【一般教育演習】プログラミングで問題を解く：集計から人工知能まで・第十三回 2018 年 7 月 12 日

以下の学生は mynewnote.ipynb のメールでの提出が確認できていません。
再送付 or 提出してください：
01185102
02180225
02180974

※この課題は単に「第 10 回」の「補足資料」で作成するものです。Web サイトからこの補足資料をチェックしてその内容にしたがって作成しメールに作成した mynewnote.ipynb を添付して瀧川 takigawa@ist.hokudai.ac.jp まで送付してください。

1. Web サイト
   スライド、資料、事務連絡などはすべてココでシェアする予定↓
   http://art.ist.hokudai.ac.jp/~takigawa/prog/
   「毎回の作業」「Jupyter 使い方」を忘れた人はここで確認

2. 瀧川のメールアドレス 提出や質問
   takigawa@ist.hokudai.ac.jp

3. 教室
   5/10（木）～8/2（木）E209 教室

4. 今日の内容
   - フォローアップ Q & A
   - AI について
   - 機械学習について
   - Numpy で配列データを扱おう

先週の質問への回答：

Q. 16 回目がまとめ Notebook 作成日ということは、提出期限は 16 回目の授業終了時までということですか。(最終授業でも提出用の Notebook の作成の時間を確保していただけるとのことでしたが提出日もその人いうことですか？それともまだ未定ですか？)
A. その日(8/2)に作業すると終わらない人もいると思うので(8/6 とか)翌週にする予定です。ホームページで提出確認ができるようにする予定です。多分お盆くらいに事務上の成績入力期限があるのでそれよりは少し前という感じですね。

Q. まとめノートには第一回からの授業で習ったことをまとめたものを書くのですか？
A. そのような想定ですが、まあお任せします。基本的には「この授業でプログラミングについて得られた知識」ということを評価するためのものです。単に規則のまとめではなくて「考察」などでも良いですし、別の自習をやってみたメモとかでも良いですし、後で自分が見返すと「この授業で自分が得たこと」が思い出せる、という点が重要かなと思います。

Q. mynewnote は practice1 – practice の最後までを自分なりにまとめた物を提出するのですか？それに加えて、自分が昔と今でプログラムについてどのように感じているかも書くのですか？
(mynewnote と最後に提出の notebook の違いは何ですか？)
A. 今提出してと行っている mynewnote は「第 10 回」の「補足資料」で作成するものです。最後に提出する notebook については、どっちでも良いです。上の質問への回答を参考にしてください。

Q. print で半角英語と日本語を交互に入力することが多いと思うのですが毎回半角全角入れ替えない方法はありますか？
A. ありません。

Q. Python などのプログラミング言語やそのライブラリを使った人はそれらを無料で配布していますが、これは作成物による収入がゼロということですか。純粋に業界の発展のために作成し配布しているのでしょうか。
A. オープンソースであればそれ自体で収入があるわけではないと思います。ただ、純粋に業界の発展のためにというよりかは、こんな良いソフトウェアをわたし作ってますよ、という名誉のためということが大きいかもしれません。本当に幅広く使われるようなソフトウェアを開発していればそれだけでその界隈では有名人になりますし、講演依頼などで収入が出たり、インパクトによっては Google をはじめとして就職の声がかかるとか、色々な波及効果がありえます。

Q. 前までは使えていたプログラムが突然使えなくなることはあるのでしょうか。またどうしてそのようなことが起こるのでしょうか。
A. 何もしていなければそのようなことはありえません。何かが変更されたとか機器が故障したとか、なんらかの原因があるということになります。Python のバージョンが勝手に上がったとか、OS のバージョンが勝手に上がったとか、勝手に影響のあるアップデートがインストールされたとか、知らないうちに何かファイルを上書きしてしまったとか消してはいけないファイルを消したり移動したりしてしまったとか、いろんな原因があるのでなんとも言えません。

Q. practice07 の問題 4 の数当てゲーム High or Low の部分で、seikai = random.randint(1, 100)の「randint」とはどういう役割を果たしているのですか？「randint」は必要ですか？
A. 必要です。Randint は、random integers でランダムな整数(1 から 100 までの)ということです。普通乱数というと、0 から 1 までの実数乱数(一様乱数)をさすことが多いです。確率・統計の話なので、詳しくは説明しませんが、全ての乱数はこの 0 から 1 の一様蘭数があれば作ることができます。

Q. プログラムの書き方をみて「このプログラムは〇〇が書いた」とある程度検討をすることは可能なのでしょうか？
A. 昔はありましたが、現代では誰でも読めるように属人性のないクリーンなコードを書くほうが美しいということになってきたので、あまりありません。むしろクセが分かるようではあまりよくないコードということになります。

Q. プログラミングはコマンドプロンプトみたいな黒い背景のところに複雑なコードを書いているイメージがありましたが、Jupyter Notebook を使うことでその煩雑さを軽減しているのでしょうか。
A. 工学部系の実習では黒い背景のところにコードは書かされますが、社会に出ると逆に「統合環境」という Jupyter のように色々な補助機能がある環境でコードを管理することが多いと思います。煩雑さを軽減しているというよりは、機能拡張という感じでしょうか？逆に画面のない遠隔のコンピュータにネットワーク接続してプログラムを仕込むとかの場合やスーパーコンピュータで何週間もかかる計算をするとかの場合など、Jupyter のような環境での開発が向いていない場合ももちろん色々とあります。

