情報デザイン専攻

# 画像情報処理論及び演習I
# -デジタル画像の表現と応用-
## Linuxの基礎、画像クラス

第２回講義
水曜日１限
教室6218情報処理実習室

吉澤　信
shin@riken.jp, 非常勤講師
大妻女子大学　社会情報学部

独立行政法人 理化学研究所

---



## 今日の内容

1. Linuxの基礎：
2. 第一回演習：Cプログラミングのつづき:
   1. Hello World
   2. argc & argv
   3. pnm画像 & 閾値

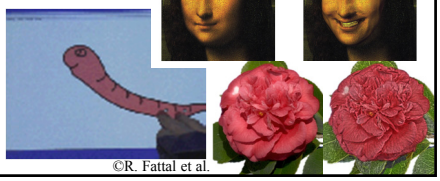
www.riken.jp/brict/Yoshizawa/Lectures

---



## 復習：第一回講義まとめ

✓ **画像処理は信号(音声)処理・CG (Computer Graphics) /CV(Computer Vision)/パターン認識の分野と密接な関連がある:**
- 情報学ではCG と並んで花形の分野.
- 目に見える結果、綺麗、技術的面白さ.

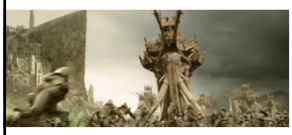
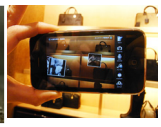




---



## 復習：第一回講義まとめ

✓ **様々な応用分野がある (データが画像):**
- デジタルカメラの爆発的普及により…
- エンターテイメント産業：映画・ゲーム等.
- 自然科学：天文学・生物学・化学・物理学等の観察・観測データ解析等.
- 工業・工学：現実世界の製品データ解析等.
- 医療：CT、MRI等の画像診断等.



---



## 復習：重要

今日必ず憶える事：ls、cd、pwd:
端末(コンソール)にて打ち込みエンターキーで実行.

- cd: ディレクトリー(フォルダー)の移動.
  「cd ディレクトリー名」
- ls:ディレクトリー内のファイル名・フォルダー名を表示.「ls ディレクトリー名」、「ls ./」「ls ../」.「ls -lh」、「ls -alh」
- pwd: 現在のディレクトリーを表示.「pwd」

✓ ファイル名・ディレクトリー名に日本語はダメ！
✓ プログラムのソースコードにコメント以外では、日本語は使わない事！

---



## コンソール(端末) とエディター(emacs)

✓ コンソール(端末)：cd, ls, pwd等のLinuxコマンドを入力し「Enter」キーを押す事でOS（オペレーティングシステム）にファイル操作やプログラムのコンパイル等の命令を与える.

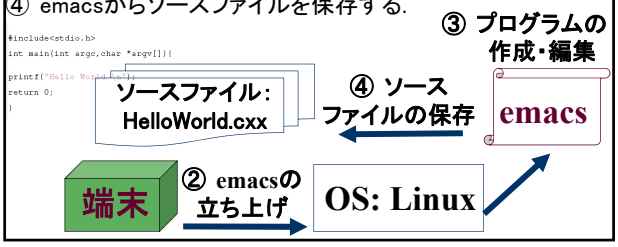
✓「Tab」キーでパスやコマンドを補間出来ます.


✓ エディター(emacs): プログラムや文章を書くツール. テキストでプログラムを入力→ファイルとして保存.

Shin Yoshizawa: shin@riken.jp

## Linuxでのプログラミングの流れ：1

① コンソール(端末)を立ち上げる:画面左下のコンソールアイコンをクリック.
② 端末：でemacsを立ち上げる: 端末に「emacs &」と打ち込み「Enter」キーを押す.
③ emacsでプログラム(ソースファイル)を書く.
④ emacsからソースファイルを保存する.

