PyRAF

Python Scripting on IRAF
(PyRAF を用いたデータ処理入門)

高田　唯史
(国立天文台・天文データセンター)

PyRAF

今日の講習内容

- Pythonについて(すごく簡単に)
- PyRAFを知る(インタラクティブに使ってみる)
- Pythonを使ったスクリプトを作ってPyRAFを用いた画像処理を行ってみる

Spiral Galaxy NGC 3370 HUBBLESITE.org

Pythonについて

- オブジェクト指向スクリプト言語
- 1990年にオランダ人**Guido van Rossum氏によって開発されたプログラミング言語**
- その可読性の高さ、利用のしやすさ、拡張性の高さなどから世界的に利用者が広がり、将来性も豊かであるとされている
- フリーソフトであり、関連する様々なライブラリも大変多く出回っている
- Python Software Foundationがライセンスを持つ

http://www.python.org/

Spiral Galaxy NGC 3370 HUBBLESITE.org

python™

PyRAFについて

- STScIの中で開発が進んでいるstsci_pythonソフトウェア群の中核をなすもの

http://www.stsci.edu/resources/software_hardware/pyraf/stsci_python

- clの短所を乗り越えるため、幅広く利用されているスクリプト言語でIRAF環境を利用できるように考えられたもの（特にエラー処理など）
- **「IDLのようなコマンドラインによるデータアクセスと小細工をできる環境の提供」**を目的とする
- 2003年ぐらいにはかなりしっかりとした形になってきていた。

PyRAFとIRAF

cl

Python

IRAF Wrapper(PyRAF)

IRAF本体（実行プログラムとかclスクリプト）

IRAF
Image Reduction and Analysis Facility

E.org

PyRAFを使うメリット

- Pythonの文法を知っていれば簡単にスクリプトが組める
- Pythonの参考書・WEBサイトは非常に多い
- 外部ソフトウェアとの連携が簡単(豊富なライブラリ：　今日の例はプロッティング・ソフト)
- clでは出来ないことがいくつか出来る
- 既に枯れていると思われるIRAFのタスクを一般的なスクリプト言語からIRAFと連携良く呼べる

PyRAFとIRAF（cl）で出来ることの違い

- PyRAFにはbyeはない(タスクunloadができない）
- PyRAFではGOTO文が使えない
- PyRAFではbackground処理は出来ない
- **Error traceback はCL line numberを出力しない**

- 上矢印キーでヒストリー参照でき、左右矢印で編集可能
- Ctrl-Rキーでhistoryのパターンマッチができる等
- Tabキーでコマンド補完
- セッション情報をファイルに書ける。またそれをrestoreできる。
- グラフ・ウィンドウを複数開ける
- パラメータエディット(epar)がGUIでできる(helpもそこで参照可能）

(詳細はPyRAF Tutorialなどを参照のこと）

PyRAFを使うための準備

- IRAFを利用できる環境に自分たちの環境を設定する

作業用のディレクトリにおいて、

% xgterm　（xgtermを起動）

% mkiraf　（IRAFの環境を整備）

　　（terminal-typeはxgtermを指定すること）

% ds9 &（SAOimage DS9が起動できることを確認）

PyRAFを
インタラクティブで使ってみる

- eparの実行・確認(helpとの連動）
- 複数のウィンドウを起動する
- Historyやコマンド補完機能の確認
- displayタスクの実行

```
--> display dev$pix 1 fill+
```

- imexamineタスクの実行

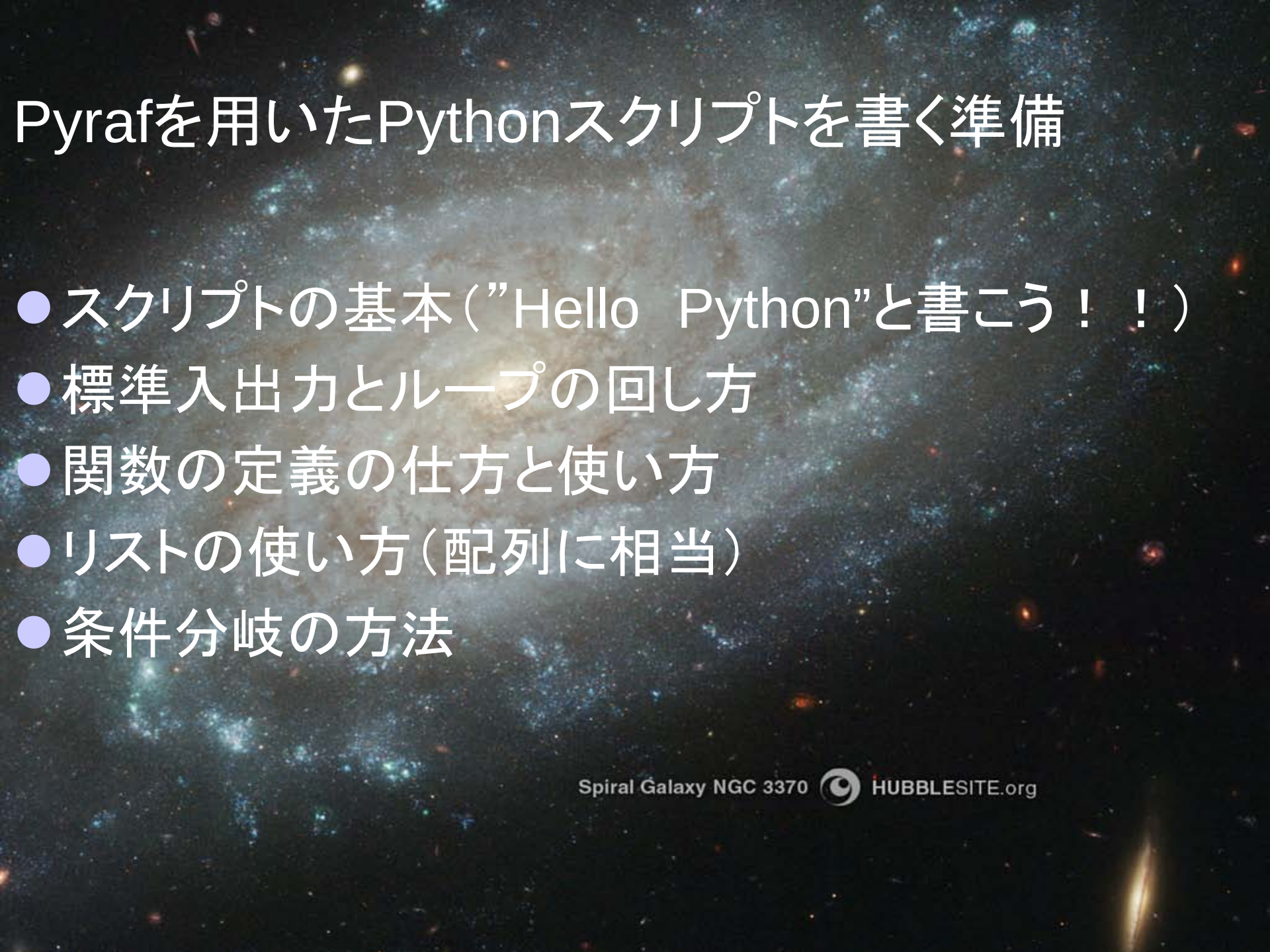
```
--> imexamine 1 （’r’や’e’といった基本的なコマンド確認）
```

- Python-likeなタスクの実行方法

```
--> iraf.display(”dev$pix”,1,fill=yes） 等
```

Pyrafを用いたPythonスクリプトを書く準備

- スクリプトの基本（”Hello　Python”と書こう！！）
- 標準入出力とループの回し方
- 関数の定義の仕方と使い方
- リストの使い方（配列に相当）
- 条件分岐の方法



