経済シミュレーション I

リリース ***1.0.0***

Kazuki Kumashiro

2020 年 **09** 月 **10** 日

Contents:

第 1 章

はじめに

1.1 この資料について

この資料は岡山商科大学経済学部で開講される経済シミュレーション I のために作成した．内容は随時更新予定である．授業が始まる前に本章と第 2 章を通読し，各自自分の PC で Python が使える環境を準備しておくこと．

1.2 Python とは

コンピュータ上で計算を行うにはプログラミング言語を用いる．プログラミング言語には数えきれないほど種類があり，それぞれ特徴を持っている．Python もプログラミング言語のひとつである．Python は比較的初学者でも習得しやすい言語であるにもかかわらず，使いこなせばかなり高度なことができるという特徴を持つ．そのため世界中の様々な分野で広く使われている．例えばマーケティングのデータ分析や（今流行りの）人工知能などでもよく使われるし，身近な所では YouTube や Instagram なども Python を使って作られている．

この授業では Python を使って経済学のモデルを可視化する方法を学ぶ．つまり数式で表されていたものにデータを与えることで，何が起こるのかをその目で確かめてみようということである．このような手法を数値解析などと呼ぶ．経済学では MATLAB や Mathematica というソフトウェアが使われることが多かったが，近年はその扱いやすさと汎用性の高さ，そして **無料** であるという利点から，Python を使う人が増えてきたようである．

1.3 Python 習得のために

Python に限らずプログラミング言語を習得する最短の方法は，とにかく触って慣れる事である．日本語や英語などの言語を，教科書を読むだけで使いこなせる人はいないだろう．それと同じで，言語は使わないと身につかないのである．よって，毎週 90 分の授業時間にだけ Python を触っていれば Python をマスター出来て分析がバリバリできるようになるということはあり得ない．自宅の PC にも Python をインストールして，暇さえあればコードを触るようにしよう．

コードを触れと言われても何をすれば良いかわからない人は，授業で解説したコードやインターネット上で公開されているコードを写経するのが良い．自分の手でコードを書き写し，それぞれの命令でどんなことをやろうとしているのか考えるのである．さらに，数値の設定を変えたり，機能を付け加る方法を考えたりするのも良い練習になる．コードの改変の仕方が間違っていればもちろんコードが動かなくなることはあるが，その場合はエラーメッセージを熟読しよう．コードの中のどの部分が間違っているのか特定するヒントになる（経験上，学生の間違いで最も多いのは単なるタイプミスである）．コードが動かなくても PC 自体が壊れることはまず無いので，失敗を恐

れずにとにかく触ってみよう．

もし自宅に自由に使える PC が無いと言う人は，残念ながらこの授業の履修はオススメできない．もちろん履修を禁止するわけではないが，履修しても十分には理解できないし，単位を習得するには練習時間が少なすぎる．最近は PC の高性能・低価格化が進んでいるので 5 万円前後である程度まともな PC が手に入る．今後も他の授業のレポートや卒論作成，就活や就職後も使えるものなのでこの機会に購入してはどうだろうか．

もう一つ，Python に限らず他のプログラミング言語や PC の操作を身に着けるには，ウェブ検索を最大限に活用することが大事である．Python のユーザーは世界中に星の数ほどいる上，解説を無料で公開している人も多い．実行したい操作や関数名，エラーメッセージなどを Google 検索にかけると大体のことは解決する．もちろん必要とする情報が一番上の検索結果に来るとは限らないので，ある程度時間をかけて情報を探す必要はある．加えて，英語のウェブサイト上で情報が提供されていることも多い．英語の勉強にもなると考え，怖気づかずに読んでみよう．

1.4 Python の勉強・練習に便利な web サイト

Python はユーザーが多いことから，学習用の web サイトも多く存在する．いくつか紹介するので上手く活用して Python に慣れよう．

この授業の主要部分は Thomas J. Sargent（New York 大学）と John Stachurski（オーストラリア国立大学）を中心とする非営利組織 QuantEcon が作成した講義ノートをベースにしている．以下のページの Lectures 内にある Python Programming for Economics and Finance と Quantitative Economics with Python は非常に参考になるだろう．

- https://quantecon.org/

以下は東京大学の数理・情報教育研究センターが作成している Python のマニュアルである．基礎的な内容から中級者向けの使い方まで解説されている．データ分析や機械学習のライブラリまで網羅されている．

