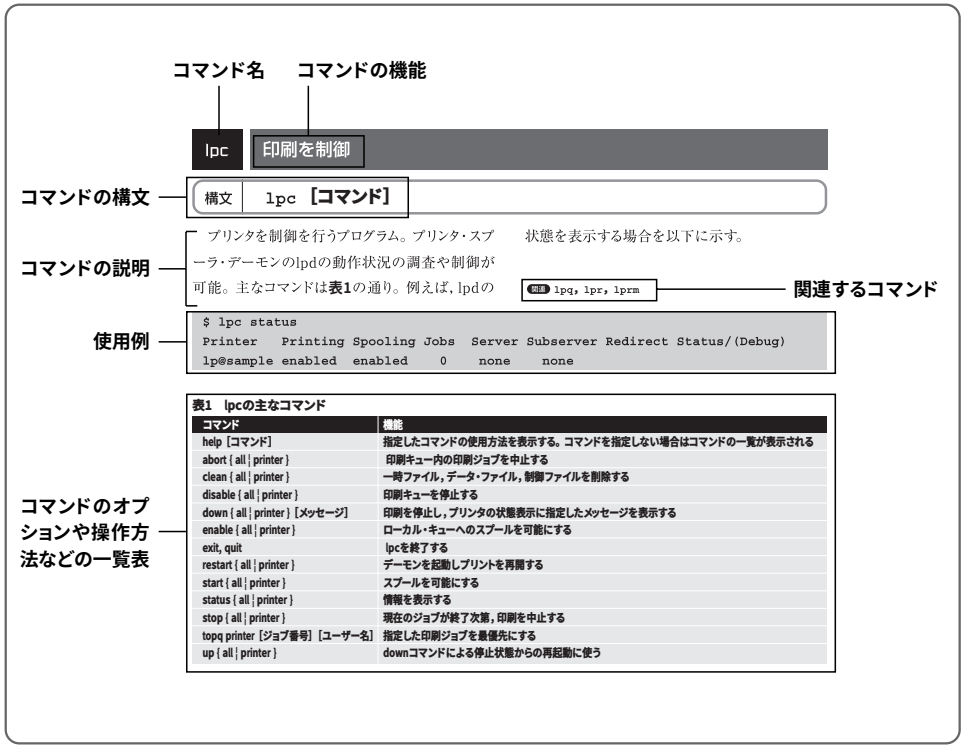
日経Linux
NIKKEI Linux
日経Linux2003年3月号 別冊付録

コマンド・リファレンス 2003

シェル・コマンドの解説を大幅追加

- ▼シェル・コマンド
- ▼ファイル管理
- ▼システム管理
- ▼ジョブ・プロセス管理
- ▼テキスト・ファイル操作
- ▼ネットワーク関連
- ▼SSH関連
- ▼デバイス関連
- ▼印刷関連
- ▼符号化操作
- ▼mtools関連
- ▼その他

●コマンド名 ……コマンドの名前
●コマンドの機能 ……コマンドの主な機能
●コマンドの構文 ……コマンドの基本的な構文を紹介している
●コマンドの説明 ……コマンドの機能や使用上の注意点などを解説している
●関連するコマンド ……関連するコマンドを挙げた
●使用例 ……使い方の具体例を示した
●コマンドのオプションや操作方法などの一覧表 ……コマンド実行時の操作方法や代表的なオプションを紹介している



| コマンドの出力を次のコマンドの入力として渡す

構文 （コマンド1）|（コマンド2）

左側のコマンドの標準出力を右側のコマンドの標準入力に引き渡す。例として，lsコマンドでディレクトリ内のファイルを一覧表示した結果をlessコマンドに渡して閲覧する場合を以下に示す。

関連 <, >, >>

```
$ ls -l | less
```

> 出力のリダイレクト

構文 （コマンド）> （ファイル/デバイス名）

コマンドの出力をファイルやデバイスにリダイレクトする。指定したファイルが既に存在する場合は上書き保存する。存在しない場合はファイルを作成して出力を書き込む。例として，テキスト・ファイル「text.txt」からgrepコマンドで文字列“ linux ”を検索した結果を「result_new.txt」に出力する場合を以下に示す。

関連 |, <, >>

```
$ grep "linux" text.txt > result_new.txt
```

>> 出力をファイルへ追加

構文 （コマンド）>> （ファイル名）

コマンドの出力を指定したファイルの末尾に追加する。テキスト・ファイル「text.txt」からgrepコマンドで文字列“ linux ”を検索した結果を「result_list.txt」に追加出力する場合を以下に示す。

関連 >, |

```
$ grep "linux" text.txt >> result_list.txt
```

< 入力のリダイレクト

構文 （コマンド）< （ファイル/デバイス名）

プログラムやファイルに対する入力をファイルやデバイスから受け入れる。例として，ログ・ファイル「access.log」から“ 192.168.0.1 ”という文字列を検索する場合を以下に示す。

関連 |, >, >>

```
$ grep "192.168.0.1" < access.log
```

<< 入力の終端を通知する

構文 **（コマンド） << 文字列**

入力が終わりであることをプログラムに通知するために使用する。<<の右側に来る文字列が終了を通知するワードとなる。catコマンドを利用して最後の文字列が“ End ”となるテキスト・ファイル「text.txt」を作成する場合を以下に示す。

```
$ cat << End > text.txt
> NikkeiBP
> NikkeiLinux
> End
```

& コマンドをバックグラウンドで実行

構文 **（コマンド） &**

コマンドの最後に付加することにより，そのコマンドをバックグラウンドで動かすことができる。例として，"sample"という文字列をfindコマンドで検索して「result.txt」に出力する処理をバックグラウンドで動かす場合を以下に示す。

関連 **bg，fg，jobs，kill，ps，stop，wait**

```
$ find / -name "sample" > result.txt &
```

alias コマンドの別名を登録

構文 **alias（登録する別名）='（コマンド名）'**

コマンドなどの別名を登録する。「（登録する別名）='（コマンド名）'」を省略すると現在登録している別名をすべて表示する。また，「='（コマンド名）'」を省略すると指定した別名の登録状況を表示する。例として，ディレクトリを中にあるファイルごとrmdirで削除（-rfオプションを付けてrmコマンドを実行）できるようにする場合を以下に示す。

関連 **unalias**

```
$ alias rmdir='rm -rf'
```

bg　ジョブをバックグラウンドの実行に切り替える

構文　**bg［コマンドのジョブ番号］**

指定した番号のジョブをバックグラウンドで実行する。ジョブ番号を省略した場合はカレント・ジョブ（+の付いたジョブ）をバックグラウンドの実行に切り替える。例えば“ ssh ”という文字列を検索しているジョブを探してバックグラウンドの実行に切り替える場合を以下に示す。

関連 **&, fg, jobs, kill, ps, stop, wait**

```
$ jobs
[1] -  Stopped                 emacs hoge.txt
[2]    Stopped                 man ls
[3]    Stopped                 find / -name "ssh"
[4] +  Stopped (signal)        w3m http://www.nikkeibp.co.jp/
$ bg 3
$ jobs
[1] -  Stopped                 emacs hoge.txt
[2]    Stopped                 man ls
[3]    Stopped                 find / -name "ssh" &
[4] +  Stopped (signal)        w3m http://www.nikkeibp.co.jp/
```

break　ループ構造から抜け出す

構文　**break（ループの段数）**

for, select, until, whileを用いたループ構造から脱出する。以下に示したような構文がシェル・スクリプト中にある場合，条件式1にある終了条件が満たされるまで，コマンド1を実行，加えて条件式2を満たした場合のみコマンド2を実行，コマンド1を実行，…というループを実行し続ける。例えば，コマンド2を実行した場合のみ，ループ全体から抜け出すようにするには，コマンド2とfiの間にbreakと書かれた行を追加すればよい。n重ループから一挙に脱出する場合は，break nとする。

```
while [条件式1]
do
  コマンド1
  if [条件式2]
  then
    コマンド2
  fi
done
```

builtin　シェル・コマンドを優先して実行する

構文　**builtin（コマンド名）**

スクリプト中で関数を定義した場合，同名のシェル・コマンドを優先して実行する。例えば，aliasという自作の関数内で，aliasを呼び出すと，再起呼び出し（自分自身を呼び出す）が起こって，終了しなくなる。このとき以下のように記述すると，シェル・コマンドであるaliasを呼び出すことができる。

関連 **enable，command**

```
builtin alias
```

case　条件分岐構文を作る

構文　**case 式 in 式）コマンド esac**

条件分岐構文を作る。以下の例のように，シェル変数の値が1のときは，2のときは，…という条件分岐を実現したい場合は，if文を組み合わせるより，case文を使ったほうが分かりやすいコードを記述できる。式の部分には正規表現を記述できるため，最後の条件を*としておくと，他のすべての条件に合致しないときにコマンドnを実行できる。

関連 **if**

```
case 式
in
  1)  コマンド1  ;;
  2)  コマンド2  ;;
  3)  コマンド3  ;;
   （中略）
  *)  コマンドn  ;;
esac
```

cd　ディレクトリの移動

構文　**cd［ディレクトリ名］**

指定したディレクトリに移動する。ディレクトリ名は絶対パス，相対パスのいずれも指定できる。ディレクトリ名を指定しないとホーム・ディレクトリに移動する。ディレクトリの表記方法には“ / ”（ルート・ディレクトリ），“ . ”（現在のディレクトリ），“ .. ”（親ディレクトリ），“ ~ / ”（ホーム・ディレクトリ）が使える。親ディレクトリに移動した場合を以下に示す。

関連 **ls，pwd**

```
$ pwd
/home/test
$ cd ..
$ pwd
/home
```

command　コマンドを優先して実行する

構文　**command（コマンド名）**

シェル・スクリプト中で関数を定義した場合，同名のシェル・コマンドを優先して実行する。builtinコマンドと逆の働きをする。以下に例を示す。

関連 **enable，builtin**

```
command ailas
```

continue　ループ内の特定の行を飛ばす

構文　**continue（ループの段数）**

for, select, until, whileを用いたループ構造の途中で，残りの行の実行を中止し，次のループを開始する。以下に示したような構文がシェル・スクリプト中にある場合，条件式1にある終了条件が満たされるまで，コマンド1を実行，加えて条件式2を満たした場合のみコマンド2を実行，コマンド3を実行，コマンド1を実行…というループを実行し続ける。例えば，コマンド2を実行した場合は，コマンド3を実行したくない場合，コマンド2とfiの間にcontinueと書かれた行を追加すればよい。n重ループ中で一挙に先頭から実行を開始するには，continue nとする。

```
while [条件式1]
do
  コマンド1
  if [条件式2]
  then
    コマンド2
  fi
  コマンド3
done
```

dirs　記録しているディレクトリを表示する

構文　**dirs（オプション）**

ディレクトリ・スタックに入っているディレクトリ・リストを表示する。以下に，ディレクトリ・スタックをクリアする例を示す。

関連 **popd，pushed**

```
$ dirs -c
```

enable　シェルの組み込みコマンドを有効化

構文　`enable （オプション）（コマンド名）`

シェルの組み込みコマンドを無効にする。再び有効にすることもできる。以下に，aliasコマンドを無効にする例を示す。この場合でも既存のalias設定は有効になっている。

関連 builtin，command

```
$ enable -n alias
```

表1　enableの主なオプション

オプション	機能
-a	有効，無効の表示を付けてシェル・コマンドの一覧を表示する
-n	コマンドを無効にする
-p	現在有効なシェル・コマンドの一覧を表示する。nオプションと組み合わせると，現在無効なシェル・コマンドだけを一覧表示する

exit　プロセスを終了，ログアウト

構文　`exit`

漢字コンソールなどのアプリケーションやシェルなどのプロセスを終了するときに使用する。ほかのプロセスがないときに入力するとログアウトする。例えば，シェル上で漢字コンソールを実行してからシェルを終了するまでの処理を以下に示す。

関連 login，su

```
$ kon  ←漢字コンソールの開始
  •
  •
$ exit  ←漢字コンソールの終了
$ exit  ←シェルの終了
```

export　変数を大域変数として追加する

構文　`export （変数名）`

シェル・スクリプト中などでシェル変数の値を設定しても，スクリプトが終了してしまうと，シェル変数の値が元に戻ってしまう。スクリプト中でPATH変数を設定後，以下のように実行すると終了後も新しいPATH設定が有効になる。

```
export PATH
```

fg　ジョブをフォアグラウンドの実行に切り替える

構文　**fg［コマンドのジョブ番号］**

バックグラウンドで実行しているジョブをフォアグラウンドに切り替える。ジョブ番号を省略した場合はカレント・ジョブ（+の付いたジョブ）を切り替える。例えば，“ html ”という文字列を検索しているバックグラウンドのジョブをフォアグラウンドの実行に切り替える場合を以下に示す。

関連 **&, bg, jobs, kill, ps, stop, wait**

```
$ jobs
[1] -  Stopped                 vi sample.txt
[2] +  Stopped (signal)        lynx http://linux.nikkeibp.co.jp/
[3]    Stopped                 find / -name "html" &
$ fg 3
```

for　ループ制御構造を作る

構文　**for 変数名 in リスト名 do コマンド done**

select, until, whileなどと並び，ループ制御構造を作る。シェル・スクリプトで用いるforは，C言語などのfor文とは機能が異なり，実行回数や実行する条件を指定できない。例えば，コマンドライン引数が残っている間はループを回すという使い方に向く。以下の例では，calコマンドの出力結果から日付などの単語を切り出す。

関連 **select, until, while**

```
for loop in `cal`
do
  echo $loop
done
```

history　コマンドの実行履歴を表示する

構文　**history（オプション）**

コマンドの実行履歴を表示するほか，ファイルから履歴を読み出したり，書き込んだりできる。以下の例では，履歴をすべて消去している。

```
$ history -c
```

表1　historyの主なオプション

オプション	機能
-c	履歴を消去する
-d[履歴番号]	履歴のうち，1カラム目に表示された番号の履歴を削除する
-r[履歴ファイル]	履歴ファイルを読み込み，現在の履歴と置き換える
-w[履歴ファイル]	現在の履歴を履歴ファイルに上書きする

if　条件分岐構造を作る

構文　**if 条件 then コマンド fi**

条件分岐構造を作る最も基本的なコマンド。if 条件式 then コマンド fi, if 条件式 then コマンド1 else コマンド2 fi, if 条件式1 then コマンド1 elif 条件式2 then コマンド2 else コマンド3 fi, という3つの構文がある。以下の例は, 3番目の例である。この場合, 条件式1を満たした場合はコマンド1を, 条件式1を満たしていないが, 条件式2を満たした場合はコマンド2を, 条件式1, 条件式2とも満たさなかった場合はコマンド3を実行する。

関連 case

```
if [条件式1]
then
  コマンド1
elif [条件式2]
then
  コマンド2
else
  コマンド3
fi
```

jobs　実行中のジョブを表示

構文　**jobs（オプション）［プロセスのジョブ番号］**

ジョブの稼動状況を表示する。ジョブ番号に続き, ジョブの状態をRunning(実行中), Stopped(停止中), Done(終了)と表示する。状態の後に(tty output)や(tty input)とあるのは, それぞれ画面に文字を表示する直前と入力待ちを表す。ジョブ番号の直後には, カレント・ジョブを表す" + "や前のジョブを表す" - "が付く。例として, 現在のジョブをプロセスID付き(-lオプション)で表示する場合を以下に示す。

関連 &, bg, fg, kill, ps, stop, wait

```
$ jobs -l
[1] -  4325 Stopped (tty output)
emacs sample.txt
[2] +  4372 Running
find / -name "html" &
```

popd　スタックに保存したディレクトリに戻る

構文　**popd**

pushdコマンドでディレクトリ・スタックに保存したディレクトリに復帰する。ディレクトリ・スタックとは, カレント・ディレクトリとして, これまで使っていたディレクトリを記憶しておくメモリー・メモリー領域のこと。以下の例では, スタックの先頭にあるディレクトリをスタックから削除する。

関連 cd, dirs, pushd

```
$ popd +1
```

pushd　カレント・ディレクトリをスタックに保存して移動

構文　**pushd ［移動先のディレクトリ］**

現在のディレクトリをスタックに保存して，指定したディレクトリに移動する。スタックに保存したディレクトリにはpopdコマンドで戻ることができる。カレント・ディレクトリをスタックに追加する場合には，pushd .ではなく，pushd `pwd`を用いる。以下に，ディレクトリ移動の例を示す。

関連 **cd，dirs，popd**

```
$ pushd /home/user1/test
/home/user1/test ~
$ pushd /usr/local/bin
/usr/local/bin /home/user1/test ~
$ popd
/home/user1/test ~
$ popd
```

read　読み込んだファイルを解釈

構文　**read （シェル変数）**

ファイルから入力行を読み出して解釈する。例のような内容がシェル・スクリプト中にあった場合，入力を読み出し，単語に分離する。その後，シェル変数first，second，thirdに入力する。例えばabcdefと入力すると，firstにはa，secondにはb，thirdにはcdefが入る。

```
read first second third
```

select　ループ制御構造を作る

構文　**select 変数 in リスト do コマンド done**

メニューを作成するのに使えるコマンド。リストに指定された文字列を番号付きで表示し，ユーザーからの番号入力を待つ。番号を指定すると，番号自体は組み込み変数REPLYに，選択された文字列が変数に入る。in リストを省略することもできる。その後，コマンドを実行する。変数を参照するコマンドを記述する場合が多い。

関連 **for，until，while**

```
select 変数 in リスト
do
  コマンド
done
```

set　シェルのオプションを設定

構文　**set （オプション）**

シェルにオプションを設定する。以下の例の場合は，CTRLキーを押しながらDキーを押しても，シェルを終了しないように設定している。「-」の代わりに「+」を用いると意味が逆になる。

