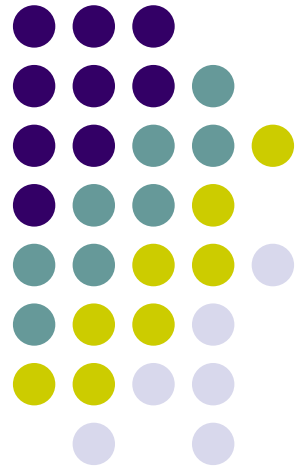
# 情報処理概論

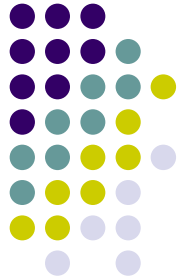
工学部 物質科学工学科
応用化学コース
機能物質化学クラス



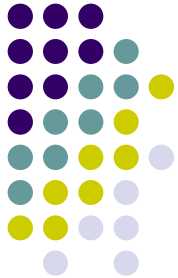
## 第2回

2005年　4月21日

# プログラムを添付したメールの送信



- 以下の人は次回の講義までに
  z5nt02in@s.kyushu-u.ac.jp
  宛にプログラムを添付したメールを送信
  - まだメールを上記のアドレスに送ってない人
  - 送ったけど web に到着が載っていない人
  - web の提出状況で　×　が付いている人
    （メールの返事に「再度提出」もしくは
    「もう一度提出」と書かれていた人）
- 注意：ファイル名に ～　や # がついてないものを
  添付する.
  - ～や＃が付いたファイルは一つ古い版なので送らない
  - コンパイル後の実行ファイルではなく test.f90 や hello.f90 のようなソースプログラム（後述）を添付する

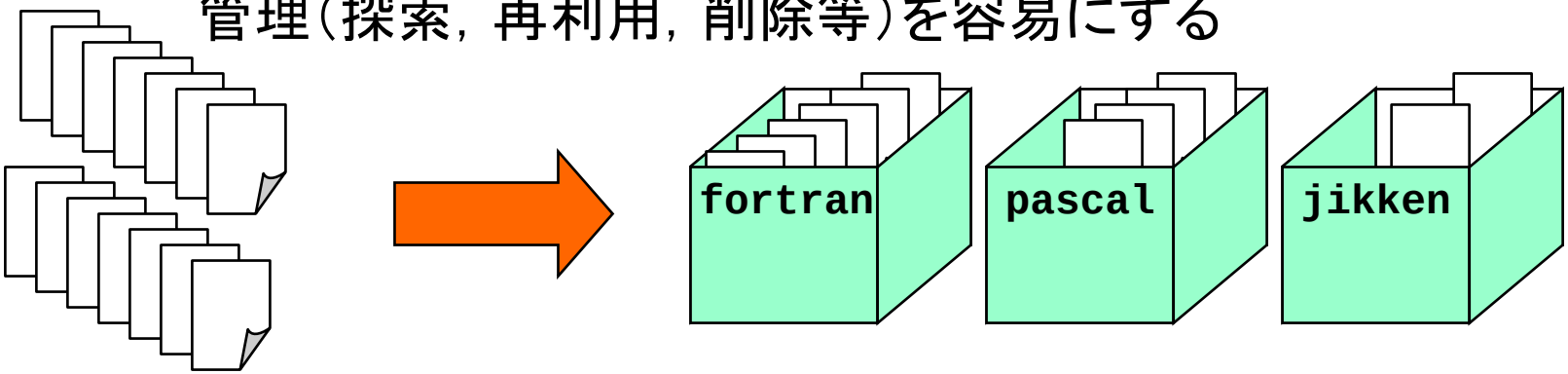


○　UNIXでのファイル管理

- Fortranプログラムのコンパイル（翻訳）及び実行

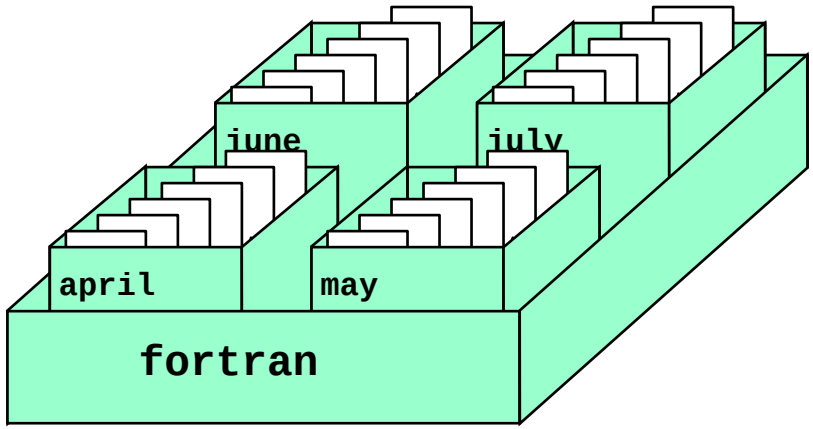
# ディレクトリ

- ファイル分類用の箱
  - Windowsのフォルダとほぼ同じ
- この講義（半期）で作るファイルの数： 50個以上
  - 作成するプログラム数：10～20本
    ＋ 編集時の予備ファイル ＋ 実行ファイル
  - 他の講義などで作成したファイルも混在
- ディレクトリを利用して整理
  - ファイルを目的別に分類して格納することにより
    管理（探索，再利用，削除等）を容易にする



# ファイルの分類法

- 自分が管理しやすいように
  - 例えば
    1. この講義で作成するファイルは fortran ディレクトリの中に置く
    2. さらに fortran ディレクトリの中に 4, 5, 6, 7月のディレクトリを用意し作成した月で分類する

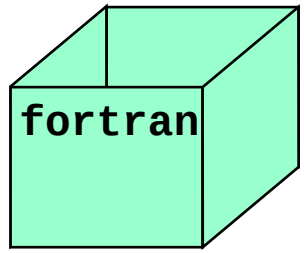


# ディレクトリの作成
# mkdir コマンド

- 使い方：
  **% mkdir** 作成したいディレクトリ名

空白（スペース）を
忘れない

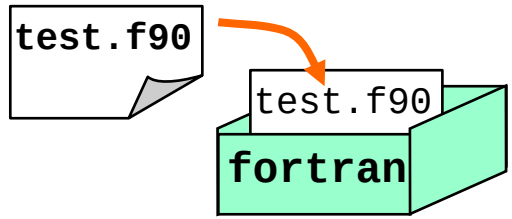
  - 例：fortranディレクトリの作成
    **% mkdir fortran**