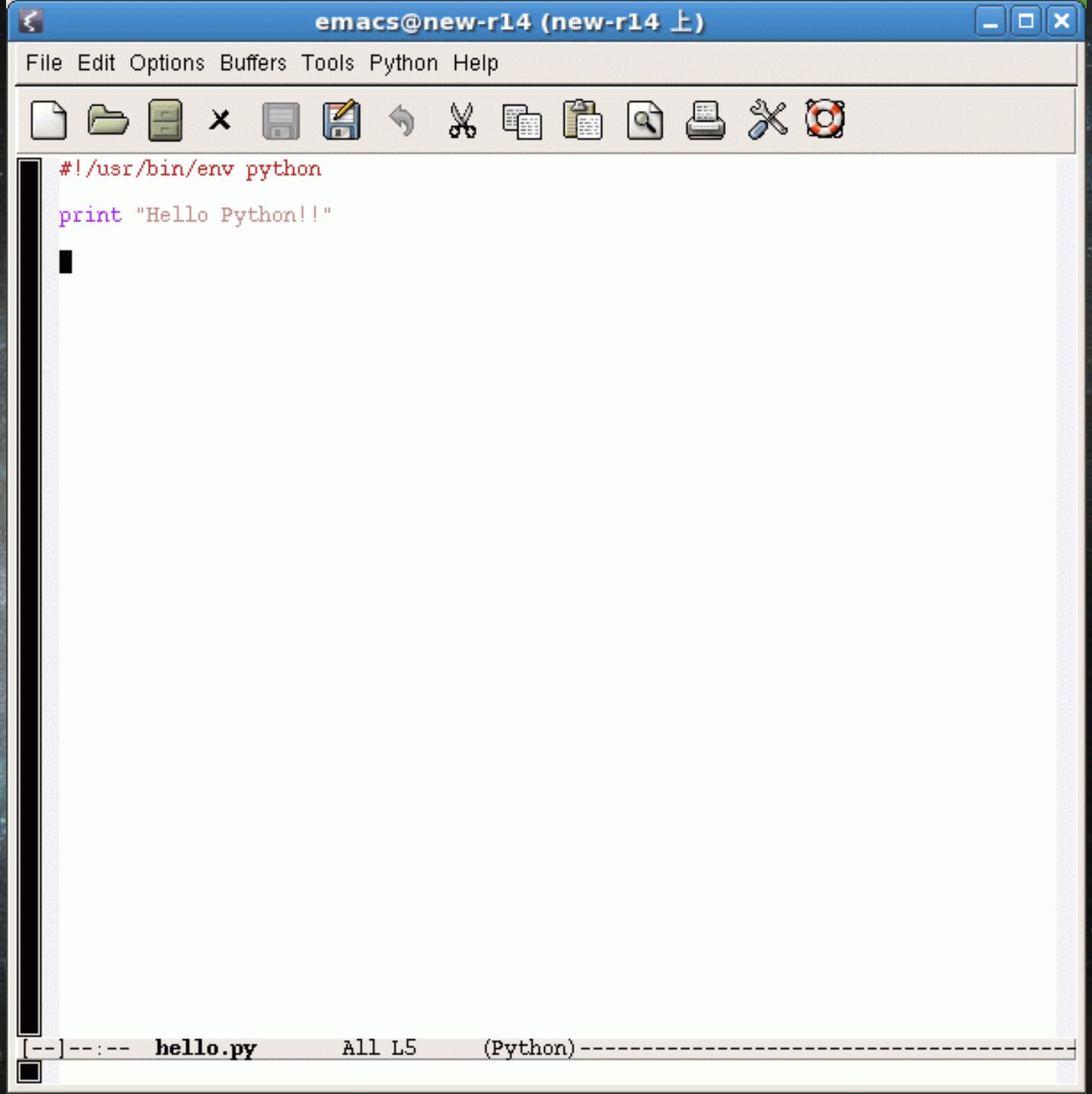
実習1-1：Hello Pythonと出力しよう

- Pythonプログラムの走らせ方を学ぶ
- 他の文字を出力したり、色々試してみよう

参考プログラム

/data/takata/Python/hello.py

Spiral Galaxy NGC 3370 HUBBLESITE.org


emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
print "Hello Python!!"
[--]--:-- hello.py All L5 (Python)

