# SCLS計算機システムの基礎

独立行政法人理化学研究所
HPCI計算生命科学推進プログラム

チーム員　波内 良樹
yoshiki.namiuchi@riken.jp

# この講習について

## 講習の対象の方

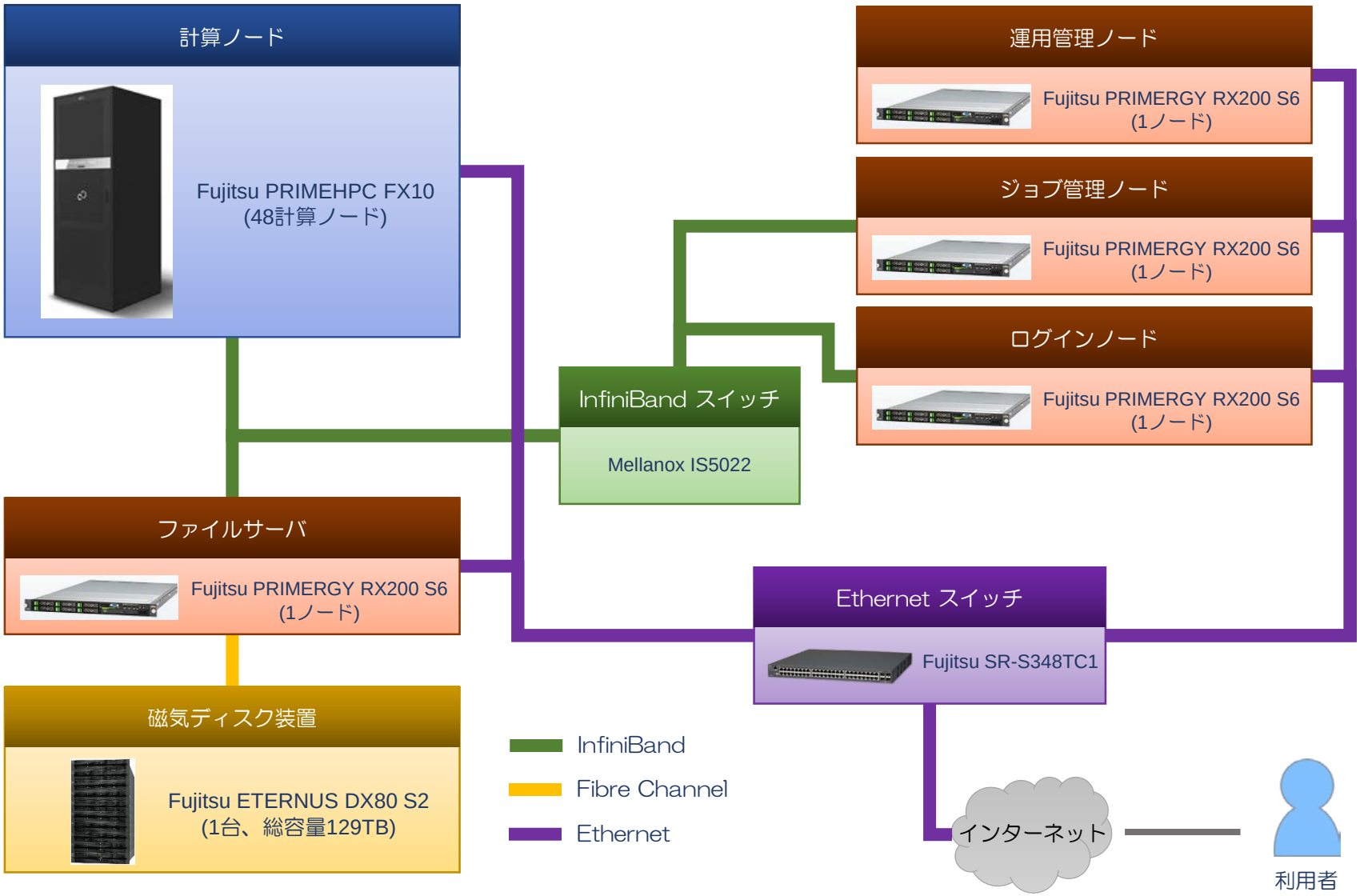
- SCLS計算機システムをはじめて使う
- UNIX/Linuxはあまり使ったことがない

## 講習の内容

1. SCLS計算機システムの概要
2. UNIX/Linuxの基礎
    a. ファイルとディレクトリ
    b. 基本的なコマンド (実習)
    c. ファイルとディレクトリのアクセス権
    d. シェルの機能 (実習)
3. ジョブの実行 (実習)