# ファイルの移動
# mv コマンド

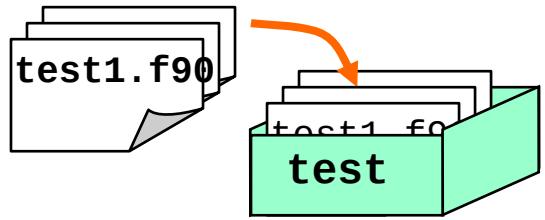
- 使い方：
  `% mv　移動元　移動先`

  - 例1：test.f90 を fortranディレクトリに移動
    `% mv test.f90 fortran`




  - 例2：test.f90 の名前を aaa.f90 に変更
    `% mv test.f90 aaa.f90`

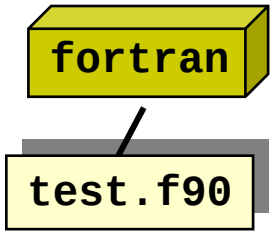

  - 例3：test1.f90, test2.f90, test3.f90　を test ディレクトリに移動
    `% mv test1.f90 test2.f90 test3.f90 test`



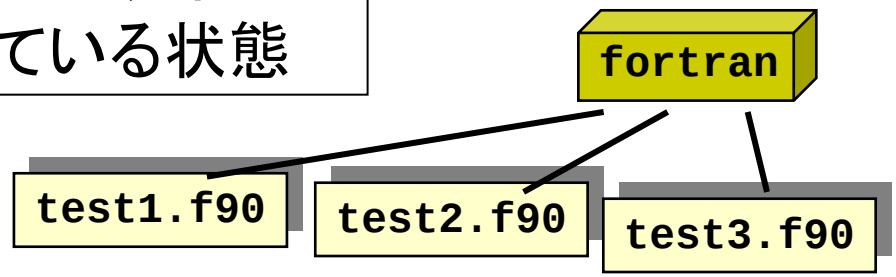
# ファイルとディレクトリの構造

- 準備:
  ディレクトリにファイルが入っている状態を
  このように表すことにする.