関連 **enable**

```
$ set -o ignoreeof
```

表1　setの主なオプション

set -oのオプション	bashのオプション	機能
noclobber	-C	既存のファイルに対する出力リダイレクトを禁じる。「>|」を用いればリダイレクトできる
noexec	-n	コマンドを実行せず，構文エラーだけを確認する
noglob	-f	ファイル名用のワイルド・カードによる展開（「*」や「?」）を無効にする
nounset	-u	未定義の変数を参照するとエラーを表示する
verbose	-v	コマンドを実行する前にコマンド名などを表示する
xtrace	-x	-x コマンド行の展開処理を表示する

shift　引数を1つずらす

構文　**shift （オプション）**

シェル・スクリプトに与えられた引数を1つずつ処理したい場合に役立つ。例えば，以下のようなシェル・スクリプトを実行し，引数に「a b c d e」を与えたとすると，aからeまでを1行に1文字ずつ出力することができる。

```
while [ "$1" != "" ]
do
  echo $1
  shift
done
```

suspend　シェルを実行停止

構文　**suspend （オプション）**

現在のシェルの実行を停止する。シェル内部からはCtrl+Zなどキー操作ではシェル自体を中断できないために用いる。以下の例では，一時的にrootから元のユーザーに戻り，fgコマンドでrootに復帰している。

関連 **exit**

```
$ su
# suspend
$ fg
#
```

test　条件式の真偽を判定

構文　`test （式）`

以下の例では，直前に実行したコマンドが正常終了したかどうかを確認している。この行をifやwhileの条件式の部分に埋め込むと応用が効く。

```
test $? -eq 0
```

times　コマンドの実行時間を測定

構文　`times （コマンド）`

コマンドの実行時間を測定し，表示する。以下の例では，lsコマンドの実行に要した時間が分かる。1行目左にシェルが使用したユーザー時間，右に同システム時間が表示され，2行目左にlsコマンドが使用したユーザー時間，右に同じくシステム時間が表示される。

```
$ times ls
```

trap　システム割り込み時の処理を設定

構文　`trap （処理内容）（シグナル番号）`

システム割り込みが発生した場合の処理内容を設定する。以下の例では，CTRL+Cを押すと，"CTRL+C is pushed."というメッセージが出力される。設定できるシグナル番号は，0〜30である。

```
$ trap "echo CTRL+C is pushed." 2
```

表1　主なシグナル

シグナル番号	意味
0	シェルが終了
2	CTRL+Cなどによる割り込み発生
8	浮動小数点演算で例外発生
15	killコマンドなどによる終了
17	子プロセスが終了
21	バックグラウンド・プロセスがtty入力待ち

type　コマンドの型を検査

構文　**type（コマンド名）**

各種のコマンドの型を検査する。型にはalias（エイリアス），built-in（シェル・コマンド），function（関数），keyword（予約語）がある。以下の例の場合は，結果がaliasとなり，「ls is aliased to `ls --color=tty`」などと表示される。psやmkdirなどのコマンドを引数に取ると，パスを表示する。

```
$ type ls
```

ulimit　コマンドに割り当てる資源を制限

構文　**ulimit（オプション）（コマンド名）**

以下の例ではfindコマンドのCPU時間を20秒に限る。20秒を超えると「強制終了」と表示されて，実行が停止し，シェルに制御が戻る。

```
$ ulimit -t20 find
```

表1　ulimitの主なオプション

オプション	機能
-a	現在の制限の内容を表示
-f	シェルが作成するファイルのサイズ（Kバイト単位）
-s	スタック・サイズ（Kバイト単位）
-t	CPU時間（秒単位）
-u	ユーザーあたりのプロセス数

unalias　コマンドの別名を抹消

構文　**unalias［登録する別名］**

aliasで登録したコマンドの別名を削除する。rmdirのエイリアスを削除する場合を以下に示す。

関連 **alias**

```
$ alias
alias rmdir='rm -rf'
$ unalias rmdir
```

until　ループ制御構造を作る

構文　`until 条件式 do コマンド done`

条件式が満たされるまで，doとdoneの間に書かれたコードを繰り返し実行する。条件式の成立条件を正反対にすれば，whileコマンドと置き換え可能。

関連 for，select，while

```
until [条件式]
do
  コマンド
done
```

wait　プロセスおよびジョブの終了を待つ

構文　`wait［プロセスIDまたはジョブ番号］`

バックグラウンドで実行しているプロセスまたはジョブの終了を待つ。ジョブ番号を指定する場合は番号の前に%が必要。プロセスID/ジョブ番号で指定しなかった場合はバックグラウンドで実行しているすべてのプロセス/ジョブの終了を待つ。例えば「address.txt」というテキスト・ファイルから“ tokyo ”という文字列で始まる行をgrepコマンドで抜き出した後にソートするよう，waitコマンドで指示する場合を以下に示す。

関連 &，fg，bg，jobs，kill，ps，stop

```
$ grep "tokyo" address.txt &dt;
temp &
[1]  3682
$ wait 1 ; sort temp > list
```

while　ループ制御構造を作る

構文　`while 条件式 do コマンド done`

条件式が満たされるまで，doとdoneの間に書かれたコードを繰り返し実行する。条件式の成立条件を正反対にすれば，untilコマンドと置き換え可能。

関連 for，select，until

```
while [条件式]
do
  コマンド
done
```

basename　ファイル名からディレクトリや末尾の文字列を削除したものを返す

構文　`basename　［ファイル名］　［取り除く末尾の文字列］`

ファイル名だけ指定した場合はディレクトリを取り除いたものを戻り値として返す。また文字列を指定すると，ディレクトリと該当する文字列を削除したものを返す。例として，ディレクトリだけを取り除く場合とディレクトリと末尾の拡張子を取り除く場合を以下に示す。

```
$ basename /home/user1/test/sample.txt
sample.txt
$ basename /home/user1/test/sample.txt .txt
sample
```

chgrp　ファイルやディレクトリのグループを変更

構文　`chgrp（オプション）［変更後のグループ名］［ファイル名］`

ファイルやディレクトリのグループを変更する。ただし，変更できるのは対象となるファイルやディレクトリの所有者またはスーパーユーザーのみ。例えば，「test」というディレクトリとその中のファイルのグループをすべて変更する場合を以下に示す。

関連 `chmod，chown，ls`

```
$ chgrp -R user-grp test
```

表1　chgrpの主なオプション

オプション	機能
-c，--changes	グループが変更されたファイルのみ詳細に表示する
-f，--silent，--quiet	所有者を変更できなかったファイルについて，エラー・メッセージを表示しない
-v，--verbose	グループの変更を詳細に表示する
-R，--recursive	ディレクトリとその中身のグループを再帰的に変更する

chmod　ファイルやディレクトリのアクセス権を変更

構文　**chmod（オプション）［変更後のアクセス権］［ファイル名］**

ファイルやディレクトリのアクセス権限を変更する。所有者，グループ，その他のユーザーに対して，それぞれ読み込み，書き込み，実行権限を与えられる。ファイルに実行権限を与えるとコマンドやスクリプトなどとして使用できる。

関連 chgrp，chown，ls

表1　chmodの主なオプション

オプション	機能
-c，--changes	アクセス権グループが変更されたファイルのみ詳細に表示する
-f，--silent，--quiet	所有者を変更できなかったファイルについて，エラー・メッセージを表示しない
-v，--verbose	アクセス権の変更を詳細に表示する
-R，--recursive	ディレクトリとその中身のグループを再帰的に変更する

なお変更後のアクセス権は以下のように記述する。アクセス権はカンマで区切ることで複数記述できる。最初のユーザーの設定項目を省略したときは，すべてのユーザーと同等の意味になる。

［権限を与えるユーザー］［権限の設定/付加/削除の指定］［権限または特殊モード］

各項目の記述方法を**表2**に示す。

表2　権限の記述方法

権限を与えるユーザーの表記	
u	所有者の権限
g	グループの権限
o	その他のユーザーの権限
a	すべての権限
権限の設定/付加/削除を指定する表記	
+	後に記述した権限を付加する
-	後に記述した権限を削除する
=	後に記述した権限にする
与える権限および特殊モードの表記	
r	読み込み権限
w	書き込み権限
x	実行権限
s	セットID
t	スティッキー・ビット

表3　権限の数字による記述方法

数字	権限
0	---
1	--x
2	-w-
3	-wx
4	r--
5	r-x
6	rw-
7	rwx

権限の設定は**表3**のように3たけの8進数を使用して指定することもできる。左から順に所有者，グループ，その他のユーザーの権限を表す。

このほか, chmodは読み込み・書き込み・実行以　　の指定もできる。外に**表4**に示すスティッキー・ビットおよびセットID

表4　特殊なモード

権限の表示	意味
drwxrwxrwt	スティッキー・ビット。指定したディレクトリ以下のファイル名の変更やファイルの削除は所有者のみ実行できる
-rwsr-xr-x	セット・ユーザーID。所有者以外のユーザーが実行する際，所有者の権限で実行する
-rwxrwsr-x	セット・グループID。所有者以外のユーザーが実行する際，グループの権限で実行する

例えば「sample」というファイル, または「sample-　　下に示す。dir」というディレクトリの権限を変更する場合を以

```
$ ls -l
-rw-r--r--   1 user1     user-grp      216 4月  4 12:36 sample
$ chmod a+w sample     ←すべてのユーザーに書き込み権限を与える
$ ls -l
-rw-rw-rw-   1 user1     user-grp      216 4月  4 12:36 sample
$ chmod g+x,o= sample  ←グループに実行権限を与え，その他のユーザーのアクセスはすべて禁止する
$ ls -l
-rwxrwx---   1 user1     user-grp      216 4月  4 12:36 sample
$ chmod 444 sample     ←数字表記ですべてのユーザーに読み込み権限だけを与える
$ ls -l
-r--r--r--   1 user1     user-grp      216 4月  4 12:3 sample
$ chmod 1777 sample-dir        ←スティッキービットを付加する
$ ls -l
drxwrxwrxt   1 user1     user-grp      324 4月  4 12:39 sample-dir
$ chmod u+s sample     ←ファイルにセット・ユーザーIDを付加する
$ ls -l
-rwsr-xr-x   1 user1     user-grp      216 4月  4 12:36 sample
$ chmod g+s sample     ←ファイルにセット・グループIDを付加する
$ ls -l
-rwxrwsr-x   1 user1     user-grp      216 4月  4 12:36 sample
```

chown　ファイルやディレクトリの所有者を変更

構文　chown（オプション）［変更後の所有者］［ファイル名］

スーパーユーザーだけが実行できる，ファイルやディレクトリの所有者を変更するコマンド。所有者の後ろに“:（または.）［グループ名］”を付けるとグループも同時に変更できる。例えば「sample-dir」というディレクトリ配下のファイルの所有者を変更する場合を以下に示す。

関連 chmod，chgrp，ls

```
# chown -R user1 sample_dir
```

表1　chownの主なオプション

オプション	機能
-c，--changes	所有者が変更されたファイルのみ詳細に表示する
-f，--silent，--quiet	所有者を変更できなかったファイルについて，エラー・メッセージを表示しない
-v，--verbose	所有者の変更を詳細に表示する
-R，--recursive	ディレクトリとその中身の所有者を再帰的に変更する

cp　ファイルやディレクトリをコピー

構文　cp（オプション）［コピー元のファイル/ディレクトリ名］［コピー先のファイル/ディレクトリ名］

ファイルやディレクトリをコピーする。コピー元のファイル/ディレクトリ名にワイルドカードを使うこともできる。使用例を以下に示す。

関連 ls，mv，rm

```
$ cp sample1 sample2    ←ファイルを違う名前でコピーする
$ cp -r sample_dir1 sample_dir2  ←ディレクトリの内容をすべてコピーする
```

表1　cpの主なオプション

-オプション	機能
-a，--archive	できる限り属性やディレクトリ構造を保持してコピーする
-b，--backup	上書きや削除されるファイルについて，バックアップ・ファイルを作成する
-d，--no-dereference	シンボリック・リンクをコピーするときは，シンボリック・リンクの実体をコピーする
-f，--force	コピー先に同名ファイルがあるとき警告せずに上書きする
-i，--interactive	上書きされるファイルがあるときは問い合わせる
-l，--link	ファイルをコピーするときは代わりにハード・リンクを作成する
-P，--parents	指定したディレクトリを付けてコピーする
-p，--preserve	オーナー，グループ，パーミッション，タイムスタンプを保持したままコピーする
-r	ディレクトリを再帰的にコピーする
-s，--symbolic-link	ディレクトリ以外のファイルをコピーする際，シンボリック・リンクを作成する
-u，--update	同名のファイルが存在する場合，タイムスタンプを比較して同じまたは新しいときはコピーしない
-v，--verbose	コピーの前にファイル名を表示する
-x，--one-file-system	コピーする際，異なったファイル・システムのサブ・ディレクトリはコピーしない
-R，--recursive	ディレクトリを再帰的にコピーする

dd　ファイルを変換してコピー

構文　**dd（オプション）**

ファイルを変換をしながら指定したブロック・サイズでコピーする。また，デバイスからデバイスへ直接コピーする際にも使用する。例として，Linuxカーネルをフロッピー・ディスクにコピーする場合を以下に示す。

```
$ dd if=/boot/vmlinuz of=/dev/fd0
```

表1　ddの主なオプション

オプション	機能
if=[ファイル名]	読み込むファイルを指定する。指定がない場合は標準入力を表す
of=[ファイル名]	出力ファイルを指定する。指定がない場合は標準出力を表す
ibs=[バイト数]	一度に指定したバイトのブロックを読み込む
obs=[バイト数]	一度に指定したバイトのブロックを書き込む
bs=[バイト数]	一度に指定したバイトのブロックを読み書きする
cbs=[バイト数]	一度に指定したバイトのブロックを変換する
skip=[ブロック数]	入力ファイルの先頭から指定したブロックをスキップする
seek=[ブロック数]	出力ファイル中の指定したブロックをスキップする
count=[ブロック数]	入力から出力へ指定したブロックをコピーする
conv=[コード]	コード変換を実行する。指定できるコードは表2に示す

表2　convオプションで指定できるコード

コード	意味
ascii	EBCDICをASCIIに変換
ebcdic	ASCIIからEBCDICに変換
ibm	ASCIIから別のEBCDICに変換
block	改行で区切られたレコードをcbsで指定したサイズに合わせる。不足分は空白が使われる
unblock	cbsで指定したブロックの末尾の連続した空白を改行に変換
lcase	大文字を小文字に変換
ucase	小文字を大文字に変換
swab	奇数と偶数バイトを入れ替える
noerror	読み込みエラーが発生したとしても継続する
notrunc	出力ファイルを丸めない
sync	isbで指定したブロック数に合わせる。不足分はNULLが使われる

df　ディスク・ドライブの使用量を表示

構文　df （オプション）［ファイル名］

ファイル・システムの使用状況を表示する。標準ではファイル・システム名，全容量，使用量，残量，使用割合，マウント・ポイントの順に表示する。なおオプションの後にファイル名を記述すると，指定したファイルが属しているディスク・ドライブの容量を表示する。使用例を以下に示す。 関連 du, ls

```
ファイルシステム    1k-ブロック    使用済    使用可    使用率%  マウント場所
/dev/hdc9           10082852       3179592   6391072   34%      /
/dev/hdc1           36566          7320      27358     22%      /boot
/dev/hdc5           8066164        423852    7232564   6%       /home
```

表1　dfの主なオプション

オプション	機能
-a，--all	空のファイル・システムも含めたすべてのファイル・システムの情報を表示する
-i，--inodes	iノードの使用量を表示する
-k，--kilobytes	容量をキロ・バイト単位で表示する
-t，--type=fstype	表示するファイル・システムを指定する
-x，--exclude-type=fstype	表示しないファイル・システムを指定する
-h	容量を適当な単位で表示する

du　ディレクトリ内のファイル容量を表示

構文　du （オプション）［ファイル/ディレクトリ名］

指定したファイルやディレクトリの使用容量を表示する。ファイル/ディレクトリ名を省略するとカレント・ディレクトリの容量を集計する。使用例を以下に示す。

関連 df, ls

```
$ du sample/
92      sample/.xvpics
968     sample/test
21292   sample
$ du -a  sample/
144     sample/photo1.png
28      sample/photo2-1.png
12      sample/photo2-2.png
72      sample/photo2.png
20      sample/photo3-1.png
8       sample/photo3-2.png
20      sample/photo4.png
308     sample
```

表1　duの主なオプション

オプション	機能
-a, --all	ディレクトリだけではなくファイルについても表示する
-c, --total	検索したすべての容量の総計を表示する
-b, --bytes	単位をバイトにする
-k, --kilobytes	単位をキロ・バイトにする
-m, --megabytes	単位をメガ・バイトにする
-l, --count-links	リンクも集計に含める
-s, --summarize	引数で指定した物のみの総計を表示する
-x, --one-file-system	違うファイルシステムの物は集計から外す
-D, --dereference-args	シンボリック・リンク・ファイルは元ファイルの容量を集計する
-S, --separate-dirs	個々のディレクトリでサブ・ディレクトリを含めずに表示する
-h	容量を適当な単位で表示する

find　ファイルやディレクトリを検索

構文　`find（オプション）［検索するディレクトリ］［判別式＋アクション］`

ファイルやディレクトリを判別式に基づいて検索し，指定されたアクションを実行する。複数の判別式を用いるときは判別式を演算子で結ぶ。主な判別式とアクション，演算子は**表1**に示す。なお，数字を判別式に用いるとき，数字の前に“＋”を付けると，n以上の数が検索対象となり，“－”を付けると以下の数が検索対象となる。使用例を以下に示す。

関連　`locate`，`ls`

```
$ find /home/user1/ -name "*.png"
/home/user1/sample.png
/home/user1/test/photo1.png
/home/user1/work/photo2.png
$ find ~/public_html -name "*.png" -and -size +50k -fprint pic_50kover_list
←50キロ・バイト以上のPNGファイルを検索，結果をリストに書き出す
$ find ~/ -name "~*" -ok rm {} ¥;      ←“～”を含むファイルを検索して削除する
```