- https://utokyo-ipp.github.io/index.html

以下の三つはゲーム感覚で Python を学べる．与えられたパズルや課題を解決するコードを書いて送信すると，自動で採点が行われる．checkio では他のユーザーの解答例も見ることができ，非常に参考になる．paiza は色々なキャラクターとプログラミングが学べるコースがあるので，特にゲームやアニメが好きな人にはお勧めである．

- https://checkio.org/
- https://www.codingame.com/start
- https://paiza.jp/works

第 2 章

Python 環境の準備

本章では Python が使える環境を整える方法について解説する．自分の PC を開き，実際に操作しながら読もう．

2.1 Anaconda による Python の導入

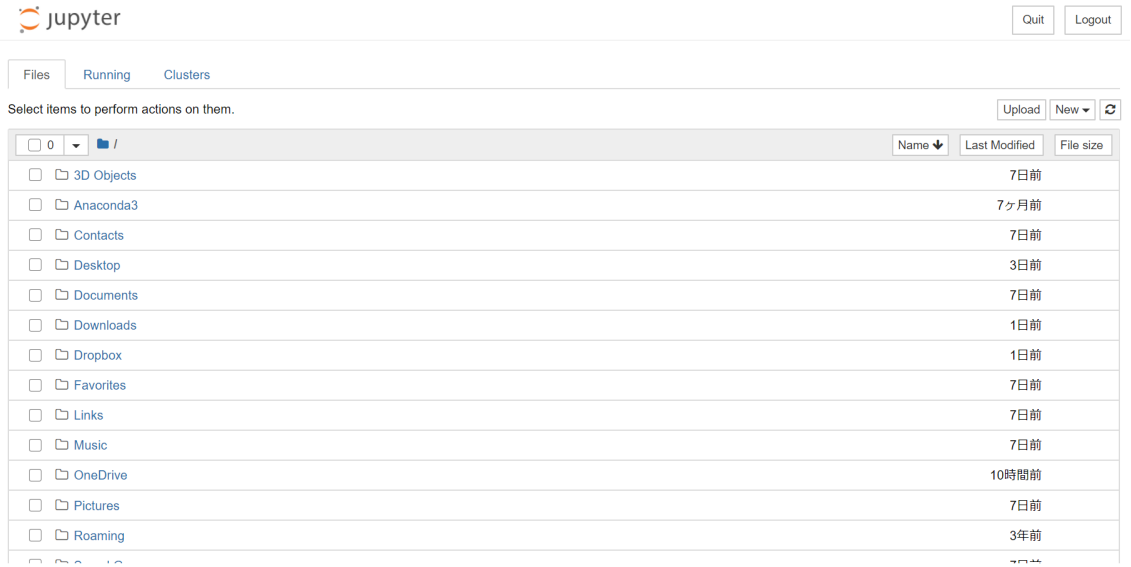
Python をインストールする方法は色々あるが，Anaconda と呼ばれるディストリビューション（Python の利用に必要な道具をまとめたもの）を利用するのが便利である．これをインストールしておけば，学術的な分野で使われる道具はだいたいまとめてインストールされる．

1. https://www.anaconda.com にアクセス
2. ページ上部の Products から Individual Edition を選択
3. ページ下部の Anaconda Installers から，自分の PC に合ったインストーラをダウンロード
   - windows PC で 64 ビットか 32 ビットのどちらかわからない人は，画面左下の windows ロゴを右クリック → システムをクリック → 開いたウインドウの右側に表示される「システムの種類」を確認しよう．
4. ダウンロードされたインストーラを開き，指示に従ってインストール．
   - 基本的には *I Agree* や *Next* ， *Install* を押していけばよい．
   - インストール先は Destination Folder で変更できる．
5. Completing Anaconda 3 のような画面が出ればインストールは終了． *Finish* を押して画面を閉じる．

