先進ゲノム支援講習会 2018 基本コマンド

ディレクトリ、ファイル操作を中心としたLinux基本コマンドを扱います

本資料においてコマンドの前の $ マークはプロンプト（コマンド入力が待機中であることを意味する記号）を示しますので、入力は不要です

誤った操作などによって、入力が受け付けなくなってしまった（プロンプトが表示されなくなった）場合には、
ctrl-Cを押すことで、処理をキャンセルし入力待機状態に戻ることができる場合があります。
q を押すことで処理を中断させられる場合もあります。

また、タブキーを素早く２度押すと、利用可能なコマンドやファイル名が補完されます（候補が多すぎて表示しきれない場合 q を押すと途中でとめられます）
矢印キーの上下で過去に入力したコマンドの履歴が表示できます。
これらをなるべく活用することで手入力による打ち間違いを避けるのがコツです。

はじめに

スパコンログイン直後は、スパコンシステム環境への入り口であるゲートウェイノード(gw)にいます。
gw では解析処理等を行うことは想定されておらず、負荷の高い処理を行うと他のユーザにも影響がでることがある。
そのため、スパコンにログインした後に
$ qlogin
コマンドを実行し、計算ノード（インタラクティブノード）に移動します。
$記号はプロンプトを示しますので入力不要です。

qlogin を実行する際に、パスワードを要求されます。
スパコンアカウントのパスワードを入力してください。
毎回パスワードを入力するのは手間なので、これを回避する方法もあります（後述）。

計算ノードから抜けるには
$ exit
を実行します。さらにgwにおいて exit を実行するとスパコンからログアウトできます。

---

1. 現在の状況を把握する (pwdとls)

pwd: (print working directory)) 現在自分がいるディレクトリ（フォルダ）を表示します
$ pwd
ログイン直後には各ユーザーのホームディレクトリ /home/userXXX/ にいるはずです。

ls: 現在のディレクトリ内にあるファイルおよびディレクトリの一覧 (list) を表示します。
$ ls
スパコンにログインしたのが初めてであれば ↑のコマンドを実行しても何も表示されないかもしれません。

不可視ファイル (. で始まるもの）を含めて表示させます。
$ ls -a
このコマンドによって表示されたディレクトリのうち、
. は現在のディレクトリ、.. は一つ上の階層のディレクトリ（親ディレクトリ）を示します。
また、-a はコマンドラインオプションのひとつです。
linuxのコマンドの多くはコマンドラインオプションを使うことで、既定の動作を変更させることができます。

ファイルの更新日時やファイルサイズも含めて表示させます。
$ ls -a -l
上記は ls -al としても同様の結果が得られます。
また、多くのシステムでは ll が "ls -l” のショートカットとして利用できます。

$ ls --help
多くのコマンドでは -h -help --help などのオプションをつけることでヘルプを表示できます。

---

2. Linuxのディレクトリの基本構造

/ (ルートディレクトリ)
　/bin（おもにLinuxシステムの動作に最低限必要な実行ファイルを格納する）
　/usr（各種アプリケーションと、それに付随するファイルを格納する）
　　/usr/local/bin ユーザーがインストールしたアプリケーションがされる場所
　　/usr/local/pkg/ NIGスパコンでは各種解析ツールがバージョンごとに置かれている
　　/usr/local/seq/ NIGスパコンではDDBJに登録されている塩基配列データやBLASTデータベースが置かれている
　/home 各ユーザーごとのホームディレクトリが格納されている

/home/userXXX/　userXXXのホームディレクトリ
/home/userYYY/　userYYYのホームディレクトリ
/home/userZZZ/　userZZZのホームディレクトリ

UNIX/Linuxではファイルやディレクトリごとに所有者や書き込み・読み込み権限（パーミッション）が設定されており、
通常は他のユーザーからはファイルが見られないようになっています。
同じグループに属するユーザーであれば一般的には閲覧可能です。
自分が所有するファイルであれあ権限を変更することも可能です（chmodコマンド）