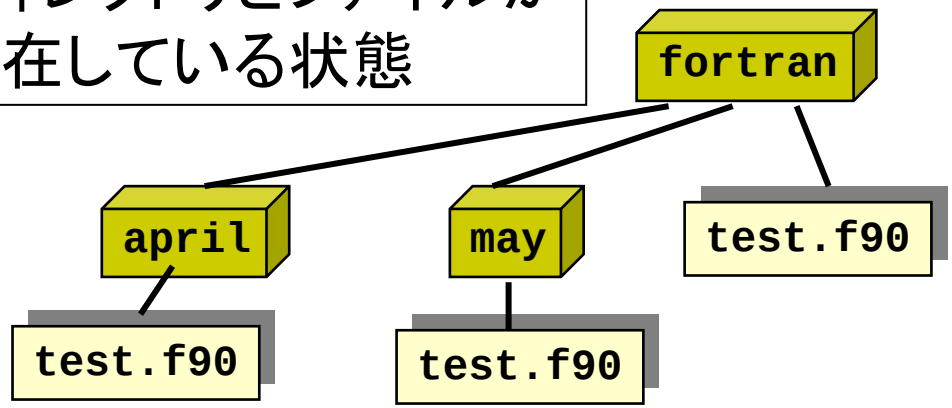
1つのディレクトリに
1つのファイルが
入っている状態




複数のファイルが
入っている状態

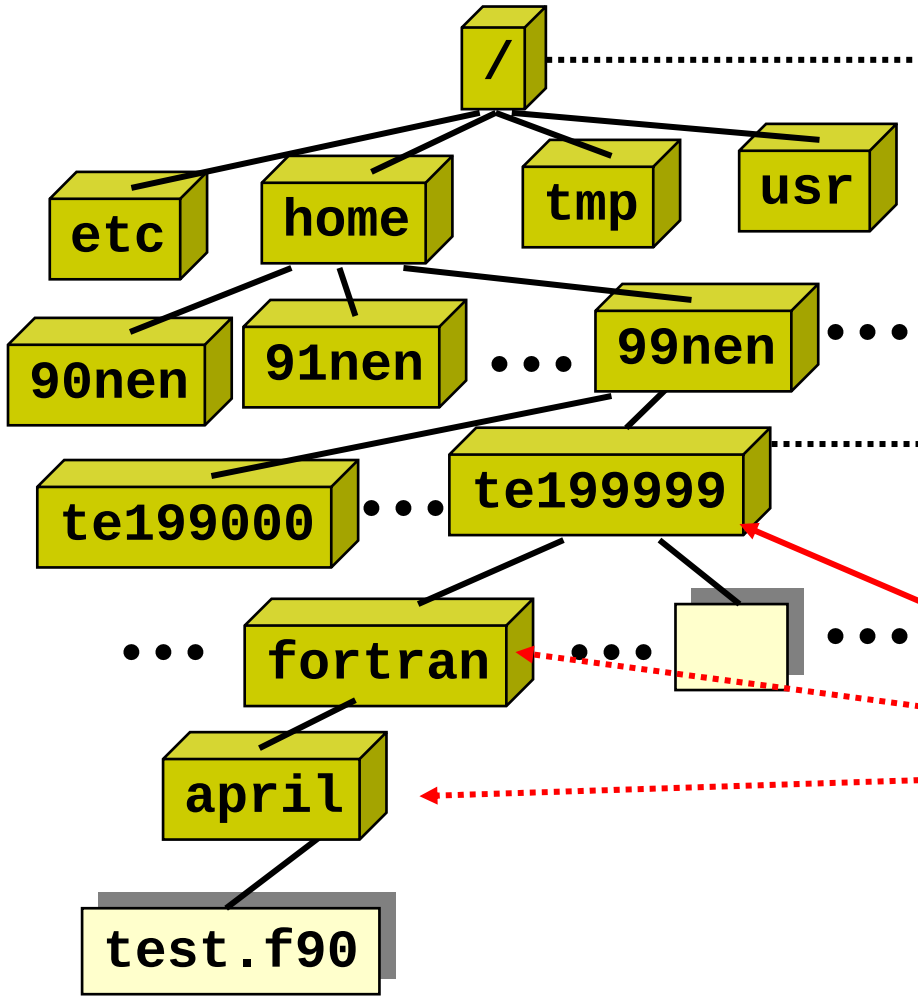



ディレクトリとファイルが
混在している状態



# サーバ全体のファイル構造

- 逆さにした木のような構造
  - 一番上にルート(根)ディレクトリ

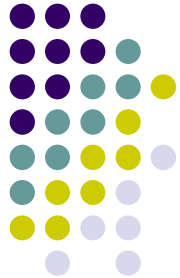



ルートディレクトリ
- 階層構造の起点

ホームディレクトリ
- ログイン時のカレントディレクトリ
- ユーザー毎に用意

カレントディレクトリ
- 作業対象となるディレクトリ
- cd コマンドで適宜変更可
- 最初(ログイン直後)は ホームディレクトリ

# ホームディレクトリと カレントディレクトリ



- カレントディレクトリ（＝現在"いる"ディレクトリ）を
  基準としてファイルやディレクトリの場所を指定する.

- ログイン直後はホームディレクトリ
  （＝自分専用のディレクトリ）に ”いる”

  例:

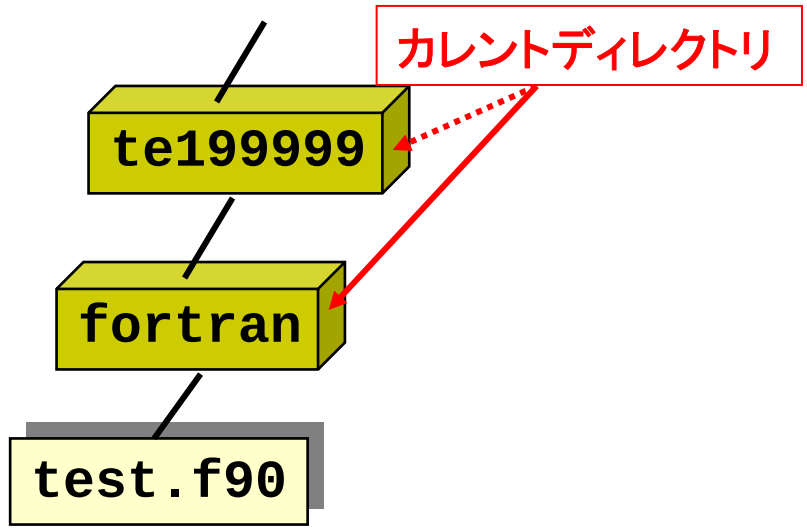
  ```
  % emacs test.f90
    → カレントディレクトリに test.f90 を作成
  % mkdir fortran
    → カレントディレクトリに fortranディレクトリ作成
  % mv test.f90 fortran
    → カレントディレクトリの test.f90 を
       カレントディレクトリの下の fortran ディレクトリに移動
  ```

# 別のディレクトリに移る cd コマンド

- 使い方：
  **% cd 移動先ディレクトリ**
  - 別のディレクトリに移る ＝ カレントディレクトリの変更
  - 移動先ディレクトリを指定しない場合，ホームディレクトリに移る

例1：fortranディレクトリに移る
```
% cd fortran
```

例2： ホームディレクトリに移る
```
% cd
```

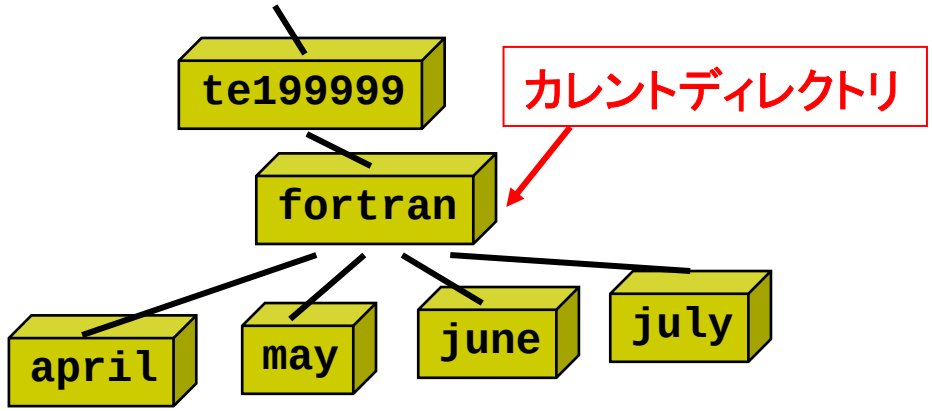


# 別のディレクトリに移って作業

- 作業対象のファイルやディレクトリがある
  ディレクトリに移って作業する

例： fortran ディレクトリに移って月別のディレクトリを作成する

```
% cd fortran
% mkdir april may june july
```

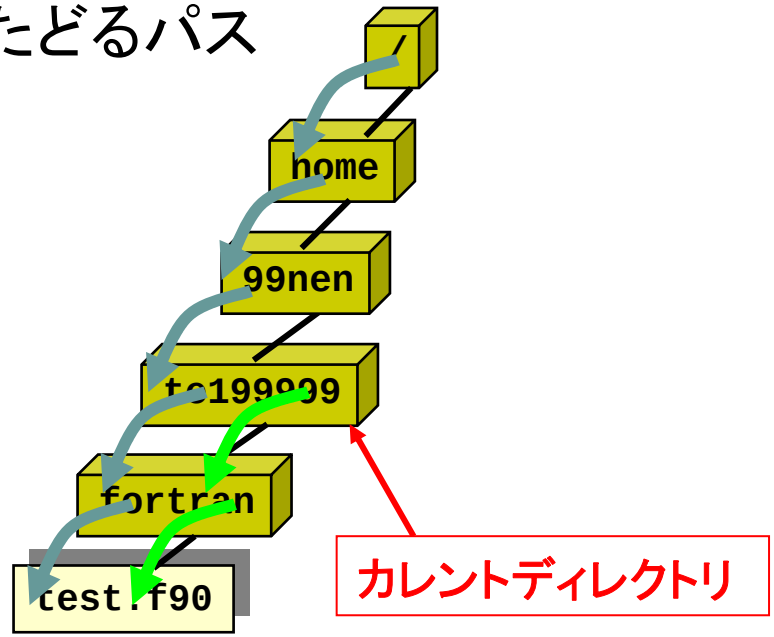



ディレクトリを移らない場合,キー入力が大変：

```
% mkdir fortran/april fortran/may fortran/jun fortran/july
```