表1　findの主な判別式，アクション，判別式の演算子

判別式	機能
-atime n	最後にアクセスされたのがn日前のファイルを検索する
-empty	空なファイルや中身のないディレクトリを検索する
-group［グループ名］	該当するグループ名のファイルを検索する（ID番号も指定可）
-mmin n	データが最後に修正されたのがn分前のファイルを検索する
-mtime n	データが最後に修正されたのが，n日前のファイルを検索する
-name［文字列］	指定した文字列のファイル名を検索する。ワイルド・カードも利用可能
-perm［アクセス権］	指定したアクセス権のファイルを検索する。アクセス権は8進数でも指定可
-type［ファイル・タイプ］	指定したファイル・タイプを検索する。ファイル・タイプの表記はディレクトリが“ d ”，通常ファイルが“ f ”，シンボリック・リンクが“ l ”
-user［ユーザー名］	該当するユーザー名のファイルを検索する（ID番号も指定可）
-size n［単位］	nのサイズのファイルを検索する。表示単位はバイトが“ c ”，キロ・バイトが“ k ”，ブロック（通常は1ブロック=512バイト）が“ b ”
アクション	**機能**
-exec［コマンド］¥;	検索後，指定したコマンドを実行する。このときコマンドの後ろに{}を付けると，検索結果をコマンドに引き渡す
-ok［コマンド］¥;	execと同様に検索後にコマンドを実行する。ただし，ユーザーに問い合わせる
-print	検索結果を標準出力する。このとき結果をフルパスで表示する
-fprint［ファイル名］	検索結果を指定したファイルに書き出す。同名のファイルがある場合は上書きをする
-ls	結果をファイルの詳細付きで表示する
演算子	**機能**
（判別式）	カッコのなかを優先する
［判別式1］-and［判別式2］	ANDで評価する
［判別式1］-or［判別式2］	ORで評価する
-not［判別式］	NOTで評価する

表2　findの主なオプション

オプション	機能
-depth	ディレクトリ本体の前にディレクトリの内容を評価する
-follow	シンボリック・リンクの参照先を検索する
-xdev	ほかのファイル・システムにあるディレクトリは探索しない

ln　ファイルやディレクトリにリンクを張る

構文　**ln（オプション）［リンク元のファイル］［リンク先のファイルまたはリンク先のファイルを作成するディレクトリ］**

ファイルやディレクトリのリンクを作成する。リンクにはハード・リンクとシンボリック・リンクが存在する。ハード・リンクは，目的のファイルと同じiノードを持つことで実現される。そのため，リンク元のファイルを削除してもリンク先のファイルの実体は存在し続ける。作成にはルート権限が必要である。ただし，リンク元とリンク先のパーティションやファイル・システムが異なる場合はハード・リンクを作成できない。

一方，シンボリック・リンクはリンク先のファイルやディレクトリのファイル名やパス名を保存することにより実現している。そのため，パーティションやファイル・システムが異なっていてもリンクを張れる。ただし，元のファイルを削除すると実体を持たないリンクになってしまう。作成には「-s」オプションを付ける。一般ユーザーの権限でも作成可能。

それぞれの作成例を以下に示す。

関連 **ls，alias**

```
$ ln -s /usr/local/bin
$ ls -F
bin@
# ln -d /home/user1/sample /home/user2/sample_link
# ls /home/user2
sample_link
```

表1　lnの主なオプション

オプション	機能
-b，--backup	上書きや削除されることになるファイルにはバックアップを作成する
-d，-F，--directory	ディレクトリのハード・リンクを作成する。スーパーユーザのみ作成可能
-f，--force	リンク先に同名ファイルがあるときも警告なく上書きする
-i，--interactive	上書きされるファイルがあるときは問い合わせる
-n，--no-dereference	リンク作成の際にリンク元に既存ディレクトリを指定した場合でも，ディレクトリ中にリンクを作成せず，ディレクトリとリンクを置き換える
-s，--symbolic	シンボリック・リンクを作成する
-v，--verbose	コピーの前にそのファイル名を表示する

locate　ファイルを高速に検索

構文　**locate（オプション）［検索パターン］**

ファイルを高速に検索できる。あらかじめ作成したデータベースを使って検索するため，findコマンドより高速に検索できる。データベースはスーパーユーザー権限でupdatedbコマンドを実行して作成する。なお，オプションに“ -d［データベースのパス］”または“ --database=［データベースのパス］”を付けることで，標準のファイル名データベースからパスに指定したデータベースに切り替えて検索を実行することもできる。

使用例を以下に示す。

関連 **updatedb，find，ls**

```
$ locate sample_file
/home/user1/work/sample_file
```

ls　ファイルやディレクトリの情報を表示

構文　**ls（オプション）［ディレクトリ/ファイル名］**

ディレクトリやファイルの情報を表示する。オプションの後の名前を省略したときはカレント・ディレクトリ以下のディレクトリ/ファイルの情報をすべて表示する。例えば，カレント・ディレクトリのすべての情報を詳細に表示した場合を以下に示す。

関連 **cd，pwd**

```
$ ls -al
合計 188
drwx------   21 user1 user1     4096  4月  1 12:16 ./
drwxr-xr-x    6 root  root      4096  3月  5 14:40 ../
-rw-------    1 user1 user1       97  4月  1 12:13 .Xauthority
-rw-r--r--    1 user1 user1     3824  3月  5 13:07 .Xdefaults
drwx------    2 user1 user1     4096  3月 13 12:56 .autosave/
-rw-------    1 user1 user1     9594  3月 28 16:24 .bash_history
-rw-r--r--    1 user1 user1       24  3月  5 13:07 .bash_logout
-rw-r--r--    1 user1 user1      247  3月  5 13:07 .bash_profile
-rw-r--r--    1 user1 user1     1363  3月  5 13:07 .bashrc
-rw-r--r--    1 user1 user1       72  3月  5 13:07 .elvisrc
```

表1　lsの主なオプション

オプション	機能
-a, --all	隠しファイル（ファイル名の先頭にピリオドがあるファイル）を含んだすべてのファイルを表示する
-b, --escape	ファイル名に表示できないコントロール・コードなどが使われている場合，8進数3けたで表示する
-c, --time=ctime, --time=status	ファイルのステータスを変更した時間順にソートして表示する。また，-lオプションと併用するとタイム・スタンプの代わりにステータス更新時間を表示する
-d, --directory	ディレクトリを，ディレクトリの内容ではなくファイルと同様に表示する
-f	ディレクトリ内の順番に沿って表示する
-i, --inode	インデックス番号をファイルの左側に表示する
-k, --kilobytes	ファイル・サイズをキロ・バイト単位で表示する
-l, --format=long, --format=verbose	ファイルの詳細（ファイル名，ファイル・タイプ，パーミッション，ハード・リンクの数，オーナー名，グループ名，ファイル・サイズ，タイムスタンプ）を表示する
-m, --format=commas	ファイルをカンマで区切って表示する
-n, --numeric-uid-gid	ユーザー/グループ名をIDで表示する
-o, --color, --colour, --color=yes	ファイル・タイプに応じて色分けして表示する
-p	ディレクトリの最後に/を付けて表示する
-q, --hide-control-chars	エスケープ・コードなど表示できない文字列が含まれていた場合は?に変換して表示する
-r, --reverse	逆順にソートして表示する
-s, --size	ファイル・サイズをファイルの左側にキロ・バイト単位で表示する
-t, --sort=time	タイム・スタンプ順にソートして表示する
-u, --time=atime, --time=access, --time=use	タイム・スタンプの代わりにアクセス時刻でソートして表示する。また，-lオプションと併用するとタイム・スタンプの代わりにアクセスした時刻を表示する
-x, --format=across, --format=horizontal	ファイルを水平方向に並べて表示する
-A	".."（親ディレクトリ）や"."（カレント・ディレクトリ）を表示しない
-B, --ignore-backups	ファイルの最後にチルダ（~）が付いているファイルを除いて表示する
-C, --format=vertical	ファイルを上下方向に並べて表示する
-F, --classify	ファイル名にファイル・タイプを表す記号（ディレクトリは"/"，実行可能ファイルは"*"，シンボリック・リンクは"@"，FIFOは"|"，ソケットは"="）を付けて表示する
-G, --no-group	-lオプションを付けて実行したときにグループを表示しない
-L, --dereference	シンボリック・リンクのリンク先のファイルを表示する
-N, --literal	エスケープ・コードを変換せずに表示する
-Q, --quote-name	ファイルをダブル・クォーテーションで囲んで表示する
-R, --recursive	ディレクトリ下を再帰的に表示する（ディレクトリ内をすべて表示）
-S, --sort=size	ファイル・サイズ順にソートする
-U, --sort=none	ディレクトリ内の順番に沿って表示する
-X, --sort=extension	拡張子でソートして表示する
-1, --format=single-column	1行に1ファイルずつ表示する
-w, --width cols	端末の幅をcolsで指定した文字数として表示する
-T, --tabsize cols	タブ幅をcolsで指定した文字数として表示する
-I, --ignore pattern	patternにマッチするファイルは表示しない
--full-time	タイム・スタンプを曜日，月，日付，時間，年すべて表示する
-h	ファイル・サイズを適当な単位（例えば1K，1M，1Gなど）で表示する

mkdir　ディレクトリを作成

構文　mkdir（オプション）［ディレクトリ名］

ディレクトリを作成する。-pオプションで目的のディレクトリに必要となる中間ディレクトリも作成してくれる。また，-mオプションを付けるとアクセス権も指定できる。使用例を以下に示す。 関連 rmdir,rm,ls

```
$ mkdir -p work/sample_dir      ←中間ディレクトリのworkも作成
$ mkdir -m 777 sample_dir2      ←アクセス権を“ rwxrwxrwx ”に指定して作成
```

mktemp　ランダムなファイル名の空ファイルを作成

構文　mktemp（オプション）［文字列］

ユニークなファイル名をした一時ファイルを作る。ファイル名の可変部分は「XXXXXX」で指定し，ファイル名の末尾部分でなければならない。作成したファイルのアクセス権は“ rw------- ”になる。なお，-dオプションをつけるとファイルではなくディレクトリを作成する。また，-qオプションを付けると警告や作成後のメッセージを表示しない。

使用例を以下に示す。

```
$ mktemp sample.XXXXXX
sample.tXuGJl
```

mv　ファイルやディレクトリを移動，または名前を変更

構文　mv（オプション）［コピー元のファイル/ディレクトリ名］［コピー先のファイルまたはコピー先のファイルを作成するディレクトリ］

ファイルやディレクトリの移動を行う。またファイル名を変更するときにも使用する。使用例を以下に示す。

関連 ls，cp，rm

```
$ mv sample /home/user1/sample_dir      ←ファイルを移動
$ mv sample2 sample3                    ←ファイル名を変更
```

表1　mvの主なオプション

オプション	機能
-b, --backup	上書きまたは削除されるファイルがある場合，バックアップを作成する
-f, --force	移動先に同名ファイルがあるときも警告せずに上書きをする
-i, --interactive	上書きされるファイルがあるときは問い合わせる
-u, --update	同名のファイルがある場合，タイム・スタンプを比較して同じまたは新しいときは移動しない
-v, --verbose	移動の前にファイル名を表示する
-x, --one-file-system	違うファイル・システムのサブ・ディレクトリには移動しない

od　バイナリ・ファイルの内容を閲覧

構文　**od（オプション）［ファイル名］**

バイナリ・ファイルの内容を表示する。例として，　下に示す。16進数で1行に16バイト形式で表示する場合を以

関連 cat

```
$ od -x -w16 -A x sample_binary
000000 020443 072457 071163 061057 067151 070057 071145 005154
000010 073412 064550 062554 024040 037074 020051 020173 063444
000020 074554 064160 022173 076461 036440 022040 020062 063151
000030 027440 027050 027056 024456 024072 025456 056051 027556
```

表1　odの主なオプション

オプション	機能
-A, --address-radix=［オフセットの基数］	表示するオフセットの基数を指定する。基数には以下の物を指定できる。dが10進数，oが8進数，xが16進数，nがオフセット非表示を表す
-j, --skip-bytes=［バイト数］	バイナリ・ファイルの指定したバイト数の場所から表示する。また，数字の後に，xを付加すると16進数，kを付加するとキロ・バイトに，mを付加するとメガ・バイトになる。数字の前に0を付加すると8進数になる
-t, --format=［表示タイプ］	表示タイプを指定する。表示タイプの表記は表2に示す
-w, --width［=バイト数］	1行に表示するバイト数を指定する
-a	文字の名前を出力する。-t aと同じ
-b	8進でバイトを出力する。-t oCと同じ
-c	ASCII文字またはバック・スラッシュ付きのエスケープ文字として出力する。-t cと同じ
-d	符号無し10進shortとして出力する。-t u2と同じ
-f	floatとして出力する。-t fFと同じ
-h	16進shortとして出力する。-t x2と同じ
-i	10進shortとして出力する。-t d2と同じ
-l	10進longとして出力する。-t d4と同じ
-o	8進shortとして出力する。-t o2と同じ
-x	16進shortとして出力する。-t x2と同じ

表2　-tオプションで指定する表示タイプの表記

オプション	機能
a	文字の名前
c	ASCII文字かバックスラッシュ付きのエスケープ文字
d	符号付き10進数
f	浮動小数点数
o	8進数
u	符号無し10進数
x	16進数

pwd　現在のディレクトリの場所を確認

構文　**pwd**

現在のディレクトリの場所を絶対パスで表示する。使用例を以下に示す。

関連 cd, ls

```
$ pwd
/home/user1
```

rm　ファイルやディレクトリを削除

構文　**rm（オプション）［ファイル名］**

指定したファイルを削除する。ファイル名にはワイルド・カードも使用できる。また，-rオプションを付けるとディレクトリごと削除できる。このとき，警告メッセージが表示されるが，メッセージを表示させたくない場合は-fオプションも同時に指定するとよい。例えば，カレント・ディレクトリのJPEGファイルをまとめて削除する場合と，ディレクトリをなかのファイルごと削除する場合を以下に示す。

関連 ls, cp, mv

```
$ rm *.jpg
$ rm -rf sample_dir
```

表1　rmの主なオプション

オプション	機能
-d, --directory	ディレクトリごと削除できる。スーパーユーザーのみ使用が可能
-f, --force	警告メッセージを表示しない
-i, --interactive	ファイルを削除してよいかを問い合わせる
-r, -R, --recursive	ディレクトリ内を再帰的に削除する
-v, --verbose	ファイルを削除する前にファイル名を表示する

rmdir　ディレクトリを削除

構文　`rmdir（オプション）［ディレクトリ名］`

ディレクトリを削除する。ただしディレクトリ内にファイルやディレクトリがあると削除できない。そのときは“ rm -rf ”を使用すると便利である。なお，実行時に-pまたは--parentsオプションを付けると削除するディレクトリの親ディレクトリが空のときは親ディレクトリも同時に削除する。

使用例を以下に示す。

関連 `mkdir, rm, ls`

```
$ rmdir sample_dir
```

split　ファイルを分割

構文　`split（オプション）［元ファイル名］［出力ファイルのベース名］`

ファイルを分割し，複数のファイルに出力する。出力はベース名にアルファベットを付けたファイル名になる。例として，テキスト・ファイルを5行ごとに切り出してファイルに出力する場合を以下に示す。

関連 `cat`

```
$ split -l 5 sample.txt out.
```

表1　splitの主なオプション

オプション	機能
-［行数］, -l［行数］, --lines=［行数］	元ファイルを指定した行ごとに区切り，出力ファイルに書き出す
-b［バイト数］, --bytes=［バイト数］	元ファイルを指定したバイト数で分割し，出力する。数字の後に記号を付加することにより単位を変えることができる。kを付加するとキロ・バイト，Mを付加するとメガ・バイト単位になる
-C［バイト数］, --line-bytes=［バイト数］	各行で区切り出力する。このとき各行で指定したバイト数を越える場合は次のファイルに書き出す

touch　ファイルのタイム・スタンプを変更

構文　`touch（オプション）［ファイル名］`

指定したファイルやディレクトリのタイム・スタンプを変更する。オプションを何も付けない場合は現在の時刻にタイプ・スタンプを書き換える。また，存在しないファイル名を指定した場合は新規に空ファイルが作成される。使用例を以下に示す。

関連 `ls`

```
$ touch sample                          ←現在の時刻に変更
$ touch -d "2002/4/7 00:00:00" sample   ←2002年4月7日の00:00:00に変更
```

表1　touchの主なオプション

オプション	機能
-a，--time=atime，--time=access，--time=use	アクセス時刻のみ変更する
-c，--no-create	ファイルが存在しない場合は新規に空ファイルを作成しない
-d，--date time	現在の時刻の代わりに指定した時刻にタイム・スタンプを書き換える。設定する時刻一般的な記法により指定できる（例："2000/1/1 00:00:00"，"Sun Apr 7 00:00:00 JST 2002"など）
-m，--time=mtime，--time=modify	修正時刻だけを変更する
-r，--reference［ファイル名］	指定したファイルと同じタイム・スタンプに書き換える
-t MMDDhhmm[[CC]YY][.ss]	引数で指定した時間にタイム・スタンプを変更する。MMは月，DDは日，hhは時，mmは分，CCは西暦年の上2けた，YYは西暦年の下2けた，ssは秒を表す

updatedb　locate用のファイル・データベースを更新

構文　`updatedb`

locateに使用するファイル・データベースを作成する。ただし実行はスーパーユーザーの権限で行う必要がある。使用例を以下に示す。

関連 `locate，find`

```
$ updatedb
```

clock　ハードウエア内部のクロックの読み出し，設定

構文　clock（オプション）

ハードウエア・クロック（CMOSクロック）の読み出しや設定を行う。使用例を以下に示す。

```
$ clock -r
2002年04月08日 00時00分00秒  0.123456秒
```

表1　clockの主なオプション

オプション	機能
-u	世界標準時（UTC）として扱う
-r	CMOSクロックを表示する
-w	CMOSクロックにシステム時刻を書き込む
-s	CMOSクロックからシステム時刻を設定する

date　日付・時間を表示・設定する

構文　date（オプション）

現在の時間を表示したり時間設定を行う。時間設定はスーパーユーザーの権限が必要。使用例を以下に示す。

関連 cal

```
現在の日付と時間を表示
$ date                          ←現在の日付と時間を表示
月  4月  8 00:00:00 JST 2002
$ date 010100002003.00          ←別方式で日付を表示する
水  1月  1 00:00:00 JST 2003
$ date +"%Y/%m/%d %p %I:%M:%S"  ←指定した形式での表示する
2002/04/08 午前 00:00:00
# date -s "04/08 0:00 2002"     ←指定した時刻にセットする
```