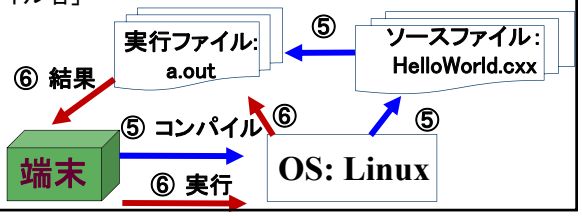



Shin Yoshizawa: shin@riken.jp

## Linuxでのプログラミングの流れ：2

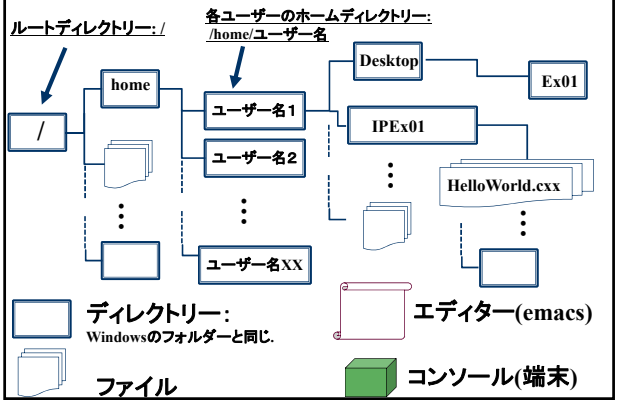
⑤ 端末でソースファイルをコンパイルして実行ファイルを作る:「g++ ソースファイル名」or
「g++ -o 実行ファイル名 ソースファイル名 」.
実行ファイル名を指定しない場合は「a.out」という名前の実行ファイルが作成される.

⑥ 端末で実行ファイルを実行する:「./a,out」or「./実行ファイル名」



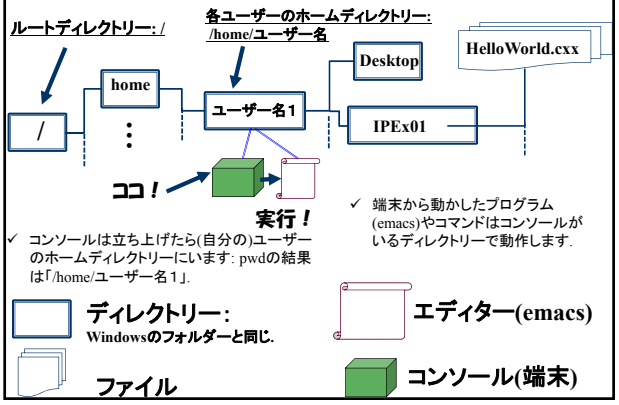

Shin Yoshizawa: shin@riken.jp

## ディレクトリー構造：絶対パス・相対パス




Shin Yoshizawa: shin@riken.jp

## ディレクトリー構造

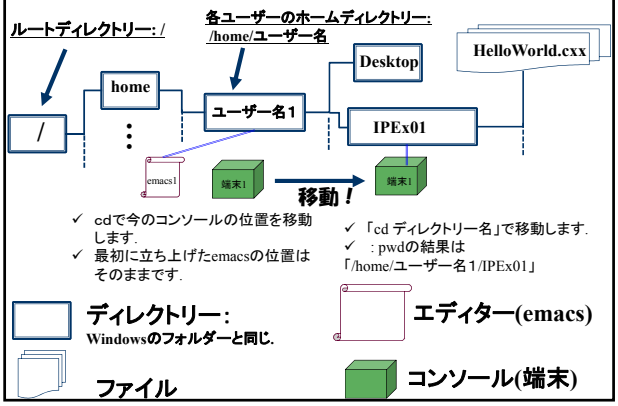



✓ コンソールは立ち上げたら(自分の)ユーザーのホームディレクトリーにいます: pwdの結果は「/home/ユーザー名1」.

✓ 端末から動かしたプログラム(emacs)やコマンドはコンソールがいるディレクトリーで動作します.

Shin Yoshizawa: shin@riken.jp

## ディレクトリー構造

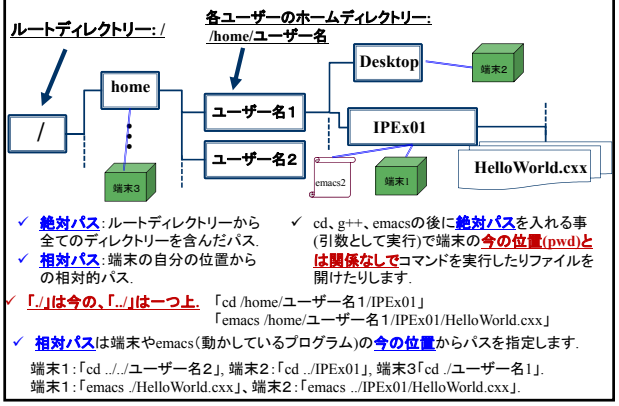



✓ cdで今のコンソールの位置を移動します.
✓ 最初に立ち上げたemacsの位置はそのままです.