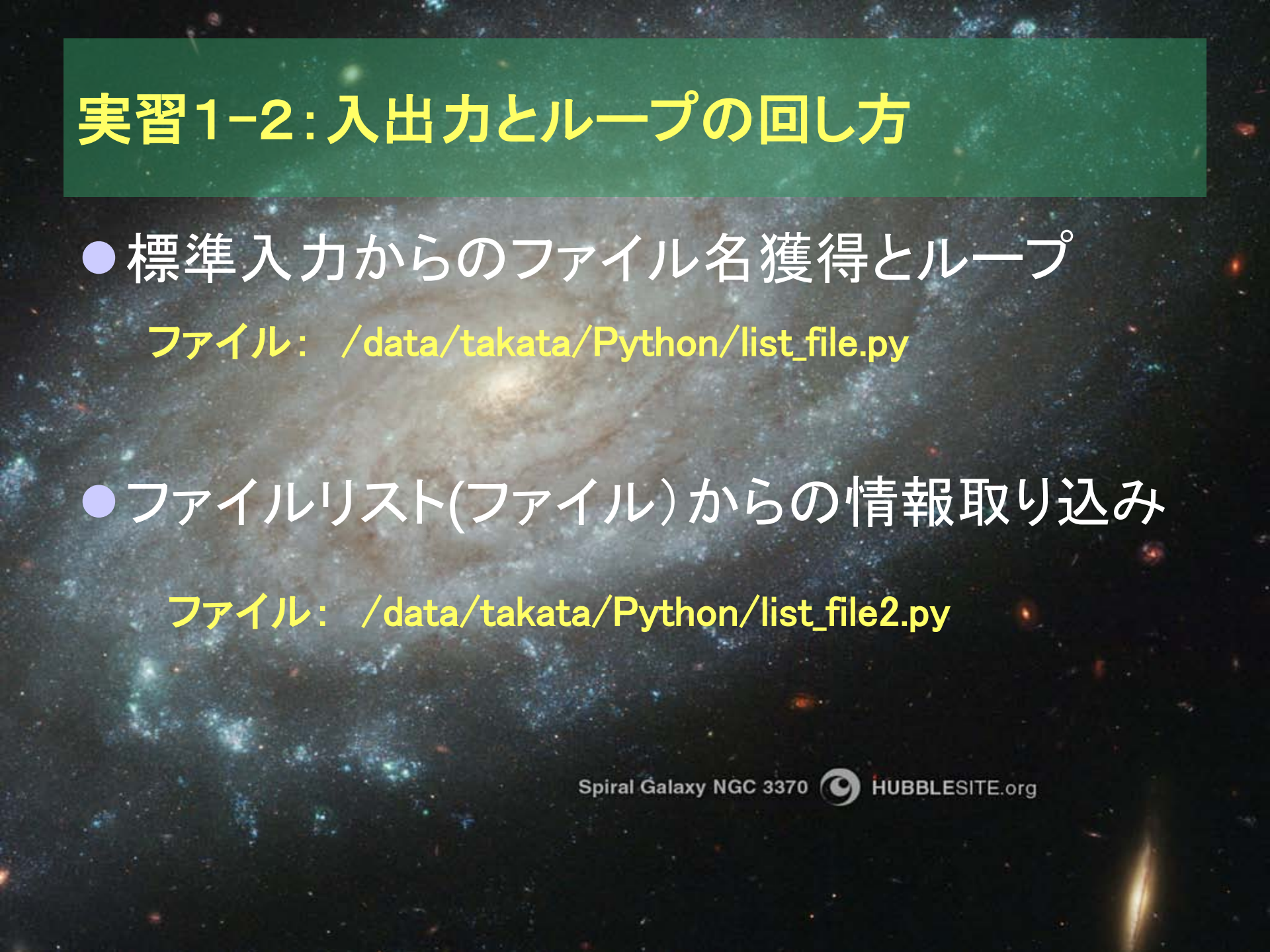
実習1-2：入出力とループの回し方

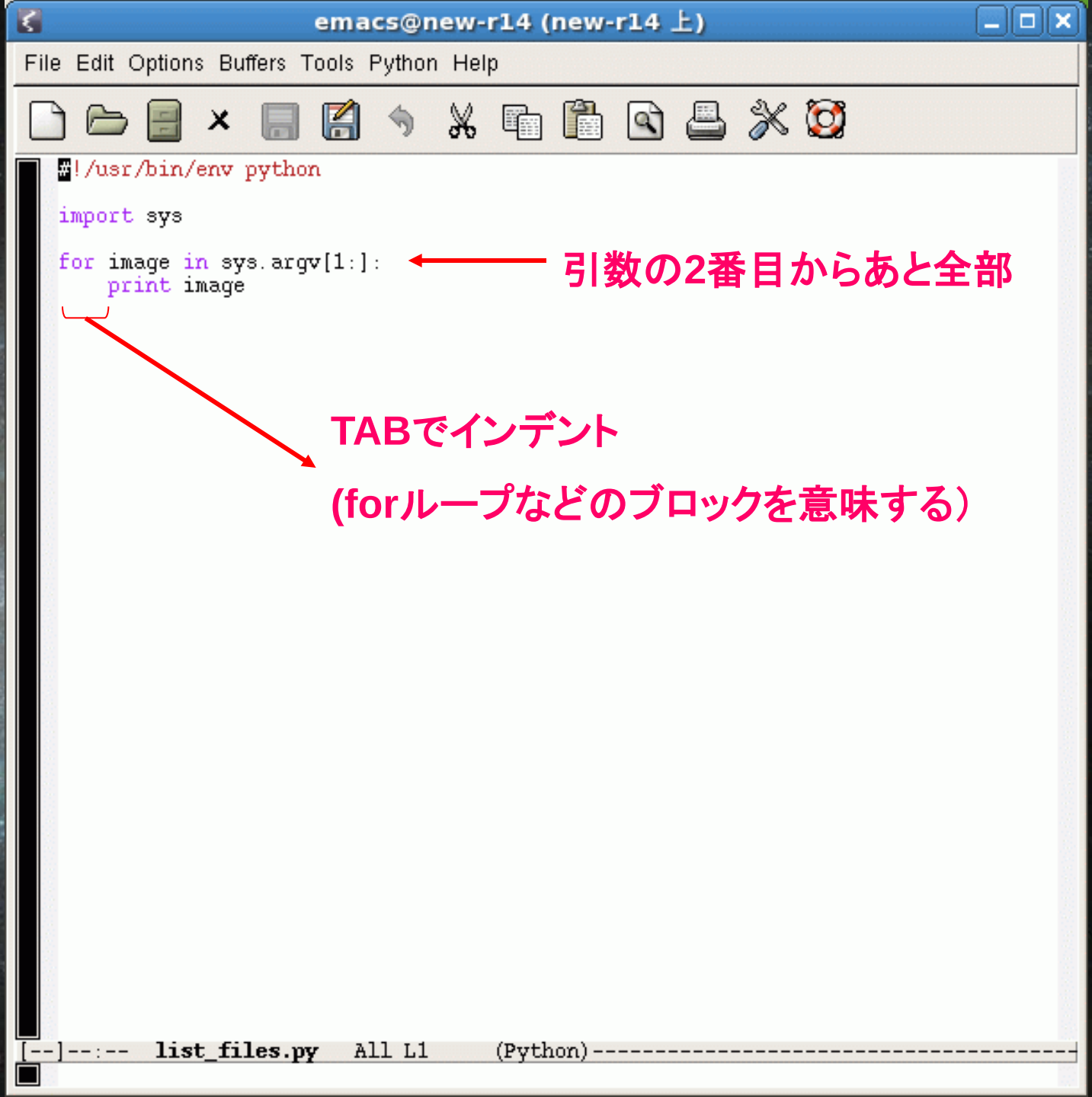
- 標準入力からのファイル名獲得とループ

  ファイル：　/data/takata/Python/list_file.py

- ファイルリスト(ファイル）からの情報取り込み

  ファイル：　/data/takata/Python/list_file2.py



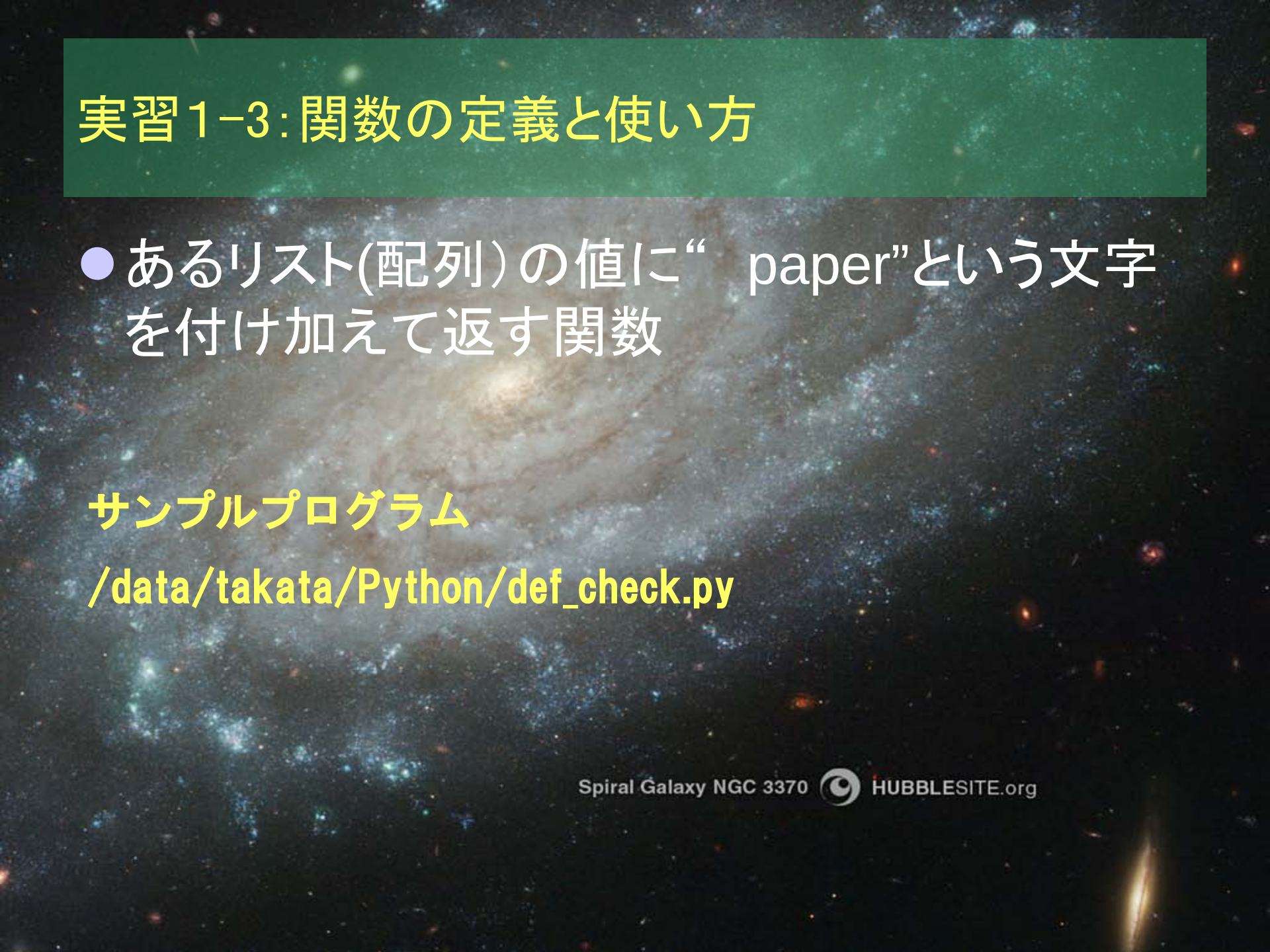

emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
import sys
for image in sys.argv[1:]:
print image
引数の2番目からあと全部
TABでインデント
(forループなどのブロックを意味する)
[--]--:-- list_files.py All L1 (Python)

