Advance Lecture of Medical Information Sciences

Life and Medical Sciences,
Medical Information Department

Tomoyuki Hiroyasu

Hiroyasu Information

Department of Life and Medical Sciences・Medical Information System Laboratory

Professor

Office 2F IN-208N

phone (0774 - 65 -) 6932

tomo@is.doshisha.ac.jp

＜概要＞

本講義では，近赤外脳機能計測法(fNIRS)で取得したデータを対象に，データ処理する手法を学ぶ。講義では，座学だけでなく，Python によるデータ処理を学生自ら行う演習も含む。

＜到達目標＞

近赤外脳機能計測法(fNIRS)で取得したデータを対象に，Pythonによるデータ処理が行えるようになる。

＜成績評価基準/Evaluation Criteria＞

授業の出席	30%	授業に参加し，積極的に発言する。内容を理解する。
宿題	70%	毎回，宿題が課される。課題を行い評価される。
備考		各回に出される宿題が一定以上のレベルで処理されていない場合には，単位取得不可の可能性あり。

クイズ形式で，理解度を確認し，クイズ結果の点数を適宜，返却する。場合によっては，個別に指導を行う。

TIOBE Index for April 2018

Apr 2018	Apr 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.777%	+0.21%
2	2		C	13.589%	+6.62%
3	3		C++	7.218%	+2.66%
4	5	↑	Python	5.803%	+2.35%
5	4	↓	C#	5.265%	+1.69%
6	7	↑	Visual Basic .NET	4.947%	+1.70%
7	6	↓	PHP	4.218%	+0.84%
8	8		JavaScript	3.492%	+0.64%
9	-	⇈	SQL	2.650%	+2.65%
10	11	↑	Ruby	2.018%	-0.29%
11	9	↓	Delphi/Object Pascal	1.961%	-0.86%
12	15	↑	R	1.806%	-0.33%
13	16	↑	Visual Basic	1.798%	-0.26%
14	13	↓	Assembly language	1.655%	-0.51%
15	12	↓	Swift	1.534%	-0.75%
16	10	⇊	Perl	1.527%	-0.89%
17	17		MATLAB	1.457%	-0.59%

- **＜備考＞**
- 生命医科学部医情報学科で身につけるべく基礎単語を知らない場合には受講不可。
  学生自身のノートＰＣを用意すること。
- Python 2.7以上を用意すること。 ＞ Python 3
- 関連のライブラリを自身でインストールできない場合には，受講不可。
  TeX（組版処理システム）の知識と利用ができない場合には，受講不可。
  生命医科学部医情報学科の学部科目である「プログラミングⅠ」および「プログラミングⅡ」の知識は必須。
  同志社大学e-learning system eclassに学生自身のノートＰＣからアクセスおよび利用ができない場合には受講不可。

  本科目は，学部大学院連携科目である。学部生が「医療情報学」として受講する場合には，「プログラミングII」を成績「B」以上 所得しておく必要がある。

Phthon

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.[26]

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.[27]

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software[28] and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

http://en.wikipedia.org/wiki/Python_%28programming_language%29

Anacondaのインストール

- Anacondaは、Pythonで科学技術計算やデータ分析をするのによく使われる125以上の拡張パッケージを、Python本体とひとまとめにしたディストリビューション。https://store.continuum.io/cshop/anaconda/
- Python本体と各種パッケージを自分ではじめから導入するのは若干面倒なのですがAnacondaは各OS向けにインストーラが用意されているので簡単です。ダウンロードしてダブルクリック（Windowsの場合）するだけ。環境変数の設定も自動で行われます。
- 経済データ分析でよく使うPythonパッケージは、Numpy、Scipy、Matplotlib, Pandas、Sympy、Statsmodels、などですが、全てAnacondaに含まれています。また、遺伝的プログラミングのためのDEAPや、Deep Learningの実装に便利なTheanoなど最近流行の分析ツールも入ってます。

Python Version

- Python2.7
- Python 3.6
  - 今年からこちらのバージョンを使用する
  - わからないかもしれない。。。
- The Python core team plans to stop supporting Python 2 in 2020.