# 1. SCLS計算機システムの概要

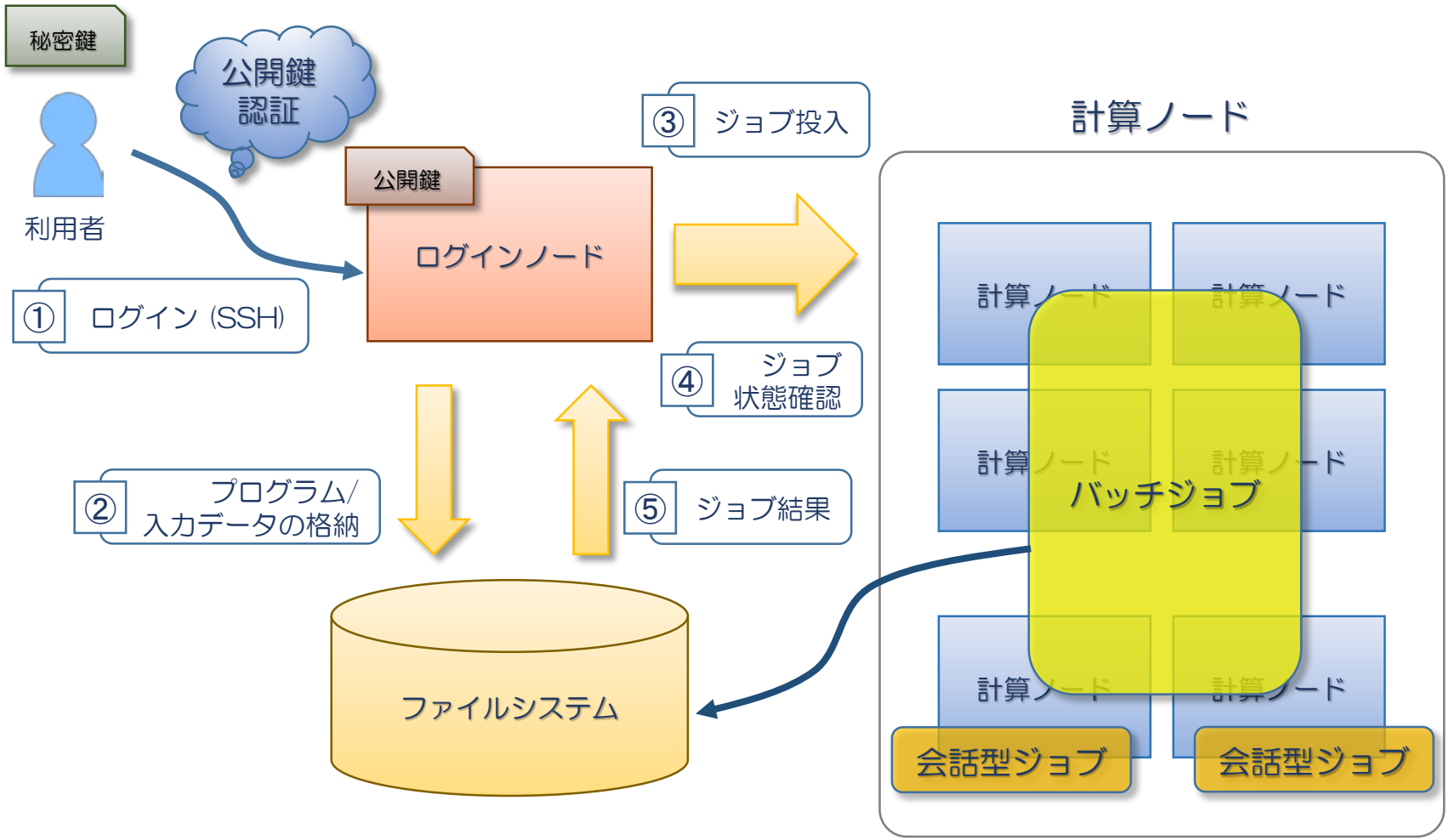
# SCLS計算機システムの構成



# スーパーコンピュータ「京」との比較

| | | スーパーコンピュータ「京」 | SCLS<br>計算機システム |
|---|---|---|---|
| CPU | 名称 | SPARC64™ VIIIfx | SPARC64™ IXfx |
| | 動作周波数 | 2 GHz | 1.65 GHz |
| | コア数 | 8 | 16 |
| | 理論性能 | 128 GFLOPS | 211 GFLOPS |
| システム全体 | ノード数 | 82,944 (計算ノードのみ)<br>88,218 (含IOノード) | 48 |
| | メモリ | 1.26 PB (計算ノード)<br>(16 GB/ノード) | 1.5 TB<br>(32 GB/ノード) |
| | ストレージ | 30 PB～ (グローバルファイルシステム)<br>11 PB～ (ローカルファイルシステム) | 129 TB |
| | 理論性能 | 10.62 PFLOPS (計算ノードのみ)<br>11.28 PFLOPS (全体) | 10.1 TFLOPS |

# 利用のながれ


秘密鍵
公開鍵
認証
利用者
公開鍵
ログインノード
① ログイン (SSH)
③ ジョブ投入
④ ジョブ
状態確認
② プログラム/
入力データの格納
⑤ ジョブ結果
ファイルシステム
計算ノード
計算ノード
計算ノード
計算ノード
計算ノード
計算ノード
バッチジョブ
会話型ジョブ
会話型ジョブ

# SCLS計算機システムを利用するためのソフトウェア

| | Windows | Linux / MacOS X |
|---|---|---|
| ログイン<br>(SSH) | PuTTY<br>Tera Term | ssh コマンド |
| ファイル転送<br>(SSH) | WinSCP | scp コマンド<br>sftp コマンド |