実習1-3:関数の定義と使い方

- あるリスト(配列)の値に“ paper”という文字を付け加えて返す関数

サンプルプログラム

/data/takata/Python/def_check.py



emacs@new-r14 (new-r14 上)

File Edit Options Buffers Tools Python Help

```
#!/usr/bin/env python

import sys

colors=["blue", "red", "yellow", "black"]

def append_chars(input):
    output = str(input) + " paper"
    return output

for cnt in range(4):
    result = append_chars(colors[cnt])
    print cnt, result

#print range(4)
```

リスト“colors”の定義

関数“append_chars”の定義

Range(4) => cnt を0～3で1ずつ増やす

[--]--:-- **def_check.py** All L1 (Python)----------

(実習2)imstatを使って複数のFITSファイルの統計量を測定する

- ファイルは以下のものを使用する

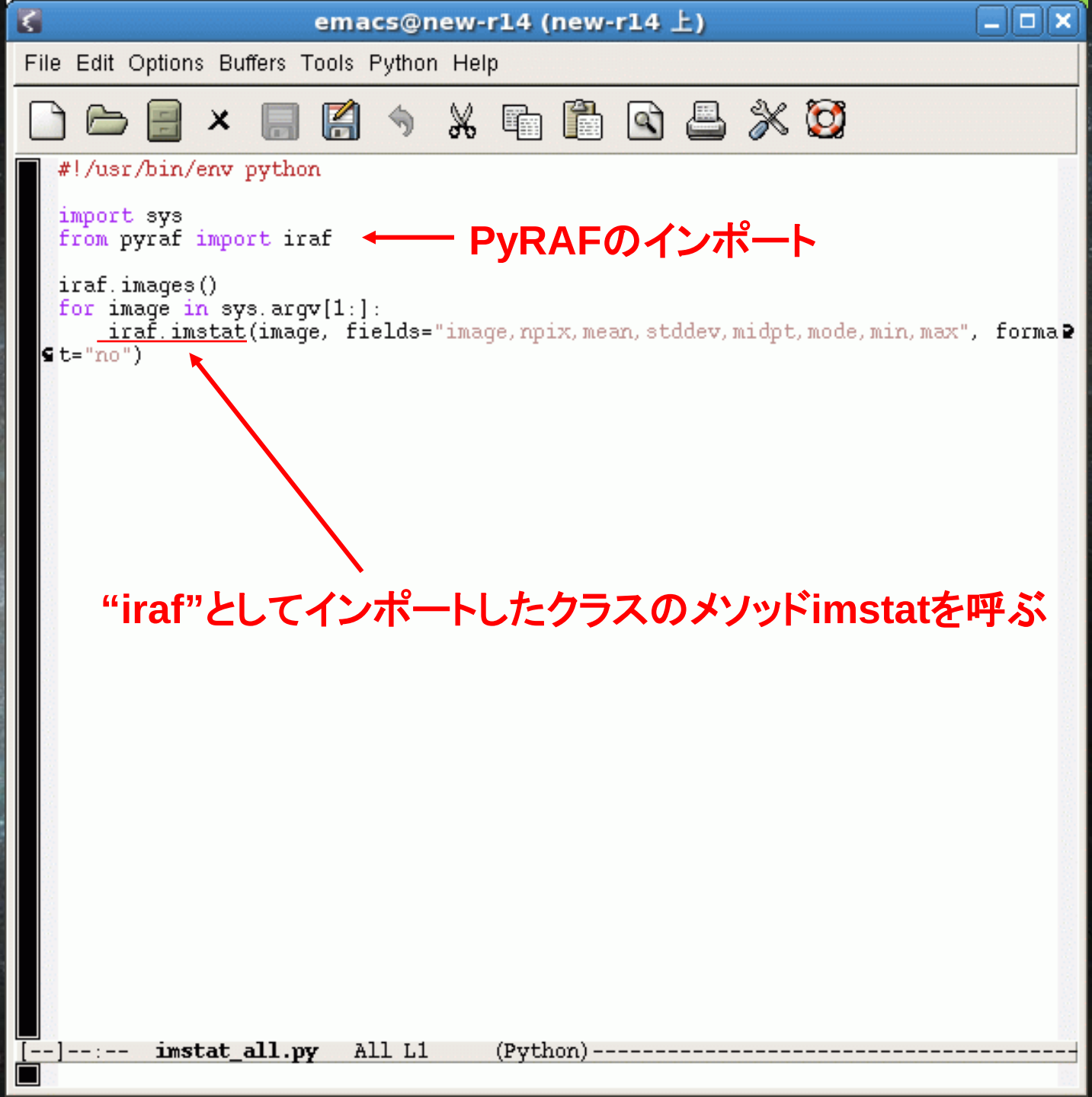
  /data/takata/Image/Fimage_R??.fits

- サンプルプログラムは以下のものを使用する

  /data/takata/PyRAF/imstat_all.py

- サンプルプログラムを改良して、各画像ファイルの[100:500,100:500]の領域の平均値とメジアン値、モード値を求めなさい

Spiral Galaxy NGC 3370 HUBBLESITE.org


emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
import sys
from pyraf import iraf
iraf.images()
for image in sys.argv[1:]:
iraf.imstat(image, fields="image,npix,mean,stddev,midpt,mode,min,max", format="no")
PyRAFのインポート
“iraf”としてインポートしたクラスのメソッドimstatを呼ぶ
imstat_all.py All L1 (Python)

実習3:複数の画像をDS9に並べて表示するプログラムを書いてみる

- IRAFのdisplayコマンドを用いる
- 最後にタイル状に並べてみる(DS9の機能)

  ds9のメニューから、“frame->tile”