Python 開発環境

- Integrated Development Environment（IDEL)
- Eclipse
  - PyDev
- Visual Studio
  - Python Tools for Visual Studio
- PyCharm
- Spyder
- Jupyter Notebook

- Pyscripter
- Komodo IDE

Web informtion

- 日本Pythonユーザ会
  - http://www.python.jp/Zope
- Python 入門
  - http://www.pythonweb.jp/tutorial/
- Pythonの入門から応用までの学習
  - http://www.python-izm.com/
- Learn Python The Hard Way
  - http://learnpythonthehardway.org/index

Web information

- Pydata
- Pystatsmodels
- Numpy-discussion
- Scipy-user

Pythonライブラリ

- NumPy
  - Numerical Pythonの略
  - Pythonで科学計算をする場合に基盤となるパッケージ
- SciPy
  - 科学計算の領域におけるパッケージ

  上記はインストーラーから
  下記は、easy_istall & pip
- Pandas
  - リッチなデータ構造と関数を提供する
- Matplotlib
  - 可視化を行うさいに使うライブラリ
- Ipython
  - Pythonを対話的に実行するためのシェル

IDLE

- IDLE は tkinter GUI ツールキットをつかって作られた Python IDE です。
- 特徴
  - tkinter GUIツールキットを使って、100% ピュア Python でコーディングされています
  - クロスプラットホーム: Windows と Unix で動作します
  - 多段 Undo、 Python 対応の色づけや他にもたくさんの機能 (例えば、自動的な字下げや呼び出し情報の表示)をもつマルチウィンドウ・テキストエディタ
  - Python シェルウィンドウ(別名、対話インタープリタ)
  - デバッガ(完全ではありませんが、ブレークポイントの設定や値の表示、ステップ実行ができます)

PyCharm

- http://qiita.com/pashango2/items/de342abc10722ed7a569

宿題その0

- Anaconda をインストールしてみる

List

- リストオブジェクト[インデックス] 引数に指定したインデックスを持つ要素を取り出します。インデックスは0から始まることに注意して下さい。
- 具体的には次のように記述します。
- list = [2005, 2006, 2007, 2008] year = list[0]

Listの切り出し

- リストオブジェクト[開始インデックス:終了インデックス]
- 具体的には次のように記述します。
- list = ["A", "B", "C", "D", "E"]
- slice1 = list[1:2]　　# ["B"]
- slice2 = list[1:-1]　　# ["B", "C", "D"]
- slice3 = list[1:]　　# ["B", "C", "D", "E"]
- slice4 = list[:2]　　# ["A", "B"]
- slice5 = list[:]　　# ["A", "B", "C", "D", "E"]

len

- len(リストオブジェクト) リストに含まれる要素数を取得します。「len」関数はリストの他に文字列やタプル、辞書などの要素数を取得できます。
- 具体的には次のように記述します。
- list = ["A", "B", "C"] print len(list)

要素の追加

- リストオブジェクト.append(オブジェクト) リストの最後に引数に指定したオブジェクトを追加します。
- 具体的には次のように記述します。
- list = ["A", "B", "C"] list.append("D") print list # ["A", "B", "C", "D"] リストの最後に新しい要素として文字列「D」が追加されます。結果的にリストは["A", "B", "C", "D"]となります。
- 別のリストの要素を追加
- 次にリストの最後に別のリストの要素を追加する方法です。リスト型で用意されている「extend」メソッドを使います。
- リストオブジェクト.extend(リストオブジェクト) リストの最後に引数に指定したリストが持つ要素が全て追加されます。
- 具体的には次のように記述します。
- list = ["A", "B", "C"] list.extend(["D", "E"]) print list # ["A", "B", "C", "D", "E"] リストの最後に別のリストに含まれている要素が追加されます。結果的にリストは["A", "B", "C", "D", "E"]となります。
- なお、「extend」メソッドの代わりに「append」メソッドの引数にリストを指定した場合、リストの要素が追加されるのではなく、リストオブジェクトそのものが1つの要素として追加されますので注意して下さい。
- list = ["A", "B", "C"] list.append(["D", "E"]) print list # ["A", "B", "C", ["D", "E"]] リストの連結
- リストとリストを連結して新しいリストを作成するには「＋」演算子を使います。
- リストオブジェクト ＋ リストオブジェクト 「＋」演算子はリストオブジェクトを変更するものではなく、リストとリストを連結して新しいリストオブジェクトを作成します。
- 具体的には次のように記述します。
- list = ["A", "B", "C"] newlist = list + ["D", "E"] print newlist # ["A", "B", "C", "D", "E"] リスト["A", "B", "C"]に別のリスト["D", "E"]を連結し、新しいリストオブジェクトを作成しています。
- また「＊」演算子はリストオブジェクトを指定の回数繰り返した新しいリストオブジェクトを作成します。
- リストオブジェクト ＊ 回数 具体的には次のように記述します。
- list = ["A", "B", "C"] newlist = list * 3 print newlist # ["A", "B", "C", "A", "B", "C", "A", "B", "C"]