# 利用上の注意

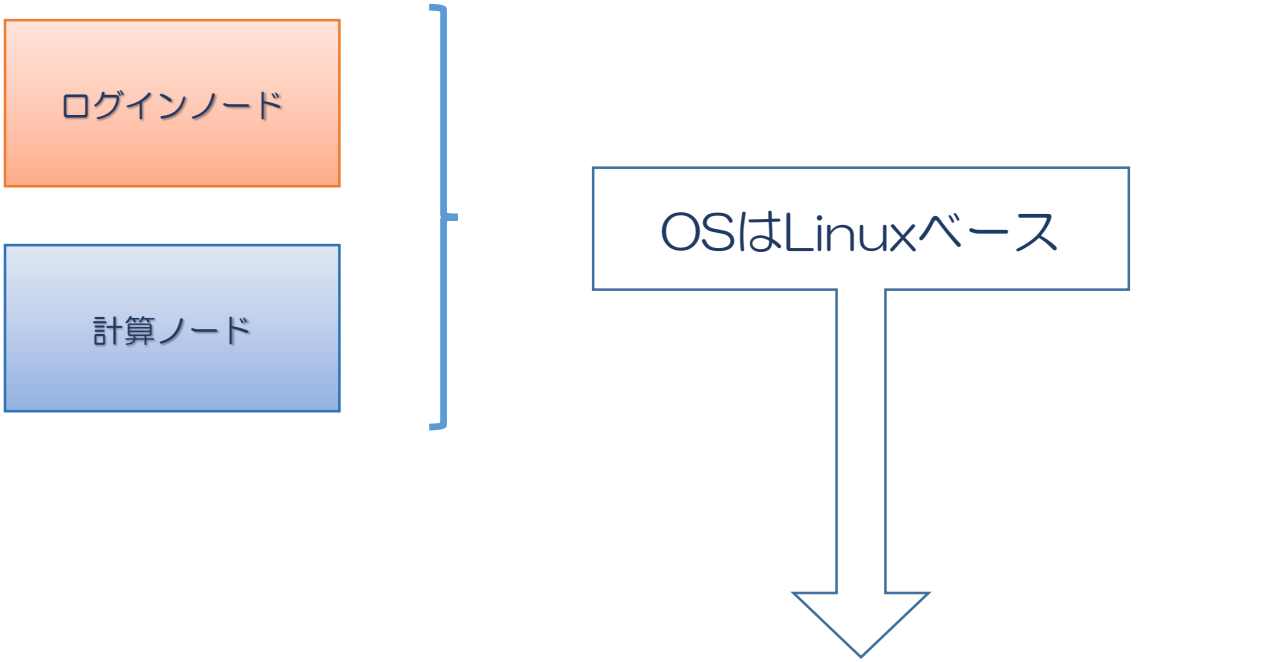
- ログインノードのリソースは、多くのユーザで共有するため、高負荷の処理は行わないようにして下さい。

- 使用状況によって、課題グループ単位で計算資源の使用制限を行う場合があります。

# 2. UNIX/Linuxの基礎

# SCLS計算機システムのOS

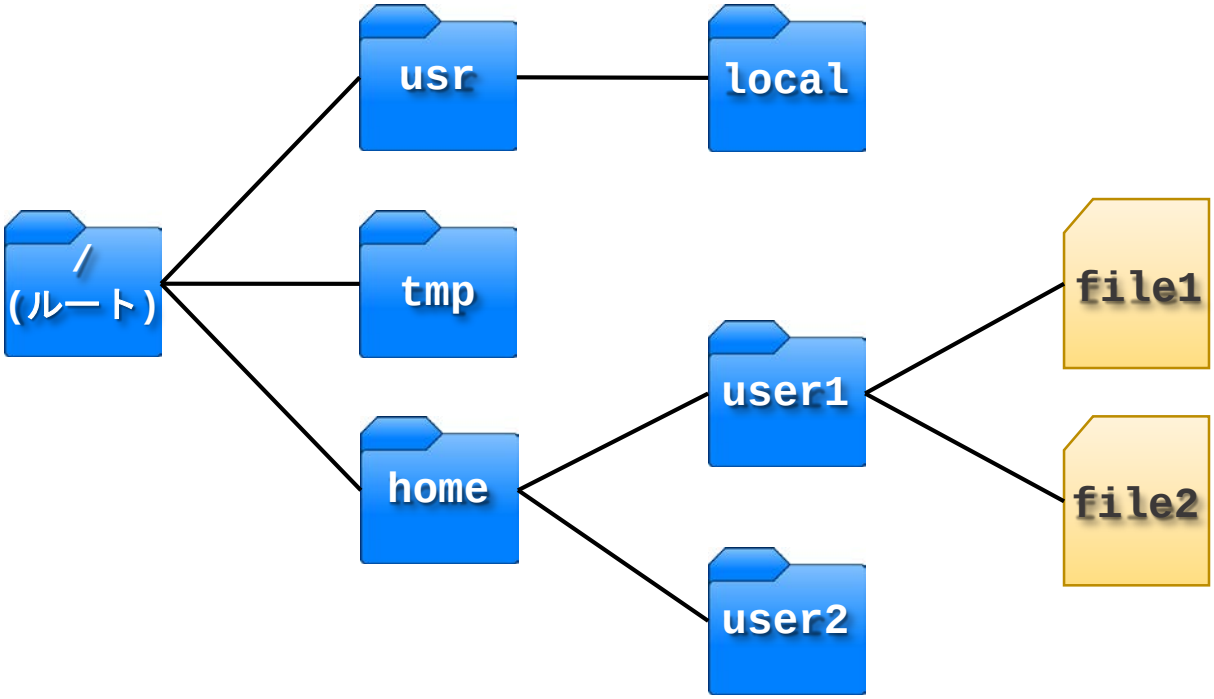
SCLS計算機システムは、



# 2.a ファイルとディレクトリ

# ファイルとディレクトリの概要

- ディレクトリはファイルの保存場所
- ディレクトリ構造は、ルートディレクトリを最上位とする木構造
- ファイルは、木構造のリーフノード
- Windowsのようなドライブの概念はない