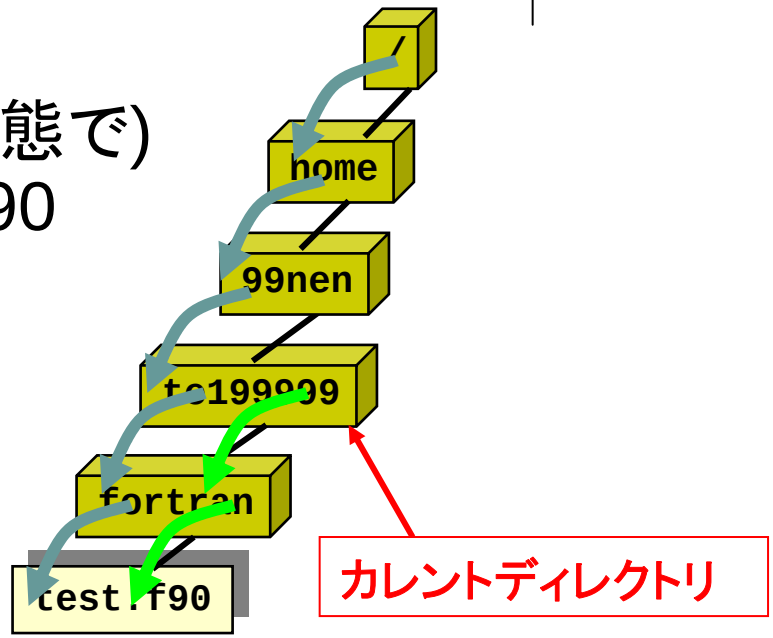
# パス： ファイルやディレクトリの場所

- 目的のファイルやディレクトリまでたどる際のパス（path:経路）を列挙
  - 二通りの記述方法（どちらでも良い）
    - 相対パス：カレントディレクトリからたどるパス
    - 絶対パス：ルートディレクトリからたどるパス



# パスの指定

- 「ディレクトリの下」は / で記述する
- 例: (今ホームディレクトリにいる状態で)
  fortran ディレクトリの下の test.f90
  - 相対パス
    （カレントディレクトリの下の）
    fortran の下の test.f90
    **fortran/test.f90**




  - 絶対パス
    ルートディレクトリの下の home の下の
    99nen の下の te199999 の下の fortran の下の　test.f90
    **/home/99nen/te199999/fortran/test.f90**
    - ルートディレクトリも　／　で記述する.

# パスの記述に用いる特殊な記号

- 一つ上のディレクトリ　　..　（ピリオド二つ）

  例：カレントディレクトリの一つ上に移る

  ```
  % cd ..
  ```


- カレントディレクトリ　　.　（ピリオド一つ）

  例：一つ上のディレクトリの **test.f90** をカレントディレクトリに移動する

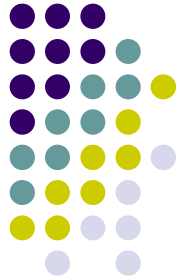
  ```
  % mv ../test.f90 .
  ```

- ホームディレクトリ　　~

  例：ホームディレクトリの **fortran** ディレクトリに移る

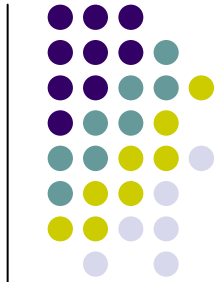
  ```
  % cd ~/fortran
  ```

# カレントディレクトリの確認 pwd コマンド



- 使い方：
  **% pwd**
- 使用のタイミング：
  - 現在自分がどこにいるかわからなくなった時
  - 重要な操作（ファイルの削除や移動等）を行う前の確認

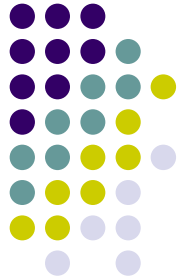
# ファイルの一覧表示 ls コマンド



- 使い方:
  - カレントディレクトリのファイルを表示
    **% ls**
  - 指定したディレクトリの下のファイルを表示
    **% ls** ディレクトリ
  - カレントディレクトリの下のファイルの詳細情報を表示
    **% ls -l**
  - 指定したディレクトリの下のファイルの詳細情報を表示
    **% ls -l** ディレクトリ
  - 指定したファイルの詳細情報を表示
    **% ls -l** ファイル

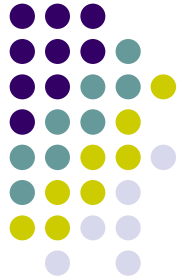
```
% ls -l
total 104
drwxr-xr-x   2 te199999  user      79 May  6 17:42 april
-rw-------   1 te199999  user      92 Apr 16  1997 test.f90
%
```

# ファイルのコピー
# cp コマンド



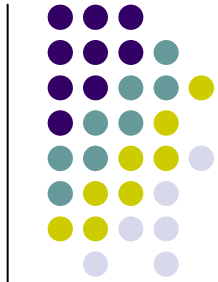
- 使い方：
  - コピー元ファイルをコピー先ファイルにコピー
    % **cp** コピー元ファイル コピー先ファイル
  - ファイル１～３ を指定したディレクトリにコピー
    % **cp** ファイル１ ファイル２ ファイル３ ディレクトリ名

# ファイルの削除
# rm コマンド



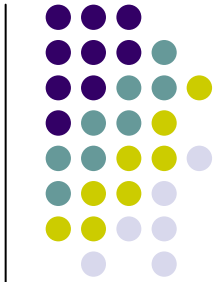
- 使い方:
  - ファイル1～3を削除
    % rm ファイル1 ファイル2 ファイル3
  - ディレクトリ 1,2とその下の全てのファイルを削除
    % rm -r ディレクトリ1 ディレクトリ2
  - 削除の前に確認
    % rm -i -r ディレクトリ1 ディレクトリ2

# UNIXのファイルに関する注意



- 削除したら（原則として）復元不可能
  - 「ごみ箱」 は存在しない
- コピーや移動で既に存在するファイルが
  上書きされる場合がある
  - この場合も復元不可能
- -i オプションを利用すると、
  各ファイルについて削除や上書きの前に確認出来る.
  `rm -i -r` ディレクトリ
  `mv -i` ファイル１ ファイル２ ファイル３ ディレクトリ
  `cp -i` ファイル１ ファイル２ ファイル３ ディレクトリ

