クレジット:
UTokyo Online Education Education コンピュータシステム概論 2018 小林克志

東京大学
THE UNIVERSITY OF TOKYO

CC BY NC ND

jupyter-notebook で Python プログラムをあつかう(再掲)

jupyter-notebook で Python プログラムをあつかうには大きく２種類の方法がある:

1. jupyter-notebook で Python プログラムを開く
2. jupyter-notebook 形式を Python プログラム(.py)に変換する

1. jupyter-notebook で Python プログラムを開く

jupyter-notebook で直接 Python プログラムファイルを開くには。 （jupyter-notebook 起動時に表示される）ファイルマネージャ画面で、 `New -> Text file` を選択すると、エディタ画面に遷移する。 エディタで .py 拡張子をもつファイル名に変更すれば良い。

2. jupyter-notebook 形式を Python プログラム(.py)に変換する

講義で利用している jupyter-notebook を .py としてセーブするには、 `File -> Download as -> Python(.py)` を選択すればよい。 コードセルだけがプログラム行として有効になりその他の行は # でコメントアウトされた Python プログラムファイルがダウンロードファイルに生成される。 環境によっては、.py ではなく .htmlファイルとして保存されるかもしれないが、ファイル名を変更すれば Python プログラムファイルとして利用できる。 jupyter-notebook を Python ファイルとして保存した場合、全てのコードセルの内容を一度に実行するプログラムとして保存されます。jupyter-notebook のようにセル単位の実行とはならないことに注意する必要があります。

講義での利用方法（＊＊重要＊＊）

講義では、とくに指定のない限り（動作する） Python プログラム形式 (.py)として提出すること。 実際には、セルで動作を確認したプログラムスクリプトをクリップボードにコピー、Python プログラムエディタにペーストする という方法が現実的と思われる。
もちろん自身が普段使い慣れているエディタを利用してもかまわない。

提出の前に（＊＊重要＊＊）

1. プログラム (.py)が動作するか、コマンドシェルから確認すること。
2. ファイル名を間違えないこと。教員は課題評価の際にファイルを漁ったりしない。

練習 1. Hello World

GitHub による Python プログラムの課題提出に慣れる目的でおこなう。自信のある受講生はこの課題をスキップしてよい。 自身のない学生が多ければ教員がデモを実施する。

`print()` 関数で文字列 Hello U-Tokyo を印字する、Python プログラム、`hello_u_tokyo.py` を作成せよ。 提出は指示にしたがって GitHub Classroom でおこなうこと。

In [ ]:
```
print("Hello U-Tokyo")
```

課題 1. 2 次方程式の解

以下の形式の２次方程式の全ての解を返す関数、`solve2(a, b, c)` をプログラムせよ。 a, b, c はいずれも実数とする。

プログラムファイルは `solve2.py` として GitHub にアップロードせよ：

$$ax^2 + bx + c = 0$$

1. 複素数の解も求めること。
2. 解なしの場合は `None` を返すこと。

Python では正数の平方根を求める関数として一般数学関数ライブラリに `math.sqrt()`、が用意されている。 利用方法は以下のとおり:

```
from math import sqrt
print(sqrt(3))
```

以下のセルのサンプルを修正してもよい。サンプルでは常に、 $x = 0$ (重解)を返す:

In [3]:
```
def solve2(a,b,c):
    return 0,0

solve2(1,0,0)
```

Out[3]: (0, 0)

課題 2. FizzBuzz

自然数を 1 から昇順に答える FizzBuzz という遊びがある（日本でも流行した）。
ただし、数が 3 で割り切れる場合 `Fizz`, 5 で割り切れる場合 `Buzz`、両方で割り切れる場合 `FizzBuzz` と答える。
引数として、自然数が与えられ、このルールで答えるべき文字列を返す関数 `fizzbuzz(n)` をプログラムせよ。

ちなみに、FizzBuzz はプログラマの採用面接に際し応募者の能力を簡単に判定するスクリーニング課題として広く知られている。
もちろん面接ではさらに高度な課題が課される。

プログラムファイルは `fizzbuzz.py` として GitHub にアップロードせよ。

以下のセルのサンプルを修正してもよい。サンプルでは常に数を返す:

In [ ]:
```
def fizzbuzz(n):
    return str(n)

for i in range(1,100):
    print(fizzbuzz(i))
```


課題 3. フィボナッチ数

フィボナッチ数列は以下のように定義される:

$$F_0 = 0$$

$$F_1 = 0$$

$$F_{n+2} = F_n + Fn + 1(n \geq 0)$$

具体的には以下の数列となる:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...$$

引数として与えられる項数, n, のフィボナッチ数列をリストとして返す関数 `fib(n)` をプログラムせよ。

プログラムファイルは `fib.py` として GitHub にアップロードせよ。

以下のセルのサンプルを修正してもよい。サンプルは 1 から n までの n 個の自然数を返す:

In [ ]:
```
def fib(n):
    return list(range(1, n + 1))

fib(10)
```

課題 4. フィボナッチ数の可視化(1)

前の課題で求めたフィボナッチ数列は、隣り合う数の比が黄金比 $(1+\sqrt{5})/2$ に収束することがよく知られている。
これを可視化によって説明する jupyter-notebook を作成せよ（証明は不要）。
ファイル名は `fib_plot.ipynb` とすること。
notebook には自分以外によって手続き（画像生成など）が再現できるような説明も加えること。

以下は説明用の図の一例：

1. x 軸に項数、y 軸にフィボナッチ数をプロット、画像ファイル `fib_plot.png` として保存する。
2. 補助線として、y 軸に黄金比を示す直線を加える。

作成した Python プログラムを含めてすべてのファイルを GitHub にアップロードせよ。

課題 5. フィボナッチ数の可視化(2)

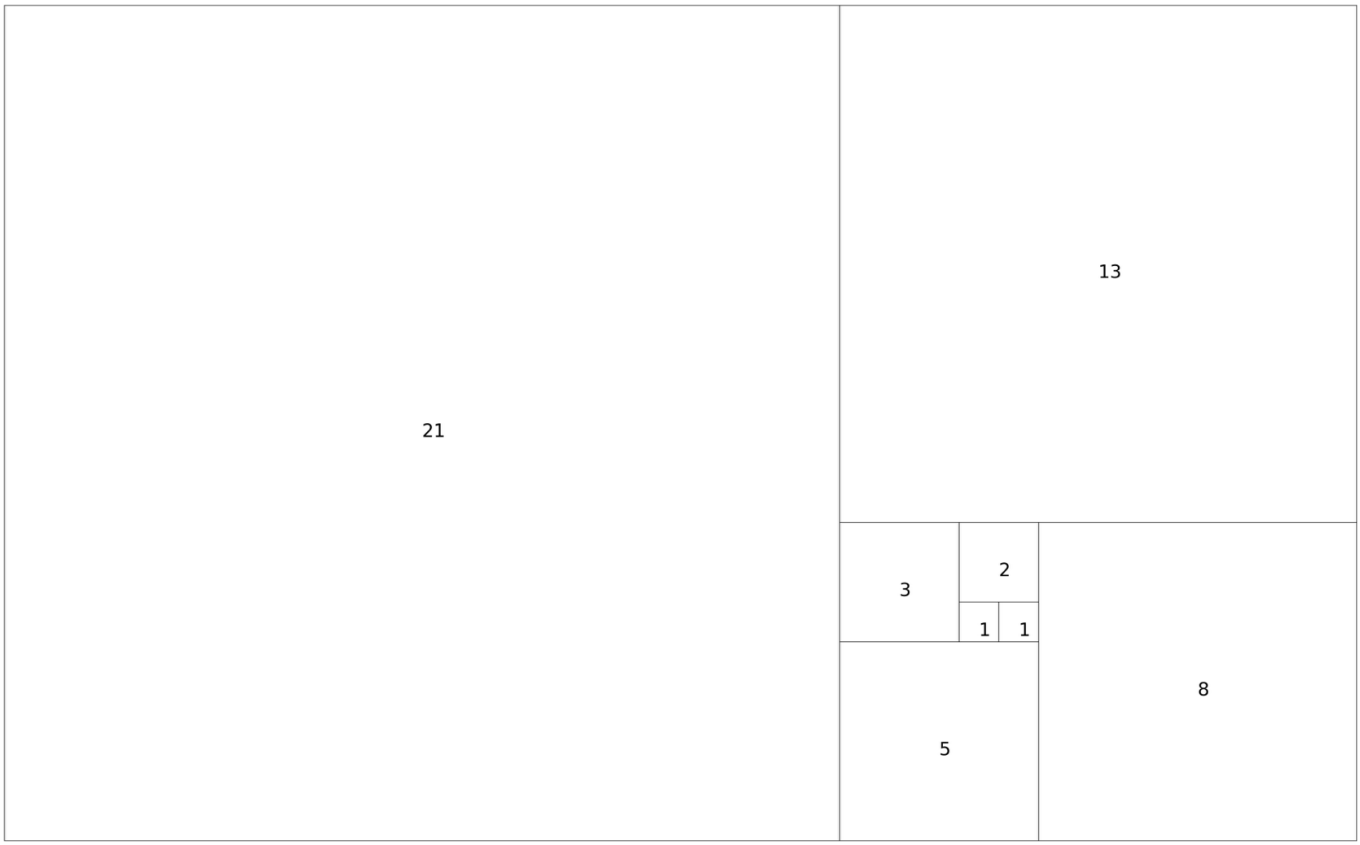
以下の画像は、フィボナッチ数列を用いて黄金比の長方形をつくる可視化例としてよく知られている。
F_{n+2} のフィボナッチ数に対応する正方形の辺が F_{n-1}, F_n の辺の合計となっている。
このような図を描画するプログラム `fib_square.py`を作成せよ。