✓ 「cd ディレクトリー名」で移動します.
✓ : pwdの結果は「/home/ユーザー名1/IPEx01」

Shin Yoshizawa: shin@riken.jp

## ディレクトリー構造：絶対パス・相対パス




✓ 絶対パス：ルートディレクトリーから全てのディレクトリーを含んだパス.
✓ 相対パス：端末の自分の位置からの相対的パス.

✓ cd、g++、emacsの後に絶対パスを入れる事(引数として実行)で端末の今の位置(pwd)とは関係なしでコマンドを実行したりファイルを開けたりします.

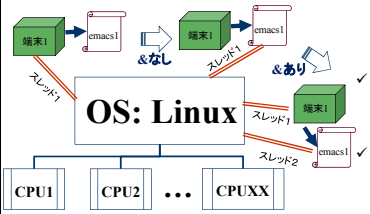
✓ 「./」は今の、「../」は一つ上.
「cd /home/ユーザー名1/IPEx01」
「emacs /home/ユーザー名1/IPEx01/HelloWorld.cxx」

✓ 相対パスは端末やemacs(動かしているプログラム)の今の位置からパスを指定します.
端末1：「cd ../../ユーザー名2」, 端末2：「cd ../IPEx01」, 端末3「cd ./ユーザー名1」.
端末1：「emacs ./HelloWorld.cxx」、端末2：「emacs ../IPEx01/HelloWorld.cxx」.

Shin Yoshizawa: shin@riken.jp

## マルチタスク vs シングルタスク

- ✓ プログラムは(古典的には)一つのCPU (Central Processing Unit:中央演算子)に一つしか動かせない！
- ✓ マルチタスク:OSの機能としてCPUの数よりも多くのプログラムをプロセス(スレッド)として動かす事.
- ✓ マルチスレッド:プログラミングとして複数のプロセスを管理して動かす事.




- ✓ 「emacs」と「&」を付けないで端末から実行した場合は端末に割り当てられていたスレッドがemacsに渡される→emacsからスレッドが戻ってこないと端末は動かない.
- ✓ 「emacs &」と実行する事で端末に割り当てられていたスレッドとは別のスレッドが割り当てられて端末もemacsも両方とも使える.
- ✓ 「&」なしで動かしていたプログラムは端末にて「Control-Z」の後に「bg」コマンドでそのプログラムをバックグランド化しできる.

Shin Yoshizawa: shin@riken.jp

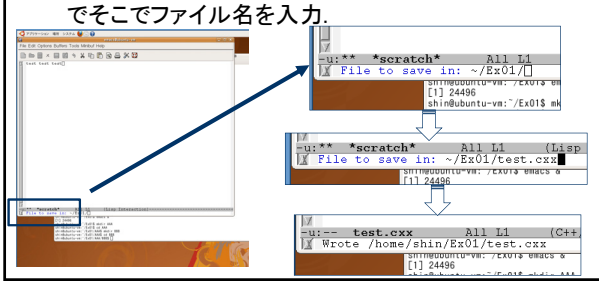
## cd, ls, mkdir, emacs

1. Linuxを立ち上げて、ログインしてください.
2. 端末(コンソール)を立ち上げてください.
3. 「pwd」と打ち込んでみてください. 自分のホームディレクトリーがでます.
4. 「ls」と打ち込んでみてください.
5. 「mkdir IPEx01」と打ち込んで第一回演習用のディレクトリーを作りましょう.
6. 「cd IPEx01」「pwd」として確認した後に「cd ../」「pwd」としてみましょう.「./」は今の、「../」は一つ上のディレクトリーの意味があります.
7. 「cd IPEx01」で先ほどのディレクトリーに戻った後に「emacs 」と打ち込んでエディターを立ち上げましょう. 端末で「Control-C」で強制終了した後に「emacs &」でもう一度立ち上げてください.