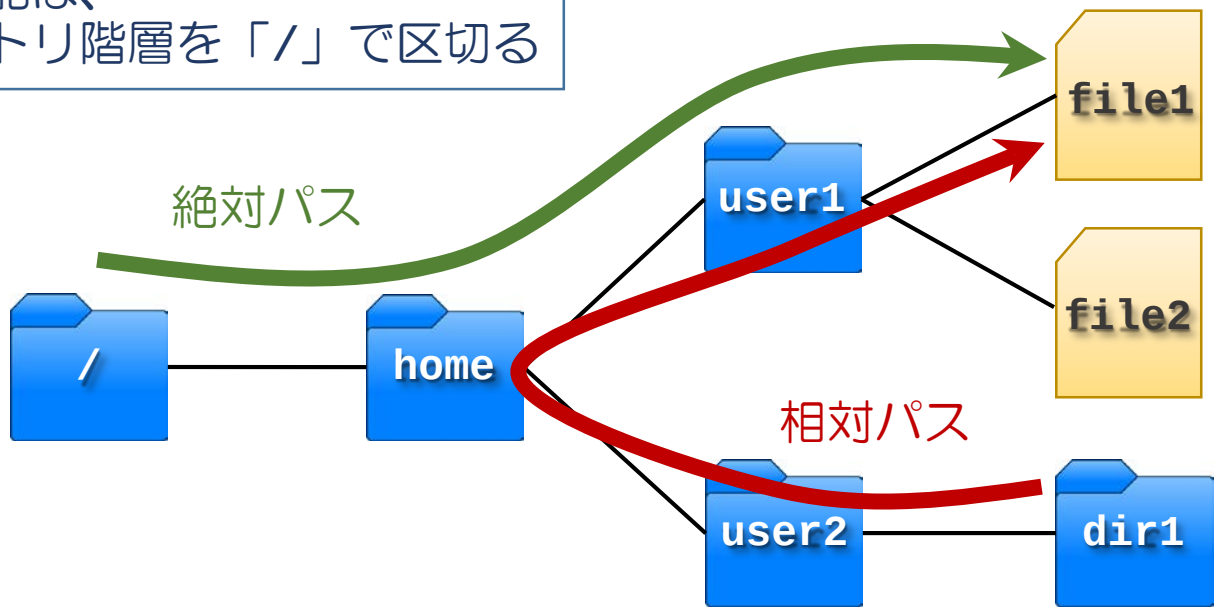


# ディレクトリの種類

- ルートディレクトリ
  - ディレクトリ構造の最上位のディレクトリ
  - パスを記述する場合の記号「/」
- カレントディレクトリ (ワーキングディレクトリ)
  - 現在作業を行っているディレクトリ
  - パスを記述する場合の記号「.」
- 親ディレクトリ (ペアレントディレクトリ)
  - ディレクトリ階層において1つ上位のディレクトリ
  - パスを記述する場合の記号「..」
- 子ディレクトリ (チャイルドディレクトリ)
  - ディレクトリ階層において1つ下位のディレクトリ
- ホームディレクトリ
  - ユーザがログインした時のカレントディレクトリ
  - パスを記述する場合の記号「~」
  - SCLS計算機システムのホームディレクトリは、/home/<ユーザ名>

# パスの表記

パスの表記は、
ディレクトリ階層を「/」で区切る




- 絶対パス（フルパス）
  - ルートディレクトリを基点とした表現
  - 「file1」の絶対パス → 「/home/user1/file1」
- 相対パス
  - 特定のディレクトリを基点とした表現
  - 「dir1」から「file1」の相対パス → 「../../user1/file1」

# 2.b 基本的なコマンド

(実習)

# 実習で使用するコマンド一覧

| | | |
|---|---|---|
| **pwd** | Print Working Directory | カレントディレクトリの絶対パスを表示 |
| **ls** | List Segments | ディレクトリの内容(ファイルとディレクトリの一覧)を表示 |
| **mkdir** | Make Directory | ディレクトリを作成 |
| **cd** | Change Directory | カレントディレクトリの変更 |
| **cp** | CoPy | ファイルやディレクトリのコピーを作成 |
| **vi** | VIsual editor / Visual Interface | テキストエディタ |
| **cat** | CATenate | 指定したテキストファイルの内容を表示 |
| **more** | MORE | テキストを1画面単位で表示 |
| **less** | LESS | moreのようなページャー |
| **diff** | DIFFerence | 2つのファイルの違いを探す |
| **mv** | MoVe | ファイルやディレクトリの移動または名前の変更 |
| **rm** | ReMove | ファイルやディレクトリを削除 |
| **rmdir** | ReMove DIRectory | 空のディレクトリを削除 |
| **man** | MANual | オンラインマニュアルを表示 |

# ディレクトリの作成

**pwd**
カレントディレクトリの絶対パスを表示

**ls** *[options] [file...]*
ディレクトリの内容(ファイルとディレクトリの一覧)を表示

**mkdir** *directory...*
ディレクトリを作成

```
[user1@scls ~]$ pwd
/home/user1

[user1@scls ~]$ ls

[user1@scls ~]$ mkdir dir1

[user1@scls ~]$ ls
dir1

[user1@scls ~]$ mkdir dir1/subdir1 dir1/subdir2

[user1@scls ~]$ ls dir1
subdir1  subdir2
```

カレントディレクトリのパスを表示

カレントディレクトリの内容を表示

ディレクトリdir1を作成

カレントディレクトリの内容を表示

ディレクトリdir1の配下に、subdir1, subdir2を作成

ディレクトリdir1の内容を表示

# カレントディレクトリの変更

**cd** *[dir]*

カレントディレクトリの変更

```
[user1@scls ~]$ cd /home/user1/dir1/subdir1

[user1@scls subdir1]$ pwd
/home/user1/dir1/subdir1

[user1@scls subdir1]$ cd ../subdir2

[user1@scls subdir2]$ pwd
/home/user1/dir1/subdir2

[user1@scls subdir2]$ cd ..

[user1@scls dir1]$ pwd
/home/user1/dir1
```