さて　注意

- 予約語や　ライブラリの名前をファイル名にしてはいけない！

- print “hello”
- for
- list
- 関数　＞　ここまで必ずいく

宿題その１

- チュートリアル
- http://docs.python.jp/2/tutorial/

- 関数とクラス
- 配列をlist形式で取り扱えるように

- 乱数を取り扱う

乱数

- import random

- 「random.random()」… 0.0～1.0までのfloat値を取得します。
- 「random.uniform(x,y)」… x～yまでのfloat値を取得します。
- 「random.randint(x,y)」… x～yまでのint値を取得します。
- 「random.choice(param)」… param内から一つの要素を取得します。
- 「random.shuffle(array)」… array内の要素をシャッフルします。

TeXとは

TeXとは

スタンフォード大学のDonald E. Knuth教授(当時)が開発した組版ソフトウェア。

正確にはTEXと表記する。

商用印刷にも使える高度な組版をコンピュータ上で実現する。フリーソフトウェアとして無償で公開されており、WindowsやMacintosh、UNIXなど様々なプラットフォームに移植されているが、標準化が行き届いているので、どの機種でもまったく同じ出力が得られる。

特に数式に強いため、大学などの学術機関を中心に普及している。

HTMLなどのように、文書の中に組版情報を埋め込み、それを処理系で印刷可能な形式に変換するというスタイルを取っている。

http://e-words.jp/w/TeX.html

IT用語辞典 e-Words

sgmlとは

エスジーエムエル

Standard Generalized Mark-up Language

マークアップ言語 の一つ

文書の論理構造、意味構造を記述する言語。タイトル、引用部分、著者など文書の中で特別な意味をもつ部分にマーク付けをすることができる。

http://e-words.jp/w/SGML.html

IT用語辞典 e-Words

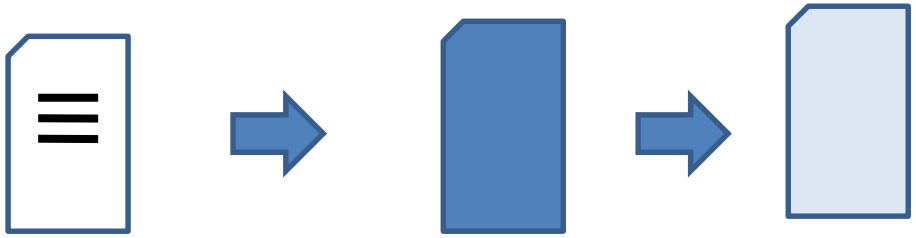
SGML

- HTML
- XML
- その他

```
<本>
<表題>SGML</表題>
<作者>三浦　淳</作者>
<電子メイル>miuraj@isc.meiji.ac.jp</電子メイル>
<はじめに>SGMLとはいったいなんぞや？・・・</はじめに>
<章>
    <章見出し>SGML概説</章見出し>
    <段落>SGMLとはStandard Generalized Markup
    Language の略で、、、、・・・
</章>

<章>
・・・
</本>
```

TeXの流れ



TeX形式
（テキスト）

div形式
（バイナリ）

印刷

ps
pdf
（バイナリ）

TeXの例

論文{article}，書籍{book}，報告書{report}といったものがありる．
日本語化に対応したものは最初にjが付く．

```
¥documentclass[12pt]{artic
  le} ¥begin{document}
  Hello ¥TeX!
  ¥end{document}
```

TeXの例２

```
¥documentclass{jarticle}        % LaTeX文書の先頭
                                % プリアンブルの記述（省略可能）
¥title{論文の題名}              % 文書の題名
¥author{著者名}                 % 文書作成者
¥date{日付}                     % 日付
¥begin{document}
¥maketitle                      % 論文題名，著者，日付を出力
    ....
    ....        本文を記述する
    ....
¥end{document}                  % LaTeX文書の最後
```

platexのインストール（windows)

- **TeXインストーラ 3**
- http://www.math.sci.hokudai.ac.jp/~abenori/soft/abtexinst.html

- platex ttt.tex
- dvipdfmx ttt.dvi

¥documentclass{jarticle} % LaTeX文書の先頭 % プリアンブルの記述（省略可能）

¥title{Texの練習} % 文書の題名
¥author{廣安　知之} % 文書作成者
¥date{4月10日} % 日付

¥begin{document}
¥maketitle % 論文題名，著者，日付を出力

¥section{はじめに} はじめに
¥section{2}

¥end{document} % LaTeX文書の最後

宿題その２

- TeXで　python の調査レポートを作成

- タイトル：　python 3.5について
  - 概要
  - 特徴
  - インストール方法
  - jarticle