Shin Yoshizawa: shin@riken.jp

## emacs補足

- ✓ emacsについて:
  - ファイルの入出力:「Control-X」、「Control-S」又は、左上のFile(ファイル)→save buffer as(名前を付けて保存)でemacsの下にファイル名を入力する所が現れるのでそこでファイル名を入力.



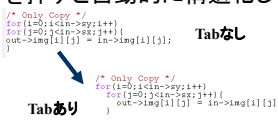

Shin Yoshizawa: shin@riken.jp

## emacs補足

- ✓ emacsについて:
  - 日本語(全角)入力・英語(半角英数)入力の切り替え「Control-¥」.

✓ プログラムはいつも半角英数で入力してください.


  - 下の部分でセーブモード等のコマンド入力状態からの脱出は「Control－G」.
  - プログラムは「Tab」キーを押すと自動的に構造化して見やすくなります.
  - Control-Kで一行削除.
  - Undoは「Control-/」




Shin Yoshizawa: shin@riken.jp

## 補足

- ✓ 「¥」はバックスラッシュ「 \ 」でキーボード右上の「¥」キーに対応しています. 以後スライドに¥が出てきたら「 \ 」に読み替えてください.
- ✓ 「Control-X」や「Control-S」の「Control」はキーボード左下の「Ctrl」キーを押しながらの意味です.
- ✓ emacsでファイルの保存や入力のときにおかしくなったら「Control-G」を押してみてください.

Shin Yoshizawa: shin@riken.jp

## Hello World

2. EmacsでC言語のHelloWorldを書いてみよう:

7. emacsに以下のプログラムを半角英数で打ち込んでください.

```
#include<stdio.h>
int main(int argc, char *argv[ ]){
    printf("Hello World ¥n");
    return 0;
}
```

8. Control-X Control-Sでセーブできますので、「HelloWorld.cxx」という名前でセーブしてください.
9. 端末で「g++ HelloWorld.cxx」と打ち込んでみましょう.「a.out」という実行ファイルが出来るので、「./a.out」と実行してみてください「Hello World」と端末にでれば成功です.「g++ -o 実行ファイル名 ソースファイル名」で名前を指定してコンパイルできます.

「g++」はGNU のC++コンパイラーです.

## C: Hello Worldの説明

stdio.hはprintf等の標準入出力関数群

```
#include<stdio.h>
```
ヘッダーファイル(.h)の読み込み

mainはLinuxでは必ずint型の関数.

```
int main(int argc,char *argv[]){
```
argcにはこのプログラムが実行されたときの引数の数が入る.
*argv[]には引数の文字列が入る.

```
printf("Hello World \n");
```
printfは端末(標準入出力)に文字や数値を出力する関数, 「¥n」は改行.

```
return 0;
```
プログラムの正常終了を表す0をOSに返す.

```
}
```



## 第一回演習:C: Hello World & pnm画像 & 閾値

3. EmacsでC言語のargc, argvを使ってみよう:

```
#include<stdio.h>

int main(int argc,char *argv[]){


printf("argc = %d\n",argc);
 int i;
 for(i=0;i<argc;i++)
   printf("argv[%d] = %s\n",i,argv[i]);


return 0;

}
```



## C: argvの説明

端末での実行結果

```
shin@ubuntu-vm:~$ ./a.out
argc = 1
argv[0] = ./a.out
shin@ubuntu-vm:~$ ./a.out abc
argc = 2
argv[0] = ./a.out
argv[1] = abc
shin@ubuntu-vm:~$ ./a.out abc efg
argc = 3
argv[0] = ./a.out
argv[1] = abc
argv[2] = efg
shin@ubuntu-vm:~$ ./a.out abc efg 1.234
argc = 4
argv[0] = ./a.out
argv[1] = abc
argv[2] = efg
argv[3] = 1.234
shin@ubuntu-vm:~$ ./a.out abc efg 1.234 HIJK
argc = 5
argv[0] = ./a.out
argv[1] = abc
argv[2] = efg
argv[3] = 1.234
argv[4] = HIJK
shin@ubuntu-vm:~$
```

ソースファイル

```
#include<stdio.h>

int main(int argc,char *argv[]){


printf("argc = %d\n",argc);
 int i;
 for(i=0;i<argc;i++)
   printf("argv[%d] = %s\n",i,argv[i]);


return 0;

}
```

- ✓ %dはint、%sはchar []やchar *の文字列を表示するときに使う.
- ✓ floatは%f, doubleは%lf、charは%c.
- ✓ argcにはこのプログラムが実行されたときの引数の数が入る.
- ✓ *argv[]には引数の文字列が入る.
- ✓ プログラム内ではargv[1],argv[2]とかで使う.argv[0]には実行ファイル名が入る.
- ✓ 文字列な事に注意！数値として使いたい場合は、stdlib.hをインクルードしてatoi()やatof()を使う.