上記は@tomoya氏のBlogを参考にしました
https://qiita.com/tomoya1993/items/c0f1592faa12478ffefa

---

3. ディレクトリの基本操作

ディレクトリの新規作成
mkdir コマンド (make directory)を使い “test"という名称のディレクトリを作ります。
$ mkdir test

ディレクトリの移動
cd コマンド (change directory) を使い作成したディレクトリに移動します
$ cd test

元の（一つ上の）ディレクトリに戻ります。
$ cd ..
.. は親ディレクトリを示します

ディレクトリを削除します。
$ rmdir test
このコマンドは空のディレクトリに対してしか使用できません。
ディレクトリの中身も含めて削除する場合には代わりに
$ rm -r test
コマンドを使用してください

/usr/local/bin ディレクトリに移動します。
$ cd /usr/local/bin
/ （ルートディレクトリ）を起点にした場所の示し方を「絶対パス」と呼びます。

/usr/local/bin から /usr/local/pkg ディレクトリに移動します。
$ cd ../pkg

現在いる場所を起点にした場所の示し方を「相対パス」と呼びます。

ホームディレクトリに戻ります
$ cd
または
$ cd ~
~ はホームディレクトリを示す記号です。

---

4. touchコマンドとrmコマンド(ファイル作成と消去)

touchコマンドを使い空のテキストファイル (a.txt) を作成します
$ touch a.txt
なおテキストファイルであることが目で見てわかりやすいように拡張子 .txt をつけていますが、
Windowsとは異なり、Linuxでは拡張子によってファイルの種類を認識することはありません。
したがって、拡張子は必須ではありません。（touch aでaという名称の空のファイルが作られます）
また、Linuxではファイル名の大文字小文字を区別して認識します。(win/macとは異なる）
したがって、a.txtとA.txtは別のファイルとして扱われます。

ll コマンド（ls -l）でファイルが作成できていることを確認します。
$ ll

なお、touch コマンドは本来はファイルのタイムスタンプを更新するためのコマンドです。
動作確認のため、再び touch コマンドを用いた後、タイムスタンプを見てみます。
$ touch a.txt
$ ll

ファイルの消去
$ rm a.txt

rm a.txt b.txt c.txt というように複数のファイルを指定して削除することも可能です。
また、ワイルドカード（後述）を利用してパターンに合うファイルを一括削除することもできます。
-r オプション（recursive）をつけて実行することでフォルダ内のファイルすべてを再帰的にすべて消去することができます。
例）rm -r test　← testディレクトリを中身を含めて消去します。

---

5. echo コマンド、cat コマンド、リダイレクト

echoコマンド：文字列を画面に出力します
$ echo Hello

> を使うと画面の出力をテキストファイルに書き込むことができます（リダイレクト）。
$ echo Hello > a.txt
a.txtがすでに存在していた場合、上書きされますので気をつけてください。

>> で追加書き込みできます
$ echo world >> a.txt

他のコマンドの出力結果もファイルに書き込めます。
$ ll > b.txt

catコマンド：ファイルの中身を画面に出力します
$ cat a.txt
catコマンドは本来は複数のファイルを連結（concatenate）するコマンドです。
$ cat a.txt b.txt > c.txt
でa.txtとb.txtを連結し、その結果をc.txtに書き込みます。

---

6. ワイルドカード

.txt で終わるファイルの一覧を表示
$ ls *.txt
*は0文字以上の任意の文字列を示します

$ ls ?.txt
?は任意の 1 文字を示します
$ ls ??.txt
と打った場合には ab.txt や XY.txt にマッチしますが、a.txtにはマッチしません

これまでに作成した .txtファイルを削除します
$ rm *.txt
(単に rm * と打つと全てのファイルが消えてしまいますのでご注意ください。
また、-r オプションをつけて rm -r * を実行すると、カレントディレクトリ以下にある
ファイルがすべて消えます。linuxでは一度削除したファイルを
復活させることはできませんので、使用する場合には注意してください)