# ファイルの内容表示 less コマンド



- 使用法:
  % **less** ファイル名
  - ファイルの内容を1ページずつ表示
  - 閲覧中は以下のキーを利用
    - 次のページへ: **SPACE** もしくは **f**
    - 前のページへ: **b**
    - 1行下へ: **Enter** もしくは **j**
    - 1行上へ: **k**
    - 閲覧終了: **q**

# 演習1

- ah にログインしホームディレクトリの
  ファイル一覧を表示

- ホームディレクトリに
  fortran ディレクトリを作成

- fortran ディレクトリに移り april ディレクトリを
  作成
- 一つ上のディレクトリ（＝ホームディレクトリ）にある
  test.f90 （先週作成したもの）を
  april ディレクトリに移動

# 注意

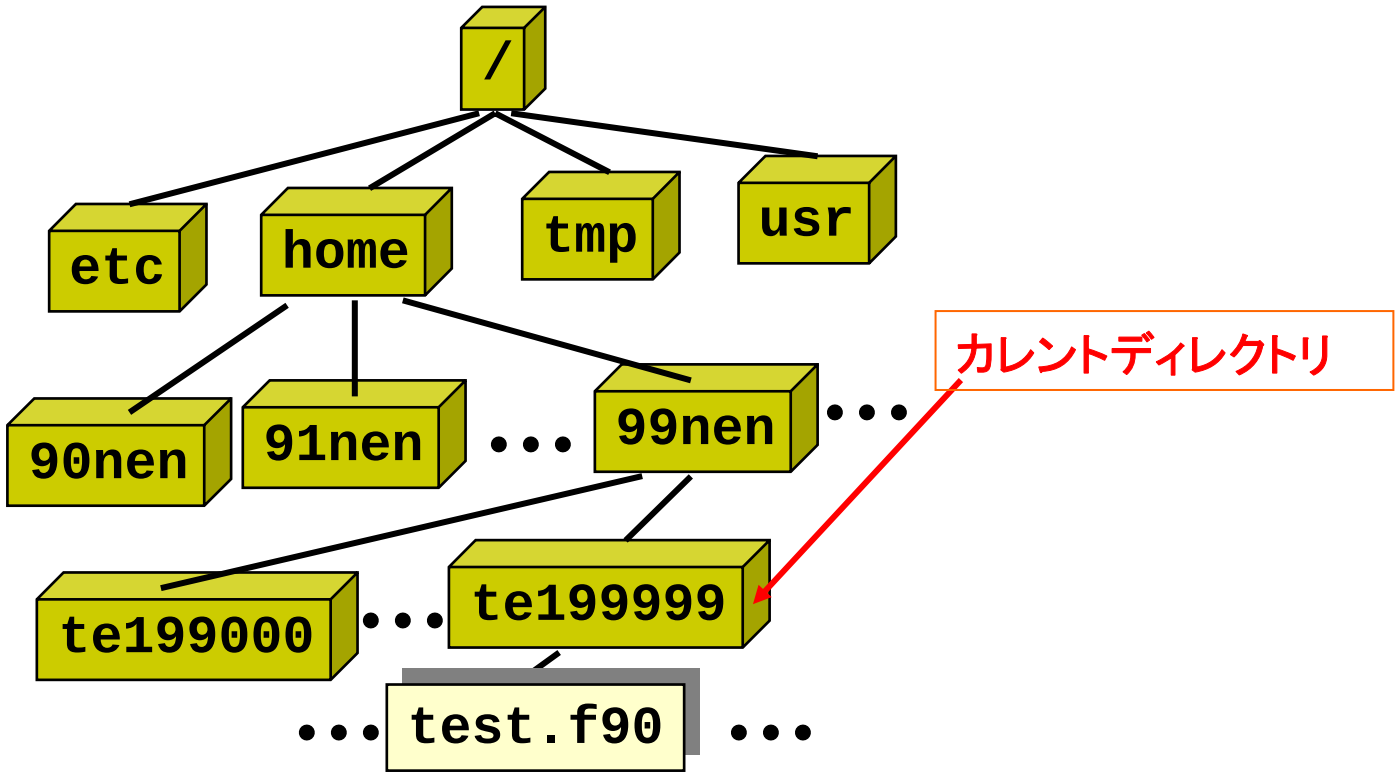
- コマンド，ファイル名の打ち間違いに注意
  - 必要なファイルが消えたり中身が変わったりする場合もある
  - Enterキーを押す前にもう一度確認

- エラーメッセージが出たら
  - まず状況確認
    **pwd　　現在のディレクトリを表示**
    **ls　　現在のディレクトリに存在するファイル一覧**
  - 必要に応じて再実行