絶対パスで指定

カレントディレクトリのパスを表示

同じ親ディレクトリ配下のsubdir2を相対パスで指定

カレントディレクトリのパスを表示

親ディレクトリを指定

カレントディレクトリのパスを表示

# ファイル/ディレクトリのコピー

**cp** *[options] file... path*

ファイルやディレクトリのコピーを作成

```
[user1@scls dir1]$ cp -r ~namiuchi/lec/lec_scls .

[user1@scls dir1]$ ls
lec_scls  subdir1  subdir2

[user1@scls dir1]$ cp lec_scls/job_seq/seqjob.c subdir1

[user1@scls dir1]$ ls subdir1
seqjob.c

[user1@scls dir1]$ cd subdir1

[user1@scls subdir1]$ cp seqjob.c seqjob1.c

[user1@scls subdir1]$ ls
seqjob1.c  seqjob.c
```

ディレクトリlec_sclsを
カレントディレクトリにコピー

カレントディレクトリの内容を表示

ファイルseqjob.cを他のディレクトリにコピー

ディレクトリsubdir1の内容を表示

カレントディレクトリをsubdir1に変更

ファイルseqjob.cを
名前を変更してコピー

カレントディレクトリの内容を表示

# テキストファイルの作成/変更

**vi** *[options] [file...]*

テキストエディタ

```
[user1@scls subdir1]$ vi seqjob1.c
```

コマンドモード
各種編集や保存/終了などのコマンドを実行するモード
入力開始コマンドで入力モードに移行

入力モード
文字を入力するモード
(画面左下に、「-- INSERT --」と表示される)
ESCキーでコマンドモードに移行

## コマンドモードの主なコマンド

| | |
|---|---|
| h | 1文字左へカーソルを移動 |
| j | 1行下へカーソルを移動 |
| k | 1行上へカーソルを移動 |
| l | 1文字右へカーソルを移動 |
| a | カーソルの右から入力開始 |
| i | カーソルの左から入力開始 |
| o | カーソルの下に1行追加し入力開始 |
| x | カーソル上の1文字を削除 |
| dd | カーソルの行を削除（バッファにコピー） |

| | |
|---|---|
| yy | カーソルの行をバッファにコピー |
| p | バッファ内のテキストを下の行に挿入 |
| u | 直前の操作のUndo |
| U | 行全体の操作のUndo |
| . | 直前の操作の繰り返し |
| :w | 現在のファイルに保存 |
| :wq | 現在のファイルに保存して終了 |
| :q | 終了 |
| :q! | 強制終了 |

# テキストファイルの内容の表示

**cat** *[options] [file...]*

指定したテキストファイルの内容を表示

**more** *[options] [file...]*

テキストを1画面単位で表示

**less** *[options] [file...]*

moreのようなページャー

```
[user1@scls subdir1]$ cat seqjob1.c
```

seqjob1.cの内容を表示

```
[user1@scls subdir1]$ more seqjob1.c
```

seqjob1.cの内容を表示

- カーソルキー上下、改行キーで行送り/行戻り
- スペースキーまたは「f」キーで画面送り
- 「b」キーで画面戻り
- 「q」キーで終了

```
[user1@scls subdir1]$ less seqjob1.c
```

seqjob1.cの内容を表示

- カーソルキー上下、改行キーで行送り/行戻り
- スペースキーまたは「f」キーで画面送り
- 「b」キーで画面戻り
- 「q」キーで終了

# テキストファイルの比較

**diff** *[options] fromfile tofile*

2つのファイルの違いを探す

```
[user1@scls subdir1]$ diff seqjob.c seqjob1.c
6c6,7
<    fprintf(stderr, "Hello SCLS World. (STDERR)¥n");
---
>    fprintf(stderr, "Hello SCLS Computer. (STDERR)¥n");
>    /* Comment */
```

| | |
|---|---|
| a | 追加 |
| d | 削除 |
| c | 置き換え |

6c6,7

*fromfile*の行番号　*tofile*の行番号

| | |
|---|---|
| < | *fromfile*の内容 |
| > | *tofile*の内容 |

# ファイル/ディレクトリの移動

**mv** *[options] source... target*

ファイルやディレクトリの移動
または名前の変更

```
[user1@scls subdir1]$ mv seqjob.c seqjob_org.c
[user1@scls subdir1]$ ls
seqjob1.c  seqjob_org.c

[user1@scls subdir1]$ mv seqjob1.c ../subdir2
[user1@scls subdir1]$ ls
seqjob_org.c

[user1@scls subdir1]$ mv ../subdir2 .
[user1@scls subdir1]$ ls -F
seqjob_org.c  subdir2/

[user1@scls subdir1]$ mv subdir2 subdir20
[user1@scls subdir1]$ ls -F
seqjob_org.c  subdir20/
```

*target*に存在しないファイルパスを指定すると、ファイルの名前を変更して移動

カレントディレクトリの内容を表示

*target*に既存のディレクトリパスを指定すると、ファイルの移動

カレントディレクトリの内容を表示

*target*に既存のディレクトリパスを指定すると、ディレクトリの移動

カレントディレクトリの内容を表示

*target*に存在しないディレクトリパスを指定すると、ディレクトリの名前を変更して移動

カレントディレクトリの内容を表示

# ファイル/ディレクトリの削除

**rm** *[options] file...*

ファイルやディレクトリを削除

**rmdir** *[options] directory...*

空のディレクトリを削除

```
[user1@scls subdir1]$ ls subdir20
seqjob1.c
[user1@scls subdir1]$ rm subdir20/seqjob1.c

[user1@scls subdir1]$ ls subdir20
[user1@scls subdir1]$ rmdir subdir20
[user1@scls subdir1]$ ls
seqjob_org.c
[user1@scls subdir1]$ cd ..

[user1@scls dir1]$ rm -r subdir1
[user1@scls dir1]$ ls -F
lec_scls/

[user1@scls dir1]$ rmdir lec_scls
rmdir: failed to remove ‘lec_scls’: Directory not empty
```