---

7. cpコマンド（ファイルのコピー）とmvコマンド（ファイルの移動）

これまでの復習を兼ねて練習用のファイルとディレクトリを作ります
$ cd # ホームディレクトリに移動
$ mkdir dirA
$ mkdir dirB
$ echo "File A" > a.txt
$ echo "File B" > b.txt

a.txtをAディレクトリに移動します。
$ mv a.txt dirA/
(最後の/はなくても構いません）

b.txtをBディレクトリにコピーします。
$ cp b.txt dirB/
(最後の/はなくても構いません）

ファイルを別名で複製することもできます。
b.txtをb2.txtとして複製します
$ cp b.txt b2.txt
(最後の/はなくても構いません）

mvコマンドはファイル名を変更するときにも使用します。
b.txtをc.txtに名前を変えます
$ mv b.txt c.txt

dirBに移動します
$ cd dirB
一つ上のディレクトリにある c.txt をカレントディレクトリに移動します。
$ mv ../c.txt ./
結果的に b.txtとb2.txtとc.txtの２つがカレントディレクトリに存在するはずです。

b.txtとc.txtの両方を dirA にコピーします。
$ cp b.txt c.txt ../dirA/
このように複数のファイルを指定してまとめてコピーすることもできます。
mvコマンドも同様です。

一つ上の階層に戻ります
$ cd ..
dirBをdirAの中にコピーします。
ディレクトリをコピーするときは -r オプションを加えます

$ cp -r dirB dirA

最後に余分なファイルを削除します
$ rm -r b2.txt dirA dirB

---

8. headコマンドとtailコマンド, wc

それぞれファイルの先頭および末端部分を表示します。

$ head -10000 hoge.txt > hoge2.txt
で先頭から10000行を抽出し、ファイルに書き込みます。
$ tail -10000 hoge.txt > hoge2.txt
で末端から10000行を抽出し、ファイルに書き込みます。

-10000の部分を省くと10行表示されます。

参考）
-f オプションをつけるとファイルの監視モードになり、
ファイルが追記されると自動で表示が更新されるようになります。
ログファイルなどをリアルタイムで監視するのに便利です。
$ tail -f hoge.txt
終了するには ctrl-C を押します。

wcで行数を数えて見ます
$ wc hoge.txt
表示される数値は順に、行数、単語数、文字数を表します。
文字数には改行コードも含まれます。

-l をつけると行数のみ表示します
$ wc -l hoge.txt

---

9. less コマンド とmoreコマンド

cat コマンドはファイルの中身を一気に表示させるのに対し、
大きなファイルを部分ごとに表示させるのがlessやmoreコマンドです。

lessとmoreは多少の違いはありますが、lessはmoreの拡張版ですので、
lessを使うことの方が多いです。

$ less hoge.txt
で起動します。

矢印キーの上下左右でスクロール。
fとbで 1 画面ずつ上下にスクロールします。

/ で前方（現在の位置からファイル末端に向けて）への検索
? で後方（現在の位置からファイル末端に向けて）への検索
繰り返し検索する場合には n または N で次候補または前候補を表示します。

q でlessを終了します。

h でヘルプが表示できます。

参考）
lessコマンドは、デフォルトでは画面の右端で自動的に折り返されて表示されます。
この設定を無視して画面 1 行にファイルの内容を 1 行ずつ表示させる場合には
-S オプションを使用します。
$ less -S hoge.txt

---

10. パイプ

“ | “ を使うとコマンドの出力を他のコマンドに連結することができます（読み方はパイプ、パイプライン、バーチカルバーなど）

例）ファイルの先頭10000行を抽出した後、その末尾100行を表示させます。
$ head -10000 hoge.txt | tail -100
結果的に9901行目から10000行目が抽出されて表示されます。

複数のパイプを連結することも可能です。
例）ファイルの先頭10000行を抽出した後、その末尾100行を、lessを用いて表示させます。
$ head -10000 hoge.txt | tail -100 | less