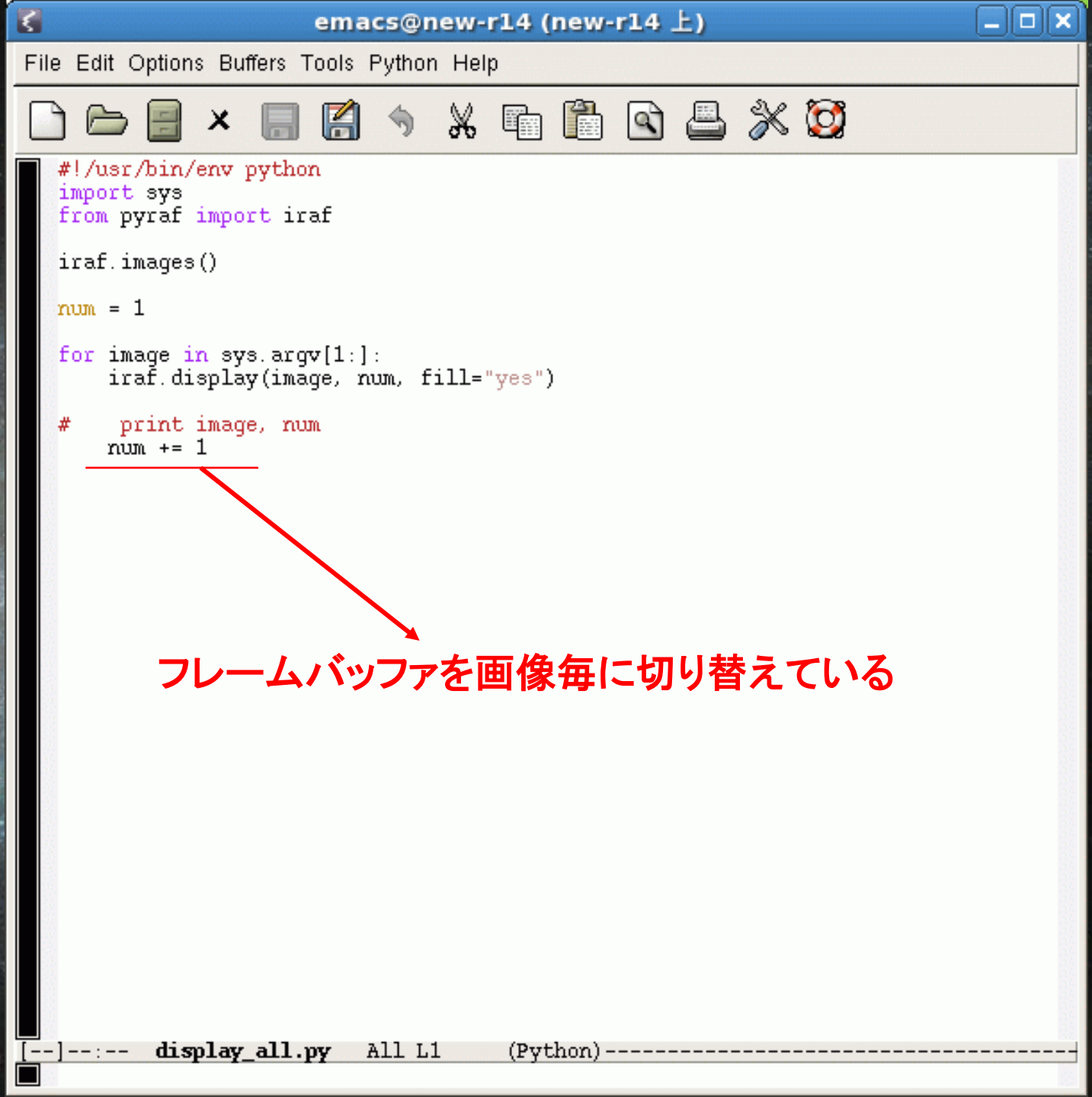
使用するデータ:　/data/takata/Image/Image_R0?.fits

サンプルプログラム:　/data/takata/PyRAF/display_all.py

注意：uparm/tvdisply.par　の中で以下のような定義になっていることが必要。

```
image,f,a," ",,,"image to be displayed"
frame,i,a,1,1,16,"frame to be written into“

bpmask,f,h,"BPM",,,"bad pixel mask"      .....
```


emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
import sys
from pyraf import iraf
iraf.images()
num = 1
for image in sys.argv[1:]:
iraf.display(image, num, fill="yes")
# print image, num
num += 1
フレームバッファを画像毎に切り替えている
[--]--:-- display_all.py All L1 (Python)

実習4：画像のヘッダーの情報を取得する

- 画像ファイルがどの日時（UT）に取得されたかをヘッダー情報から得る
- imgetsタスクを用いること

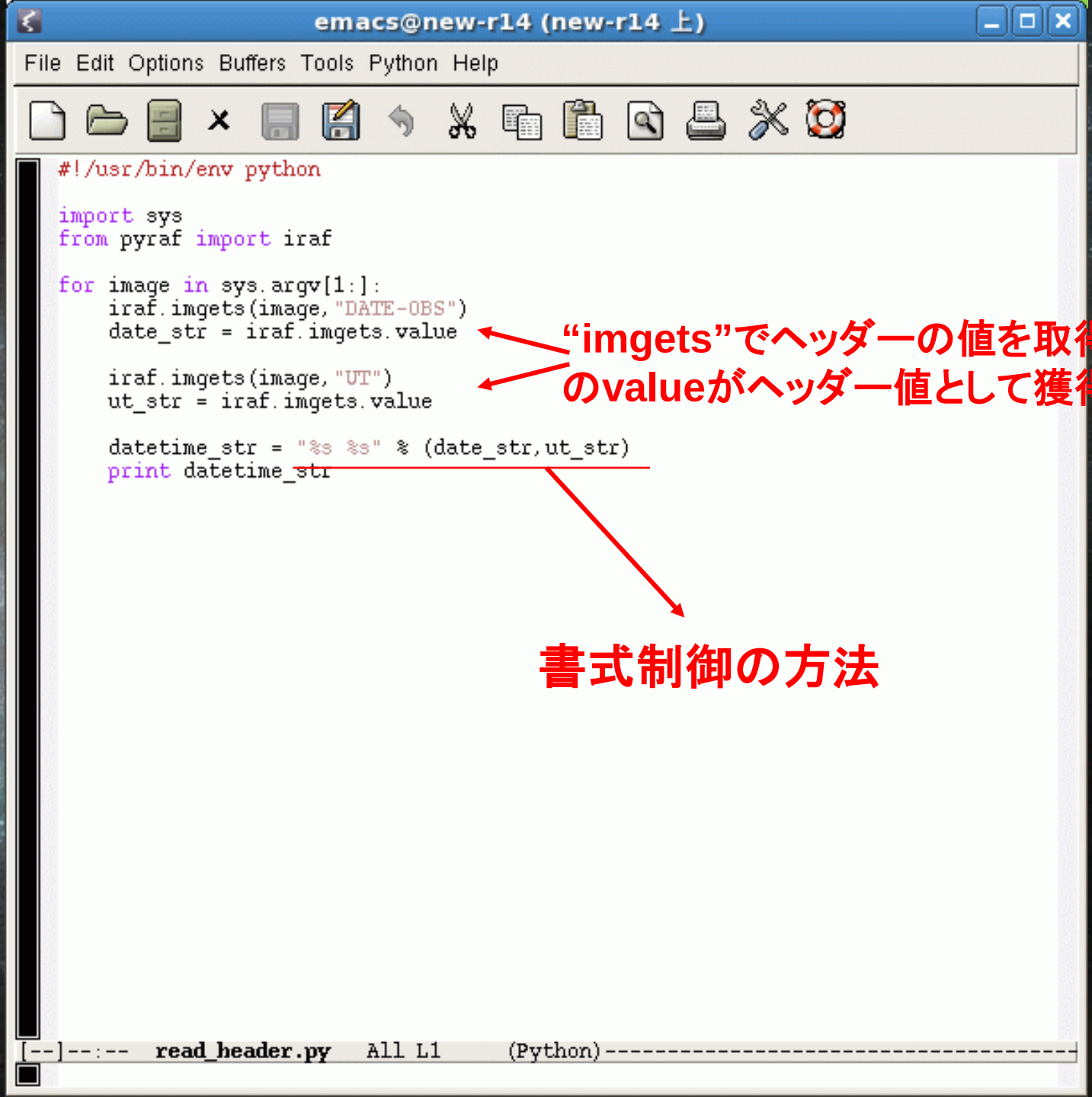
使用データ：　/data/takata/Image/Image_R0?.fits

