プログラミング1

(第8回) File I/O, Tuples, Jupyter Notebook

1. Chapter 4.6 Files
    1. File I/O (ファイル入出力)
    2. ファイル・ハンドラ
    3. with構文
2. Chapter 5.1 Tuples
    1. 変更できない、順序付きシーケンス集合
3. Jupyter Notebook (教科書にありません)
    1. 文章・コード・実行結果を一つのノートにまとめる
4. まとめ
5. 演習
    1. 演習7: doctest, while文
6. 宿題

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/

関数や仕様はどう決定したら良いか？

授業計画:
第1回～第8回

- 代表的な原則
- KISS原則
  - Keep it simple, stupid!
  - 小さく作り、組み合わせる。
    - 一つの関数は一つの作業をこなす。
  - 各部品(関数)をテストする。
    - 検証・再現性を意識する。
- DRY原則
  - Don’t repeat yourself.
  - 繰り返しを避ける。

授業計画:
第2回～

- 経験を積む
  - 様々な問題にトライする。
  - ペア・プログラミング。
    - 互いに教え合う
    - 知識の共有化
- (マルチステークホルダーの存在を意識する)

目安:
・1関数=数10行程度
・50行ぐらいになったら分割できないか考えてみよう。
・長過ぎるブロックは読みづらく、バグに気づきにくく、再利用しにくい。

4 Functions, Scoping, and Abstraction

- 4.1 Functions and Scoping
- 4.2 Specifications
- 4.3 Recursion
- 4.4 Global Variables
- 4.5 Modules
- 4.6 Files

4.6 Files (ファイル)

- ファイルを読み書きするには、
  - Step1:「ファイルハンドラ(file handle)」を準備する。
  - Step2: ハンドラに対して読み書きする。
  - Step3: 読み書きし終えたらハンドラを閉じる。
- コード例

```
name_handle = open('kinds', 'w')
for i in range(2):
    name = input('Enter name: ')
    name_handle.write(name + '¥n')
name_handle.close()
```

引数でファイル名とモードを指定。
kindsというファイルに'w'(上書きモード)でアクセスするためのハンドラを用意。

¥=バックスラッシュ。
特別な用途を指示。
¥n = 改行

「ハンドラ.write()」ハンドラに対して書き込む。
「ハンドラ.close()」ハンドラを閉じる。

代表的なFile I/O操作 (図4.11)

fn = ファイル名

- open()
  - open(fn, 'w'): write(上書きモード)でファイルを用意（あれば中身を消去、なければ新規作成）し、ハンドラを返す。
  - open(fn, 'r'): read(読み込みモード)でファイルを用意し、ハンドラを返す。
  - open(fn, 'a'): append(追記モード)でファイルを用意し、ハンドラを返す。
- ファイルハンドラへの操作
  - fh.read(): サイズ指定がない場合、EOF (End of File) まで読み込み、1つのstr型オブジェクトとして返す。
  - fh.readline(): 改行もしくはEOFまで読み込み、1つのstr型オブジェクトとして返す。
  - fh.readlines(): 行のリストを読み込んで返す。
  - fh.write(s): str型オブジェクトsを書き込む。
  - fh.writelines(S): シーケンス集合Sを書き込む。
  - fh.close(): ハンドラを閉じる。

参考: Python ドキュメント
open(): http://docs.python.jp/3/library/functions.html#open
16.2. io — ストリームを扱うコアツール: http://docs.python.jp/3/library/io.html

ファイル操作の推奨方法: with構文

```
#ファイルへ書き込む
with open('spam.txt', 'w') as file:
    file.write('Spam and eggs!')
    file.write('hogege')

#ファイルを読み込む
with open('spam.txt', 'r') as file:
    for raw_line in file:
        line = raw_line.rstrip()
        print(line)
```

- 操作したいファイル名とモードを with 文で指定。
- ブロックの処理が終わると、自動でclose()する。
  ＊途中でエラーになったとしても閉じる。

「str型オブジェクト.rstrip()」
行末の改行文字(¥n)を削除。

参考: Dive Into Python 3, 第11章 ファイル,
http://diveintopython3-ja.rdy.jp/files.html

5 Structured types, mutability, and higher-order functions

- 5.1 Tuples
- 5.2 Lists and Mutability
- 5.3 Functions and Objects
- 5.4 Strings, Tuples, and Lists
- 5.5 Dictionaries

5.1 Tuples (タプル型オブジェクト)

- Like strings, **tuples** are ordered sequences of elements. The difference is that the elements of a tuple need not be characters.
- (chap 5.2) Tuples and strings are immutable.
  - 文字列と同様に、タプルは順序のあるシーケンス集合。
  - 文字列と同様に、タプルは変更できないオブジェクト。
  - 文字列と異なり、要素が文字である必要はない。

- コード例

```
>>> t1 = ()
>>> t2 = (1, 'two', 3)
>>> print(t1)
()
>>> print(t2)
(1, 'two', 3)
>>> t2[0]
1
>>> t2[0] = 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

タプルの補足

- 「1個の要素を持つタプル」は特別な書き方が必要。