---

11. ファイル内の検索 （grep）

ファイルの中から文字列AAAAを含んだ行を表示させます。
$ grep AAAA hoge.txt

パイプと組み合わせて使用することもできます
$ head -10000 hoge.txt | grep AAAA

使用例）

スパコンの実行中ジョブの一覧からytanizawによって投入されているジョブを表示させる。
$ qstat -u "*" | grep ytanizaw

---

12. パスを通す

パスとは、コマンド検索パスを意味します。
あるコマンドを実行するためには、そのコマンドの存在するディレクトリが環境変数 PATH で
指定されている必要があります。

たとえば、BLAST検索を実行するためのコマンド blastall を例にとります。
$ blastall
と打つと、blastall コマンドが実行されヘルプ画面が表示されます。

blastall コマンドの本体（実行ファイル）がどこにあるのかは、which コマンドを用いて調べることができます。
$ which blastall
これにより本体は /usr/local/bin/blastall であることがわかります。

ここでechoコマンドでPATHの設定内容を確認します。
$ echo $PATH
（Linuxでは変数の内容を参照する際には$記号をつける必要があります。）
“: (コロン)”で区切られていくつかのディレクトリが表示されますが、/usr/local/bin が PATHの中に含まれていることがわかります。
なお、$printenv PATHでも表示可能です。

ところで、前述の blastall は Legacy Blast と呼ばれており、
現在では後継の NCBI-Blast+ の使用が推奨されています。
NCBI-Blast+ パッケージの中の blastp (タンパク質配列同士の比較) を実行してみます。
$ blastp
「コマンドが見つかりません」というメッセージが表示されます。
これはPATHの中にblastpがあるディレクトリがPATHに含まれていないのが原因です。

blastp の実行ファイルは /usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/ の中にあります。
$ ls /usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/
blastp コマンドを実行できるようにするためには、次のようにPATHを設定する必要があります。
$ export PATH=/usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/:$PATH

後ろ側のPATHには$記号をつけることを忘れないでください。
既存のPATHの設定の前に新しい設定を付け加えるという意味です。

後側に付け足すこともできますが、PATHは先頭に近い方から探索されていくため、
前につけた場合とは意味が若干異なります。
（$ export PATH=$PATH:/usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/ ）

これでblastpコマンドが実行可能になりました。
$ blastp
でヘルプが表示されることを確認してください。

このように環境変数PATHを設定し、コマンドを実行できるようにすることを
「パスを通す」といいます。

参考１）
パスを通さなくても、次のように絶対パスあるいは相対パスを使って
ファイルの場所を指定して実行することも可能です。
/usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/blastp

参考２）
ログインをすると export コマンドで設定した環境変数の内容は失われてしまいますので、
ログインごとに再設定する必要があります。
頻繁に使うツールについては、exportコマンドを .bash_profile ファイル（または .bashrc ファイル）の中に記述することで
自動で設定するようにできます。
これらのファイルはログイン時に自動で読み込まれ、設定が反映されます。

---

wget インターネット上からファイルを取得する

URLを指定してインターネット上のファイルを取得します。
例）DDBJのFTPサーバーからシークエンスデータを取得します。
$ wget
ftp://ftp.ddbj.nig.ac.jp//ddbj_database/dra/fastq/SRA117/SRA117449/SRX456287/SRR11151187_1.fastq.bz2

ファイルの展開および圧縮

zip 形式 → 展開 unzip、圧縮 zip
gz 形式 → 展開 gunzip、圧縮 gzip
bz2 形式 → 展開 bzip2、圧縮 bzip2
tar or tar.gz 形式 → 展開および圧縮 tar コマンド

例 tar.gzファイルの展開
$ tar xvfz archive_file.tar.gz
xは展開（extract）, vは展開の過程を画面に表示（verbose）を表します。
zは対象が gz 形式で圧縮されている場合につけますが、最近はつけなくても展開できます。
fは展開する対象がファイルであることを示します。