サンプルプログラム:
/data/takata/PyRAF/read_header.py

（課題）

/data/takata/Imageの中からRバンドで取得されたデータのファイル名を得て全て並べなさい。

（ヒント：FILTER01キーワードの値が“W-C-RC”のものがそれ。）


emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
import sys
from pyraf import iraf
for image in sys.argv[1:]:
iraf.imgets(image,"DATE-OBS")
date_str = iraf.imgets.value
iraf.imgets(image,"UT")
ut_str = iraf.imgets.value
datetime_str = "%s %s" % (date_str,ut_str)
print datetime_str
“imgets”でヘッダーの値を取得し、そのvalueがヘッダー値として獲得できる
書式制御の方法
[--]--:-- read_header.py All L1 (Python)

実習５：複数画像のシフトアンドアッドを行うプログラムを書いてみる

- displayで表示した画像をimexamineで調べ、全てのフレームに写っている共通の天体の（ｘ、ｙ）座標を測る。
  （Stdplot=1オプションを利用する）
- 測定した（ｘ、ｙ）座標をoffset情報を格納するファイルに書き込む
- imcombineを用いてoffset情報を使いながら足し合わせる

サンプルプログラム：　/data/takata/PyRAF/shift_add.py

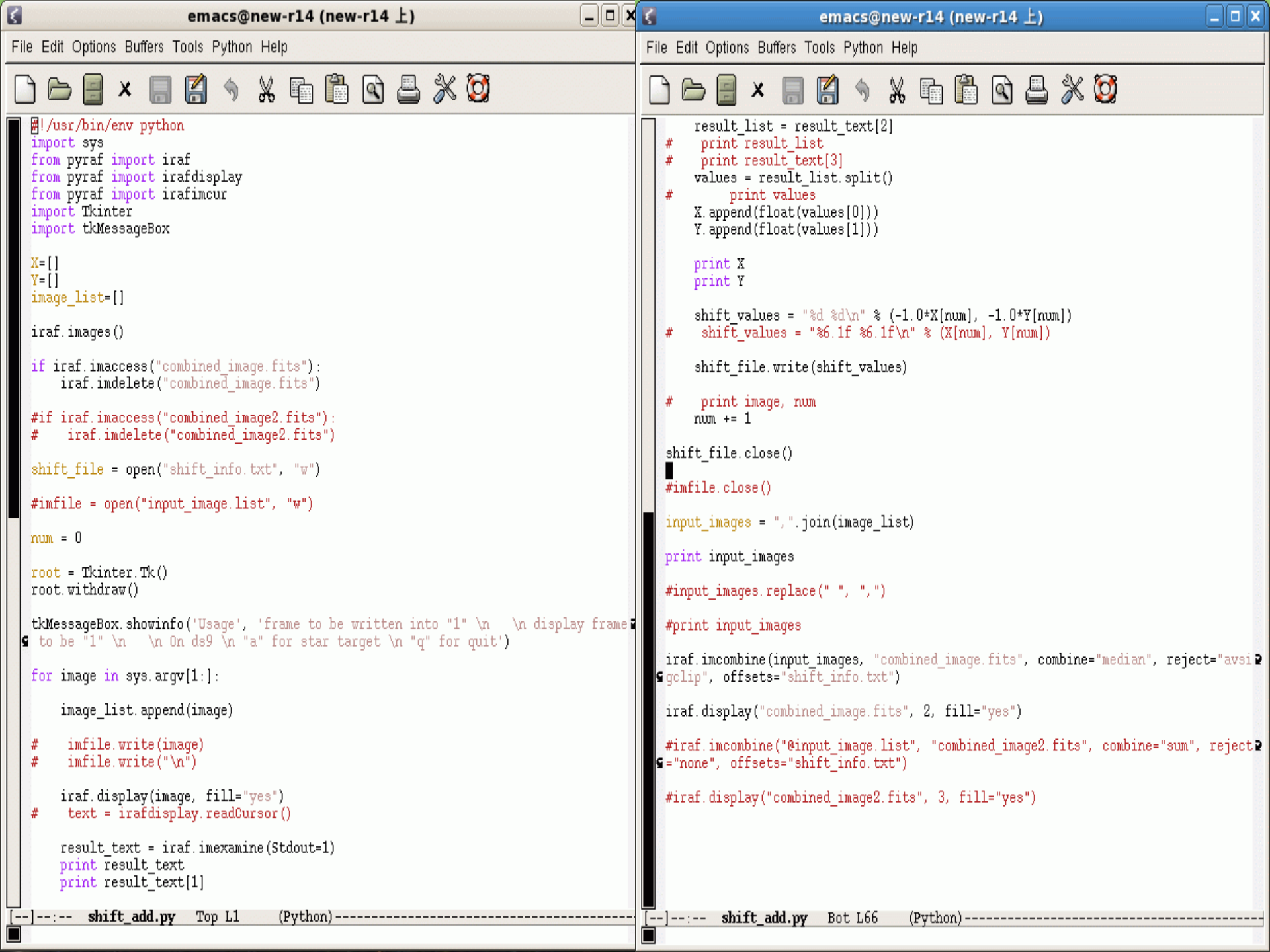
データ：　/data/takata/Image/Image_R0?.fits

ターゲットの星の上で“a”ボタン

終了するためには”q”ボタン

（imexamineのカーソルキーの使用方法と同じ）

- **応用課題：画像ファイルもリストファイルにする**


emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
#!/usr/bin/env python
import sys
from pyraf import iraf
from pyraf import irafdisplay
from pyraf import irafimcur
import Tkinter
import tkMessageBox
X=[]
Y=[]
image_list=[]
iraf.images()
if iraf.imaccess("combined_image.fits"):
iraf.imdelete("combined_image.fits")
#if iraf.imaccess("combined_image2.fits"):
# iraf.imdelete("combined_image2.fits")
shift_file = open("shift_info.txt", "w")
#imfile = open("input_image.list", "w")
num = 0
root = Tkinter.Tk()
root.withdraw()
tkMessageBox.showinfo('Usage', 'frame to be written into "1" \n \n display frame to be "1" \n \n On ds9 \n "a" for star target \n "q" for quit')
for image in sys.argv[1:]:
image_list.append(image)
# imfile.write(image)
# imfile.write("\n")
iraf.display(image, fill="yes")
# text = irafdisplay.readCursor()
result_text = iraf.imexamine(Stdout=1)
print result_text
print result_text[1]
[--]--:-- shift_add.py Top L1 (Python)
emacs@new-r14 (new-r14 上)
File Edit Options Buffers Tools Python Help
result_list = result_text[2]
# print result_list
# print result_text[3]
values = result_list.split()
# print values
X.append(float(values[0]))
Y.append(float(values[1]))
print X
print Y
shift_values = "%d %d\n" % (-1.0*X[num], -1.0*Y[num])
# shift_values = "%6.1f %6.1f\n" % (X[num], Y[num])
shift_file.write(shift_values)
# print image, num
num += 1
shift_file.close()
#imfile.close()
input_images = ",".join(image_list)
print input_images
#input_images.replace(" ", ",")
#print input_images
iraf.imcombine(input_images, "combined_image.fits", combine="median", reject="avsigclip", offsets="shift_info.txt")
iraf.display("combined_image.fits", 2, fill="yes")
#iraf.imcombine("@input_image.list", "combined_image2.fits", combine="sum", reject="none", offsets="shift_info.txt")
#iraf.display("combined_image2.fits", 3, fill="yes")
[--]--:-- shift_add.py Bot L66 (Python)

実習６：imstatで得た情報をグラフにしてみる。

- Imstatで画像の平均値を取得
- 取得した情報をPythonのリストに格納
- Matplotlibを用いてグラフを作成
- （結果のグラフをEPSファイル出力する）