演習1

# ログインしてホームディレクトリのファイル一覧を表示.

- ah へのログイン:　先週の資料を参照
- ファイル一覧の表示:　ls コマンド
  **% ls**

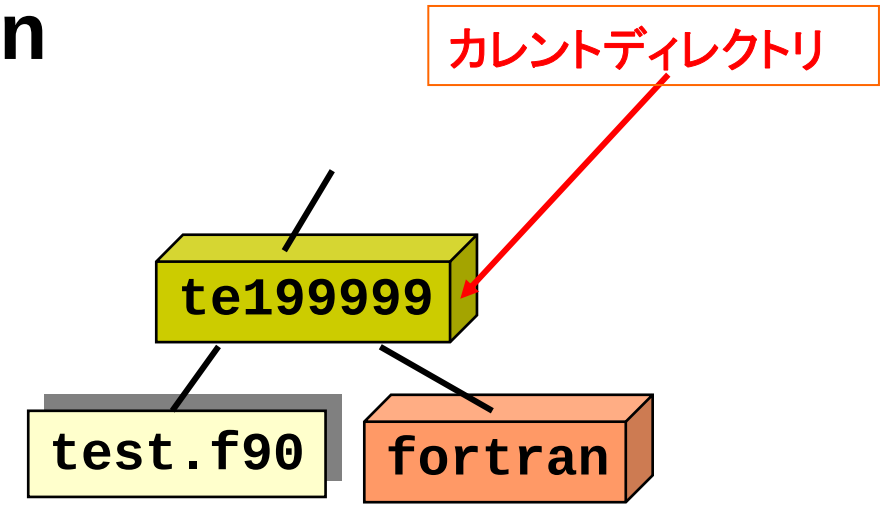


演習1

# ホームディレクトリに fortran ディレクトリを作成

- まず，カレントディレクトリを確認
  `% pwd`

- ディレクトリの作成： mkdirコマンド
  `% mkdir fortran`



演習1

# fortran ディレクトリに移り april ディレクトリを作成

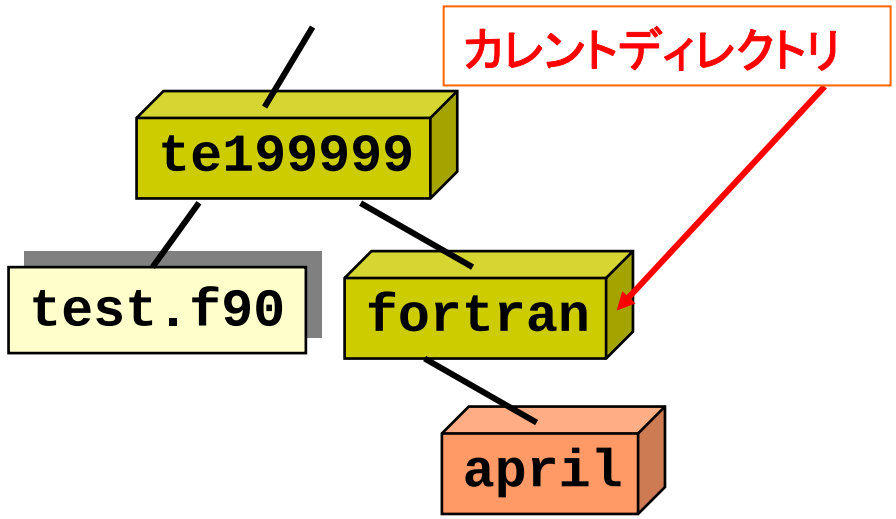
- fortranディレクトリに移る： cdコマンド

```
% cd fortran
% pwd
```