---

scp ローカルマシンとリモートサーバー（スパコン）との間でのファイルの転送を行う

本講習ではSFTPソフトを用いてファイルの転送を行いますので、割愛します。

ファイルを検索する find

find 場所 -name パターン
でパターンに一致するファイルを検索します。

/usr/local/pkgの中からblastpという名前のファイルを探す
$ find /usr/local/pkg -name blastp

カレントディレクトリ以下からファイル名が .txtで終わるものを探す
$ find . -name *.txt

リンクを張る ln

リンク（シンボリックリンク）は、windows/Macのショートカット/エイリアスに相当し、
本体のファイルの代わりに使用することができます。

リンクの作成方法
blastpへのリンクを myblastという名前で作成する
$ ln -s /usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/blastp myblast

ためしに
$ ./myblast -h
と実行するとblastpが呼び出されヘルプが表示されます。
注）カレントディレクトリにあるファイルを実行する場合には、明示的に ./ をつける必要があります。

リンクの作成方法
blastpへのリンクをカレントディレクトリに作成する
$ ln -s /usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin/blastp ./
この場合にはカレントディレクトリに、本体と同じ blastp という名前のリンクが作成されます。

ためしに
$ ./blastp
と実行するとblastpが呼び出されヘルプが表示されます。
注）カレントディレクトリにあるファイルを実行する場合には、明示的に ./ をつける必要があります。

リンクを削除するには
$ unlink myblast
$ unlink blastp
とします。rmコマンドでも削除できますが、誤って実体のファイルを削除してしまわないようunlinkをお奨めします。

参考）
シンボリックリンクの他にハードリンクというものもありますが、
使用する機会はあまり多くはないと思われます。
-s オプションをつけないとハードリンクになります。

参考）
ディレクトリに対してリンクを作成することもできます。
$ ln -s /usr/local/pkg/ncbi_blast+/ncbi-blast-2.7.0+/bin ./blastbin
カレントディレクトリにblastbinという名称で作成されます

パスワードの変更

$ passwd

のあと、現在のパスワードを入力、
続いて新しいパスワードを２回（確認のため）入力します。

---

vi と emacs

どちらもターミナル上で動作するテキストエディタです。
本講習ではSFTPクライアントソフトを使ってリモートのファイルを編集しますので、
これらは使用しません。

遺伝研スパコンを含め、一部のLinuxシステムではviが標準のエディタになっているため、
最低限の操作方法は覚えておいた方が良いです。

起動
$ vi
または
$ vi ファイル名

viはコマンドモードと入力モードに分かれています。
入力モードに切り替えるには i をコマンドモードに戻るときは ESC を押します。
コマンドモードにおいて :w で保存、:q で終了します。

JAVA で作られたプログラムを動かすときの注意点

スパコン上でJAVAで作られたプログラムを動かす際に、
#Cannot creat Virtual Machine というエラーメッセージがでて動かない場合があります。
$ java

この場合には下記のように環境変数 _JAVA_OPTIONS を指定しておくとエラーが回避できることがあります。

export _JAVA_OPTIONS="-Xmx1g -XX:ParallelGCThreads=1”
1gの部分は 256m 512m or 2g などにした方が良い場合があります。
export _JAVA_OPTIONS="-XX:ParallelGCThreads=1”
だけでも良いかもしれません。

参考）
https://supcom.hgc.jp/japanese/utili_info/manual/faq.html#2829

---

qlogin時にパスワード入力を回避する

詳細は省きますが、パスワード認証のかわりに鍵認証を行うように設定します。

$ ls ~/.ssh/id_rsa.pub
を実行してみて、ファイルが存在していないことを確認します。
ファイルが存在していなければ、
$ ssh-keygen
を実行します。ファイルの保存場所やパスフレーズを聞かれますが、
すべてデフォルトのまま（空欄のまま）エンターキーを押します。
ファイルが存在していればこの操作は不要です。