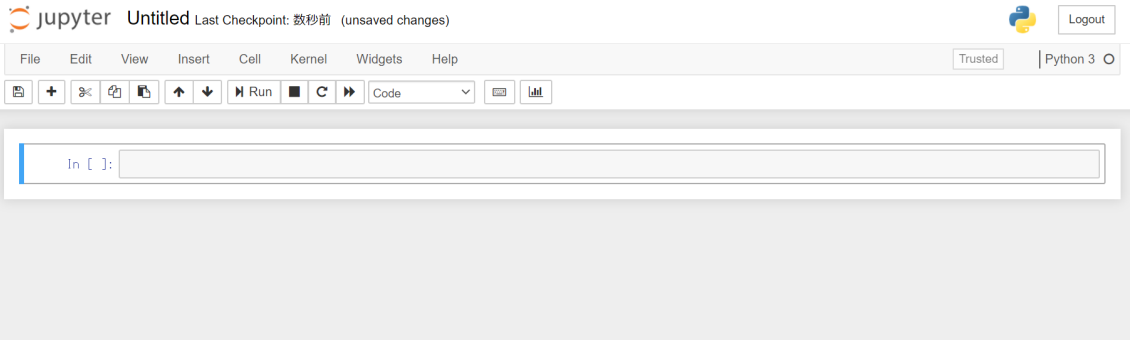
2.2 Jupyter notebook の起動

Python を実行するには色々な方法がある．例えばコマンドプロンプト上で Python を起動して一行ずつコードを実行する方法，まとまったコード（スクリプトという）をテキストエディタで書いておいて，まとめて実行する方法などである．授業では Jupyter notebook という，web ブラウザ上で Python を実行できるアプリケーションを使うので，ここでは Jupyter notebook の使用方法を解説する．

1. Anaconda prompt を起動する．
   - windows PC の場合，windows メニューの中に Anaconda3 というアプリケーションのフォルダが入っているはずなので，その中の Anaconda prompt を選択．
2. Jupyter notebook を起動する．
   - 表示される画面に `jupyter notebook` と打ち込み， `Enter` を押す．
   - ブラウザが自動で起動し，以下のような画面が表示される．



- さらにこの画面の右上の New→Python3 とクリックすることで以下の画面が表示される．これが Jupyter notebook である．



ヒント： 上記の手順に従うと，作業フォルダは windows のユーザーフォルダになるので，このフォルダ以下にあるファイルしか開くことができない．自分の PC であればそれでも特に困ることは無いと思うが，授業で使う PC は USB メモリなどにファイルを保存する必要がある．そのためには Jupyter notebook を起動する前に適切なフォルダに移動しなければならない．Anaconda prompt 上でフォルダを移動するには，`cd` **移動先のフォルダ** を実行する．例えば USB メモリが F ドライブで，その中の file フォルダに移動するなら，`cd F:\file` を実行する．ちなみに `cd` は change directory の略であり，ディレクトリはフォルダと同じような意味で使われる用語である．

2.3 Jupyter notebook での計算

難しい話は置いておいて，とりあえず Jupyter notebook を使って簡単な計算をしてみよう．Jupyter notebook では `In[]` と書かれている右側の欄（セルという）にコードを入力する．まずは以下のようにと「半角で」入力して， `Shift` キーを押しながら `Enter` キーを押してみよう． `Out[1]` の右側に 5 と表示されれば成功である．

```
2 + 3
```

言うまでもなくこれは 2+3 という足し算を，python が代わりに計算して結果を返してくれたのである．Jupyter

notebook ではこのように入力されたコードの実行結果をすぐ下に表示してくれるので非常に見やすい．要領が分かったところで，各自色々な計算を試してみよう．ただし，python（を含むプログラミング言語）では通常私たちが手計算で使う数学記号と違うものを使うことが多い．以下に代表的なものをまとめたので参考にして欲しい．剰余は普段の計算ではあまり使うことは無いかもしれないが，プログラミングをする上では知っておくと何かと便利である．

python での記号	意味	例	実行結果
+	足し算	2 + 3	5
-	引き算	2 - 3	-1
*	掛け算	2 * 3（2×3）	6
/	割り算	2 / 3（$2 \div 3$）	0.6666666666666666
**	べき乗	2 ** 3 （2^3）	8
%	剰余	2 % 3 （$2 \div 3$ の余り）	2