表1　dateの主なオプション

オプション	機能
-u	グリニッジ標準時を使用する
-s [時刻]	時間を設定する。時刻は一般的に分かる形であれば形式を問わない（例：2002/04/08 00:00:00）
+[表示形式]	表示する形式を指定する。形式の表記は表2を参照
MMDDHHmm[[CC]YY][.ss]	日付を指定して別のフォーマットで表示する。MMは月，DDは日，HHは時，mmは分，CCは西暦の上2けた，YYは西暦の下2けた，ssは秒を表す

表2　+formatで指定できる形式

文字列	指定した文字列を表示する
%H	時(00～23)
%I	時(01～12)
%k	時(0～23)
%l	時(1～12)
%M	分(00～59)
%p	AM あるいは PM のロケール
%r	12時間形式の時刻 (hh:mm:ss [AP]M)
%s	1970-01-01 00:00:00 UTC からの秒数
%S	秒(00～61)
%T	24時間形式の時刻 (hh:mm:ss)
%a	ロケールによる省略形の曜日の名前(Sun～Sat)
%A	ロケールによる完全に表記した曜日の名前(Sunday～Saturday)
%b	ロケールによる省略形の月の名前(Jan～Dec)
%B	ロケールによる完全に表記した月の名前(January～December)
%c	ロケールによる日付と時刻(Sat Nov 04 12:02:33 EST 1989)
%d	日(月内通算日数)(01～31)
%D	日付(mm/dd/yy)
%j	年内通算日数(001～366)
%m	月(01～12)
%w	週のうちの曜日(0～6)で0が日曜日に対応
%x	ロケールによる日付の表現(mm/dd/yyz)
%y	年の最後の2つの数字(00～99)
%Y	年(1970～)

fastboot　システムを高速に再起動

構文　`fastboot`

すぐに再起動をする。このときファイル・システムのチェックを行わない。「shutdown -r -q -f now」と同じ意味。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 `shutdown, halt, fasthalt, reboot`

```
# fastboot
```

fasthalt　システムを高速にシャットダウン

構文　`fasthalt`

すぐにシャットダウンをする。このときファイル・システムのチェックを行わない。「shutdown -h -q -f now」と同じ意味。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 `shutdown,halt,reboot,fastboot`

```
# fasthalt
```

finger　ユーザー情報を表示

構文　`finger（オプション）［ユーザー名］［ユーザー名@ホスト名］`

ユーザの情報を表示する。ユーザー名やホスト名を省略した場合は現在ログインしているユーザーを表示する。なお，現在のホストでは，セキュリティの強化のため外部からのfingerコマンドを通さないように設定していることが多い。使用例を以下に示す。

関連 `who，w`

```
$ finger user1
Login: user1                            Name: (sampl_user)
Directory: /home/user1                  Shell: /bin/bash
On since Mon Apr  8 01:55 (JST) on pts/1 from hoge.nikkeibp.co.jp
No mail.
No Plan.
Company : NikkeiBP
Address : xx-xx-xx Abc-cyo， Chiyoda-ku， Tokyo， Japan
TEL     : +81-3-5xxx-xxxx
FAX     : +81-3-5xxx-yyyy
```

表1　fingerの主なオプション

オプション	機能
-s	ユーザのログイン名，実名，端末名と，その端末への書き込みの可否，アイドル時間，ログイン時間，およびオフィスの場所と電話番号の情報を表示する
-l	-sオプションの情報に加え，ホーム・ディレクトリ，電話番号，ログイン・シェル，メールの状態，さらにホーム・ディレクトリに“.plan”や“.project”というファイルがあれば，その中身を表示する
-p	-lオプションのうち，“.plan”および“.project”を除いた他の情報を表示する
-m	実名とのマッチングを抑止する

free　メモリーの使用状況を表示

構文　`free（オプション）`

現在のメモリーの使用状況を知ることができる。/proc/meminfoを整形して表示するコマンド。オプションに-bを付けるとバイト，-kを付けるとキロ・バイト，-mを付けるとメガ・バイト単位で表示する（オプションを指定しないとキロ・バイトで表示）。なお，-tオプションを付けるとトータルを表示する。使用例を以下に示す。

関連 `ps,top`

```
$ free -mt
             total       used       free     shared    buffers     cached
Mem:           124        115          8          0         23         22
-/+ buffers/cache:         69         54
Swap:          101         13         88
Total:         226        128         97
```

groupadd　グループを追加

構文　`groupadd（オプション）［グループ名］`

新しいグループを作成する。作成したグループ名は/etc/groupファイルに追加される。なお，実行時に-gオプションでグループIDも指定できる。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 `groupdel,groupmod,id,vigr`

```
# groupadd new_grp
```

groupdel　グループを削除

構文　`groupdel［グループ名］`

グループを削除する。ユーザーのプライマリ・グループは即存している限り削除することはできないため，ユーザー・アカウントを削除する必要がある（ただし，ユーザを削除すると同時にプライマリ・グループも削除されるので改めてgroupdelをする必要はない）。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 `groupadd, groupmod, id, vigr`

```
# groupdel sample_grp
```

groupmod　グループ情報を変更

構文　**groupmod（オプション）［新しいグループ名］［グループ名］**

グループ情報を変更する。-gオプションでグループIDが変更できるほか，新しいグループ名を指定するとグループ名も変更できる。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 **groupadd，groupdel，id**

```
# groupmod -g 530 newname_grp sample_grp
```

halt　システムをすぐにシャットダウン

構文　**halt（オプション）**

すぐにシャットダウンする。「shutdown -h -q now」と同じ意味。このコマンドを実行できるのはスーパー・ユーザーのみ。使用例を以下に示す。

関連 **shutdown，fasthalt，reboot，fastboot**

```
# halt
```

id　ユーザーやグループIDを表示

構文　**id（オプション）［ユーザー名］**

指定したユーザーやグループのIDを表示することができる。なにも指定しないと現在のユーザーの情報が表示される。使用例を以下に示す。

関連 **groupadd，groupdel，groupmod，useradd，userdel，usermod**

```
$ id
uid=502(user1) gid=100(users) groups=100(users)
```

表1　idの主なオプション

グループ	機能
-g，--group	グループID のみを表示する
-G，--groups	属しているグループのみ表示する
-n，--name	IDの数字を表示する代わりにユーザー名やグループ名を表示する。同時に [-u]，[-g]，または[-G]を指定する必要がある
-r，--real	実際のユーザー及びグループIDを表示する。同時に[-u],[-g]，または[-G] を指定する必要がある
-u，--user	ユーザーIDのみを表示する

last　最後にログインしたユーザーを表示

構文　last（オプション）［参照するユーザー］［参照するターミナル］

ユーザーの最近のログを調べることができる。ユーザーやターミナルを指定しない場合は過去のログイン状況を一覧表示する。使用例を以下に示す。

関連 lastlog

```
$ last
user1 pts/3        linux.nikkeibp Mon Apr  8 16:38   still logged in
user2 pts/1        192.168.0.5    Mon Apr  8 07:00 - 07:10  (00:10)
```

表1　lastの主なオプション

オプション	機能
-［行数］, -n［行数］	表示行数を指定する
-R	ホスト名を表示しない
-a	ホスト名を最後に表示する
-d	リモート・ログインに対してホスト名を表示する。IP番号はリモート・ホスト名に変換される
-x	シャットダウンやラン・レベル変更のログも同時に表示する

lastlog　ユーザーの最後にログインした日付を表示

構文　lastlog（オプション）

ユーザーの最後にログインした日付を表示する。オプションに“-u［ユーザー名またはユーザーID］”を付けると指定したユーザーだけを表示する。また“-t［日数］”を付けると指定した日数以内の最終ログイン日付を表示する。何もオプションを設定しない場合は，登録されているユーザーすべてが対象となる。一度もログインしたことのないユーザーについては“**Never logged in**”と表示される。使用例を以下に示す。

関連 last

```
$ lastlog
Username         Port     From             Latest
root             tty1                      金  2月  1 07:57:32 +0900 2002
bin                                        **Never logged in**
  •
  •
user1            pts/1    linux.nikkeibp.co.jp 月  4月  8 12:05:38 +0900 2002
pop                                        **Never logged in**
```

login　ログインする

構文　**login**

現在のセッションを終了し，新しいシステムにログインする。もし，/etc/nologinファイルが存在するとroot以外のログインができなくなる。使用例を以下に示す。

関連 **exit, su**

```
$ login
ログイン：
```

passwd　ユーザーのパスワードを変更

構文　**passwd［ユーザー名］**

ユーザーのログイン・パスワードを変更する。ただし他人のパスワードを変更できるのはスーパーユーザーのみ。ユーザー名を指定しない場合はカレント・ユーザーのパスワードを変更する。使用例を以下に示す。

関連 **pwconv**

```
$ passwd
Changing password for user1
(current) UNIX password: ********
New UNIX password: *********
Retype new UNIX password: *********
passwd: all authentication tokens updated successfully
```

pwconv　shadowパスワードに移行

構文　**pwconv**

古いパスワード・ファイル情報をshadowパスワード方式に変更する。新しいパスワード・ファイルはnpasswd，シャドー・パスワードはnshadowという名前で保存される。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 **passwd**

```
# pwconv
```

reboot　システムをすぐに再起動

構文　**reboot**

すぐに再起動をする。「shutdown -r -q now」と同じ意味。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 shutdown,halt,fasthalt,fastboot

```
# reboot
```

shutdown　システムをシャットダウン・再起動

構文　**shutdown（オプション）**

シャットダウン・再起動を行う。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 halt,fasthalt,reboot,fastboot

```
# shutdown -r now
```

表1　shutdownの主なオプション

オプション	機能
-h	システムをシャットダウンする
-r	システムを再起動する
-f	高速指定。再起動の際，ファイル・システムのチェックを行わない
-q	メッセージを表示しない
-s	シングル・ユーザー・モードで再起動する
now	すぐにシステムのシャットダウン・再起動を行う
[時刻]	指定した時間にシステムのシャットダウン・再起動を行う
+[分]	現在より指定時間後にシステムのシャットダウン・再起動を行う。単位は分
[文字列]	シャットダウン・再起動時にユーザーに送るメッセージを指定する

su　ユーザー・アカウントを切り替える，管理権限を持つ

構文　su（オプション）［ユーザー名］

ユーザー・アカウントを切り替えるのに使用する。ユーザー名を指定しない場合は管理者権限に切り替えることができる。使用例を以下に示す。

関連 exit，id

```
$ su user1          ←user1に切り替え
password:*******
$ su                ←管理権限を持つ
password:*******
```

表1　suの主なオプション

オプション	機能
-f	初期設定ファイル(.cshrc)を実行しない
-l，-	ログイン・シェルを使用してユーザを切り替える
-m，-p	環境変数“ HOME ”,“ USER ”,“ LOGNAME ”,“ SHELL ”を変更しない
-c［コマンド］	ユーザーの切り替え後にコマンドを実行する
-s［シェル］	指定したシェルを実行する

uname　システム情報を表示

構文　uname（オプション）

コンピュータのシステム情報を表示する。オプションを何も指定しない場合は，-sオプションが付いた状態と同様。使用例を以下に示す。

関連 hostname

```
$ uname -a
Linux linux.localnet 2.4.17 #3 金 2月 1 08:35:08 JST 2002 i686 unknown
```

表1　unameの主なオプション

オプション	機能
-m，--machine	コンピュータ（ハードウェア）の種類を出力する
-n，--nodename	ネットワークにおけるホスト名を出力する
-r，--release	OSのリリース番号を出力する
-s，--sysname	OSの名称を出力する
-v	OSのバージョンを出力する
-a，--all	上記の情報をすべて出力する

useradd　ユーザーを追加

構文　useradd（オプション）［ユーザー名］

ホストにログインできるユーザー・アカウントを作成する。ホーム・ディレクトリを指定しない場合は，/home/［ユーザー名］に作られる。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 userdel，usermod，id，passwd

```
# useradd -G user_grp user1
```

表1　useraddの主なオプション

オプション	機能
-c［文字列］	ログイン・アカウントに対するコメント
-d［ディレクトリ名］	ホーム・ディレクトリを指定する
-e［日時］	このアカウントの有効期限を指定する。日時はmm/dd/yyで表す
-G［グループ名］	このアカウントが所属するグループを指定する。このとき，カンマで区切ることにより複数指定できる

userdel　ユーザーを削除

構文　userdel（オプション）［ユーザー名］

登録されているユーザー・アカウントを削除する。-rオプションを付けるとユーザーのホーム・ディレクトリも同時に削除する。このコマンドを実行できるのはスーパーユーザーのみ。使用例を以下に示す。

関連 useradd，usermod

```
# userdel -r hoge
```

usermod　ユーザーのアカウント情報を変更

構文　usermod（オプション）［ユーザー名］

ユーザー・アカウントの情報を変更することができる。このコマンドを実行できるのはスーパーユーザーのみ。なお，アカウント名を変更する場合は，できればアカウントを削除して新規に作成した方が安全である。使用例を以下に示す。

関連 useradd，userdel

```
# usermod -e 12/31/02 user1
```

表1　usermodの主なオプション

オプション	機能
-c［文字列］	ログイン・アカウントに対するコメント
-d［ディレクトリ名］	ホーム・ディレクトリを変更する。さらに-mを指定すると，以前のホームディレクトリを移動する
-e［日時］	このアカウントの有効期限を指定する。形式はmm/dd/yyで表す
-f［日数］	有効期限を越えた場合，使用不可能になるまでの日数を指定する。-1にしておくことでこの機能を無効にできる
-l［ユーザー名］	ユーザー・アカウント名を変更する。ただしユーザー・アカウント名のみ変更するだけで，ほかの情報は変更しない
-G［グループ名］	このアカウントが所属するグループを指定する。このとき，カンマで区切ることにより複数指定できる

vmstat　メモリーやCPUの負荷率や使用状況を表示

構文　vmstat［検査する間隔（秒）［検査する回数］］

メモリーやCPUの使用状況や負荷率を調べる。使用例を以下に示す。

関連 ps，top

```
$ vmstat 3 3
   procs                  memory    swap          io     system         cpu
 r  b  w  swpd  free  buff  cache  si  so    bi    bo   in    cs  us  sy  id
 1  0  0 13976  4036 26476 23820   0   0     1     2    5     0   0   0   2
 0  0  0 13976  4032 26476 23820   0   0     0     0  101     8   0   0 100
 0  0  0 13976  4032 26476 23820   0   0     0     0  101     8   0   0 100
```

w　ログインしているユーザー名とその作業内容を表示

構文　w（オプション）［ユーザー名］

現在ログインしているユーザー名とその利用状況を表示する。JCPUはユーザーが使用しているttyから実行されている全プロセスが使った時間を意味し，PCPUはwhat項目で示されているカレント・プロセスが使っている時間を意味する。使用例を以下に示す。関連 who，ps

```
$ w
  7:10pm  up 66 days, 10:14,  2 users,  load average: 0.00, 0.00, 0.00
USER   TTY      FROM              LOGIN@ IDLE    JCPU   PCPU  WHAT
user1 pts/1    linux.nikkeibp.co.jp 5:52pm 59:20   0.08s  0.08s -bash
user2 pts/2    192.168.0.4          6:46pm 23:45   0.05s  0.05s -bash
```

表1　wの主なオプション

オプション	機能
-h	ヘッダーを表示しない
-u	現在のプロセスとCPU時間を計算しているときに，ユーザー名の違いを無視する
-s	ログイン時刻，JCPU，PCPUを表示しない
-f	FROM項目を表示しない

who　現在ログインしているユーザーを表示

構文　**who （オプション）**

現在ログインしているユーザーの情報を表示する。通常はログイン・ユーザー名，端末名，ログインした時間，リモート・ホスト名またはX端末名を表示する。自分自身の情報を知るためには慣習的に“ who am i ”と入力することが一般的だ。使用例を以下に示す。

関連 w，ps

```
$ who -i
user1 pts/1    Apr  8 17:52 01:12
user2 pts/2    Apr  8 18:46 00:36
user4 pts/4    Apr  8 19:10   .
```

表1　whoの主なオプション

オプション	機能
-m	“ who am i ”と入力するのと同じで，自分自身の情報を“ ホスト名!whoの情報 ”で表示する
-q，--count	ログインしているユーザーの人数と，そのログイン名のみを表示する
-i，-u，--idle	ログインした時間のあとに，ユーザーが最後に端末操作を行なってからの時間（idle-time）を表示する。“ . ”が表示されている場合，ユーザーが一分以内に端末操作を行なったことを示し，“ old ”が表示された場合，ユーザーが24時間以上何の端末操作も行なっていないことを示す
-H，--heading	表示の先頭に項目名を記述した行を挿入する
-w，-T，--mesg，--message，--writable	ログイン名の後に，そのユーザーの端末に対する書き込みが可能かどうかを表す文字を付け加える。“ + ”及び“ - ”は，それぞれwriteによるメッセージの書き込みが可能/不可能を表す。また，“ ? ”はデバイスを発見できなかったときに表示される

at　指定時刻にジョブを実行

構文　**at（オプション）［日時］**

指定した時刻にジョブを実行する。実行結果はメールで登録者に送られる。atはコマンドなどを標準入力で指定することになる。コマンドを入力し終えたら，Ctrlキーとdキーを同時に押すことでatコマンドから抜け出せる。なお，日時はMMDDYY，MM/DD/YY，MM.DD.YYなどで指定できる（MMは月をDDは日をYYは年の下2けたを表す）。時刻はhh:mmで指定できるほか，midnight（真夜中），noon（正午），teatime（午後4時）なども指定できる。

例として，0時にログ・ファイルからloginに関するログを切り出す場合を以下に示す。

関連　batch，crontab

```
$ at midnight
> echo -n "Jan 10 : " >> login.log
> gerp -c "Jan 10.*: (login)" /var/log/message >> login.log
> ^d     ←Ctrlキーとdキーを同時に押す
```