次に、以下のコマンドを実行します。
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

以上です。

パーミッションの設定（chmod）

ls -lでファイルの情報を表示させると、次のように先頭に
rやwからなる10桁の文字列が表示されます。
-rw-r--r-- 1 tanizawa staff   0 3 15 14:45 a.txt
これはファイルやディレクトリのパーミッションを示しています。

最初の 1 文字目はファイル種別を表しており、
- は通常のファイル、lはシンボリックリンク、dはディレクトリを示しています。

2文字目から4文字目はファイルの所有者に対する権限を表し、
5文字目から7文字目はファイルの所有グループに対する権限を表し、
8文字目から10文字目はその他に対する権限を表しています。

- 権限なし（数値で表すと0）
r 読み取り（数値で表すと4）
w 書き込み（数値で表すと4）
x 実行可能（ディレクトリの場合は中のファイルにアクセス可能）（数値で表すと1）

たとえば、上記の a.txt の場合
ファイルの所有者は読み書き両方ができ(4+2=6)、
同一グループに属するユーザーおよびその他のユーザーは読み込みのみ可能(4)です。

これを数値で表すと 644 となります。

パーミッションを変更するには chmod コマンドを用います。
たとえば、
$ chmod a+x a.txt
とすると、すべてのユーザ (a)に対し実行権限(x)を付与する(+)という意味になり、
パーミッションは以下のようになります。
-rwxr-xr-x 1 tanizawa staff 0 3 15 14:45 a.txt

また、数値で直接指定することもでき、
$ chmod 751 a.txt
とすると、パーミッションは
-rwxr-x--- 1 tanizawa staff 0 3 15 14:45 a.txt
というようになります。

下記のリンク先を参考にしています。
https://qiita.com/shisama/items/5f4c4fa768642aad9e06
http://inaz2.hatenablog.com/entry/2013/09/01/224223

新しいアプリケーションをインストールする

ほとんどのプログラムは管理者権限を持っていなくても自分のホームディレクトリにインストールして使用することが可能です。
ソースコードをダウンロードして自分でコンパイル（＝実行可能なファイルを作成する）する場合もあれば、
コンパイル済みの実行ファイル（バイナリファイル）をダウンロードする場合もあります。
コンパイル済みのファイルをダウンロードする場合には、”Linux x86_64”用のものであればスパコンで動作します。

例）トランスクリプトームアッセンブリツール HISAT2 のインストール
配布元：https://ccb.jhu.edu/software/hisat2/index.shtml
ソースコード：ftp://ftp.ccb.jhu.edu/pub/infphilo/hisat2/downloads/hisat2-2.1.0-source.zip
バイナリファイル：ftp://ftp.ccb.jhu.edu/pub/infphilo/hisat2/downloads/hisat2-2.1.0-Linux_x86_64.zip

コンパイルの手順
ソースコード取得
$ wget ftp://ftp.ccb.jhu.edu/pub/infphilo/hisat2/downloads/hisat2-2.1.0-source.zip

展開
$ unzip hisat2-2.1.0-source.zip
ディレクトリに移動
$ cd hisat2-2.1.0
インストールの手順を確認する。
最初に読むべき重要なファイルは “README" のように大文字で記載されていることが多い。
HISATの場合は “MANUAL” がそれに該当する。
マニュアルに従い、make コマンドでコンパイルを行う
$ make

プログラムによっては make の前に configure を実行したり、後に make install を行うものもあるが、
基本的には マニュアルに従って行う。

Python、Ruby、Perlをホームディレクトリにインストールする（中級以上）

スパコンにはこれらがすでにインストールされているが、バージョンが古かったり、
必要な追加モジュールが不足していたりする場合がある。

自分のホームディレクトリ内であれば、管理者権限がなくても自由にインストールできる。
インストールには pyenv や rbenv といったツールを利用するのが簡単で、
かつ、複数のバージョンをインストールして異なるバージョンを切り替えて使うといった自由が効いて良い。

また、様々なプログラミング言語に対応した anyenv というものもある。

インストール方法については anyenv, pyenvなど各ツールで検索すると
日本語の情報が多く得られる。