ここで一つ注意点だが，python では何か計算結果を画面上に表示したいときには `print` 関数を使う必要がある．例えば先ほどのように $2 + 3$ の計算結果を表示したければ `print(2 + 3)` として実行しなければならない．Jupyter notebook では `print` を使う手間を多少軽減してくれており，セル内の最後の行に入力された内容は `print`` 無しで表示してくれる．一つのセル内で複数の計算を行い，どちらも表示したい場合には通常通り :code:`print` を使う必要があるので注意しよう．例えば $2 + 3$ と $7 - 2$ を両方表示したい場合には，以下のようにしなければならない．

```
print(2 + 3)
print(7 - 2)
```

2.4 ファイルの保存の仕方・開き方

Jupyter notebook では，作業を新しく始めた時点で作業フォルダ内に Untitled.ipynb という名前でファイルが作られている．ipynb というのは Jupyter notebook のファイルの種類を表す記号（拡張子という）である．これは余談だが，Jupyter notebook は以前は ipython notebook という，python に特化したアプリだったので，その名前が名残として残っている．

話を戻すと，Jupyter notebook での作業中は定期的に自動で上書き保存される．画面上部に autosaved と表示されていれば，その時点での入出力の内容は保存済みである．とは言え，自分でも定期的に作業内容を保存する習慣をつけるのが望ましい．上書き保存をする方法は実に簡単で， `Ctrl` キーを押しながら `s` を押せばよい．これはほとんどの windows 用ソフトウェアで共通のショートカットキーなので覚えておこう．

ファイル名を変更するには，Jupyter notebook の画面上部に太字で書かれているファイル名（デフォルトでは Untitled）をクリックして表示されるテキストボックスに新しい名前を入力して *Rename* をクリックすれば良い．ファイル名は日本語も使えるがお勧めしない．基本的にプログラミング言語は英数のファイル名を想定して作られているので日本語のファイル名は不具合の元である．

ファイルを開くのも簡単で，作業フォルダ内に既に ipynb ファイルがあれば Jupyter notebook のホーム画面に表示されているはずである．それをクリックすれば ipynb ファイルを開くことができる．

2.5 Jupyter notebook の便利な使い方

2.5.1 コマンドモード

Jupyter notebook では，セルの中にコードを入力し，実行する作業を繰り返す．その過程で必要の無いセルができてしまったり，セルを複製したいということが起こり得る．このような，セルの操作を行う時にはコマンドモードを使う．コマンドモードに入るには，操作したいセルのコード入力箇所以外の部分をクリックするか，コード入力中に Esc キーを押せばよい．以下の画面のように，セルの左端が青く表示されていればコマンドモードに入っていることを表している．

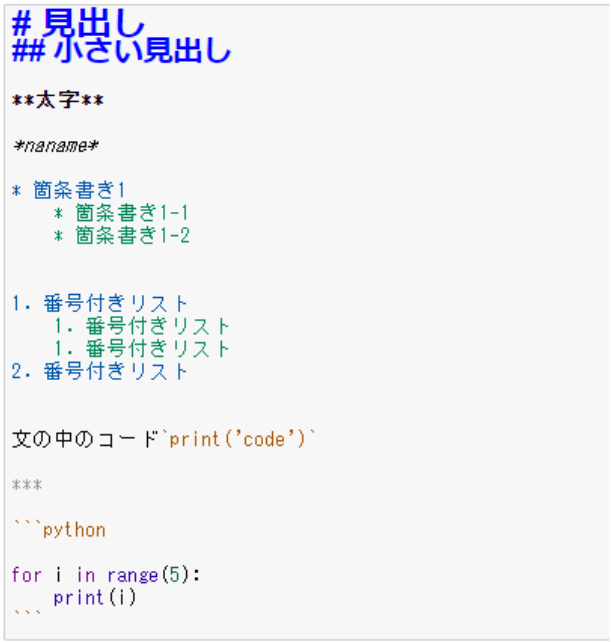
この状態で，コマンドを入力すればセルの操作が可能である．以下の表に挙げたものは使う機会が多いと思われる．他にもいろいろあるので，確認するには Jupyter notebook の画面上部の Help の中にある Keyboard Shortcuts を見よう．

コマンド	実行結果
c	選択したセルをコピーする
x	選択したセルを切り取る
v	選択した箇所の一つ下にコピー・切り取りしたセルの貼り付け
d を 2 回	選択したセルの削除
z	セルの削除を元に戻す
s	現在の作業内容を保存する
m	マークダウンセルに変換
y	コードセルに変換

