Programming for Data Science (with Python)

Le Trong Ngoc – http://fit.iuh.edu.vn/giangvien@letrongngoc

Contents

- Introduction to Data Science and Data Analysis
- **Introduction to Python for Data Science**
- Data Visualization with Python
- Statistical Thinking in Python
- Applied Machine Learning in Python

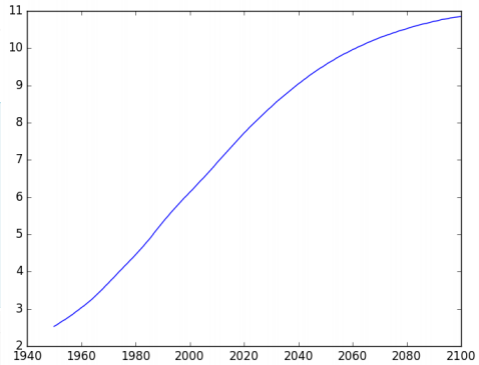
Introduction to Python for Data Science

- Set up the Lab Environment
- Python basics
- List – A Data Structure
- Functions and Packages
- Numpy
- **Plotting with Matplotlib**
- **Control Flow and Pandas**

Plotting with Matplotlib

my_script.py

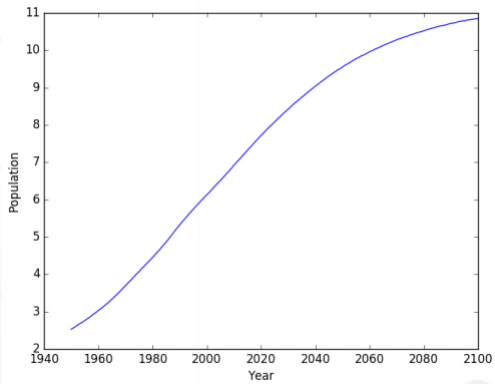
```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out

plt.plot(year, population)
plt.show()
```



Plotting with Matplotlib

my_script.py

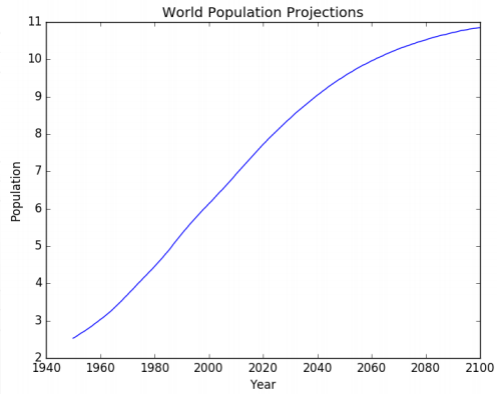
```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out

plt.plot(year, population)

plt.xlabel('Year')
plt.ylabel('Population')

plt.show()
```



Plotting with Matplotlib

my_script.py

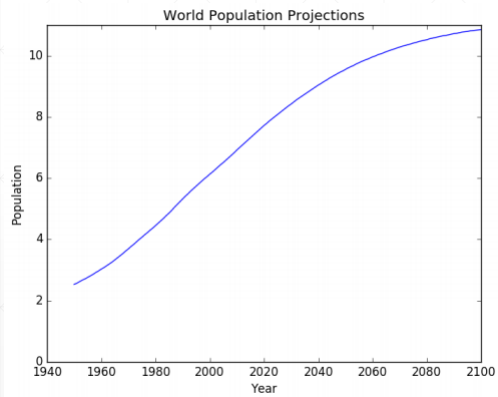
```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out

plt.plot(year, population)

plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population Projections')

plt.show()
```



Plotting with Matplotlib

my_script.py

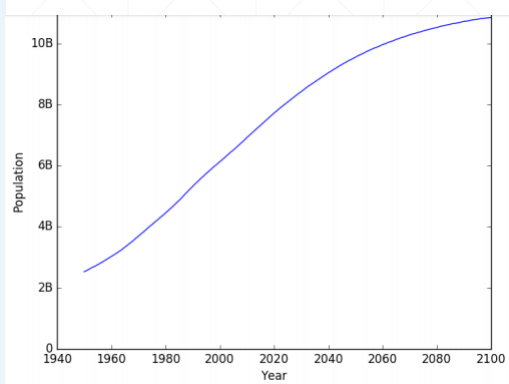
```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out

plt.plot(year, population)

plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population Projections')
plt.yticks([0,2,4,6,8,10])

plt.show()
```