表1　atの主なオプション

オプション	機能
-q［クエリー］	指定したキューを使用する。クエリーにはa～z，A～Zのうち1文字を指定できる。ただし，aはatが，bはbatchコマンドが使用する。優先度はaが一番高く，Zが最も低くなる
-f［ファイル名］	ファイルからジョブを入力する
-m	ジョブが終了した後，ジョブの出力がなくてもメールで終了を知らせる
-l	実行待ちのジョブを表示する
-d	ジョブを削除する
-b	自動的にジョブを実行する

batch　自動的にジョブを実行

構文　`batch（オプション）［日時］`

自動的にジョブを実行する。このとき，batchはシステムに高負荷をかけないよう順次ジョブを実行してゆく。実行結果はメールで登録者に知らされる。batchはコマンドなどを標準入力で指定することになる。コマンドを入力し終えたら，Ctrlキーとdキーを同時に押すことでbatchコマンドから抜け出せる。なお，日時の指定や使用例はatコマンドを参照。 関連 at，crontab

表1　batchの主なオプション

オプション	機能
-q［クエリー］	指定したキューを使用する。クエリーにはa〜z，A〜Zのうち1文字を指定できる。ただし，aはatが，bはbatchコマンドが使用する。優先度はaが一番高く，Zが最も低くなる
-f［ファイル名］	ファイルからジョブを入力する
-m	ジョブが終了した後，ジョブの出力がなくてもメールで終了を知らせる

crontab　プログラムを定期的に実行するcrondの設定ファイルを編集する

構文　`crontab（-u［ユーザー名］）（オプション）`

プログラムを指定した時間に定期的に実行するためのデーモンcrondの設定を行う。設定はユーザーごとに用意されたcrontabという設定ファイルに記述する。crontabには**表2**に示すような書式で，各コマンドについて1行ずつ記述する。なお，"-u［ユーザー名］"を省略した場合はカレント・ユーザーのcrontabを設定する。例として，crontabの情報を表示する場合を以下に示す。

関連 at，batch

```
$ crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.1455 installed on Wed Dec  1 02:14:18 1999)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
0 0 * * *   /home/user1/sample.pl      ←毎日0時0分にsample.plを実行
```

表1　crontabの主なオプション

オプション	機能
-l	登録されているcrontabを表示する
-r	登録されているcrontabを削除する
-e	crontabを編集する

表2　crontabの書式

min hour day month week［コマンド］	
min	分を指定する。ワイルド・カードを使用できる
hour	時を指定する。ワイルド・カードを使用できる
day	日を指定する。ワイルド・カードを使用できる
month	月を指定する。ワイルド・カードを使用できる
week	曜日を指定する。0を日曜日とし，順に数字に当てはめていく。また，ワイルド・カードも使用できる
［コマンド］	実行するコマンドを記述する

kill　プロセスおよびジョブを強制終了

構文　**kill（オプション）［プロセスIDまたはジョブ］**

プロセスおよびジョブを終了させる。終了の方法はオプションに“-s［シグナル］”を付けて指定する。使用できるシグナルは**表1**に示す。通常，よく使うのはSIGHUP（終了して再起動），SIGKILL（強制終了）など。シグナルの指定には数値か，シグナル名（先頭のSIGを除いたもの）が使用できる。なお，オプションに-lを付けるとシグナルを一覧表示する。

使用例を以下に示す。

関連 **&,fg,bg,jobs,stop,ps**

```
$ kill -9 583    ←583番のプロセスを強制終了
$ kill -HUP pop ←POPを再起動
```

表1　代表的なシグナル

シグナル番号	シグナル	シグナル番号	シグナル
1	SIGHUP	9	SIGKILL
2	SIGINT	11	SIGSEGV
3	SIGQUIT	13	SIGPIPE
4	SIGILL	14	SIGALRM
6	SIGABRT	15	SIGTERM
8	SIGFPE		

nice　優先順位を決めてコマンドを実行

構文　**nice（オプション）［コマンド］**

先順位（ナイス値）を指定してコマンドを実行する。ナイス値はオプション“-n［ナイス値］”または“-［ナイス値］”，“--adjustment=［ナイス値］”で指定する。ナイス値は-20～19であり，-20が最も優先される。オプションを指定しないとナイス値は10になる。一般ユーザーはナイス値を下げることはできない。使用例を以下に示す。

関連 **ps,top**

```
$ nice -n -15 grep "test" sample.txt > result.txt &
```

nohup　ログアウトした後もコマンドを実行し続ける

構文　**nohup［コマンド］**

nohupでコマンドを実行した場合は，ログアウトしてもプログラムを実行し続ける。長い処理を行っており，席を外したいときなどに使用すると便利である。ログアウトした後は，ジョブではなくプロセスとして管理することになる。また，処理中に出力されたメッセージは“ nohup.out ”に保存される。使用例を以下に示す。

関連 **ps，jobs**

```
$ nohup grep "test" sample.txt > result.txt &
```

pidof　プロセスのpidを調べる

構文　**pidof（オプション）［プログラム］**

プログラムのプロセスpidを調べる。オプションに“ -s ”を付けると，同一の名前のプロセスがあるときに最も番号の大きい1つのpidを表示する。また“ -o［プロセスpid］”を付けると指定したpidの持つプロセスについては表示しない。使用例を以下に示す。関連 **ps**

```
$ pidof httpd
9799 9347 8754 8640 8550 3655 3653 3652 3651 3650 7400
```

ps　実行中のプロセスを表示

構文　**ps（オプション）**

システムで実行しているプロセスを表示する。表示される要素の意味は**表2**の通り。使用例を以下に示す。

関連 **&，fg，bg，jobs，stop，kill**

```
$ ps alx
  F   UID   PID  PPID PRI  NI    VSZ  RSS WCHAN  STAT TTY        TIME COMMAND
100     0     1     0   8   0   1368  176 do_sel S    ?          0:03 init [3]
040     0     2     1   9   0      0    0 contex SW   ?          0:00 [keventd]
  •
  •
100   502 15296 15295  12   0   2660 1332 wait4  S    pts/2      0:00 -bash
000   502 15324 15296  18   0   3004 1136 -      R    pts/2      0:00 ps alx
```

表1　psの主なオプション

オプション	機能
l	詳細を表示する
u	ユーザー名と開始時刻を表示する
j	pgidとsidを表示する
s	シグナル形式で表示する
v	vm 形式で表示する
f	ツリー形式で表示する
a	自分以外のユーザーのプロセスも表示する
x	制御端末のないプロセスの情報も表示する
S	子プロセスのCPU消費時間とページ・フォルトを合計する
c	task_structに格納されているコマンド名を表示する
e	「実行命令＋」に環境変数を付加する
w	1行追加して表示を拡大する。wを増やすことによって行数をさらに増やせる
h	ヘッダーを表示しない
r	実行中のプロセスだけ表示する
n	USERとWCHANを数字で表示する
txx	tty xxのプロセスのみ表示する
pids	表示するプロセスIDを指定する

表2　psで表示される要素の意味

要素	意味
PID	プロセスID
PPID	親プロセスID
TTY	制御端末の種類及び番号
STAT	プロセスのステータス。Rは実行可能，Sは停止，Dは割り込み不可の停止，Tは停止またはトレース中，Zはゾンビ・プロセス，Wはスワップ・アウトしたプロセス，Nはナイス値が正であることを表す
TIME	プロセスが開始された時間
COMMAND	プロセスのコマンド名
UID	ユーザーID
PRI	優先度
NI	ナイス値
SIZE	仮想イメージの大きさ
RSS	使用中の物理メモリー量
WCHAN	プロセスが休眠状態の時のカーネル関数名
PAGEIN	主要なページ・フォルト数
TRS	テキストの常駐しているサイズ
SWAP	スワップ・デバイスの量
SHARE	使用中の共有メモリーの量

stop　バックグラウンドのジョブを停止する

構文　`stop［ジョブ番号］`

バックグラウンドで実行しているジョブを停止する。ジョブ番号を省略した場合はカレント・ジョブ（+のついたジョブ）を停止する。例えば，検索ジョブを停止する場合を以下に示す。

関連　`&, fg, bg, jobs, kill, ps`

```
$ jobs
[1]-  Running                 find / -name "sample" &
[2]+  Stopped (signal)        w3m http://linux.nikkeibp.co.jp/
$ stop 1
$ jobs
[1]-  Stopped                 find / -name "sample"
[2]+  Stopped (signal)        w3m http://linux.nikkeibp.co.jp/
```

top　現在のシステム状況を表示する

構文　top（オプション）

CPUのプロセスをリアルタイムで表示する。実行中はシステム上でリソースを最も使用しているプロセスを上から順に表示する。CtrlキーとCキーを同時に押すことで終了する。なお，オプションに“d［秒数］”をつけると指定した間隔で検査する。またスーパーユーザーの権限でqオプションをつけて実行するとtopを最高の優先順位で実行する。表示されるステータスは**表1**以下の通り。使用例を以下に示す。

関連 ps

```
$ top
 3:08pm  up 55 days, 18:58,  1 user,  load average: 0.03, 0.01, 0.00
53 processes: 51 sleeping, 2 running, 0 zombie, 0 stopped
CPU states:  0.0% user,  0.0% system,  0.0% nice, 100.0% idle
Mem:   255844K av, 251404K used,   4440K free,  0K shrd,  14916K buff
Swap:  208804K av,   5160K used, 203644K free            123636K cached

  PID USER PRI NI SIZE RSS SHARE STAT %CPU %MEM TIME COMMAND
    1 root   8  0  532 488   468 S     0.0  0.1 0:07 init
    2 root   9  0    0   0     0 SW    0.0  0.0 0:00 keventd
      •
      •
  499 root  11  0  872 776   712 S     0.0  0.3 0:10 sshd
  514 root   9  0  804 608   608 S     0.0  0.2 0:00 xinetd
```

表1　表示される要素の意味

要素	意味
PID	プロセスID
USER	プロセスを実行しているユーザー名
PRI	優先度
NI	ナイス値
SIZE	仮想イメージの大きさ
RSS	使用中の物理メモリー量
SHARE	使用中の共有メモリー量
STAT	プロセスのステータス。Rは実行可能，Sは停止，Dは割り込み不可の停止，Tは停止またはトレース中，Zはゾンビ・プロセス，Wはスワップ・アウトしたプロセス，Nはナイス値が正であることを表す
LIB	ライブラリが使用するページ・サイズ
%CPU	CPU占有率
%MEM	メモリー占有率
TIME	プロセス開始からの実行時間
COMMAND	タスクのコマンド名

cat　ファイルを連結して標準出力に出力

構文　cat （オプション）［ファイル名］

ファイルの内容を閲覧する。またファイルを複数指定した場合は，ファイルを連結することもできる。例えば，テキスト・ファイルを連結して別のファイルに出力する場合を以下に示す。

関連 more，less

```
$ cat sample1.txt sample2.txt > sample3.txt
```

表1　catの主なオプション

オプション	機能
-b，--number-nonblank	初めの行を1行目として，空白行以外に行番号を付加する
-n，--number	初めの行を1行目として，すべての行に行番号を付加する
-s，--squeeze-blank	連続した空行を1行の空行にまとめる
-v，--show-nonprinting	タブを“ ^I ”に置き換える。また，表示不可能な文字（最上位ビットが1の文字）は“ M- ”に置き換える
-E，--show-ends	各行の最後に“ $ ”を表示する
-T，--show-tabs	タブを“ ^I ”に置き換える
-e	-vEと同じ意味
-t	-vTと同じ意味
-A，--show-all	-vETと同じ意味

cut　テキスト・ファイルの各行から文節を取り出す

構文　cut （オプション）［ファイル名］

ファイルの各行から指定した文節を取り出す。取り出すとき，指定する数字は複数指定できる。また，“ m-n ”のように数字の間にハイフンを入れるとmからnまでのバイト，文字およびフィールドを指定できる。使用例を以下に示す。

関連 cat

```
$ cat sample.txt
03-xxxx-xxxx   tokyo   user1   Man
092-yyy-yyyy   fukuoka user2   Man
06-zzzz-zzzz   osaka   user3   Woman
$ cut -c 12- sample.txt ←12文字目以降を取り出す
tokyo          user1   Man
fukuoka        user2   Man
osaka          user3   Woman
$ cut -f 2 sample.txt     ←第2フィールドを取り出す
user1
user2
user3
```

表1　cutの主なオプション

オプション	機能
-b, --bytes［バイト数］	指定した位置のバイトだけ表示する
-c, --characters［文字数］	指定した位置の文字だけ表示する
-f, --fields［フィールド数］	field-listで指定したフィールドだけ表示する
-d, --delimiter［区切りの指定］	フィールドの区切りを設定する。初期設定値はタブ
-s, --only-delimited	フィールドの区切りのない行を無視する

grep　文字列を検索する

構文　`grep（オプション）［ファイル名］`

ファイル中の文字列を検索する。検索パターン　　例を以下に示す。
として**表2**のような正規表現を使用できる。使用

```
$ grep "/usr/bin" *.cfg     ←拡張子がcfgのファイルから“/usr/bin”を検索
$ ps aux | grep "httpd"     ←httpdのプロセス情報を表示
$ grep -ci "test" sample.txt  ←“test”のある行数を表示
```

表1　grepの主なオプション

オプション	機能
-G	検索に正規表現を使用できる
-E	検索に拡張正規表現を使用できる
-F	固定文字列の検索を行う
-［行数］	マッチした行から前後，指定した行数を同時に検索結果として表示する
-A［行数］	マッチした行から後，指定した行数を同時に検索結果として表示する
-B［行数］	マッチした行から前，指定した行数を同時に検索結果として表示する
-C	マッチした前後2行を同時に検索結果として表示する
-b	各行の前に，ファイルの先頭からバイト単位のオフセット数を表示する
-n	各行の前に行番号を表示する
-c	検索条件にマッチした行数を表示する。-cvとするとマッチしなかった行数を表示する
-e［パターン］	検索条件を指定する
-f［ファイル名］	検索パターンとしてファイルの内容を使用する
-h	検索結果の先頭にマッチしたファイル名を同時に表示する
-i	検索条件に大文字と小文字の区別をなくす
-l	検索条件にマッチしたファイル名を表示する。-lvとするとマッチしなかったファイル名を表示する
-q	検索結果を表示しない
-s	エラー・メッセージを表示しない
-v	マッチしない行を検索結果として表示する
-w	パターン・マッチを，単語全体で行うようにする
-x	行全体を検索対象にする

表2　正規表現の表記

表記	意味
.	改行文字以外の任意の1文字
*	直前の1文字の0回以上の繰り返しに一致。直前の文字は正規表現でも構わない
^	行の先頭を表す
$	行の末尾を表す
[]	かっこ内の任意の文字に一致。ハイフン(-)で範囲指定もできる。かっこ内の最初の文字に^を使用すると，意味が逆転する
+	直前の文字の1個以上の連続
?	直前の文字の0または1文字にマッチ
[パターン1]|[パターン2]	パターン1またはパターン2のいずれかにマッチ
([パターン])	パターンをグループ化する
\	正規表現に使われる記号を普通の文字として扱う

less　テキスト・ファイルの内容を閲覧

構文　`less [ファイル名]`

ファイルの内容をページ単位で自由に閲覧できる。閲覧中の操作方法は**表1**の通り。使用例を以下に示す。

関連 `cat,more`

```
$ less sample.txt
```

表1　less実行時の操作方法

コマンド	機能
h, H	lessの操作ヘルプを表示する
q, Q	終了する
e, j	1行進める。コマンドを入力する前に数字を入力すると指定した行数ずつ進む
y, k	1行戻る。コマンドを入力する前に数字を入力すると指定した行数ずつ戻る
f, SPACE	次の画面へ進む。コマンドを入力する前に数字を入力すると，指定した画面数ずつ進む
b, ESC-v	前の画面へ戻る。ただし，コマンドを入力する前に数字を入力すると，指定した画面数ずつ戻る
d	半画面進む。ただし，コマンドを入力する前に数字を入力すると，指定した半画面数ずつ進む
u	半画面戻る。ただし，コマンドを入力する前に数字を入力すると，指定した半画面数ずつ戻る
r	画面を再度書き換える
/[パターン]	現在より先のパターンを検索し，移動する
?[パターン]	現在より前のパターンを検索し，移動する
n	検索を再度行う
g, <	ファイルの先頭へ移動する
G, >	ファイルの末尾に移動する
:e [ファイル名]	指定したファイルを閲覧する
:n	次のファイルを閲覧する
:p	前のファイルを閲覧する
:x	初めのファイルを閲覧する

more　テキスト・ファイルの内容をページ単位で閲覧

構文　more（オプション）［ファイル名］

テキスト・ファイルの内容をページ単位で閲覧する。閲覧中の操作方法は**表2**の通り。例えば，検索結果をページ単位で出力する場合を以下に示す。

関連 cat，less

```
$ grep "exe" ./error_log | more
```

表1　moreの主なオプション

オプション	機能
-d	ページが停止時にメッセージを表示する
-l	改ページを表すキャラクター(^L)を無視する
-f	実際の行数を集計する
-p	画面クリアしてページを切り替える
-c	上書きでページを切り替える
-s	連続した空行を1行にまとめる
-u	アンダーラインの表示を禁止する
-［文字数］	画面に表示する幅を指定する
+/［パターン］	パターンを検索し，その場所から表示を始める
+［行数］	表示開始行を指定する

表2　more実行時の操作方法

コマンド	機能
h，?	moreの操作ヘルプを表示する
SPACE	次のページに進める
z	次のページに進める。ただし，zを入力する前に数字を入力すると，その行数だけ行送りされる
RETURN	1行進める。ただし，RETURNを入力する前に数字を入力すると，その行数ずつ行送りされる
d，^D	11行進める。ただし，dまたは^Dを入力する前に数字を入力すると，その行数ずつ行送りされる
q，Q，INTERRUPT	終了する
s	1行進める。ただし，sを入力する前に数字を入力すると，その行数ずつ行送りされる
f	1画面進める。ただし，fを入力する前に数字を入力すると，その画面数ずつ画面送りされる
b，^B	1画面戻る。ただし，bを入力する前に数字を入力すると，その画面数ずつ画面を戻す
'	検索を開始した位置に戻る
=	現在の行番号を表示する
/［パターン］	パターンを検索し，その場所に移動する。ただし，/［パターン］を入力する前に数字を入力することで，その回数だけ検索し，最後の検索がマッチした画面を表示する
n	/［パターン］の検索を再度行う
!または:!	シェルを起動しcmdで指定したコマンドを実行する
v	現在の行から以降をviで編集する
^L	画面を再描画する
:n	次のファイルに進める。ただし，:nを入力する前に数字を入力すると，そのファイル数だけ進める
:p	前のファイルに戻る。ただし，:pを入力する前に数字を入力すると，そのファイル数だけ戻る
:f	現在のファイル名と行番号を表示する
.	直前のコマンドを繰り返す

nkf　文字コードを変換

構文　**nkf（オプション）［ファイル名］**

文字コードを変換する。オプションに何も指定しない場合はJISコードに変換する。例えば，シフトJISコードのファイルをEUCコードに変換する場合を以下に示す。

```
$ nkf -e sample_sjis.txt > sample_euc.txt
```