F_n に対応する一片の長さ F_n の正方形の左下座標 (x_n, y_n) は直前の座標 (x_{n-1}, y_{n-1}) を利用して以下のように場合分け できる。ここで、$m = x \bmod 4$：

$$
\begin{array}{ll}
(x_{n-1} - F_n, y_{n-1} - F_{n-1}) & (m = 0) \\
(x_{n-1}, y_{n-1} - Fn) & (m = 1) \\
(x_{n-1} + F_{n-1}, y_{n-1}) & (m = 2) \\
(x_{n-1} - F_{n-2}, y_{n-1} + F_{n-1}) & (m = 3)
\end{array}
$$

以下に任意の場所・大きさの長方形(Rectangle)および文字列をを描画するサンプルを示した。

個々の関数・メソッドの意味・利用方法は Matplotlib (https://matplotlib.org)を参考にすること。



In [ ]:
```
import matplotlib.pyplot as plt
import matplotlib.patches as patches

fig, ax = plt.subplots(1)

x, y = 3, 3
width = 4
height = 2
r = patches.Rectangle((x, y), width, height,
    fill=False, transform=ax.transAxes, clip_on=False, lw=3)
ax.add_patch(r)

ax.text(x + width / 2, y + height / 2, "sample", horizontalalignm
ent="left",
        verticalalignment='top', size = 24, transform=ax.transAxe
s)

ax.set_axis_off()
ax.set_aspect(1)

plt.savefig("sample.png", bbox_inches="tight")
plt.show()
```


課題 6. フィボナッチ数の可視化(3)

以下の画像は、前の黄金比の長方形に内接する螺旋を加えた可視化例である。 この図を出力するプログラム　`fib_spiral.py` を作成せよ。 F_n に対応する円弧の半径は F_n、中心点は、**正方形の左下座標**を利用して 以下のように場合分けできる。ここで、$m = x \, mod \, 4$：

$$
\begin{array}{ll}
(x_{n-1},\ y_{n-1} - F_{n-2}) & (m = 0) \\
(x_{n-1} + F_n,\ y_{n-1}) & (m = 1) \\
(x_{n-1} + F_{n-1},\ y_{n-1} + F_n & (m = 2) \\
(x_{n-1} - Fn - 2,\ y_{n-1} + F_{n-1}) & (m = 3)
\end{array}
$$

円弧の描画には、`matplotlib.patches.Arc()` が利用できる。

個々の関数・メソッドの意味・利用方法は Matplotlib (https://matplotlib.org)を参考にすること。