Plotting with Matplotlib

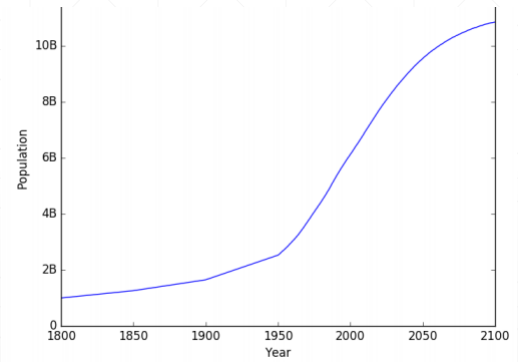
my_script.py

```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out

plt.plot(year, population)

plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population Projections')
plt.yticks([0,2,4,6,8,10],
           ['0','2B','4B','6B','8B','10B'])

plt.show()
```



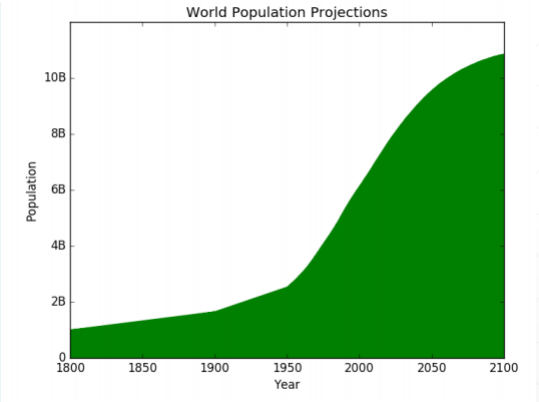
Plotting with Matplotlib

```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out
population = [1.0,1.262,1.650] + population
year = [1800,1850,1900] + year

plt.plot(year, population)

plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population Projections')

plt.yticks([0,2,4,6,8,10],
           ['0','2B','4B','6B','8B','10B'])

plt.show()
```

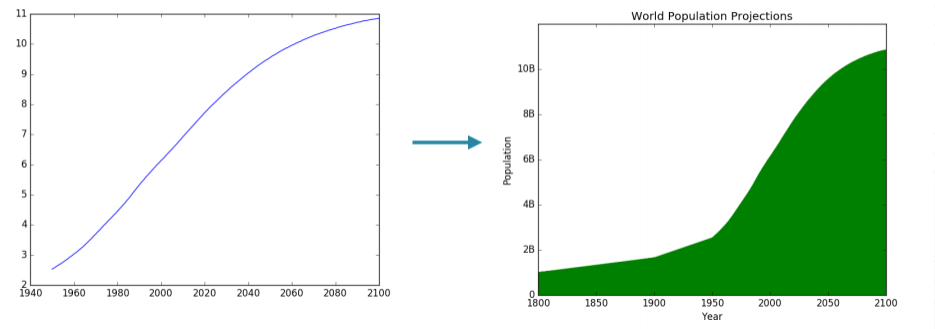


Plotting with Matplotlib

```
import matplotlib.pyplot as plt
year = ... # Implementation left out
population = ... # Implementation left out
population = [1.0,1.262,1.650] + population
year = [1800,1850,1900] + year

plt.fill_between(year,population,0,color='green'

plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population Projections')

plt.yticks([0,2,4,6,8,10],
           ['0','2B','4B','6B','8B','10B'])

plt.show()
```



Plotting with Matplotlib



Control Flow

- Different Python types
- `bool`: boolean

```
In [1]: bmi = ... # Implementation left out

In [2]: bmi
Out[2]: array([ 21.852,  20.975,  21.75 ,  24.747,  21.441])

In [3]: bmi[bmi > 23]
Out[3]: array([ 24.747])
```

Control Flow

```
In [4]: 2 < 3
Out[4]: True

In [5]: 2 == 3
Out[5]: False

In [6]: x = 2

In [7]: y = 3

In [8]: x < y
Out[8]: True

In [9]: x == y
Out[9]: False
```


Control Flow

<	strictly less than
<=	less than or equal
>	strictly greater than
>=	greater than or equal
==	equal
! =	not equal