サンプルプログラム：/data/takata/PyRAF/plot_imstats.py

データ： /data/takata/Image/Image_R0?.fits

- 応用課題：横軸をデータ取得時間にしてプロットしてみよう。

Spiral Galaxy NGC 3370 HUBBLESITE.org

※ Matplotlibについては http://matplotlib.sourceforge.net/index.html を参照

emacs@new-r14 (new-r14 上)

File Edit Options Buffers Tools Python Help

```
#!/usr/bin/env python

import matplotlib
matplotlib.use('TkAgg')

import sys
from pyraf import iraf
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg, NavigationToolbar2TkAgg
from matplotlib.figure import Figure

import Tkinter as Tk

mean=[]
midpt=[]
mode=[]
count=0
frame_no=[]
min=[]
max=[]
stddev=[]

def destroy(e): sys.exit()

def get_imstat(image):
    iraf.images()

    result_text = iraf.imstat(image, fields="image,npix,mean,stddev,midpt,mode,min,max", format="no", Stdout=1)

    return result_text

for image in sys.argv[1:]:
        print image
        result_list = get_imstat(image)
        print result_list

        for line in result_list:
            values = line.split()
            mean.append(float(values[2]))
            stddev.append(float(values[3]))
            midpt.append(float(values[4]))
            mode.append(float(values[5]))
            min.append(float(values[6]))
            max.append(float(values[7]))
            frame_no.append(int(count))
            count += 1
```

-:--- **plot_imstats.py** Top L1 (Python)----------

emacs@new-r14 (new-r14 上)

File Edit Options Buffers Tools Python Help

```
#frame_no = len(sys.argv[1:])

#print frame_no, midpt

root = Tk.Tk()
root.wm_title("Embedding in TK")

f = Figure(figsize=(5,4), dpi=100)
a = f.add_subplot(111)

#a.plot(frame_no, max, 'g^')
#a.plot(frame_no, max, 'b--')
#a.plot(frame_no, min, 'ro')
#a.plot(frame_no, min, 'y--')

a.plot(frame_no, mean, 'ro')
a.plot(frame_no, mean, 'y--')

#a.plot(frame_no, midpt, 'ro')
#a.plot(frame_no, mode, 'bs')
a.set_xlabel('frames')
a.set_ylabel('Mean values')
a.set_title('Counts stats on 9 frames')

#plt.draw()
#a.show()

# a tk.DrawingArea
canvas = FigureCanvasTkAgg(f, master=root)
canvas.show()
canvas.get_tk_widget().pack(side=Tk.TOP, fill=Tk.BOTH, expand=1)

toolbar = NavigationToolbar2TkAgg( canvas, root )
toolbar.update()
canvas._tkcanvas.pack(side=Tk.TOP, fill=Tk.BOTH, expand=1)

button = Tk.Button(master=root, text='Quit', command=sys.exit)
button.pack(side=Tk.BOTTOM)

Tk.mainloop()
```

-:--- **plot_imstats.py** Bot L68 (Python)----------

実習7（時間があれば）
マルチCPUコアを意識した並列処理をしてみる

- Python2.6より利用可能になったmultiprocessingクラスを用いて、imstatやimarithの並列処理を行ってみる
- 2並列にしたからといって必ず処理効率が2倍になるわけではない