表1　nkfの主なオプション

オプション	機能
-b	バッファリング出力を行う
-u	出力時にバッファリングを行わない
-j	JISコードに変換する
-e	EUCコードに変換する
-s	シフトJISコードに変換する
-i?	JIS漢字を指示するシーケンスとしてESC-'$'-?を使用する
-o?	1バイト英数文字セットを指示するシーケンスとしてESC-'('-?を使用する
-r	ROT13/47の変換する
-T	テキスト・モードで出力する
-l	0x80-0xfeのコードをISO-8859-1（Latin-1）として扱う。ただし，JISコードの時のみ有効
-f?	一行?文字になるように簡単な整形を行う
-Z	X0208中の英数字と一部の記号をASCIIに変換する
-J	JIS（ISO-2022-JP）と仮定して処理する
-E	日本語EUCと仮定して処理する
-S	シフトJISと仮定して処理する
-X	シフトJISと仮定して処理する。ただし，X0201仮名があるものとする
-B	壊れた（ESCが欠損した）JISと仮定して処理する。-B1の場合はESC-（およびESC-$）の後のコードを問わない。-B2の場合は改行の後に強制的にASCIIに戻す
-x	通常行われるX0201仮名→X0208の仮名変換しない

sort 行を並び替える

構文 `sort（オプション）［ファイル名］`

テキスト・ファイルの内容を並び替える。例として，テキスト・ファイルを逆順に並び替えてファイルに出力する場合を以下に示す。

関連 uniq

```
$ sort -r sample.txt > sample_rev.txt
```

表1　sortの主なオプション

オプション	機能
-c	ソートされているファイルかをチェックする。もし，ソートされていないのならばエラー・メッセージを出力する（戻り値として1を返す）
-m	複数のファイルをソートしながら1つのファイルにまとめる。ただし，それぞれのファイルはあらかじめソートされている必要がある
-b	各行の先頭に空白があった場合は，その空白を無視する
-d	アルファベット・数字・空白だけを使用してソートする
-f	小文字を大文字としソートする
-i	ASCIIコードの040以上，0176以下（8進数）に含まれない文字は無視する
-M	先頭に現れた3文字を月表記とみなしてソートする。ただし，先頭になる空白は無視される
-n	先頭の数字や記号（“＋”,“－”,“．”等）を数値とみなしてソートする。ただし，先頭の空白は無視される
-r	逆順にソートする
-o［ファイル名］	ソート結果をファイルに出力する
-t［separator］	ソートの区切りとして指定したseparatorを使用する。文字が指定されない場合は空白
-u	デフォルトや-mオプションを指定した際，同じ行が存在したときは，1行だけ表示する。-cオプションを指定した際，連続した行で等しい物がないかをチェックする
+POS1［-POS2］	各行でソート対象となるキーの場所を指定する。設定方法はf.cの形をとり，fがフィールドをcはそのフィールド内での文字数を表す。ただし，先頭は0となる
-k POS1［,POS2］	上と同じでソート・キーの場所を設定する。ただし，先頭は1となる

uniq　ソート済みのファイルから重複した行を削除

構文　`uniq（オプション）［入力ファイル名］［出力ファイル名］`

ファイルで重複している行を削除する。ただしファイルはソートしてある必要がある。なお，出力ファイルを指定しない場合は標準出力に出力する。使用例を以下に示す。

関連 sort

```
$ cat sample.txt
03-xxxx-xxxx   tokyo   user1   Man
092-yyy-yyyy   fukuoka user2   Man
06-zzzz-zzzz   osaka   user3   Woman
03-xxxx-xxxx   tokyo   user1   Man
$ soft sample.txt | uniq
03-xxxx-xxxx   tokyo   user1   Man
092-yyy-yyyy   fukuoka user2   Man
06-zzzz-zzzz   osaka   user3   Woman
```

表1　uniqの主なオプション

オプション	機能
-u，--unique	ユニークな行を表示する
-d，--repeated	重複行を表示する
-c，--count	行表示の際，それぞれの行の数も表示する
-［行数］，-f，--skip-fields=［行数］	判断を開始するフィールドを指定する。指定したフィールド以降が判断に使用される
+［文字数］，-s，--skip-chars=［文字数］	判断を開始する文字数を指定する。指定した文字以降が判断に使用される
-w，--check-chars=［文字数］	判断の終了文字数を指定する。指定しない場合は行末とする

vi　テキスト・ファイルを編集

構文　**vi（オプション）［ファイル名］**

テキスト・ファイルを編集する。viは，UNIXで標準のスクリーン指向なテキスト・エディタである。使用例を以下に示す。

```
$ vi sample.txt
```

表1　viの主なオプション

オプション	機能
-R	読み込み専用モードでファイルをオープンする
-b	バイナリ・モードで編集を行う
-r	中断から復帰する。実行には編集中断したファイル名の指定が必要
+line	編集開始時に指定した行に移動する

viには「モード」という概念がある。起動直後の「編集モード」では，カーソルの移動，文字(行)の削除などのみが可能。aやiといった文字入力コマンドを入力すると，追加/挿入モードになり文字の入力が可能になる。ファイルの書き出しや検索などにはコマンド・モードで行う。追加/挿入モード終了するにはESCキーを，編集モードからコマンド・モードへ移行するには「:」キーをタイプする。各モードの操作方法を**表2**に示す。

表2　viの操作方法

編集モード	
追加/挿入モード	
a	カーソルの右側から入力を始める
i	カーソルの左側から入力を始める
o	次行を追加し入力を始める
ESC	追加/挿入モードを終了
文字の削除	
x	カーソル位置の文字を削除する
dd	カーソル位置の行を削除する
D	カーソル位置から行の末尾まで削除する
カーソルの操作	
h	1文字左に移動(←)
j	1文字下に移動(↓)
k	1文字上に移動(↑)
l	1文字右に移動(→)
-	前行の先頭へ移動
+	次行の先頭に移動
0	行頭に移動
$	行末に移動
コマンド・モード	
ファイルの読み込み・保存コマンド	
ZZ	ファイルに保存し終了する(編集モードで使用)
:w	ファイルに保存する
:w [ファイル名]	指定したファイル名に変更して保存する
:r [ファイル名]	現在のカーソルの場所に指定したファイルを読み込む
:e [ファイル名]	指定したファイルを新規ファイルとして編集する
:q	viを終了する
検索	
/[文字列]	カーソルより後の文字列を検索し,移動する
?[文字列]	カーソルより前の文字列を検索し,移動する
n	検索を後方へ繰り返す
N	検索を前方へ繰り返す

wc　テキスト・ファイルの行数,単語数,バイト数を表示

構文　wc （オプション）［ファイル名］

ファイル内のバイト数, 単語数および行数を集計し表示する。また, 空白で区切られたものを単語として扱う。表示は左から行数, 単語数, バイト数。オプションに-cまたは--bytes, --charsを付けるとバイト数だけ, -wまたは--wordsを付けると単語数だけ, -lまたは--linesを付けると行数だけを集計して表示する。使用例を以下に示す。

```
$ wc sample.txt
      3       6      65 sample.txt
```

ftp　FTPサーバーに接続してファイル転送を行う

構文　`ftp（オプション）［ホスト名］`

リモート・ホストに接続してファイルの送受信を行う。実行後，対話型のコマンド入力により操作を行う。主なコマンドは**表2**のようなものがある。

使用例を以下に示す。

関連 `cp`，`telnet`，`scp`

```
# ftp sample_ftp.nikkeibp.co.jp
```

表1　catの主なオプション

オプション	機能
-v	リモートサーバーからすべての情報を表示する
-n	.netrcを使用した自動ログインを行わない
-i	複数のファイルの転送中にプロンプトを出力しない
-g	正規化表現を無効にする

表2　ftpの主なコマンド

コマンド	機能
open［ホスト名］	リモート・ホストに接続する
close	リモート・ホストから切断する
quit	ftpを終了する
binary	転送モードをバイナリにする。バイナリ・ファイルの転送時は必ずbinaryを実行しておく必要がある。
ascii	転送モードをASCIIモードにする
ls［ディレクトリ名］［出力ファイル名］	リモート・ホストのディレクトリのファイルを一覧する。ファイル名を指定すると，ローカル・マシン上のファイルに一覧を出力する。また，ホストの中にはlsではなくdirを用いることがある
dir［ディレクトリ名］［出力ファイル名］	lsを参照
cd［ディレクトリ名］	ディレクトリを移動する
pwd	現在のディレクトリの位置を表示する
mkdir	ディレクトリを作成する
chmod［アクセス権］［ファイル名］	ファイルやディレクトリのアクセス権を指定する。アクセス権の指定方法はchmodを参照
lcd［ディレクトリ名］	ローカル・マシンのディレクトリを移動する
get［ファイル名］［出力ファイル名］	リモート・ホストのファイルをローカル・マシンへ受信する。出力ファイル名を指定するとファイル名が変換される
put［ファイル名］［出力ファイル名］	ローカル・マシンのファイルをリモート・ホストへ送信する。出力ファイル名を指定するとファイル名が変換される
mget［ファイル名］	リモート・ホストの複数のファイルをローカル・マシンへ受信する。このとき，ファイル名にはワイルドカードを使用できる
mput［ファイル名］	ローカル・マシンの複数のファイルをリモート・ホストへ送信する。このとき，ファイル名にはワイルドカードを使用できる

hostname　ホスト名を表示，設定

構文　`hostname （オプション）［ホスト名］`

ホスト名の参照，変更を行う。ホスト名を指定するとホスト名が変更される。ただし変更にはスーパーユーザー権限が必要。使用例を以下に示す。

関連 uname

```
# hostname sample_host
```

表1　hostnameの主なオプション

オプション	機能
-a	ホストの別名（alias）があれば表示する
-d	DNSドメインを表示する
-f	FQDNを表示する
-i	IPアドレスを表示する
-s	短い形式でホスト名を表示する
-y	NISドメインを表示・設定する

ping　パケットを送り，リモート・ホストの状況を調べる

構文　`ping （オプション）［ホスト名］`

リモート・ホストにパケットを送り，リモート・ホストが動作しているかどうかを調べる。ネットワークの状態確認やホストの存在を調べる際に使用する 。使用例を以下に示す。

```
$ ping -c 3 sample.xxxx.jp
PING sample.xxxx.jp (xxx.xx.xxx.xxx) from xxxx.xx.x.x : 56(84) bytes of data.
64 bytes from sample.xxxx.jp (xxx.xx.xxx.xxx) : icmp_seq=0 ttl=245
time=57.633 msec
64 bytes from sample.xxxx.jp (xxx.xx.xxx.xxx) : icmp_seq=1 ttl=245
time=71.242 msec
64 bytes from sample.xxxx.jp (xxx.xx.xxx.xxx) : icmp_seq=2 ttl=245
time=56.590 msec

--- sample.xxxx.jp ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 56.590/61.821/71.242/6.680 ms
```

表1　pingの主なオプション

オプション	機能
-q	開始時と終了時の情報のみを表示する
-r	ゲートウェイやルーター等を経由せず，直接接続を行う
-c［回数］	指定した回数パケットを送信する。指定しない場合はパケットを送り続ける。その場合はCtrlキーとcキーを同時に入力して終了できる
-i［秒］	パケットの送信間隔を指定する
-s［バイト数］	転送するパケットのサイズを指定する

telnet　他のホストと通信する

構文　`telnet（オプション）［ホスト名 ［ポート番号］］`

リモート・ホストに接続する。ポートを省略した場合ポート23番で接続を行う。ctrl+]（-eオプションにより文字を変更可能）を入力することでコマンド・モードに移行することができる。コマンド・モードでの主なコマンドは**表2**の通り。使用例を以下に示す。

関連　`slogin，ssh，ftp`

```
$ telnet sample_host.nikkeibp.co.jp
```

表1　telnetの主なオプション

オプション	機能
-8	8ビット・バイナリ・モードで接続する。シフトJISを使用している際などに指定する
-e［キャラクタ］	telnetコマンド・モードに移行する際のキャラクターを設定する
-l［ユーザー名］	指定したユーザー名でログインする

表2　telnetで実行できるコマンド

コマンド	意味
open	指定したホストにログインする
close	現在の接続を切断する
status	現在の状況を表示する
quit	telnetを終了する
?コマンドを一覧を表示する	

scp　リモート・マシン間でファイルを安全にコピー

構文　**scp（オプション）［ユーザー名@コピー元のホスト名：コピー元のファイルのパス］［ユーザー名@コピー先のホスト名：コピー先のファイルのパス］**

SSHプロトコルを用いてリモート・ホスト間で安全にファイルをコピーする。ローカル・ホストの場合は［ユーザー名@コピー先のホスト名］を省略できる。使用例を以下に示す。

関連 ssh，slogin，ssh-keygen，cp

```
$ scp sample_file user1@sample.nikkeibp.jp:/home/user1/temp
```

表1　scpの主なオプション

オプション	機能
-i［ファイル名］	RSA公開かぎファイルを指定する。初期設定は“~/.ssh/identity”
-p	更新時間，アクセス時間，モードを保持する
-r	ディレクトリ内を再帰的にコピーする
-P［ポート番号］	接続するポートを設定する

slogin　リモート・マシンに安全にログイン

構文　**slogin（オプション）［ホスト名］［コマンド］**

SSHプロトコルを用いてリモート・ホストへ安全にログインする。ホスト名は“ ユーザー名@ホスト名 ”とすることで-lオプションと同様な効果を得ることができる。またホスト名の後にコマンドを付けるとsshコマンドと同じ働きをするので，リモート・ログインしたいときは指定してはいけない。使用例を以下に示す。

関連 ssh，scp，ssh-keygen，telnet

```
$ slogin user1@sample.nikkeibp.co.jp
```

表1　sloginの主なオプション

オプション	機能
-l［ユーザー名］	ログインに使用するユーザー名を指定する
-i［ファイル名］	RSA公開鍵ファイルを指定する。初期設定は“~/.ssh/identity”
-p［ポート番号］	接続するポートを設定する
-X	Xのポート・フォワーディングを有効にする。リモート・マシンのXアプリケーションを実行できる
-x	Xのポート・フォワーディングを無効にする

ssh　SSHでリモート・マシンのコマンドを実行

構文　**ssh（オプション）［ホスト名］［コマンド］**

SSHプロトコルを用いてリモート・ホストのコマンドを安全に実行する。ホスト名は“ユーザー名@ホスト名”とすることで-lオプションと同様な効果を得ることができる。またコマンドを指定しないとリモート・ホストへのログインとなる。例として，リモート・ホストで検索を実行する場合を以下に示す。

関連 scp，slogin，ssh-keygen

```
$ ssh user1@sample.nikkeibp.co.jp locate sample.txt
```

表1　sshの主なオプション

オプション	機能
-l［ユーザー名］	ログインに使用するユーザー名を指定する
-i［ファイル名］	RSA公開鍵ファイルを指定する。初期設定は“~/.ssh/identity”
-p［ポート番号］	接続するポートを設定する
-X	Xのポート・フォワーディングを有効にする。リモート・マシンのXアプリケーションを実行できる
-x	Xのポート・フォワーディングを無効にする

ssh-keygen　SSH用の公開かぎ，秘密かぎのペアを作成

構文　**ssh-keygen（オプション）［-P 古いパス・フレーズ］［-N 新しいパス・フレーズ］［-C コメント］**

SSHの暗号かぎを作成，変更する。オプションに何も指定しない場合は，かぎの作成を対話式に進めていく。例えば，新規にかぎを作成する場合を以下に示す。

関連 ssh，slogin，scp

```
$ ssh-keygen -b 512 -f key_file -N sample -C test_key_make
```

表1　ssh-keygenの主なオプション

オプション	機能
-b［ビット数］	かぎの長さを指定する。最低値は512ビットで初期設定値は1024ビット
-c	かぎ内のコメントを変更する
-f［ファイル名］	かぎのファイル名を指定する
-p	パス・フレーズを変更する。古いパス・フレーズと新しいパス・フレーズの両方を指定する必要がある

fdformat　フロッピ・ディスクを初期化

構文　**fdformat（オプション）［デバイスの種類］**

フロッピ・ディスクをフォーマットする。物理フォーマットを行うだけで，実際にフロッピ・ディスクを使用するには論理フォーマットでファイル・システムを構築する必要がある。物理フォーマットにはmkfsやmformatコマンドを使用する。なお，実行時に-nオプションを付けるとベリファイを行わない。実行時に指定できるデバイスの種類は**表1**の通り。

例えば，1.44Mバイトでフォーマットする場合を以下に示す。

```
# fdformat /dev/fd0H1440
```

関連　mkfs，mformat

表1　fdformatで指定するデバイスの種類

/dev/fd0h1200	FDドライブ0の1.2Mバイト2HDフォーマット
/dev/fd0D720	FDドライブ0の720Kバイト2DDフォーマット
/dev/fd0H1440	FDドライブ0の1.44Mバイト2HDフォーマット
/dev/fd1h1200	FDドライブ1の1.2Mバイト2HDフォーマット
/dev/fd1D720	FDドライブ1の720Kバイト2DDフォーマット
/dev/fd1H1440	FDドライブ1の1.44Mバイト2HDフォーマット

fdisk　ハード・ディスクのパーティションを設定

構文　**fdisk（オプション）［デバイス名］**

ハード・ディスクのパーティションの構築・削除・変更を行う。実行にはスーパー・ユーザーの権限が必要。コマンド実行後は対話式にパーティションを設定する。なお，-lオプションを付けるとマウントされているデバイスのパーティション情報を表示する。

fdis実行時に使用できる主なコマンドは**表1**の通り。使用例を以下に示す。

```
# fdisk /dev/hda1
Command (m for help):
```