2.5.2 マークダウン

Jupyter notebook はコードを実行するだけではなく，文書の作成にも適している．ただし普通のコードセルの中に文章を書くのは適切な使い方ではないし，読みづらい．Jupyter notebook 上での文書作成にはマークダウン記法を使う．マークダウンとは，テキストの装飾や箇条書きなど，テキストの見栄えを整えるときに使われる記法である．これを上手く使えば授業中のメモ書きなどにも使えるし，レポート作成の時にも便利である．前節のショートカットの表にもある通り，セルをマークダウン用のセルに変えるにはコマンドモードで m を，コードセルに戻すにはコマンドモードで y を押せばよい．以下のマークダウン記法とその実行結果を見比べてみよう．

表記	実行結果
# 見出し	見出しをつける．#が多いほど文字が小さくなる．
テキスト	太字になる
text	斜体になる
* テキスト	箇条書きにする．*の後に半角スペースを入れる．記号は-でも可．一つ下のレベルの箇条書きを作るには改行して Tab キーで字下げする．
1. テキスト	番号付きのリストになる．行頭の数字は全部 1 でも勝手に調整してくれる．. の後に半角スペースを入れる．一つ下のレベルの箇条書きを作るには改行して Tab キーで字下げする．
***	水平線
`コード`	行の中にコードを書き，強調表示する．
``python と ```で囲む	別行でコードを表示してハイライトをつける



```
# 見出し
## 小さい見出し

**太字**

*name*

* 箇条書き1
    * 箇条書き1-1
    * 箇条書き1-2

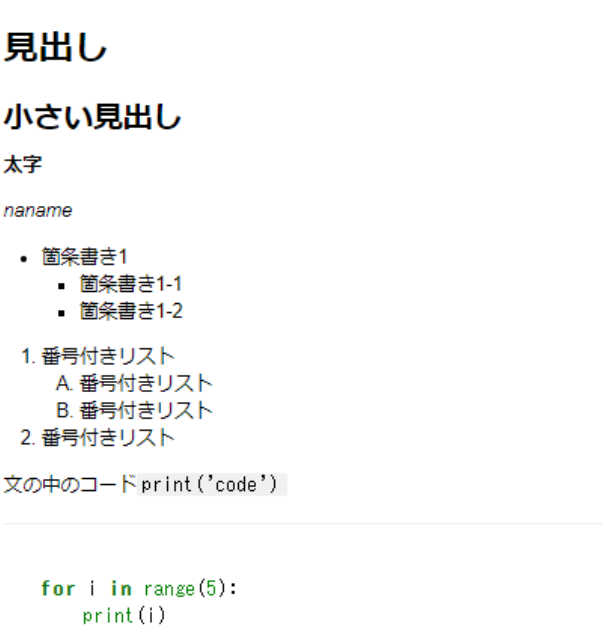

1. 番号付きリスト
    1. 番号付きリスト
    1. 番号付きリスト
2. 番号付きリスト

文の中のコード`print('code')`

***

```python

for i in range(5):
    print(i)
```
```



見出し

小さい見出し

太字

name

- 箇条書き1
  - 箇条書き1-1
  - 箇条書き1-2

1. 番号付きリスト
   A. 番号付きリスト
   B. 番号付きリスト
2. 番号付きリスト

文の中のコード `print('code')`

```python
for i in range(5):
    print(i)
```

2.6 Jupyter notebook が反応しなくなったら…

色々試しながらコードを書いていると，Jupyter notebook が反応しなくなることがある．これは，ループと呼ばれる繰り返し処理の終了条件が満たされずに永遠に繰り返し計算をし続けていたり，正しいコードであっても非常に大規模な計算を実行させたときに起こりやすい．計算を実行中の場合はそのセルの左に `In [*]` と表示される．大規模な計算になることが最初から想定されていた場合は仕方ないが，そうでなければコードの修正が必要である．そのためにはまずは一度計算を停止しなければならない．計算を強制的に停止させるには，Jupyter notebook の画面上部にある C をクリックし，表示されるダイアログで `Restart` を選択すればよい．しばらく待つと `In [ ]` の表示に変わり，命令を受け付ける状態になる．

例として，以下のコードを実行して試してみよう．なお，詳しくは後で述べるが，二行目の `pass` の前にはインデント（字下げ）が必要である．インデントを挿入するには `Tab` キーを押せばよい．

```
while True:
    pass
```