データ: /data/takata/SUPA????????.fits

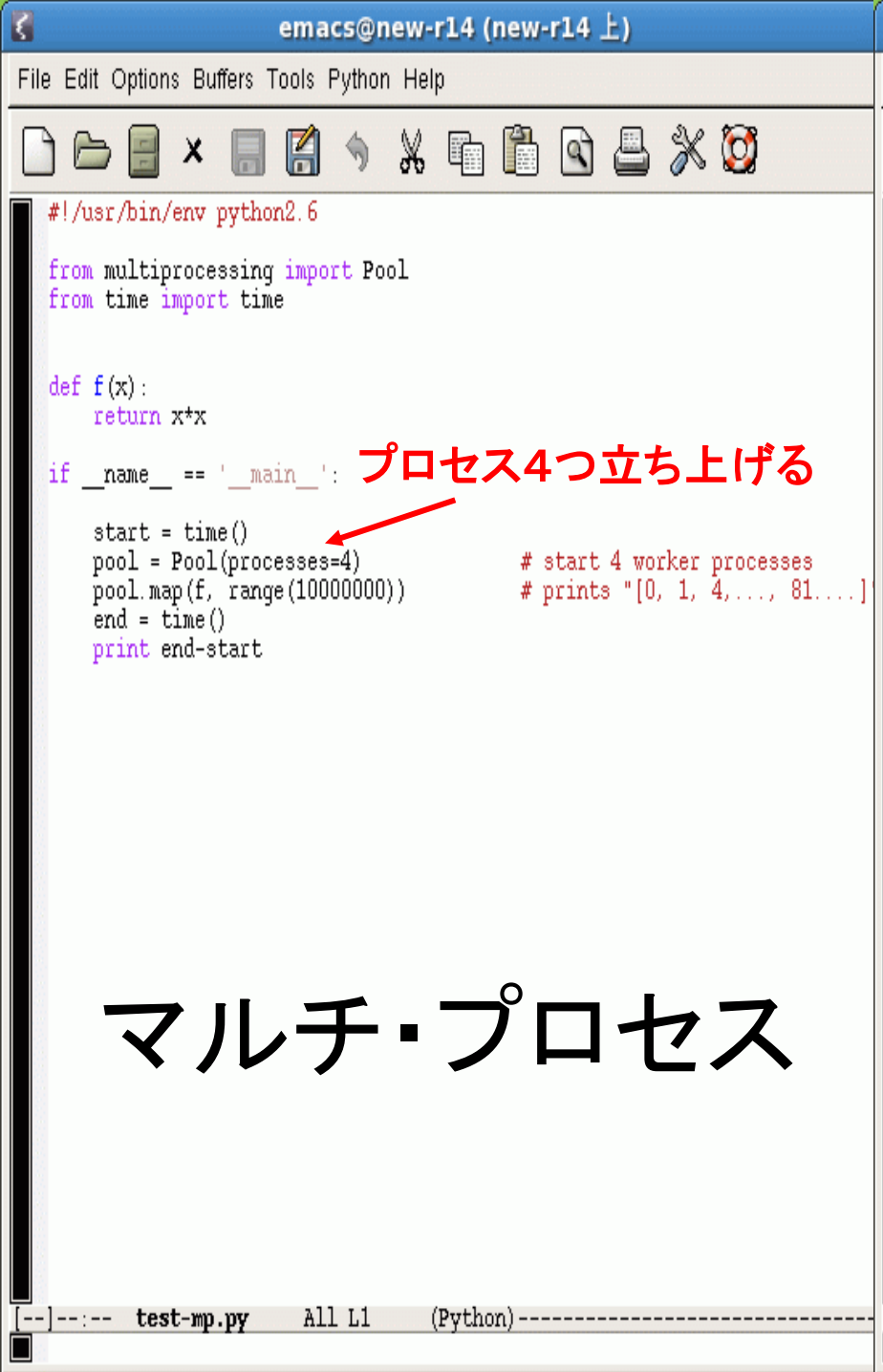
プログラム： /data/takata/Python/test-mp.py

/data/takata/PyRAF/test-mp_pyraf.py

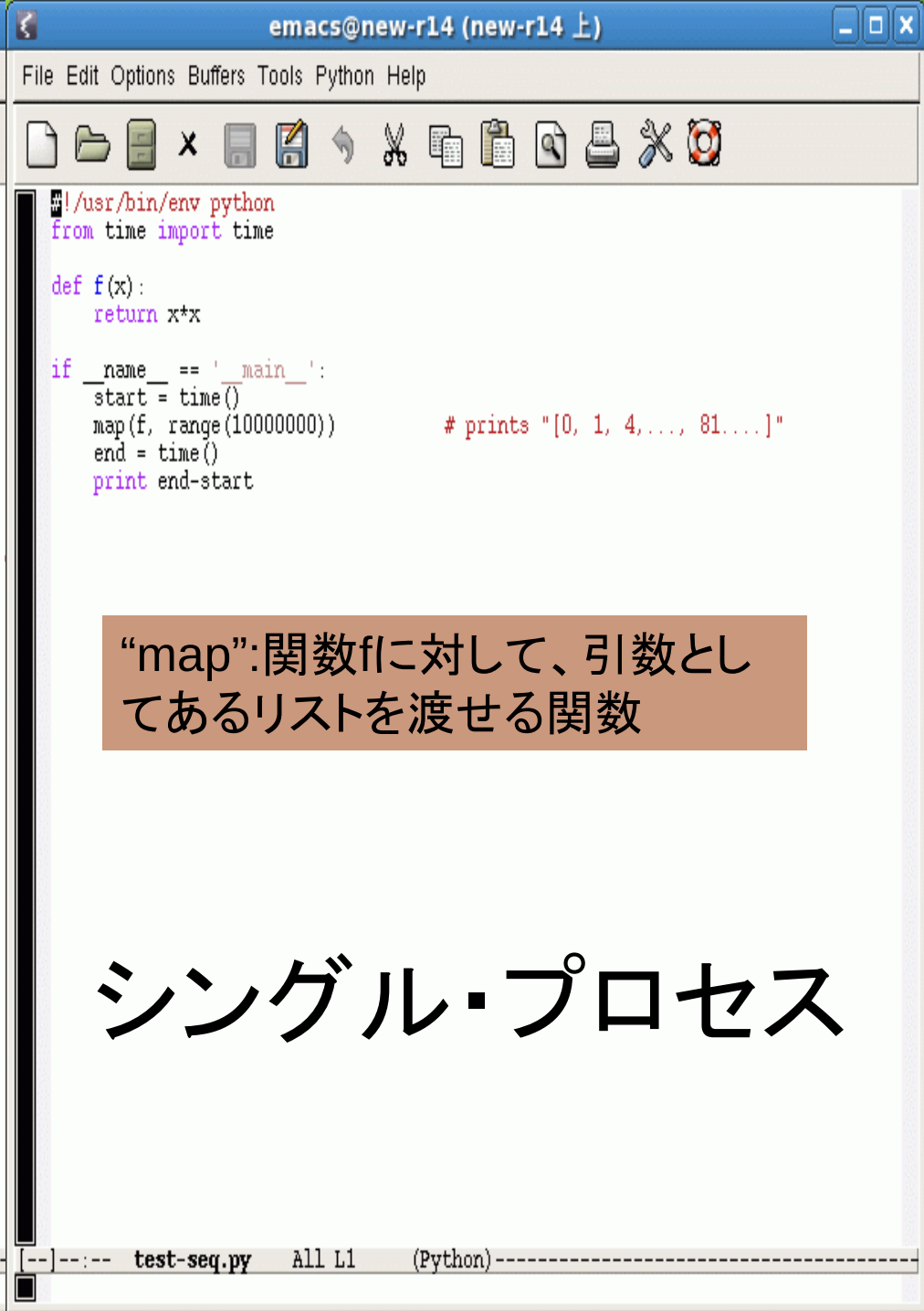
比較用のシーケンシャル処理用のスクリプト

/data/takata/Python/test-seq.py

/data/takata/PyRAF/test-seq_pyraf.py



マルチ・プロセス

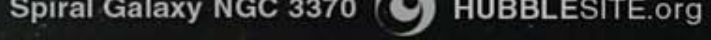


“map”:関数fに対して、引数としてあるリストを渡せる関数

シングル・プロセス

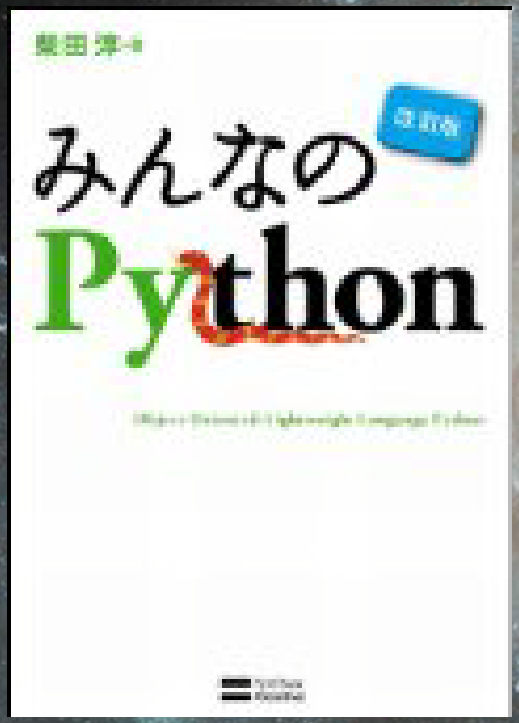
終わりに

- これはあくまでもPyRAFへの入り口に過ぎない
- これ以外にもいくつもの便利な機能と合体して、使いやすいプログラムを比較的簡単にできる(ハズ)、、、（ネットワーク経由型など)
- Pythonの外部ライブラリとうまく共用すれば、画像処理結果の分かり易い記述や、GUIを用いた画像処理＋表示ソフトを組める
- PyRAFへのタスクの登録、エラー処理、イベント処理などは本講習の範疇に含めなかったが、PyRAFのTutorialを読むと結構色々わかります。



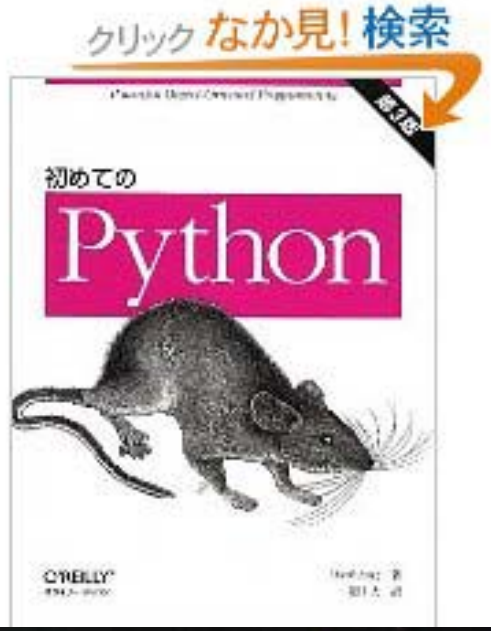
より良い使い方を知った方は、是非とも本講習会で披露して欲しい。

参考資料(Python)



みんなのPython

はじめてのPython3



初めてのPython

Spiral Galaxy NGC 3370 HUBBLESITE.org

PyRAFの参考資料

PyRAFホームページ

http://www.stsci.edu/resources/software_hardware/pyraf

PyRAFドキュメント群

- http://stsdas.stsci.edu/stsci_python_epydoc/
  - PyRAF Tutorial (May 2002)
  - PyRAF Programmer's Guide (Version 1.0, May 2004)
  - Interactive Data Analysis with Python Tutorial (May 2007)

The PyRAF Tutorial

http://stsdas.stsci.edu/pyraf/doc.old/pyraf_tutorial/pyraf_tutorial.html

A Quick Tour of Python

http://stsdas.stsci.edu/pyraf/doc.old/python_quick_tour/

Spiral Galaxy NGC 3370 HUBBLESITE.org