関連　mkfs，mount

表1　fdisk実行時のコマンド

コマンド	機能
a	ブートの可否を切り替える
d	パーティションを削除する
l	利用可能なパーティション・タイプを表示する
m	メニュー（コマンド集）を表示する
n	新しいパーティションを作成する
o	DOSパーティションを作成する
p	パーティション情報を表示する
q	保存せずに終了する
t	パーティション・タイプを変更する。 初期値は“Linux native”（83）
v	パーティションを検査する
w	パーティション情報を書き込み，終了する

fsck　ディスク検査と修復を行う

構文　`fsck（オプション）［デバイス名］`

ディスクの検査と修復を行う。ただし修復は完全ではないので，ファイルを破壊してしまう可能性も含んでいる。エラーが発生した場合は，できる限りバックアップを行った上で修復することをお勧めする。使用例を以下に示す。

関連 mbadblocks

```
$ fsck /dev/hda1
```

表1　fsckの主なオプション

オプション	機能
-A	/etc/fstabに記述されているすべてのファイル・システムの検査と修復を行う
-R	ルート・ファイル・システム以外のすべてのファイル・システムに対して検査と修復を行う
-N	実際には実行せず，実行内容だけを表示する
-t［ファイル・システムの種類］	検査するファイル・システムを指定する。指定時の表記はmount（p.56）の表2を参照

mkfs　ファイル・システムを構築

構文　`mkfs（オプション）［デバイス名］［ブロック数］`

指定したデバイスにファイル・システムを構築する。フロッピ・ディスクを使用するには，fdformatで物理フォーマットした後に，このコマンドでファイル・システムを作成する必要がある。例えば，フロッピ・ディスクをvfatでフォーマットする場合を以下に示す。

関連 fdformat，fdisk

```
# mkfs -t vfat /dev/fd0
```

表1　mkfsの主なオプション

オプション	機能
-V	詳細情報を表示する
-t［ファイル・システムの種類］	作成するファイル・システムを指定する。指定時の表記はmount（p.56）の表6を参照

mount　ファイル・システムをマウント

構文　**mount（オプション）［デバイス名］［マウント・ポイント］**

ファイル・システムをマウントする。Linuxではマウントという作業をしない限りデバイス上のファイル・システムを使用することは原則的にできない。マウント作業により，デバイス上のファイル・システムをディレクトリの1つとして使用できる。マウント作業はスーパー・ユーザー権限がないと行えないが，“ /etc/fstab ”での設定次第では通常のユーザーでのマウントも可能。使用例を以下に示す。

関連 **umount，df**

```
# mount -r -t iso9660 /dev/cdrom /mnt/cdrom  ←CD-ROMをマウント
# mount -t ext2 /dev/fd0 /mnt/floppy         ←Linux方式のFDをマウント
# mount -t vfat /dev/fd0                     ←Windows方式のFDをマウント
```

表1　mountの主なオプション

オプション	機能
-a	/etc/fstabに記述されているファイル・システムをマウントする。ただし，noautoのファイル・システムはマウントから除外される
-n	マウントをする際，/etc/mtabに情報を書き込まない
-r	ファイル・システムを読み込み専用でマウントする
-w	ファイル・システムを読み書き可能な状態でマウントする
-v	マウントの詳細を表示する
-t［ファイル・システムの種類］	ファイル・システムの種類を指定する。指定時の表記は表2を参照

表2　主なファイル・システムの種類

ファイル・システム・タイプ	詳細
ext2	Linux標準のファイル・システム
ext3	ext2にジャーナリング機能を付加したファイル・システム
ReiserFS	ジャーナリング機能を持ったファイル・システム
jfs	ジャーナリング機能を持ったファイル・システム
xfs	ジャーナリング機能を持ったファイル・システム
msdos	MS-DOSファイル・システム
vfat	Windows95以降に搭載されているファイル・システム
iso9660	ISO9660準拠のファイル・システム（一般的なCD-ROMの形式）
nfs	ネットワーク・ファイル・システム
samba	Windowsのネットワーク共有のファイル・システム

umount　ファイル・システムをアンマウント

構文　umount （オプション）［マウント・ポイントまたはデバイス名］

ファイル・システムをアンマウントする。フロッピ・ディスクやCD-ROMのイジェクトなどデバイスを切り離す前には必ず行う必要がある。例えば，/mnt/cdromにマウントしているCD-ROMをアンマウントする場合を以下に示す。

関連 mount，df

```
# umount /mnt/cdrom
```

表1　umountの主なオプション

オプション	機能
-a	/etc/fstabに記述されているファイル・システムをすべてアンマウントする
-n	アンマウントをする際，/etc/mtabに情報を書き込まない
-r	アンマウントが失敗した場合は，読み込み専用で再マウントを試みる
-v	アンマウントの詳細を表示する
-t［ファイル・システムの種類］	対象を指定したファイル・システムの種類のみアンマウントする。カンマで区切ることで複数指定できる

lpc　印刷を制御

構文　**lpc［コマンド］**

プリンタを制御を行うプログラム。プリンタ・スプーラ・デーモンのlpdの動作状況の調査や制御が可能。主なコマンドは**表1**の通り。例えば，lpdの状態を表示する場合を以下に示す。

関連 lpq，lpr，lprm

```
$ lpc status
Printer   Printing Spooling Jobs  Server Subserver Redirect Status/(Debug)
lp@sample enabled  enabled    0    none    none
```

表1　lpcの主なコマンド

コマンド	機能
help［コマンド］	指定したコマンドの使用方法を表示する。コマンドを指定しない場合はコマンドの一覧が表示される
abort { all | printer }	印刷キュー内の印刷ジョブを中止する
clean { all | printer }	一時ファイル，データ・ファイル，制御ファイルを削除する
disable { all | printer }	印刷キューを停止する
down { all | printer }［メッセージ］	印刷を停止し，プリンタの状態表示に指定したメッセージを表示する
enable { all | printer }	ローカル・キューへのスプールを可能にする
exit, quit	lpcを終了する
restart { all | printer }	デーモンを起動しプリントを再開する
start { all | printer }	スプールを可能にする
status { all | printer }	情報を表示する
stop { all | printer }	現在のジョブが終了次第，印刷を中止する
topq printer［ジョブ番号］［ユーザー名］	指定した印刷ジョブを最優先にする
up { all | printer }	downコマンドによる停止状態からの再起動に使う

lpq　印刷ジョブを確認

構文　**lpd（オプション）［ジョブ番号］　［ユーザー名］**

プリンタの印刷ジョブの情報を表示する。表示するプリンタは-Pオプションで指定する。ただし，-Pオプションを省略すると標準プリンタの情報を表示する。なお，-lオプションを付けると詳細情報も表示する。使用例を以下に示す。

関連 lpc，lpr，lprm

```
$ lpq
lp is ready and printing
Rank   Owner      Job  Files             Total Size
active user1      325  (standard input)  992865 bytes
```

lpr　プリンタで印刷

構文　**lpr（オプション）［ファイル名］**

プリンタで印刷をするためのコマンド。ファイル名が指定されていないときは標準入力を出力する。lprの出力はプリンタの管理キューに送られ，順番になり次第，印刷が開始される。使用例を以下に示す。

関連 lpc，lpq，lprm

```
$ lpr sample.txt
```

表1　lprの主なオプション

オプション	機能
-d	ファイルがTeXのDVIフォーマットであるとみなす
-l	コントロール・キャラクタを用いて，ページの区切りを抑制するフィルタを利用する
-p	ファイルの整形にprを利用する
-v	ファイルがラスタ・イメージを含んでいるとする
-P	指定したプリンタに出力する。指定しない場合は，デフォルト・プリンタが使用されるか，環境変数PRINTERの値が使われる
-m	印刷終了時にメールを送って知らせる
-# ［部数］	印刷部数を指定する
-T ［タイトル］	prのタイトル名に指定したタイトルを使う
-i［数値］	出力をインデントする。数値を指定したときはその分インデントする。指定がない場合は8
-w ［数値］	prのページ幅を指定した数値にする

lprm　印刷キュー内の印刷ジョブを取り消す

構文　**lprm（オプション）［ジョブ番号］［ユーザー名］**

キューで印刷待ちしている印刷ジョブを削除する。-Pオプションで対象となるプリンタを指定する。また-オプションを付けると自分のすべての印刷ジョブを削除する。スーパーユーザーの場合は印刷キューを完全に空にする。例えば，lp1のプリンタのジョブ番号123の印刷を中止する場合を以下に示す。

関連 lpc，lpq，lpr

```
$ lprm -P lp1 123
```

bunzip2　.bz2ファイルの展開

構文　**bunznip2（オプション）［ファイル名］**

LZ77/LZ78をベースとしたアルゴリズムを使用して圧縮ファイルを展開する。-kオプションを付けると元の圧縮ファイルが削除されずに残る。また，ワイルドカードなどで複数のファイルをまとめて展開するとき，-vオプションを付けるとファイルが展開されるたびにメッセージが表示される。使用例を以下に示す。

関連 bzip2

```
$ bunzip2 sample.bz2
```

bzip2　.bz2ファイルの圧縮・展開

構文　**bzip2（オプション）［ファイル名］**

LZ77/LZ78をベースとしたアルゴリズムを使用してファイルを圧縮・展開する。bzip2で圧縮されたファイルには拡張子にbz2が付加されて書き換えられる。オプションを省略した場合は圧縮と同じ。展開は-dオプションを付けて実行するか，bunzip2を使用する。

例えば，ファイルを圧縮する場合を以下に示す。

関連 bunzip2

```
$ bzip2 sample
```

表1　bzip2の主なオプション

オプション	機能
-d	圧縮ファイルを展開する。bunzip2と同じ
-v	圧縮状況のメッセージを表示する
-c	圧縮ファイルを標準出力し，元ファイルを残す
-n	元のファイル名とタイムスタンプは保存しない
-z	圧縮する
-k	圧縮および展開する際，元ファイルを削除しない
-1〜9	圧縮の際のブロック数を100K〜900Kバイトにする

cpio ファイルをバックアップ

構文

```
cpio -o(cBv) (-H [データ方式]) (< [アーカイブ(コピー)するファイル名]) (< [展開するアーカイブ・ファイル名])
cpio -i(cdv) (-H [データ方式]) [展開するファイル名] (> [出力ファイル名])
cpio -p(adm) [コピー先のディレクトリ名] < [アーカイブ(コピー)するファイル名]
```

ファイルのバックアップを作成・展開できる。例えば，カレント・ディレクトリのファイルをフロッピ・ディスクにバックアップする場合を以下に示す。

関連 tar

```
$ ls -1 | cpio -o > fd0H1440
```

表1 cpioの主なオプション

オプション	機能
-o	アーカイブを作成する
-i	アーカイブからファイルを取り出す。ファイルを指定しない場合はすべて取り出す
-p	ファイルを別のディレクトリにコピーする
-c	ヘッダ情報をACII文字として読み書きする
-d	必要に応じてディレクトリを作成する
-v	ファイル名のリストを表示する
-B	5,120バイトを1レコードとしてコピーする
-a	入力ファイルのアクセス時間をリセットする
-m	ファイルの変更時間をそのままにする
-H [データ形式]	出力するデータ方式を指定する。データ方式は表2を参照
< [アーカイブ(コピー)するファイル名]	アーカイブ(コピー)するファイル名を指定する
> [出力ファイル名]	出力ファイルを指定する
< [展開するアーカイブ・ファイル名]	展開するアーカイブを指定する

表2 cpioで指定できるデータ形式

表記	形式
bin	バイナリーフォーマット(デフォルト)
odc	旧ASCII(旧POSIX.1ポータブル)フォーマット
newc	新ASCII(新SVR4ポータブル)フォーマット
crc	CRC付き新ASCIIフォーマット
tar	旧tarフォーマット
ustar	POSIX.1 tarフォーマット
hpbin	HPUXのcpioで使用されている古いバイナリ・フォーマット
hpodc	HPUXのcpioで使用されているポータブル・フォーマット

gunzip　.gzファイルを展開

構文　**gunzip（オプション）［ファイル名］**

Lempel-Ziv（LZ77）アルゴリズムを使用したファイルを展開する。展開後は拡張子のgzを取り除き，ファイルを置き換える。使用例を以下に示す。

関連 **gzip，zcat**

```
$ gunzip sample.gz
```

表1　gunzipの主なオプション

オプション	機能
-d	圧縮ファイルを展開する
-v	ファイルが展開されるたびにメッセージを表示する
-c	圧縮ファイルを標準出力し，元ファイルを残す
-r	再帰的にディレクトリ内を展開する
-l	圧縮ファイルのリストを表示する
-n	元のファイル名とタイムスタンプは保存しない

gzip　.gzファイルの圧縮・展開

構文　**gzip（オプション）［ファイル名］**

Lempel-Ziv（LZ77）アルゴリズムを使用して圧縮する。gzipで圧縮されたファイルは拡張子にgzが付加されて書き換えられる。展開は-dオプションを付けて実行するか，gunzipを使用する。

例えば，ファイルを圧縮する場合を以下に示す。

関連 **gunzip，zcat**

```
$ gzip sample
```

表1　gzipの主なオプション

オプション	機能
-d	圧縮ファイルを展開する。gunzipと同じ
-v	ファイルが圧縮されるたびにメッセージを表示する
-c	圧縮ファイルを標準出力し，元ファイルを残す
-r	再帰的にディレクトリ内を圧縮する
-l	圧縮ファイルのリストを表示する
-n	元のファイル名とタイムスタンプは保存しない

lha　.lzhファイルの圧縮・展開

構文　`lha（オプション）［出力ファイル名］［元ファイル名］`

lzh方式を使用して圧縮する。日本のWindowsでのファイルの圧縮方式としてよく使われている。lhaで圧縮したファイルには一般的に拡張子にlzhを付けることになっている。使用例を以下に示す。

```
$ lha a sample.lzh *  ←カレント・ディレクトリのファイルをすべて圧縮する
$ lha e sample.lzh    ←lhaファイルを展開する
```

表1　lhaの主なオプション

オプション	機能
a	アーカイブにファイルを追加・上書きする。アーカイブ・ファイルが存在しない場合は新規作成する
e	展開する
l	アーカイブ内容を表示する
u	更新されたファイルを更新する
d	アーカイブから削除する

mimencode　MIME（base64）にエンコード・デコード

構文　`mimencode（オプション）［入力ファイル名］（-o［出力ファイル名］）`

バイナリ・ファイルをMIMEコードにエンコードしたり，MIMEコード・ファイルをデコードしたりする。エンコードするときは-bオプション，デコードするときは-uオプションを付ける。オプションを指定しなかった場合はBase64方式でエンコードする。“-o［出力ファイル名］”を指定しなかった場合は標準出力に出力する。

使用例を以下に示す。

```
$ mimencode sample -o sample_mime      ←エンコードする
$ mimencode -u sample_mime  -o sample  ←デコードする
```

tar　.tarファイルの圧縮・展開

構文　**tar（オプション）**

tar形式のファイルを圧縮・展開する。また，zまたはZオプションを併用することで，tar.gzおよびtar.Zファイルを処理できる。cpioと同様にバックアップ・ファイルの作成も可能。使用例を以下に示す。

関連 **cpio,gzip,zip**

```
$ tar cvf sample.tar ~/     ←ホーム・ディレクトリ内を圧縮する
$ tar xzvf sample.tar.gz    ←tar.gzファイルを展開する
$ tar cvf /dev/fd0 ~/*      ←ホーム・ディレクトリ内をフロッピ・ディスクにバックアップ
```

表1　tarのオプション1

オプション	機能
A	tarファイルをアーカイブに追加する
c	アーカイブを新規に作成する
d	アーカイブとファイルシステムを比較する
r	アーカイブの後ろにファイルを追加する
t	アーカイブの内容を表示する
u	アーカイブ内のファイルより新しいファイルのみを追加する
x	アーカイブからファイルを取り出す
--delete	アーカイブからファイルを削除する
[ファイル名]	ファイル名を指定する
[ディレクトリ名]	ディレクトリ名を指定する

表1　tarのオプション2

オプション	機能
--atime-preserve	ダンプしたファイルのアクセス時刻を変更しない
-b, --block-size［数値］	ブロック・サイズを指定した数値*512バイトにする。数値の初期値は20
-B, --read-full-blocks	読み込みと同時にブロック化し直す
-C, --directory［ディレクトリ名］	ディレクトリを移動してから動作を始める
--checkpoint	アーカイブの読み込み中にディレクトリ名を表示する
-f, --file［ファイル名またはデバイス］:F	指定したアーカイブ・ファイルまたはデバイスを使用する。初期値は/dev/rmt0
--force-local	アーカイブ・ファイル名にコロンが存在してもローカル・ファイルとして扱う
-F, --info-script F, --new-volume-script F	各テープの最後にスクリプトを実行する
-G, --incremental	旧GNU形式のインクリメンタル・バックアップにより作成/一覧表示/抽出を行う
-g, --listed-incremental F	新GNU形式のインクリメンタル・バックアップにより作成/一覧表示/抽出を行う
-h, --dereference	シンボリック・リンクをダンプしない
-i, --ignore-zeros	ゼロのみからなるブロックを無視する
--ignore-failed-read	読み込みが不能なファイルに対して，ステータスコートが0でない限りは終了しない
-iまたは-I*1, --bzip	bzip2を通して処理する
-k, --keep-old-files	ファイルが即存する場合は上書きしない
-K, --starting-file F	作業開始位置を指定する
-l, --one-file-system	ローカルファイル・システムに限定してアーカイブを作成する
-L, --tape-length［数値］	指定した数値*1024バイト書き込んだ後にテープの交換する
-m, --modification-time	ファイルの変更時間を抽出しない
-M, --multi-volume	マルチボリュームのアーカイブの作成/一覧表示/抽出を行う
-N, --after-date［日時］, --newer［日時］	指定した日より新しいファイルだけ格納する
-o, --old-archive, --portability	V7形式のアーカイブを書き込む
-O, --to-stdout	ファイルを標準出力に書き出す
-p, --same-permissions, --preserve-permissions	すべてのアクセス情報を書き出す
-P, --absolute-paths	ファイル名の先頭のスラッシュ(/)を取り除かない
-R, --record-number	メッセージと共にレコード数を表示する
--remove-files	アーカイブに追加後にファイルを削除する
-s, --same-order, --preserve-order	抽出するファイル名をソートし，表示する
--same-owner	所有者属性を保持したまま抽出する
-S, --sparse	少ないファイルを効率的に修理する
-T, --files-from F	指定したファイルから抽出または作成するファイル名を読み込む
--null	文字コード0で終了したファイル名を読み込む。ただし，-Cオプションを指定することはできない
--totals	書き込まれた要領を表示する
-v, --verbose	処理したファイルの一覧を詳細に表示する
-V, --label［ボリューム名］	指定したボリューム名のアーカイブを作成する
-w, --interactive, --confirmation	動作するごとに確認する
-W, --verify	アーカイブを書き込み後に照合する
--exclude［ファイル名］	指定したファイルを除外する
-X, --exclude-from［ファイル名］	指定したファイルに記述されているファイルを除外する
-Z, --compress, --uncompress	compressを通して処理する
-z, --gzip, --ungzip	gzipを通して処理する
--use-compress-program	PROG 指定したプログラムを通して処理する
--block-compress	ブロック化してテープに書き込む
-［0〜7］［lmh］	ドライブと密度を指定する

