanese と入力します.

```
Language: Japanese
```

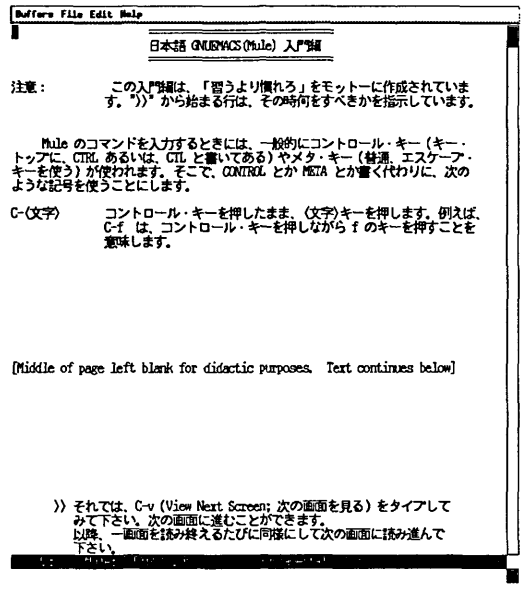

図 12: Mule のチュートリアル

オンラインヘルプ

キー操作	機能
C-h ?	オンラインヘルプを呼び出す

UNIX におけるオンラインマニュアルと同様に Mule にはその時点でのキー操作などを説明するオンラインヘルプが用意されています (図 13). 見たい項目にカーソルを合わせて ↵ を打つと読むことができます. 終了させたいときは, C-x C-k でそのバッファを終了させてください.

トラブルシューティング

Mule には簡単なコマンドで呼び出すことができる機能が, この章で紹介した他にも数多くあります. 誤ってそのような機能を呼び出した時でも, ほとんどの場合, C-g を何回か入力すると, 元のモードに戻ることができます.

9.9 Mule に関する参考文献

- 初心者の人
  宮城史朗, 富田圭介: 初めて使う GNU Emacs/Mule, テクノプレス, 1995.

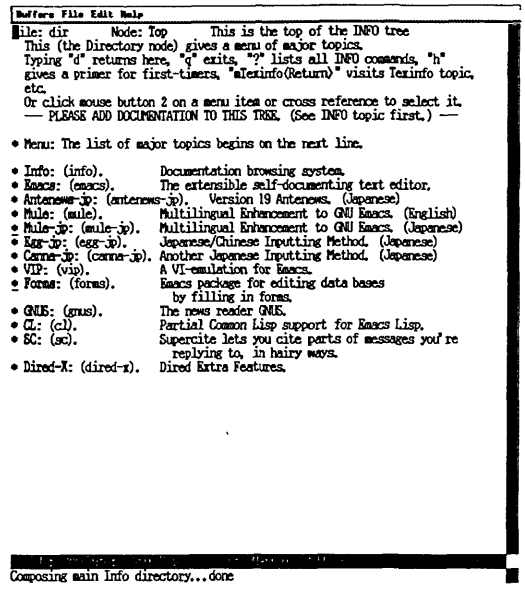

図 13: Mule のオンラインヘルプ

- 更に詳しい情報を求めている人
  R.M. Stallman (竹内、天海監訳): GNU Emacs マニュアル (bit 別冊), 共立出版, 1988.

10　トラブルの対処法

UNIX は基本的に非常に安定した OS ですが, 使い方によっては様々なトラブルが発生します. 本節では, そのようなトラブルのうち頻繁に起るものについて紹介します.

10.1　ログインできない

まず, ユーザー ID とパスワードを正しく入力しているかどうかもう一度確かめてください. ユーザー ID やパスワードは大文字と小文字を区別しますので, 特に Caps Lock キーがロックされたままになっていないか確認してください.

また, 予算を使いきった場合や利用期限が切れた場合もログインできません. これはセンターに問い合わせて頂くと確認することができます.