← 確認

- ディレクトリの作成： mkdir コマンド

```
% mkdir april
```



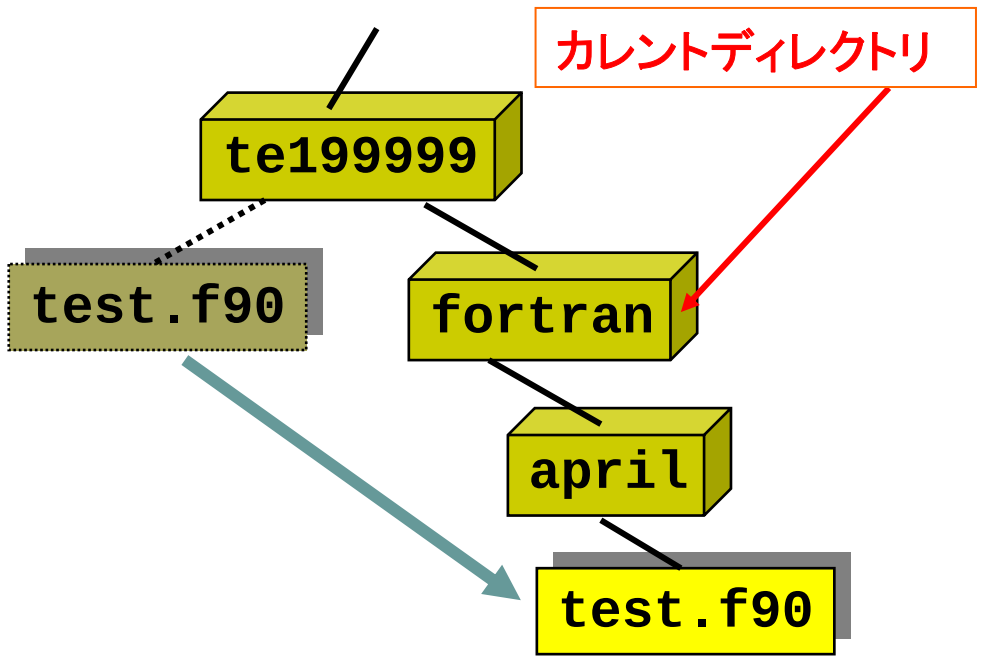
演習1

# 一つ上のディレクトリの test.f90を april ディレクトリに移動

- ファイルの移動:　　mv コマンド

```
% mv ../test.f90 april
```

一つ上のディレクトリの test.f90

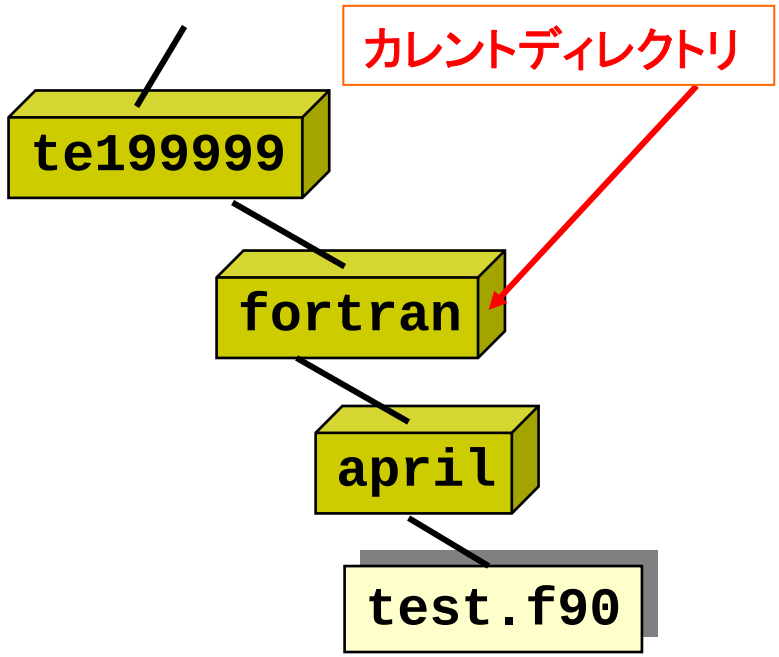


演習1

# 作業結果の確認

- ファイルの表示: ls コマンド

```
% ls
 april/
% ls april
 test.f90
```



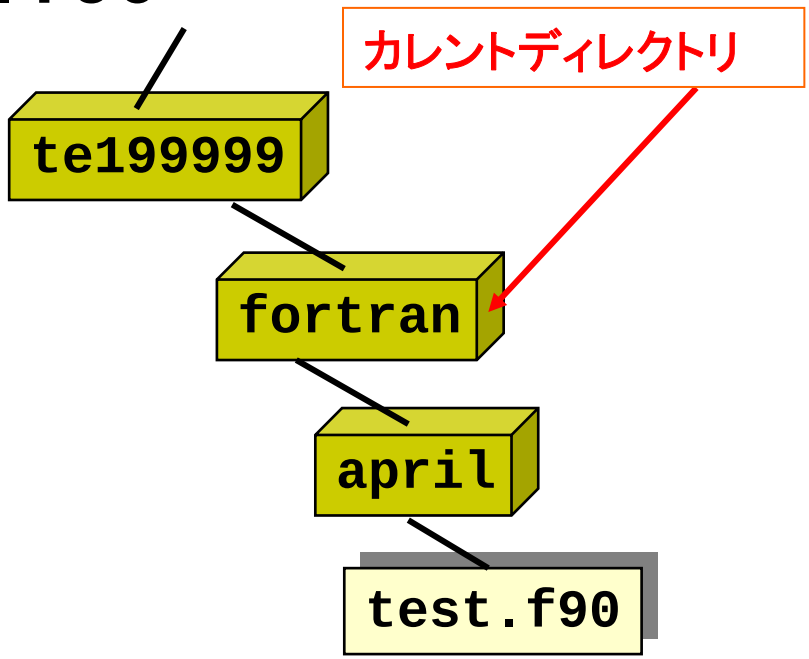
# 終了

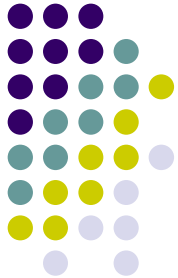
- 時間に余裕があったら他のコマンドも試す
  例)

```
% less april/test.f90
```

- 終わったらログアウト

```
% exit
```

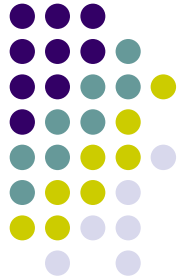




- UNIXでのファイル管理

○ Fortranプログラムのコンパイル（翻訳）及び実行

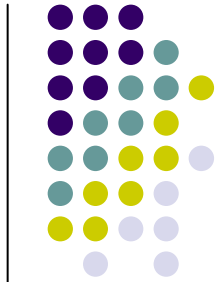
# Fortranプログラムの翻訳と実行



- Fortranプログラムを実行するには機械語へのコンパイル（翻訳）が必要.
  - 機械語:
    - コンピュータがそのまま実行できる言葉（命令）
    - 基本的な命令で細かく指示する必要があるため,
      人間が直接編集するには適さない.
  - Fortranは「高級言語」（＝人間に分かり易い言葉）の一種
    - 他に C言語, Pascal, Java, Basic 等
    - 実行するには機械語への翻訳が必要

  - 機械語とFortranの例
    - Fortran: ５ ＋ ７
    - 機械語: 以下の3つの命令
      - 場所A に 5を格納
      - 場所B に 7を格納
      - 場所A の値と場所B の値を加算

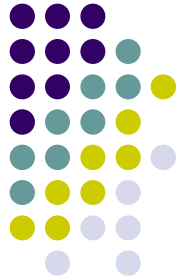
# ソースファイルと実行ファイル



- ソースファイル:
  - 翻訳前のプログラムファイル
  - レポート提出時にはこちらを添付する.

- 実行ファイル:
  - コンパイルの結果得られる　　機械語のプログラムを格納したファイル

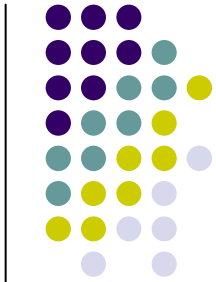
# Fortranプログラムのコンパイル
# f90 コマンド



- 使い方:
  - コンパイルの結果作成される実行ファイルの名前を指定
    % **f90** ソースファイル **-o** 実行ファイル

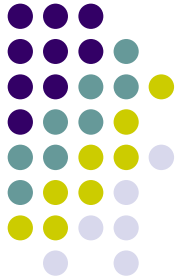

例)
**% f90 test.f90 -o test**

# ファイルの拡張子について



- 拡張子： ファイル名の末尾に付ける文字列
  - ファイルの種類の識別に利用
  - 拡張子が違うと正しく動作しない場合もある.
    例） Fortran90 のコンパイル時
    ソースファイルの拡張子は .f90
  - 他に:
    - .html, .htm　　HTML形式ファイル（Webページ）
    - .jpeg, .jpg　　JPEG形式ファイル（デジタルカメラ等の画像）
    - .mpg　　MPEG形式ファイル（動画）
    - .wav　　WAV形式ファイル（音楽）
    - .mp3　　MP3形式ファイル（音楽）
    - 等

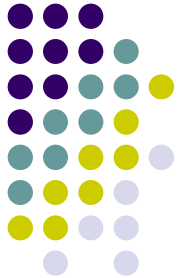
# プログラムの実行



- コンパイルによって得られた実行ファイルの名前をコマンドとして利用する
  - 例) カレントディレクトリの実行ファイル test の実行
    ```
    % ./test
    ```

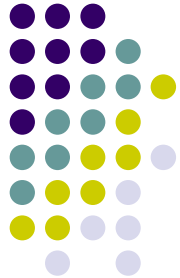
- UNIXのコマンドと区別するために実行ファイルの前に必ず ./ を付ける

# 主なUNIXのコマンド



| コマンド | 動作 |
|---|---|
| pwd | カレントディレクトリの表示 |
| ls | ファイル一覧の表示 |
| less | ファイルの内容表示 |
| cd | カレントディレクトリの変更 |
| mkdir | ディレクトリの作成 |
| mv | ファイルの移動, ファイル名の変更 |
| cp | ファイルのコピー |
| rm | ファイルの削除 |
| f90 | コンパイル |
| ./実行ファイル名 | プログラムの実行 |