Control Flow

```
In [10]: True and True
Out[10]: True

In [11]: False and True
Out[11]: False

In [12]: True and False
Out[12]: False

In [13]: False and False
Out[13]: False

In [14]: x = 12
                True        True
In [15]: x > 5 and x < 15
Out[15]: True
```


Control Flow

```
In [16]: True or True
Out[16]: True

In [17]: True or False
Out[17]: True

In [18]: False or True
Out[18]: True

In [19]: False or False
Out[19]: False

In [20]: y = 5
                True        False
In [21]: y <= 7 or y > 13
Out[21]: True
```

Control Flow

```
In [22]: not True
Out[23]: False

In [24]: not False
Out[24]: True
```


Control Flow

```
if condition :
    expression
```

```
z = 4
if z % 2 == 0 :
    print("z is even")
```

True

```
z = 4
if z % 2 == 0 :
    print("checking "  + str(z))
    print("z is even")
```

Control Flow

```
if condition :
    expression
else :
    expression
```

```
z = 5
if z % 2 == 0 :
    print("z is even")
else :
    print("z is odd")
```

False

Control Flow

```
z = 6
if z % 2 == 0 :
    print("z is divisible by 2")
elif z % 3 == 0 :
    print("z is divisible by 3")
else :
    print("z is neither divisible by 2 nor by 3")
```

True

Never reached

Pandas

- Huge amounts of data are common
- 2D Numpy array?
  - Only one type possible
- Pandas
  - High-level data manipulation
  - `DataFrame`

Pandas

brics

```
In [1]: brics = ... # declaration left out

In [2]: brics
Out[2]:
```

column labels

	country	population	area	capital
BR	Brazil	200	8515767	Brasilia
RU	Russia	144	17098242	Moscow
IN	India	1252	3287590	New Delhi
CH	China	1357	9596961	Beijing
SA	South Africa	55	1221037	Pretoria

row labels

Pandas

brics.csv

```
,country,population,area,capital
BR,Brazil,200,8515767,Brasilia
RU,Russia,144,17098242,Moscow
IN,India,1252,3287590,New Delhi
CH,China,1357,9596961,Beijing
SA,South Africa,55,1221037,Pretoria
```


Pandas

CSV file -> DataFrame

```
In [3]: import pandas as pd

In [4]: brics = pd.read_csv("path/to/brics.csv")

In [5]: brics
Out[5]:
  Unnamed: 0       country  population      area    capital
0         BR        Brazil         200   8515767   Brasilia
1         RU        Russia         144  17098242     Moscow
2         IN         India        1252   3287590  New Delhi
3         CH         China        1357   9596961    Beijing
4         SA  South Africa          55   1221037   Pretoria
```

Pandas

```
In [8]: brics["country"]
Out[8]:

BR          Brazil
RU          Russia
IN           India
CH           China
SA    South Africa
Name: country, dtype: object
```

```
In [9]: brics.country
Out[9]:

BR          Brazil
RU          Russia
IN           India
CH           China
SA    South Africa
Name: country, dtype: object
```


Pandas

```
In [10]: brics["on_earth"] = [True, True, True, True, True]

In [11]: brics
Out[11]:
         country  population      area    capital on_earth
BR        Brazil         200   8515767   Brasilia     True
RU        Russia         144  17098242     Moscow     True
IN         India        1252   3287590  New Delhi     True
CH         China        1357   9596961    Beijing     True
SA  South Africa          55   1221037   Pretoria     True
```

Pandas

```
In [12]: brics["density"] = brics["population"] / brics["area"] * 1000000

In [13]: brics
Out[13]:

         country  population      area    capital on_earth     density
BR        Brazil         200   8515767   Brasilia     True   23.485847
RU        Russia         144  17098242     Moscow     True    8.421918
IN         India        1252   3287590  New Delhi     True  380.826076
CH         China        1357   9596961    Beijing     True  141.398928
SA  South Africa          55   1221037   Pretoria     True   45.043680
```


Pandas

```
In [14]: brics.loc["BR"]
Out[14]:

country          Brazil
population          200
area            8515767
capital        Brasilia
density        23.48585
on earth           True
Name: BR, dtype: object
```

Pandas

```
In [15]: brics.loc["CH","capital"]
Out[15]: Beijing

In [16]: brics["capital"].loc["CH"]
Out[16]: Beijing

In [17]: brics.loc["CH"]["capital"]
Out[17]: Beijing
```