## 第一回演習:C: Hello World & pnm画像 & 閾値

4. EmacsでC言語のargc, argvを使ってみよう:

```
#include<stdio.h>
#include<stdlib.h>

int main(int argc,char *argv[]){


  printf("argv[1] = %s\n",argv[1]);

  int a = atoi(argv[1]);
  printf("atoi(argv[1]) = %d\n",a);

  double b = atof(argv[1]);

  printf("atof(argv[1]) = %lf\n",b);

  return 0;
}
```



## C: argvの説明

端末での実行結果

```
shin@ubuntu-vm:~$ ./a.out abc
argv[1] = abc
atoi(argv[1]) = 0
atof(argv[1]) = 0.000000
shin@ubuntu-vm:~$ ./a.out 1
argv[1] = 1
atoi(argv[1]) = 1
atof(argv[1]) = 1.000000
shin@ubuntu-vm:~$ ./a.out 2
argv[1] = 2
atoi(argv[1]) = 2
atof(argv[1]) = 2.000000
shin@ubuntu-vm:~$ ./a.out 1.2
argv[1] = 1.2
atoi(argv[1]) = 1
atof(argv[1]) = 1.200000
shin@ubuntu-vm:~$ ./a.out 0.567
argv[1] = 0.567
atoi(argv[1]) = 0
atof(argv[1]) = 0.567000
shin@ubuntu-vm:~$ ./a.out 123.456
argv[1] = 123.456
atoi(argv[1]) = 123
atof(argv[1]) = 123.456000
shin@ubuntu-vm:~$
```

ソースファイル

```
#include<stdio.h>
#include<stdlib.h>

int main(int argc,char *argv[]){


  printf("argv[1] = %s\n",argv[1]);

  int a = atoi(argv[1]);
  printf("atoi(argv[1]) = %d\n",a);

  double b = atof(argv[1]);

  printf("atof(argv[1]) = %lf\n",b);

  return 0;
}
```

- ✓ 文字列な事に注意！数値として使いたい場合は、stdlib.hをインクルードしてatoi()やatof()を使う.



## 演習

5. C言語でpnm画像の入出力を書いてみよう:

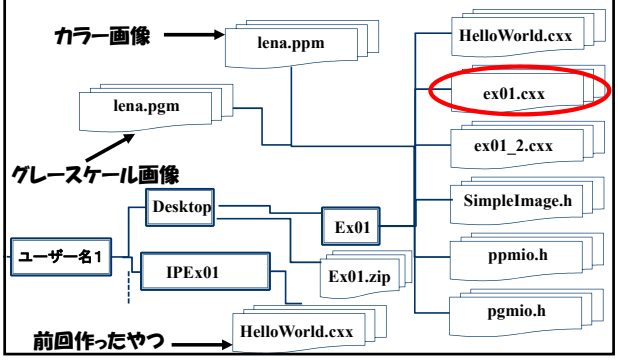
10. 端末でfirefoxを立ち上げて
www.riken.jp/brict/Yoshizawa/Lectures/Ex01.zip
を開いて、学籍番号_Ex01のディレクトリーにダウンロードしてください.
**Firefox:編集→設定→一般→ダウンロード→「ファイルごとに保存先を指定する」にチェックを入れてください.**

11. 端末で「unzip Ex01.zip」として展開後に「cd Ex01」、「emacs ex01.cxx &」でプログラムを開いてください. ex01.cxxはpgm画像を読み込んでそのままセーブするプログラムです.