Q. Code セルでは「#」を使ってメモを残すことができますが、長いプログラムを書く際にプログラマもこのようなメモを貼っていたりするのでしょうか。
A. これらは「コメント」と呼ばれますが、コメントのないソースコードというのは実用的にはあまりありません。書いている本人でも、間が空くと、すぐ細かいことは分からなくなってしまいがちです。なので、何を意図したコードなのかというコメントはむしろ推奨されています。Jupyter の場合は Markdown セルに文章がかけるのでそちらで代用できますが、python コード自体にコメントを書くなら#で書いておくことが一般的です。

Q. 100 回 Hello で、
```
for i in range(100):
    Print('Hello…')
```
と入力するとたくさん Hello がプリントされたのですが、一行目の i はなんでも良いのですか。
なんでも良いです。なんか使わないのに定義されるようで気持ち悪ければ、
```
for _ in range(100):
    Print('Hello…')
```
とかでも良いです。

Q. jupyter notebook で kernel から選べる各コマンドの意味の違いは何ですか？というか、なぜ「kernel」という名前がついているのかも疑問です。
A. jupyter notebook の使い方の時や過去の Q and A でも少し説明しましたが下記の感じです。
Interrupt → 実行を停止します。今まで評価した部分はそのまま残ります。
Restart → Kernel を再起動します。
Restart & Clear Output → Kernel を再起動して、今実行結果が出力されているセルを全て削除します。
Restart & Run All → Kernel を再起動して、全てのセルを評価実行します。
Reconnect → 何らかの理由で Kernel との接続が切れた場合に再接続しますがあまり使いません。
Shutdown → Kernel を停止します。今まで評価した部分もメモリから消えます。
変なコマンドを誤って実行してしまって入力できなくなった時には Interrupt で停止、それでも受け付けなければ、Shutdown するか Restart します。Clear Output や Run All は「Cell」メニューにある機能です。

「Kernel」というのはコンピュータ科学の専門用語で、英和辞典で出てくる英語自体の意味の通り「(問題などの)核心, 要点, 肝心な部分(core)」という感じです。Jupyter においては、裏で動いている「python インタプリタと結果をやりとりしている部分」のことです。Jupyter は Web ブラウザ自体の昨日で何か計算しているわけではなく、Web ブラウザに入力した文字を裏で「別のプログラム」が Python インタプリタに渡して都度実行してその結果をまた Web ブラウザに表示しています。この Python 実行の中核を担う「別のプログラム」を「Kernel」と呼びます。

Q. Scratch は 8 歳から 16 歳程度を対象としたプログラミング学習ソフトだと HP に書かれていました。Scratch と今我々が扱っている Python (Jupyter)とでは、やはりプログラミングを学ぶことができる範囲に違いがあるのでしょうか。
A. あります。端的に言えば、Python で Scratch は作れますが、逆はできません。また、Python や Jupyter は「プログラミングを学ぶソフト」ではなくて、プログラミング言語や環境そのものです。

Q. 情エレではプログラミング以外に何かやっていますか？また、４年間でどのレベルのプログラミングを習得できますか？
A. カリキュラムやシラバスは全て公開されていますし、学科パンフレットも工学部や情報科学研究科の建物 1F に行けば置いてあって持って行って OK なので、詳しくはそれを見てください。数学も物理も電気関係もやっています。４年間で授業自体で学べるプログラミングはまあ基礎的素養くらいですが、卒業研究などでプログラミングが必要な場合が多いので自動的にそこそこのスキルはつくと思います。

情報エレクトロニクス学科での学び「４年間の学び」
https://infoele.eng.hokudai.ac.jp/gakusei/

Q. Python でスクラッチのような物体を動かすようなプログラミングはできるのでしょうか？
A. 授業ではやりませんが、もちろんできます。興味があれば本や各種 Web サイトをみて遊んで見てください。Jupyter の中でアニメーションやインタラクティブな表示をすることもできます。

Q. ナベアツ問題に取り組んで見れば FizzBuzz についてもできるように(考え方がわかるように)なるでしょうか。
A. なります。ナベアツの方がむしろちょっと面倒です。

Q. あり得ない文字で最初に定義して初期化するという作業がよく出てきましたが、同一 Notebook 上で同じ文字や単語を使っていなければ初期化の必要はありませんか。
A. 初期化の必要があります。初期化しないで実行して見てください。そうすればなぜ初期化が必要かわかると思います。
```
sum = 0
sum = sum + 1
sum = sum + 2
```
の実行結果は sum=3 ですが、もし初期化 sum = 0 をしなければ、最初の sum = sum + 1 はエラーになります。右辺の sum + 1 が不定なので。

Q. イマイチ、付属機能というのがわかりません。Import で召喚しているのはわかるが…。Anaconda を開いた時に jupyter 以外の奴があるが、あれを呼び出しているのか？
A.「モジュール」についての回を復習したり、そのキーワードで調べてみてください。他人が書いたか、自分で書いたかの違いはありますが、モジュールを読み込んでいるだけです。

Q. お二人が今まで作ったプログラミングの中で一番やりがいを感じたのはどんなプログラミングですか？
A. 日常的な道具になるとなかなか難しい質問です。お金をもらって作る規模の大きいプログラム、視覚的に見栄えのする CG や UI のプログラミング、研究を進めるときのプログラミングなんかは基本的にやりがいは感じるんでないでしょうか。