- ディレクトリsubdir20の内容を表示
- ファイルseqjob1.cを削除
- ディレクトリsubdir20の内容を表示
- rmdirコマンドで空のディレクトリを削除
- カレントディレクトリの内容を表示
- カレントディレクトリを親ディレクトリに変更
- rmコマンドで空でないディレクトリを削除
- カレントディレクトリの内容を表示
- rmdirコマンドでは空でないディレクトリは削除できない

# オンラインマニュアルの表示

**man** *[options] name*

オンラインマニュアルを表示

```
[user1@scls ~]$ man <コマンド名>

[user1@scls ~]$ man ls
```

lsコマンドのオンラインマニュアルを表示

- カーソルキー上下、改行キーで行送り/行戻り
- スペースキーまたは「f」キーで画面送り
- 「b」キーで画面戻り
- 「q」キーで終了

# 2.c ファイルとディレクトリのアクセス権

# アクセス権の変更

**chmod** *[options] mode file*
(CHange MODe)
ファイルやディレクトリのアクセス権を変更

```
[user1@scls job_seq]$ ls -l
total 88
-rw-rw-r-- 1 user1 group1     150 Jul 24 15:45 job.sh
-rwxrwxr-x 1 user1 group1 1063723 Jul 24 15:45 seqjob
-rw-rw-r-- 1 user1 group1     134 Jul 24 15:45 seqjob.c
```

アクセス権　所有者/グループ　サイズ　タイムスタンプ

```
-rwxrwxr-x.
```

所有者　グループ　その他

| | |
|---|---|
| r | 読み取り許可 |
| w | 書き込み許可 |
| x | 実行許可 |

```
[user1@scls job_seq]$ chmod 600 job.sh
[user1@scls job_seq]$ ls -l job.sh
-rw------- 1 user1 group1     150 Jul 24 15:45 job.sh
```

← ファイルjob.shのアクセス権を変更

# ACL(Access Control Lists)によるアクセス権の設定

**getfacl** *[options] file...*
(GET File Access Control Lists)
ACLの確認

**setfacl** *[options] [acl_spec] file...*
(SET Access Control Lists)
ACLを設定

```
[user1@scls job_seq]$ getfacl seqjob.c
# file: seqjob.c
# owner: user1
# group: group1
user::rw-
group::rw-
other::r--

[user1@scls job_seq]$ setfacl –m u:user2:rw seqjob.c

[user1@scls job_seq]$ getfacl seqjob.c
# file: seqjob.c
# owner: user1
# group: group1
user::rw-
user:user2:rw-
group::rw-
mask::rw-
other::r--
```

ファイルseqjob.cのアクセス権を確認

ファイルseqjob.cに対し、ユーザuser2に読み取り権と書き込み権を与えるように設定

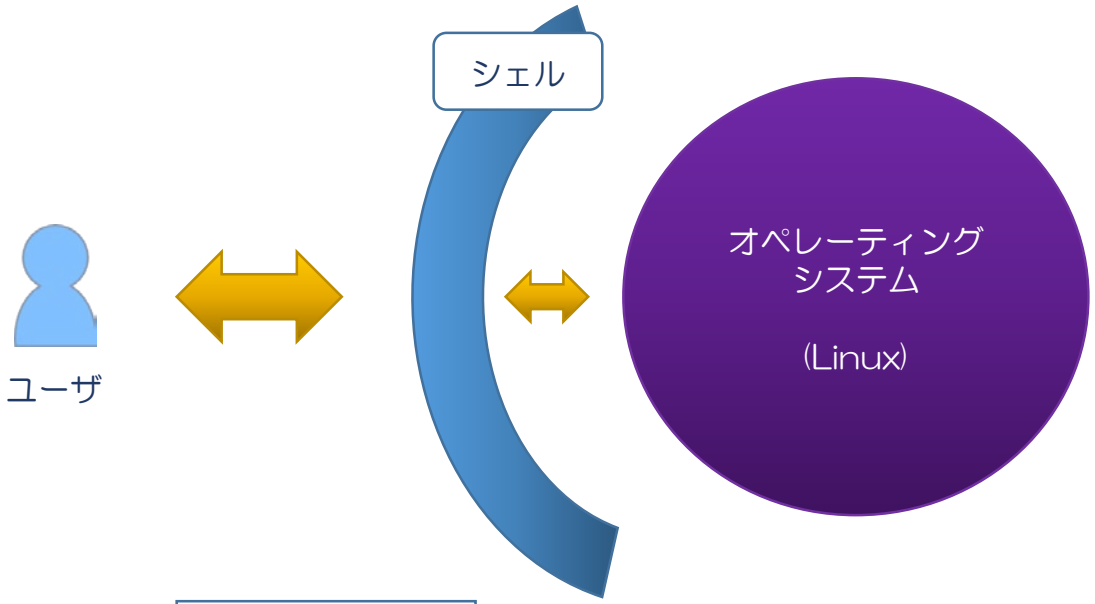
ユーザuser2に読み取り権と書き込み権が付与されている

# 2.d シェルの機能

(実習)