Shin Yoshizawa: shin@riken.jp

## 演習

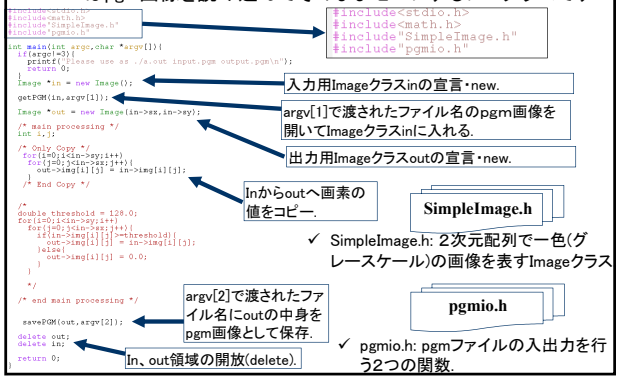
ex01.cxxはpgm画像を読み込んでそのままセーブするプログラムです.




Shin Yoshizawa: shin@riken.jp

## 演習:pgm入出力

ex01.cxxはpgm画像を読み込んでそのままセーブするプログラムです.




Shin Yoshizawa: shin@riken.jp

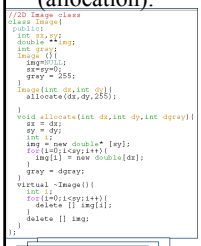
## 演習:Imageクラス

SimpleImage.h: 2次元配列で一色の画像を表すImageクラス.

#include"SimpleImage.h"**した後の使い方例:**

**宣言・メモリ確保**(allocation):

```
Image *in = new Image();
Image *out = new Image(in->sx,in->sy);
```



SimpleImage.h

**処理:**

```
getPGM(in,argv[1]);
```

```
/* Only Copy */
for(i=0;i<in->sy;i++)
 for(j=0;j<in->sx;j++){
   out->img[i][j] = in->img[i][j];
 }
```

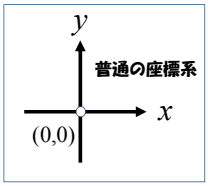
✓ 画像サイズ:縦:sy、横sx.
✓ **(座標(i,j)での)画素値:img[i][j]**

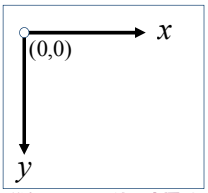
**メモリの開放:**

```
delete out;
delete in;
```

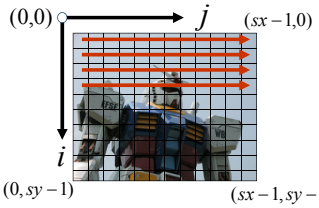
Shin Yoshizawa: shin@riken.jp

## 復習:デジタル画像の座標と配列




画像処理でよく使う座標系




**輝度値の配列表現:**

int $I[sy][sx]$;
double $I[sy][sx]$;

$for(i=0;i<sy;i++)\{$
$for(j=0;j<sx;j++)\{$
$I[i][j]=\ldots$
$\}$
$\}$

Shin Yoshizawa: shin@riken.jp

## 演習:getPGM(), savePGM()

pgmio.h: pgmファイルの入出力を行う2つの関数.

**画像入力:**

```
void getPGM(Image *in,char *filename)
```

**画像出力:**

```
void savePGM(Image *in,char *filename)
```

pgmio.h

#include"pgmio.h"**した後の使い方例:**

**入力:**

```
getPGM(in,argv[1]);
```

✓ argv[1]で渡されたファイル名のpgm画像を開いてImageクラスinに入れる.
✓ **注意:inは下記の様に画像サイズなしでnewされていないといけない!**

```
Image *in = new Image();
```

**出力:**

```
savePGM(out,argv[2]);
```

✓ argv[2]で渡されたファイル名にoutの中身をpgm画像として保存.
✓ **注意:outは下記の様に画像サイズありでnewされていないといけない!**

```
Image *out = new Image(in->sx,in->sy);
```

Shin Yoshizawa: shin@riken.jp

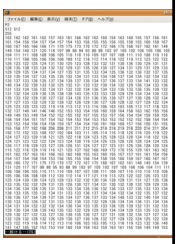
## 演習資料:pnm画像フォーマット

✓ 一番簡単な画像フォーマットです:
- グレースケール画像は「.pgm」、カラー画像は「.ppm」でテキスト形式とバイナリー形式があります.
- グレースケール(.pgm):

1行名: テキストで「P2」
2行目: 画像サイズ(横:width 縦:height)
3行目:画素の階調(最大値) 8bitの場合は255
4行目から: integerで画素値スペース画素値…

- カラー(.ppm):

1行名: テキストで「P3」
2行目: 画像サイズ(横:width 縦:height)
3行目:画素の階調(最大値) 8bitの場合は255
4行目から: integerでR G B R G B R G B…



Shin Yoshizawa: shin@riken.jp

## 演習資料：pgmを端末上でmoreしてみよう！

pgmの非圧縮形式はテキストファイルなのでmoreで中身が見れます.