＊1　今のところどのオプションでも指定できるが，将来-iオプションには他の機能が割り当てられる予定。

unzip　.zipファイルを展開

構文　**unzip（オプション）［圧縮ファイル名］［出力ファイル］**

Imploding法を使用した圧縮ファイルを展開する。Windowsを初めとした多くの機種で使われている圧縮方式。zipという拡張子の付いたファイルを展開するために使用する。使用例を以下に示す。

関連 zip

```
$ unzip sample.zip
```

表1　unzipの主なオプション

オプション	機能
-Z	圧縮情報を表示する。zipinfoと同等
-p	標準出力する
-l	圧縮情報を短い表示方式で表示する
-z	アーカイブ・コメントを表示する

uudecode　エンコードされているファイルをバイナリ・ファイルに戻す

構文　**uudecode［ファイル名］**

uuencodeでエンコードされたASCIIファイルをバイナリ・ファイルに戻す。変換後はuuencodeで指定したファイル名のファイルに保存される。使用例を以下に示す。

関連 uuencode

```
$ uudecode sample_encode
```

uuencode　バイナリ・ファイルをエンコードする

構文　**uuencode ［ファイル名］［エンコード・ファイルに添付する名前］**

バイナリー・ファイルをASCIIコードのみを使った形式に変換（エンコード）する。読める文字のみを使用するため，メールなどに添付して送ることができる。このとき，標準出力として表示されるので，ファイルとして残すにはリダイレクトを使用してファイルへ書き出す必要がある。デコードはuudecodeで行う。使用例を以下に示す。

関連 **uudecode**

```
$ uuencode sample_binary binary > sample_encode
```

zcat　gzipやcompressで圧縮されたファイルの内容を表示

構文　**zcat ［ファイル名］**

zgipなどで圧縮されているファイルの内容を表示する。zcatのほかにzmore，zlessがあり，moreおよびlessと同様な動作をすることができる。使用例を以下に示す。

関連 **gzip，gunzip，cat，more，less**

```
$ zcat sample.gz
```

zip .zipファイルに圧縮

構文 `zip（オプション）［出力ファイル名］［入力ファイル］（-xi ［リスト名］）`

Imploding法を使用して圧縮する。Windowsを初めとした多くの機種で使われている圧縮方式。zipで圧縮したファイルには一般的に拡張子にzipを付けることになっている。オプションを省略した場合は圧縮と同じ。使用例を以下に示す。

関連 `unzip`

```
$ zip sample.zip sample
```

表12　zipの主なオプション

オプション	機能
-d	アーカイブファイル内から指定したファイルを削除する
-m	アーカイブファイル内から指定したファイルを移動する
-k	ファイル名をMS-DOS（8.3）フォーマットにする
-q	クイック圧縮をする
-z	コメントを追加する
-t	指定した日付以降のファイルを対象とする
-u	変更または新しいファイルのみアップデートする
-l	リターンコードをLFからCR+LFに変換する
-ll	リターンコードをCR+LFからLFに変換する
-xi ［リスト名］	指定より圧縮するファイルのリストを読み込む

mattrib　MS-DOSファイルの属性を変更

構文　**mattrib（オプション）［ファイル名］**

MS-DOS方式ファイルの属性を表示・変更する。例えば，フロッピ・ディスク内のテキスト・ファイルを読み込み専用属性にする場合を以下に示す。

関連 mdir

```
$ mattrib a:*
             A:/disk_dir
  A          A:/sample.txt
  A          A:/sample.jpg
$ mattrib +r a:sample.txt
$ mattrib a:*
             A:/disk_dir
  A   R      A:/sample.txt
  A          A:/sample.jpg
```

表1　mattrib

オプション	機能
-?	ファイルの?属性を削除する。ただし，?はa，h，sのいずれか
+?	ファイルに?属性を付ける。ただし，?はa，h，sのいずれか

表2　属性

表示方式	オプション方式	意味
A	a	アーカイブ属性
R	r	読み込み専用属性
S	s	システム・ファイル属性
H	h	隠しファイル属性

mbadblocks　フロッピ・ディスクを検査し，不良ブロックにマークをつける

構文　**mbadblocks［デバイス名］**

フロッピ・ディスクをテストして不良ブロックにマークを付ける。フロッピ・ディスクのマウント作業は必要ない。使用例を以下に示す

関連 fsck

```
$ mbadblocks a:
```

mcd　MS-DOSディレクトリの移動

構文　**mcd［ディレクトリ名］**

MS-DOS方式のディレクトリに移動する。フロッピ・ディスクのマウント作業は必要はない。使用例を以下に示す。

関連 **mdir, cd**

```
$ mcd sample_dir
$ mdir
 Volume in drive A has no label
 Volume Serial Number is 272C-A82T
Directory for A:/sample

.            <DIR>      4-11-2002   1:19
..           <DIR>      4-11-2002   1:19
SAMPL~1  txt     2553  4-11-2002   3:21  sample.txt
        3 files               2553 bytes
                         1 264 227 bytes free
```

mcopy　MS-DOSファイルのコピー

構文　**mcopy（オプション）［コピー元のファイル名］［コピー先のファイル名またはディレクトリ］**

MS-DOSファイルをコピーする。フロッピ・ディスクのマウント作業は必要はない。-tオプションを付けるとテキストの改行コードを変換してコピーする。また, -vオプションを付けるとコピー後に照合する。例えば, ディスクのテキスト・ファイルを改行コードを変換してユーザーのホーム・ディレクトリにコピーする場合を以下に示す。

関連 **mdir,mdel,mmove,cp**

```
$ mcopy -t a:*.txt /home/user1
```

mdel　MS-DOSファイルの削除

構文　**mdel［ファイル名］**

MS-DOSファイルを削除する。フロッピ・ディスクのマウント作業は必要はない。例えば，フロッピ・ディスクの特定のディレクトリ内のファイルをすべて削除する場合を以下に示す。

関連 mcopy，mdir，mmove，mrd，rm

```
$ mdir sample_dir
$ mdel *
```

mdir　MS-DOSファイルやディレクトリの情報を表示

構文　**mdir（オプション）［ファイル/ディレクトリ名］**

MS-DOS方式のディスクのファイル情報を表示する。フロッピ・ディスクのマウント作業は必要はない。-aオプションを付けると隠し属性が付いているファイルも表示し，-wオプションを付けると詳細情報を表示せずにファイル名を横並びに表示する。また，-Xオプションはディスク内のすべてのファイルを対象にし，ファイル名だけを表示する。使用例を以下に示す。

関連 mcopy，mcd，mdel，mmd，mrd，mmove，mtype，mren，mattrib，ls

```
$ mdir
 Volume in drive A has no label
 Volume Serial Number is 272C-A82T
Directory for A:/sample

.            <DIR>      4-11-2002   1:19
..           <DIR>      4-11-2002   1:19
SAMPL~1  txt     2553  4-11-2002   3:21  sample.txt
        3 files              2553 bytes
                         1 264 227 bytes free
```

mformat　MS-DOSフォーマットを行う

構文　**mformat（オプション）［ドライブ］:**

フロッピ・ディスクなどをMS-DOS方式でフォーマットする。使用例を以下に示す。

関連 fdformat, mdir, mlabel

```
$ mformat a:
```

表3　mformatの主なオプション

オプション	機能
t	シリンダ数を設定する
h	ヘッダー番号を指定する
s	トラック中のセクター数を指定する
l	ボリューム・ラベルを指定する
2	2メガ・バイト・フォーマットする
1	2メガ・バイト・フォーマットを使用しない
M	ソフトウエア・セクター・サイズを指定する
X	XDFディスクのフォーマットを行う
C	MS-DOSファイル・システムを作成する
n	シリアル・ナンバーを設定する
F	FAT32フォーマットをする
c	シリンダ・サイズを設定する
r	ブート・ディレクトリのサイズを設定する

mlabel　フロッピ・ディスクにボリューム・ラベルを付ける

構文　**mlabel（オプション）［ドライブ］:［ラベル］**

MS-DOSディスクのボリューム・ラベルを変更する。ラベルを指定しない場合は対話方式でボリューム・ラベルを入力する。フロッピ・ディスクのマウント作業は必要はない。

なお-sオプションを付けるとラベルを表示，-cを付けるとラベルを消去する。使用例を以下に示す。

関連 mdir,mformat

```
$ mlabel -s a:
 Volume has no label
$ mlabel a: DOS-FD
```

mmd　MS-DOSディレクトリの作成

構文　**mmd ［ディレクトリ名］**

MS-DOSディスクにディレクトリを作成する。フロッピ・ディスクのマウント作業は必要はない。使用例を以下に示す。

関連 **mdir, mrd, mkdir**

```
$ mmd sample_dir
```

mmove　MS-DOSファイルを移動する

構文　**mmove ［移動元のファイル名］［移動先のファイル名またはディレクトリ］**

MS-DOSファイルを移動する。また, ファイル名を変更する場合にも使用できる。フロッピ・ディスクのマウント作業は必要はない。例えば, ディスクのJPEGファイルをすべてユーザーのホーム・ディレクトリに移動する場合を以下に示す。

関連 **mcopy, mdir, mv**

```
$ mmove *.jpg /home/user1
```

mrd　MS-DOSディレクトリの削除

構文　**mrd ［ディレクトリ名］**

MS-DOSディスクのディレクトリを削除する。ただし, ディレクトリ内は空でなくてはならない。フロッピ・ディスクのマウント作業は必要はない。使用例を以下に示す。

関連 **mdir, mmd, rmdir**

```
$ mrd sample_dir
```

mren　MS-DOSファイルのファイル名を変更

構文　**mren［ファイル名］［新しいファイル名］**

MS-DOSファイルの名前を変更する。フロッピー・ディスクのマウント作業は必要はない。使用例を以下に示す。

関連 **mdir, mv**

```
$ mren sample.txt rename.txt
```

mtype　MS-DOSファイルの内容を表示

構文　**mtype（オプション）［ファイル名］**

MS-DOSファイルの内容を表示する。フロッピー・ディスクのマウント作業は必要はない。-tオプションをつけるとテキストの改行コードを変換して表示する。使用例を以下に示す。

関連 **mdir, cat**

```
$ mtype a:sample.txt
このファイルはMS-DOSファイルです。
```

cal　カレンダを表示

構文　cal（オプション）（[月]［年]）

カレンダをテキスト・ベースで形成して表示する。-iオプションを付けるとユリウス暦で表示する。また，-yオプションを付けると現在の年のすべてのカレンダを表示する。なお，オプションを指定しないと現在の月のカレンダを表示する。使用例を以下に示す。

関連 date

```
$ cal
      4月 2002
日 月 火 水 木 金 土
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
```

echo　引数に与えられた文字列を表示

構文　echo（オプション）［文字列]

文字列に記述された内容を標準出力に出力する。また，-eオプションを指定することでエスケープ・コードを使用できる。エスケープ・コードには**表1**のようなものがある。なお，-nオプションを付けると最後の改行を出力しない。使用例を以下に示す。

```
$ echo -e "sample1¥tsample2¥nsample3"
sample1 sample2
sample3
```

表1　エスケープ・コード

表記	機能
\a	アラーム（ベル）を鳴らす
\b	バック・スラッシュ
\c	最後の改行の出力をしない
\f	フォーム・フィード（form feed）を作る
\n	改行（LF）
\r	改行（CR）
\t	水平タブ
\v	垂直タブ
\\	バック・スラッシュ
\nnn	ASCIIコードが（8進数で）nnnの文字

man　オンライン・マニュアルを参照

構文　**man（オプション）［コマンド］**
man（-k ［キーワード］）

コマンドの使用方法を表示する。passwdなど複数のセクションに登録されているものは，セクション番号でどれについて知りたいのかを指定する。使用例を以下に示す。

関連 cat，more，less

```
$ man 5 passwd
```

表1　manの主なオプション

オプション	機能
-C［設定ファイル］	使用するman.confを指定する。初期設定では“ /usr/lib/man.conf ”になっている
-P pager	使用するページャを指定する。初期設定ではlessになっている
-t	印刷用にページを整形する。プリンタにリダイレクトする必要がある
［セクション番号］	セクション番号（表2）を指定する
-k［キーワード］	すべてのマニュアル・ページからキーワードを検索する

表2　セクション番号

番号	セクション
1	ユーザー・コマンド
2	システム・コール
3	関数やライブラリ・ルーチン
4	特殊ファイル，デバイス・ドライバ，ハードウエア
5	設定ファイルとファイル形式
6	ゲームとデモ
7	その他（文字セット，ファイル・システム・タイプなど）
8	システム管理用コマンド

tee　標準入力を標準出力とファイルに出力

構文　tee［オプション］［ファイル名］

標準入力を標準出力とファイルに同時に出力する。-aまたは--appendオプションを付けるとファイル内容の上書きを禁止する(標準では既存のファイルがあったときに上書きする)。また，-iまたは--ignore-interruptsオプションを付けると割り込みシグナルを無視する。例えば，teeを利用してtelnetのログを保存する場合を以下に示す。

関連 |

```
$ telnet sample.nikkeibp.co.jp | tee telnet.log
```

which　コマンドを探す

構文　which［コマンド］

コマンドを探し出してフル・パスで表示する。エイリアスがあるときはそれも表示する。ただし，パスが通っている物でないと表示されない。パスが通っていないファイルを探すときはfindやlocateを使用するとよい。使用例を以下に示す。

関連 find，locate，ls

```
$ which ls
alias ls='ls -F --color=auto'
        /bin/ls
```

	コマンド	機能	ページ数
ファイル管理	ls	ファイルやディレクトリの情報を表示する	23
	mkdir	ディレクトリを作成する	25
	mktemp	ランダムなファイル名の空ファイルを作成する	25
	mv	ファイルやディレクトリの移動・名前の変更をする	25
	od	バイナリ・ファイルの内容を閲覧する	26
	pwd	現在のディレクトリの場所を確認する	27
	rm	ファイルやディレクトリを削除する	27
	rmdir	ディレクトリを削除する	28
	split	ファイルを分割する	28
	touch	ファイルのタイムスタンプを変更する	29
	updatedb	locate用ファイル・データベースを更新する	29
システム管理	clock	ハードウエア内部のクロックを設定する	30
	date	日付・時間を表示・設定を行う	30
	fastboot	システムを高速に再起動する	31
	fasthalt	システムを高速にシャットダウンする	32
	finger	ユーザー情報を表示する	32
	free	メモリーの使用状況を表示する	33
	groupadd	グループを追加する	33
	groupdel	グループを削除する	33
	groupmod	グループ情報を変更する	34
	halt	システムをすぐにシャットダウンする	34
	id	ユーザーやグループIDを表示する	34
	last	最近ログインしたユーザーの情報を表示する	35
	lastlog	各ユーザーの最後にログインした日付を表示する	35
	login	ログインする	36
	passwd	ユーザーのパスワードを変更する	36
	pwconv	Shadowパスワードに移行する	36
	reboot	システムをすぐに再起動する	37
	shutdown	システムをシャットダウン・再起動する	37
	su	ユーザーを切り替える	38
	uname	システム情報の表示	38
	useradd	ユーザーを追加する	39
	userdel	ユーザーを削除する	39
	usermod	ユーザーのアカウント情報を変更する	40
	vmstat	メモリーやCPUの負荷率や使用状況を表示する	40
	w	ログインしているユーザー名とその作業の内容を表示する	41
	who	現在ログインしているユーザーを表示する	41
ジョブ・プロセス管理	at	指定時刻にジョブを実行する	42
	batch	自動的にジョブを実行する	42
	crontab	プログラムを定期的に実行するcrond用の設定ファイルを編集する	43
	kill	プロセスおよびジョブを強制終了する	44
	nice	優先順位を決めてコマンドを実行する	44
	nohup	ログアウトした後もコマンドを実行し続ける	44
	pidof	プロセスのpidを調べる	45
	ps	実行中のプロセスを表示する	46
	stop	バックグラウンドのジョブを停止する	47
	top	現在のシステムの状況の表示やプロセスの管理をする	47
テキスト・ファイル操作	cat	テキスト・ファイルの内容を閲覧する	48
	cut	テキスト・ファイルの各行から文節を取り除く	48
	grep	文字列を検索する	49

	コマンド	機能	ページ数
mtools関連	mtype	MS-DOSファイルの内容を表示する	82
その他	cal	カレンダを表示する	83
	echo	引数に与えられた文字列を表示する	83
	man	オンライン・マニュアルを参照する	84
	tee	標準入力を標準出力とファイルに出力する	85
	which	コマンドを探す	85

コマンド・リファレンス 2003

日経Linux2003年3月号
別冊付録

日経 NIKKEI Linux

©日経BP社 2003
■日経BP社に無断で転載・複製することを禁じます
2003年3月8日発行　編集:日経Linux編集部　制作:株式会社マップス　発行:日経BP社　発売:日経BP出版センター