10.2　画面の表示がおかしい

UNIX で画面が乱れる場合としていくつかあげられます.

less や more 等でスクロールが正しく行われない場合

シェルが仮定しているウィンドウサイズと実際に画面上に存在するウィンドウの大きさが違う場合, アプリケーションの出力が正しく表示されません. この場合, 以下のようにしてシェルで正しい画面サイズを設定します.

```
kyu-cc% eval `resize`
kyu-cc%
```

シェルのメッセージが読めない

日本語を表示できない端末エミュレータで日本語のメッセージを表示させている場合, メッセージが変なコードに化けてしまいます. この場合, 環境変数 LANG を C にして, メッセージを英語で表示させるようにします.

画面に変な漢字が表示されて元に戻らない

バイナリファイルを画面に表示させると, 端末エミュレータが混乱して変な漢字しか表示しないようになります. この場合, Control キーを押しながらマウスの真ん中のボタンを押し続け, メニューが出たらそのまま Do Full Reset の項目までマウスを移動させて, ボタンから手を離します. すると画面がクリアされるので, ↵ を押してください.

10.3 コマンドが見つからない

シェル変数 path に含まれていないディレクトリにある実行ファイルは, パスを指定する必要があります. 絶対パスや相対パスを使って実行するか, path にその実行ファイルがあるディレクトリを追加してください. シェル変数の設定については, 5.1.2節を参照してください.

10.4 コマンドの利用法が分からない

殆んどのコマンドは UNIX のオンラインマニュアルで調べることができます. 7節を参照してください.

10.5 キーボードからの入力を受け付けない

まず, マウスのポインタがウィンドウのなかに入っているか確認してください. それでも駄目な場合は以下を試してください.

シェルのプロンプトが出ない場合

直前に実行したフォアグラウンドのジョブが終了していません. 強制的に終了させたい場合は C-c を入力してください.

シェルのプロンプトが出ている場合

C-s を入力するとキーボードからの入力が受け付けられなくなります. この場合は, C-r を入力すると元に戻ります.

Mule を利用している場合

C-r を入力してみてください. それでも駄目なら C-g を入力してみてください.

10.6 変な名前のファイルができている

ファイル名に *, |, ?, ,等の記号が入っていたり, ファイル名の先頭が - である場合, これらの記号がシェルやコマンドに対する特別な指定として扱われるため, 通常の方法ではこのような記号のファイルを操作することはできません.

このようなファイルは作らないにこしたことはないのですが, もし出来てしまった場合は, ' (クオート) や '' (ダブルクオート) でファイル名を囲むと操作できる場合があります. また, rm コマンドに - オプションを付けると, それ以降の文字列を全てファイル名として扱い, 削除します.

10.7 誤ってファイルを消してしまった

基本的にはどうしようもありません. 最悪の事態にならないよう,

- 大事なファイルは, コピーを別の計算機に置いておく
- alias rm rm -i 等としておく (5.4.3節参照)

- 徹夜明け等の意識がはっきりしない時は大事なファイルの操作を避ける

といったことに気をつけてください.

もし消してしまったファイルが Mule で作成したものであれば, バックアップファイルや一時ファイルが残っている可能性があります. 9.4節を参照してください.

10.8　どうやってもうまくいかない

センターでは以下のようなサポートを行っています. UNIX に限らずセンターの計算機に関する質問はこちらをご利用下さい.

電子メール `request@cc.kyushu-u.ac.jp`

ホームページ `http://www.cc.kyushu-u.ac.jp`

プログラム相談員 主にプログラムに関する質問をセンター 2 階で受け付けています.

11　参考文献

より詳しい使い方を知りたい方は以下のような文献を参考にしてください.

- 坂本 文: “たのしい UNIX,” アスキー, ISBN 4-7561-0785-0, 1990
- 坂本 文: “続・たのしい UNIX,” アスキー, ISBN 4-7561-0789-3, 1995
- 青柳 竜也: “Mule,” クオリティ (発売: 工学図書), ISBN 4-7632-0381-0, 1996

索引