1. Ex01.zipを展開したディレクトリーEx01に「cd」を使って端末の位置を動かす「cd /home/ユーザー名/学籍番号_Ex01/Ex01」.
2. 「ls」でディレクトリーの中身を確認(lena.pgmが入っていればOK). 入ってなかったら「pwd」で今の端末の位置を確認して、「cd」と「ls」を使ってEx01の場所を探す.「cd」だけでエンターキーを押すと自分のホームディレクトリーに戻ります.
3. 端末にて「more lena_s.pgm」でエンターキーを押してみる.

✓ moreはスペースキーで先に進みます. 途中で終了するのはControl-Cです.

Shin Yoshizawa: shin@riken.jp

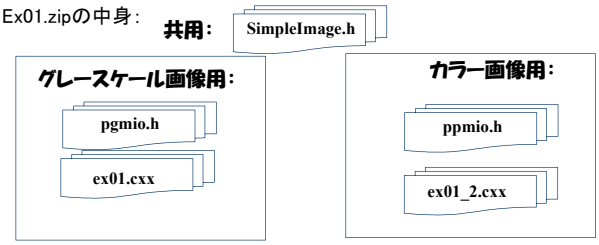
## 第一回演習：C: Hello World & pnm画像 & 閾値

5. C言語でpnm画像の入出力を書いてみよう:
   10. 端末でfirefoxを立ち上げて www.riken.jp/brict/Yoshizawa/Lectures/Ex01.zip を開いて、学籍番号_Ex01のディレクトリーにダウンロードしてください.
   11. 端末で「unzip Ex01.zip」として展開後に「cd Ex01」、「emacs ex01.cxx &」でプログラムを開いてください. ex01.cxxはpgm画像を読み込んでそのままセーブするプログラムです。
   12. 端末で「g++ ex01.cxx」として実行ファイルa.outを作成後に「./a.out lena.pgm test.pgm」としてください。その後に「display test.pgm &」と「display lena.pgm &」を実行して同じ画像である事を確認してください.
   13. 同様に「g++ ex01_2.cxx」、「./a.out lena.ppm test.ppm」、「display lena.ppm &」、「display test.ppm &」として同じカラー画像である事を確認してください.

Shin Yoshizawa: shin@riken.jp

## 演習

Ex01.zipの中身：




✓ 今後の全ての演習はこれらのファイル中のプログラム構造を雛形として使っていきますので中身をよく見ておいてください.

✓ カラー画像用ではImageクラスをR,G,B3つ使っているだけです.

Shin Yoshizawa: shin@riken.jp

## 第一回演習：C: Hello World & pnm画像 & 閾値

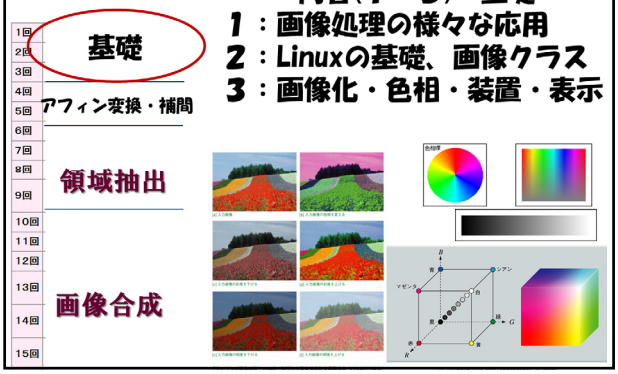
6. C言語でpnm画像の入出力を書いてみよう:
   14. ex01.cxxで画像をCopyしているところをコメントアウトして、その下の既にコメントアウトしてある部分をコメントアウトを外してください. 再度コンパイル→実行してどんな画像が生成されたか確認してみてください. そのときに「./a.out lena.pgm test1.pgm」と名前を変えてください。
   15. Threshould=128.0となっているところを32.0、64.0、160.0、192.0と変えた場合にどんな画像が生成されるか確認してみてください.同様に「test2.pgm,test3.pgm,test4.pgm,test5.pgm」違う名前でセーブしてください.
   16. 同様にex01_2.cxxの方でも閾値を変えて実行してみてください.

