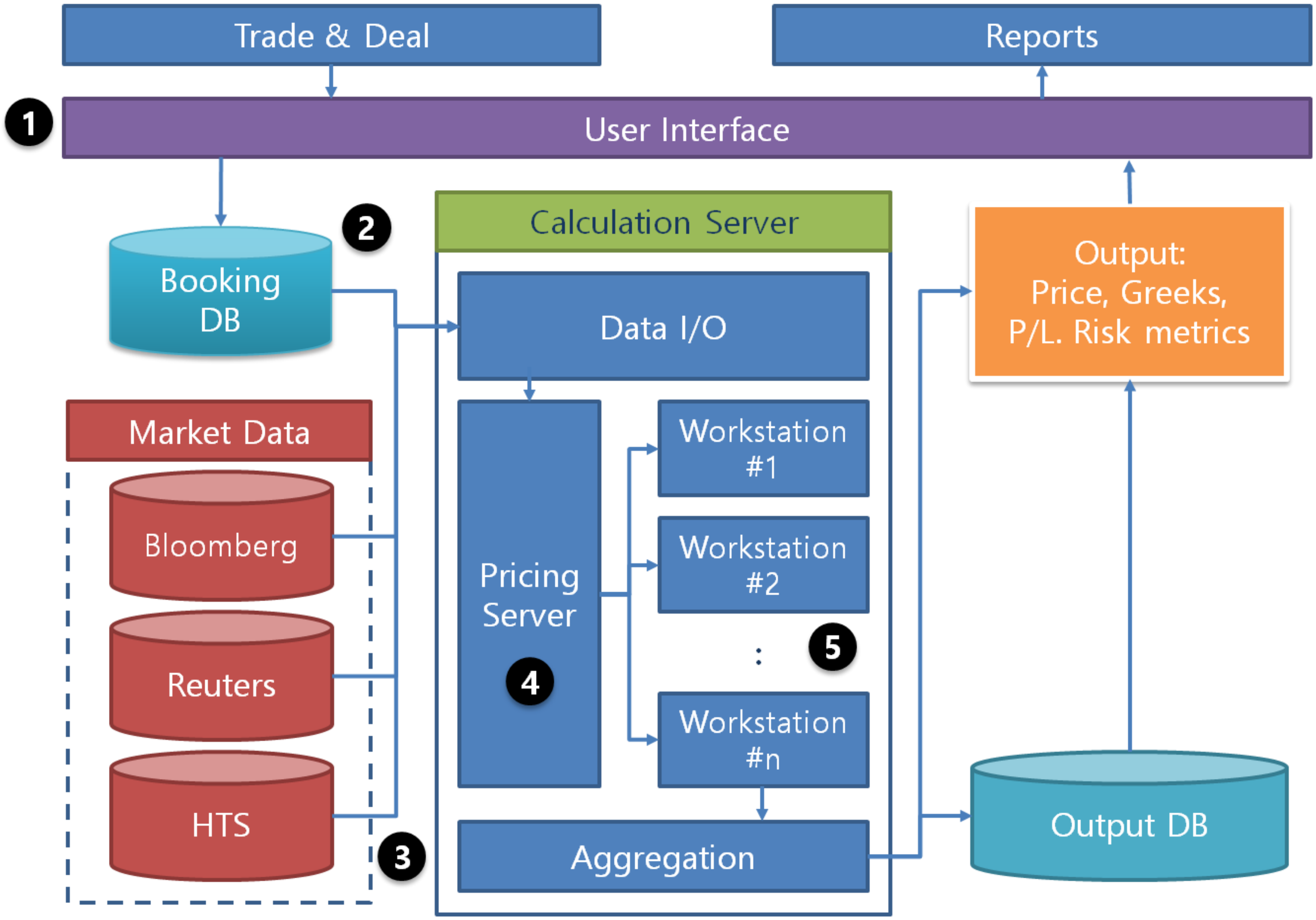
Python 1: Introduction

금융공학 프로그래밍 I

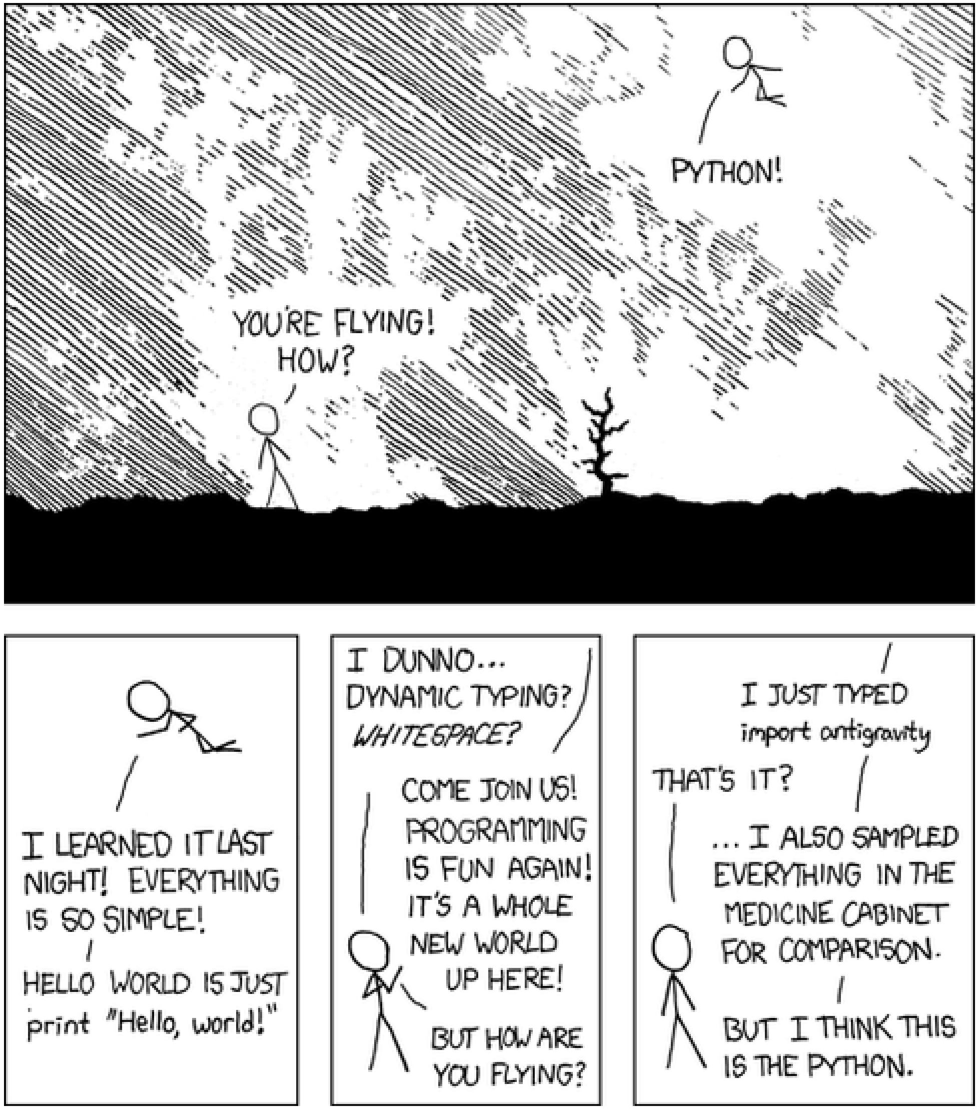
IT environment in the Front Office



Need to know

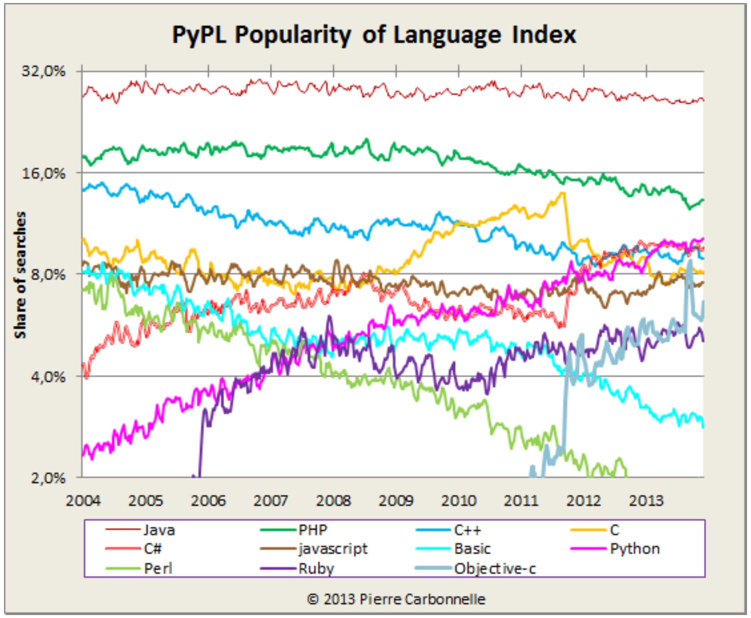
- **Operating System**
  - Windows, Linux, OS X, UNIX, iOS, Android …
- **Programming language**
  - C, C++, Java, C#, Python, Go, R, Objective-C, VB, PHP, JS …
- **Database**
  - SQL (MySQL, Oracle, SQL Server), NoSQL (MongoDB)
  - Big data (Hadoop)
- **IDE (Integrated Development Environment)**
  - Visual Studio, Eclipse, Xcode, Vim …
- **API & Libraries**
  - Bloomberg, HTS, Open sources …
- Finally, "**EXCEL**"