# シェルとは

オペレーティングシステム(OS)とユーザとの間のインタフェース

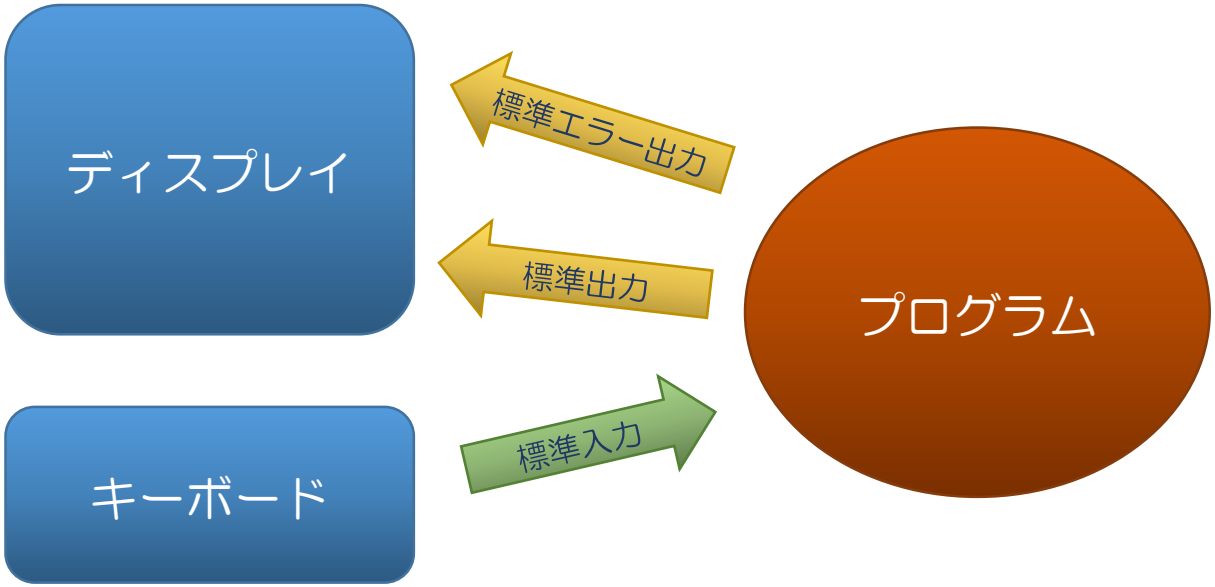



シェルの機能

- プログラム名を指定して起動
- 入力コマンドラインのワイルドカード等の展開
- リダイレクト、パイプ
- 入力補完機能、ヒストリ機能
- シェルスクリプト(複数の処理が記述されたスクリプト)の実行
- etc.

# UNIX/Linuxの標準入出力

| 標準入力 | 標準出力 | 標準エラー出力 |
| --- | --- | --- |
| プログラムへの入力<br>（キーボード） | プログラムからの出力<br>（ディスプレイ） | プログラムからのエラー出力<br>（ディスプレイ） |



# リダイレクト/パイプ

## リダイレクト

コマンドの標準入力元や
標準出力先を切り替える

## パイプ

コマンドの標準出力を
他のコマンドの標準入力に連結

```
[user1@scls dir1]$ ls -l > files.txt

[user1@scls dir1]$ ls -l >> files.txt

[user1@scls dir1]$ cat < files.txt

[user1@scls dir1]$ cat > in.txt <<EOF
> test message 1.
> test message 2.
> EOF

[user1@scls dir1]$ ls -l | less
```

lsコマンドの標準出力先をfiles.txtへ切り替え(上書き)

lsコマンドの標準出力先をfiles.txtへ切り替え(アペンド)

catコマンドの標準入力元をfiles.txtに切り替え

ヒアドキュメント (here document)

「EOF」で囲まれた範囲を、catコマンドの標準入力元とする

lsコマンドが標準出力が、lessコマンドの標準入力と連結される

# 入力補完機能/ヒストリ機能

## 入力補完機能

ファイル名やディレクトリ名、
コマンド名を補完

## ヒストリ機能

過去に入力したコマンドの
呼び出し

```
[user1@scls dir1]$ cd l(ここでTABキーを押す)

[user1@scls dir1]$ cd lec_scls/
[user1@scls dir1]$ cd lec_scls/ (ここでTABキーを押す)
[user1@scls dir1]$ cd lec_scls/job_

[user1@scls dir1]$ cd lec_scls/job_
job_hybrid/ job_seq/

[user1@scls dir1]$ ls
[user1@scls dir1]$ !!

[user1@scls dir1]$ !l
```

ディレクトリ名「lec_scls」が補完される

候補が複数存在する場合は、一意な部分まで補完

候補が複数存在する場合は、TABキーを2回推すと一覧が表示される

直前のコマンドを再実行

「l」で始まる最も最後に実行したコマンドを再実行

カーソルキーの上下で、コマンド入力履歴を辿ることができる

# 3. ジョブの実行

(実習)

# ジョブの実行

## SCLS計算機システムの計算ノードを使用したプログラムの実行


| 実行形態によるジョブの種類 | |
|---|---|
| バッチジョブ | バッチ形式で処理を実行<br>ジョブスクリプトに処理内容を記述 |
| 会話型ジョブ | 利用者端末から会話形式で実行<br>主にデバッグ用 |

# ジョブで実行するプログラムの種類

| ジョブの種類 | 実行する<br>プログラム | 説明 |
|---|---|---|
| 逐次ジョブ | 逐次プログラム | • 単一スレッドかつ単一プロセスで実行 |
| 並列ジョブ | スレッド並列<br>プログラム | • 1個のプロセスで複数のスレッドを生成<br>• プロセスのメモリ空間を共有<br>• 単一ノードで実行<br>(自動並列化プログラム, OpenMPプログラム) |
| | プロセス並列<br>プログラム | • 複数のプロセスを生成<br>• プロセス間通信<br>• 複数のノードを使用して実行可能<br>(MPIプログラム) |
| | ハイブリッド並列<br>プログラム | • スレッド並列 + プロセス並列 |