ファイル名「test1.ppm,…test5.ppm」.

以上で第一回演習は終了です.

Shin Yoshizawa: shin@riken.jp

## 次回の予定




Shin Yoshizawa: shin@riken.jp

## 演習資料：UNIX コマンド/ソフト入門

✓ よく使うコマンド

- exit: 終了コマンド.
- Control-C: 動作中のプログラムの強制終了.無限ループの時とかに使います.
- man: マニュアル.「man ls」
- cd: ディレクトリー(フォルダー)の移動.「cd ディレクトリー名」
- ls:ディレクトリー内のファイル名・フォルダー名を表示.「ls ディレクトリー名」、「ls ./」「ls ../」.「ls -lh」、「ls -alh」
- pwd: 現在のディレクトリーを表示.「pwd」
- mv; ファイルやディレクトリーを移動・上書き.「mv AAA BBB」AAAをBBBに上書き・移動はAAAとBBBがファイルなのかディレクトリーなのかで動作が異なります.
  - AAA(ファイル)、BBB(ファイル)のときは上書き:BBBが消されてAAAがBBBという名前になります.
  - AAA(ファイル)、BBB(ディレクトリー)及びAAA,BBB共にディレクトリーのときはBBBの下にAAAが移動します.
  - BBB(ディレクトリー)、AAA(ファイル)のときはエラーです.
- mkdir; ディレクトリーの作成.「mkdir ディレクトリー名」

Shin Yoshizawa: shin@riken.jp

## 演習資料：UNIX コマンド/ソフト入門

✓ よく使うコマンド
- rmdir: ディレクトリーの削除.「rmdir ディレクトリー名」
- rm: ファイルやディレクトリーの削除.「rm ファイル名」、「rm –r ディレクトリー名」.
- more: テキストファイルの中身の表示.「more ファイル名」バイナリーファイルはmore で見るとエラーで端末がおかしくなるので注意です.
- zip: ファイル圧縮.「zip ファイル名.zip ファイル名」、「zip –r ディレクトリー名.zip ディレクトリー名」
- unzip: ファイル解凍.「unzip ファイル名」
- cp: ファイルやディレクトリーのコピー.「cp AAA BBB」AAAとBBBのファイルかディレクトリーの違いは「mv AAA BBB」と同じです.

- コマンドの後に「＆」を付けるとバックグラウンド処理になるのでemacsやfirefox等のプログラムを動かす場合は「firefox &」とするとよい.
- 「|」はパイプと言ってコマンドを繋げる「ls | more」など.

Shin Yoshizawa: shin@riken.jp

## 演習資料：UNIX コマンド/ソフト入門

✓ よく使うソフト：
- 端末：xterm
- WEBを見る：firefox
- 画像を見る・変換する：display、convert:
「display 画像ファイル名」でGUI付ソフト(ImageMagick)が立ち上がる「convert –quality 100 画像ファイル名　画像ファイル名」で画像のフォーマット変換：「convert –quality 100 –compress none AAA.ppm AAA.pgm」等
- プログラムを書く：emacs
- C/C++言語：gcc, g++, make
- Java言語：javac, java
- レポート・文章作成： platex. xdvi,
- ps・pdfファイルを見る: evince, acroread

Shin Yoshizawa: shin@riken.jp

## 演習資料：pnm画像フォーマット

✓ 一番簡単な画像フォーマットです：
- グレースケール画像は「.pgm」、カラー画像は「.ppm」でテキスト形式とバイナリー形式があります.
- グレースケール(.pgm):

1行名: テキストで「P2」
2行目： 画像サイズ(width height)
3行目：画素の階調(最大値) 8bitの場合は255
4行目から: integerで画素値スペース画素値…

- カラー(.ppm):

1行名: テキストで「P3」
2行目： 画像サイズ(width height)
3行目：画素の階調(最大値) 8bitの場合は255
4行目から: integerでR G B R G B R G B…