```
>>> (1)
1
>>> type((1))
<class 'int'>
>>> (1,)
(1,)
>>> type((1,))
<class 'tuple'>
```

- どこで使われる？
  - e.g., 関数の戻り値。
- 他にどういう時に使う？
  - リストよりも高速。
  - 変更を許可したくない場合。
  - （後で出てくる）辞書型オブジェクトのキーとして使える。

参考: Dive Into Python 3, 第2章 ネイティブデータ型, http://diveintopython3-ja.rdy.jp/native-datatypes.html

Jupyter Notebook

- http://jupyter.org
  - The Jupyter Notebook is a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, machine learning and much more.
- コード、実行結果、コメント等を一つのノートにまとめることができる。

Jupyter Notebookを使ってみる1
（授業ページにコピペできるコマンド書いてます）

- インストール（赤字はサンプルを動かすためのおまけ）
  - % sudo pip3 install jupyter
  - % sudo pip3 install matplotlib
  - % curl https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/jupyter_example.ipynb -o jupyter_example.ipynb
- アプリの起動
  - % jupyter notebook
- アプリを起動すると
  - PC上でWebサーバが起動。
  - アプリを起動したディレクトリの一覧がブラウザに表示される。
  - 後はブラウザ上であれこれ操作するだけ。
- アプリを終了する
  - アプリを起動したターミナル上で「Ctrl+C」。

Jupyter Notebookを使ってみる2

- ノートを書いてみる
  - 画面上部の「Untitled」をクリックして、ファイル名を変更。（拡張子.ipynbのファイルで保存される）
  - 右上の「New」を選択。
  - 「Python3」を選択。
- 基本単位は「Cell」
  - CellにPythonコードを書く。
  - 「再生ボタン（右向き三角▶）」を押すか、Shift+Enterを押すと、コードの実行結果がCellの下に出力される。
  - 上部メニュー、左の「+」でCellを追加できる。
  - Cellはメニュー内の上下矢印で移動可能。
  - 上部メニュー中央の「Code」を「Markdown」にすると、Markdown形式で文章を書ける。
  - Markdown形式のセルを実行すると、プレビューになる。（ダブルクリックで再編集可能）

Jupyter Notebookでレポート提出？

- Case 1: そのまま .ipynb ファイルを提出。
  - 注意: 読み手にJupyterが必須。
- Case 2: ファイル形式を変換して提出。
  - Fileメニューから「Print Preview」を選択するか、「Download as」を選択して、HTMLなど好みの形式に変換。

参考:
Jupyterはデータとか数式いじる人には最強のツールかもしれない,
http://fj.hatenablog.jp/entry/2015/11/10/203924
Jupyter Notebook の使い方,
http://www.geocities.jp/penguinitis2002/computer/programming/Python/jupyter_notebook/jupyter_notebook.html

まとめ

- File I/O
  - open()でファイルハンドラを用意
  - ハンドラに対して読み書き操作
  - 終えたらハンドラを閉じる
    - with構文推奨
- Tuples
  - 変更できない、順序のあるシーケンス集合
  - 高速
- Jupyter Notebook
  - お好みで利用ください

演習

演習6をベースに修正しました。

演習7: doctest, while文の利用

補足1

- ペアプログラミングを始める前に
  - 記入漏れ
    - 「実施日」と「報告者」
  - 意思疎通しながら取り組む（独立して作業しない）
    - driver: メイン作業者。
    - observer: 観察者。
  - 前回の復習確認
    - 「何をやったっけ？」
    - 「これはこうやれば良いんだっけ？」

補足2

- ペアプログラミングのやり方
  - 7ステップ
    - 作業を決める
    - 最初の目標を決める
    - パートナーを頼りにし、支えてやる
      - driver: 仕事を終わらせることに専念
      - observer: 横から観察し、疑問・改善・簡潔化など大局的な問題について考える
    - 喋る
      - 「一人で悩む」のは十秒程度に留める
      - 一緒に相談しながら考える練習
    - お互い何をやっているか把握する
      - 頻繁に同期をとる
    - 喜ぶ
    - 交代する

分業ではない（observer=観察しながら気づいたことをコメント、分からないことを質問）

二人で2,3分考えても分からない場合には、手を上げて質問しよう

演習

- 演習7: while文
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week7-ex.html
- ペア・プログラミング
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week2-pair-programming.html
  - driver, observer (navigator)

宿題

- 復習: 適宜(これまでの内容)
- 予習: 教科書読み　＊今回は指定なし
  - 余裕がある人
    - 教科書5章の続き
    - バージョン管理: Mercurial, Git
- 復習・予習(オススメ): paiza
  - プログラミングスキルチェック　＊レベル設定のある課題集
    - https://paiza.jp/challenges/info

参考文献

- 教科書: Introduction to Computation and Programming Using Python, Revised And Expanded Edition
- Python ドキュメント
  - open(): http://docs.python.jp/3/library/functions.html#open
  - 16.2. io — ストリームを扱うコアツール: http://docs.python.jp/3/library/io.html
- Dive Into Python
  - 3第11章 ファイル, http://diveintopython3-ja.rdy.jp/files.html
  - 第2章 ネイティブデータ型, http://diveintopython3-ja.rdy.jp/native-datatypes.html
- Jupyter Notebook, http://jupyter.org
- Jupyterはデータとか数式いじる人には最強のツールかもしれない, http://fj.hatenablog.jp/entry/2015/11/10/203924
- Jupyter Notebook の使い方, http://www.geocities.jp/penguinitis2002/computer/programming/Python/jupyter_notebook/jupyter_notebook.html