# ジョブスクリプト

実行したい処理の内容を記述したファイル

## ジョブスクリプトに記述する内容

- ジョブ投入オプション
  （リソースグループ、使用ノード数、最大経過時間、etc.）
- 実行するコマンド列

## 逐次ジョブのジョブスクリプトの例

[lec_scls/job_seq/job.sh]

```
#!/bin/sh
#------ pjsub option --------#
#PJM –L “rscgrp=small”
#PJM –L “node=1”
#PJM –L “elapse=10:00”
#------ Program execution -------#
./seqjob
```

リソースグループ「small」を使用

使用ノード数は「1」

最大経過時間は「10分」

逐次プログラムを実行

# ジョブスクリプト

## 並列ジョブ(ハイブリッドMPI並列)のジョブスクリプトの例

[lec_scls/job_hybrid/job.sh]

```
#!/bin/sh
#------ pjsub option ------#
#PJM -L "rscgrp=small"
#PJM -L "node=1"
#PJM --mpi "proc=2"
#PJM -L "elapse=10:00"
#PJM -j
#------ Program Execution ------#
export OMP_NUM_THREADS=4
mpiexec ./hybridjob
```

- リソースグループ「small」を使用
- 使用ノード数は「1」
- MPIプロセス数は「2」
- 最大経過時間は「10分」
- 標準エラー出力を標準出力に向ける
- OpenMPのスレッド数は「4」(1MPIプロセスあたり)
- ハイブリッドMPI並列プログラムの実行

# ジョブ操作コマンド

| コマンド名 | 機能 |
| --- | --- |
| pjsub | ジョブ投入 |
| pjstat | ジョブの状態を参照 |
| pjdel | ジョブ削除 |
| pjhold | ジョブ保留 |
| pjrls | ジョブ保留解除 |
| pjcat | ジョブスクリプト表示 |

# ジョブ投入の実習

## 実習で使用するプログラム

```
[user1@scls dir1]$ ls -lR lec_scls/
lec_scls/:
total 0
drwxrwxr-x 2 user1 group1 53 Aug  2 18:45 job_hybrid/
drwxrwxr-x 2 user1 group1 47 Jul 24 15:25 job_seq/

lec_scls/job_hybrid:
total 36
-rwxrwxr-x 1 user1 group1 1064931 Aug  2 18:45 hybridjob*
-rw-rw-r-- 1 user1 group1     420 Aug  2 18:45 hybridjob.c
-rw-rw-r-- 1 user1 group1     210 Jul 24 15:39 job.sh

lec_scls/job_seq:
total 88
-rw-rw-r-- 1 user1 group1     150 Jul 24 15:23 job.sh
-rwxrwxr-x 1 user1 group1 1063723 Jul 24 15:21 seqjob*
-rw-rw-r-- 1 user1 group1     134 Jul 24 15:18 seqjob.c
[user1@scls dir1]$
```

並列ジョブ（ハイブリッド並列）

逐次ジョブ

実行可能ファイル

ソースファイル

ジョブスクリプト

ジョブスクリプト

実行可能ファイル

ソースファイル

# バッチジョブの投入

## ジョブの投入

```
[user1@scls job_seq]$ pjsub job.sh
[INFO] PJM 0000 pjsub Job 7874 submitted.
```

## ジョブの状態の確認

```
[user1@scls job_seq]$ pjstat
 ACCEPT QUEUED  STGIN  READY RUNING RUNOUT STGOUT   HOLD  ERROR   TOTAL
      0      0      0      0      1      0      0      0      0       1
s     0      0      0      0      1      0      0      0      0       1

JOB_ID     JOB_NAME   MD ST  USER     START_DATE      ELAPSE_LIM NODE_REQUIRE
6293       job.sh     NM RUN user1    07/26 17:37:18  0000:10:00 1
```

## バッチジョブの実行結果

バッチジョブの実行が終了すると、標準出力ファイルと標準エラー出力ファイルが出力される。

| | |
|---|---|
| 標準出力ファイル | ジョブ名.oXXXXX（XXXXXはジョブID）<br>（例 job.sh.o12345） |
| 標準エラー出力ファイル | ジョブ名.eXXXXX（XXXXXはジョブID）<br>（例 job.sh.e12345） |

# 会話型ジョブの投入

## ジョブの投入

```
[user1@scls job_seq]$ pjsub --interact [option...]
```

## 会話型ジョブの実行例

```
[user1@scls job_seq]$ pjsub --interact
[INFO] PJM 0000 pjsub Job 1234 submitted.
[INFO] PJM 0081 .connected.
[INFO] PJM 0082 pjsub Interactive job 1234 started.
[user1@a01-001 job_seq]$ ./seqjob
Hello SCLS World.
Hello SCLS World. (STDERR)
[user1@a01-001 job_seq]$ exit
exit
[INFO] PJM 0083 pjsub Interactive job 1234 completed.
[user1@scls job_seq]$
```

会話型ジョブの投入

会話型ジョブ開始

プログラム実行

会話型ジョブ終了

ログインノード復帰

# SCLS計算機システムについて

## ドキュメント

- SCLS計算機システム利用手引書
- ジョブ運用ソフトウェア エンドユーザ向けガイド

https://hpci-scls.riken.jp/

## お問い合せ

scls-adm@riken.jp


2013年8月6日 (第2版)

独立行政法人理化学研究所
HPCI計算生命科学推進プログラム