# 演習2

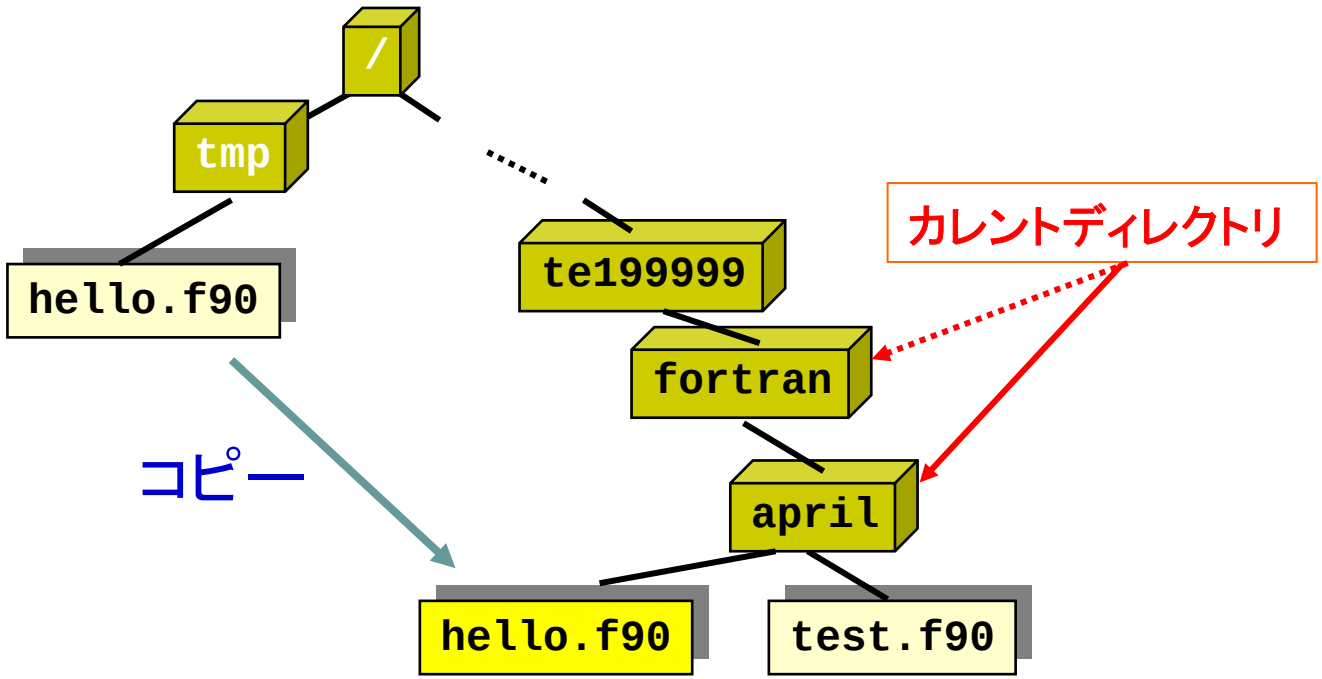


- ログインし, fortranディレクトリの下の aprilディレクトリ(fortran/april) に移る
- ソースファイル /tmp/hello.f90 をカレントディレクトリ（＝april ディレクトリの下）にコピー
- カレントディレクトリの hello.f90 をコンパイル（実行ファイル名を hello と指定）
- カレントディレクトリの hello を実行

演習2

# april ディレクトリに移り /tmp/hello.f90 をコピー

- カレントディレクトリの変更： cd コマンド
  **% cd fortran/april**
- ファイルのコピー： cpコマンド
  **% cp /tmp/hello.f90 .**

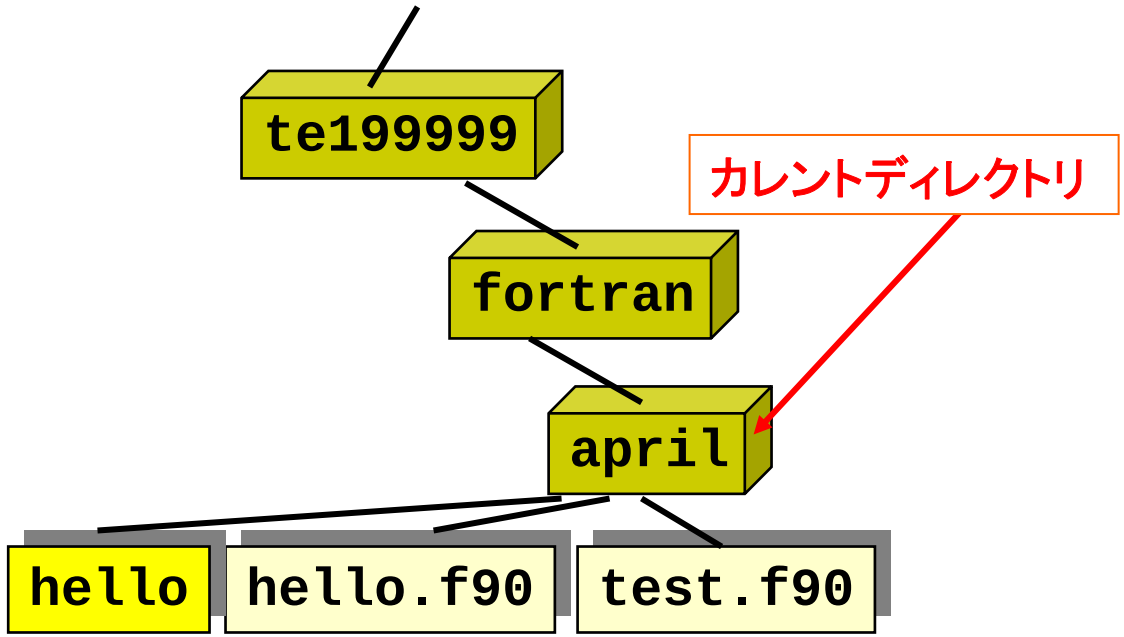


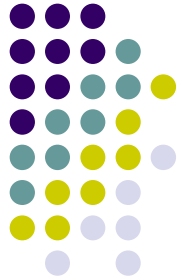
演習2

# hello.f90 をコンパイル（実行ファイル名　hello）

- Fortranプログラムのコンパイル：　f90 コマンド

```
% f90 hello.f90 -o hello
```





# hello の実行

- プログラムの実行： 実行ファイル名
  ```
  % ./hello
  ```

- 時間があったら先週入力したプログラムも試す
  ```
  % f90 test.f90 -o test
  ```
  エラーが出なければ
  ```
  % ./test
  ```

- 終わったらログアウト
  ```
  % exit
  ```