About Python


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

About Python

- Python is
  - a general purpose programming language conceived in 1989 by Dutch programmer Guido van Rossum
  - free and open source
  - supported by a vast collection of standard and external software libraries
  - now one of the most popular programming languages



© 2013 Pierre Carbonnelle

About Python

- Common Uses
  - Communications
  - Web development
  - CGI and graphical user interfaces
  - Games
  - multimedia, data processing, security, etc., etc., etc.
- High tech companies
  - Google
  - Dropbox
  - Reddit
  - YouTube
- Python is particularly popular within the scientific community
  - NASA, CERN, Wall St.

About Python

- Features
  - A high level language suitable for rapid development
  - Relatively small core language supported by many libraries
  - A multiparadigm language, in that multiple programming styles are supported (procedural, object-oriented, functional, etc.)
  - Interpreted rather than compiled (script language)

About Python

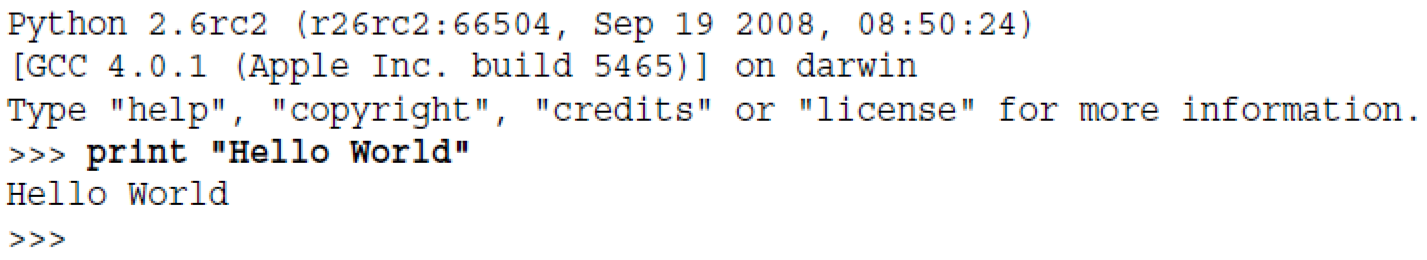
- Scientific Programming
  - Machine learning and data science
  - Astronomy
  - Artificial intelligence
  - Chemistry
  - Computational biology
  - Meteorology

About Python

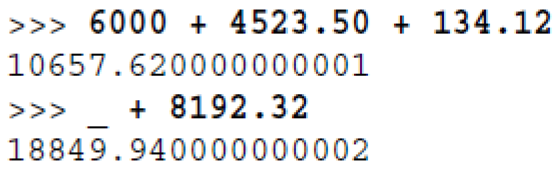
- Python Ecosystem
  - NumPy
    - Matrix & Array
  - SciPy
    - Linear algebra
    - Integration
    - Interpolation
    - Optimization
    - Distribution & random number generation
    - Signal processing
  - Matplotlib: figures and graphs
  - Pandas, SymPy, statsmodels, scikit-learn …

Running Python

- Python Prompt

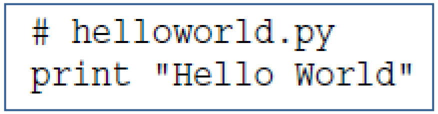


```
Python 2.6rc2 (r26rc2:66504, Sep 19 2008, 08:50:24)
[GCC 4.0.1 (Apple Inc. build 5465)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print "Hello World"
Hello World
>>>
```



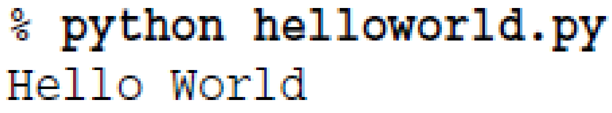
```
>>> 6000 + 4523.50 + 134.12
10657.620000000001
>>> _ + 8192.32
18849.940000000002
```

- Run repeatedly



```
# helloworld.py
print "Hello World"
```

Save as helloworld.py
And then run from terminal prompt.



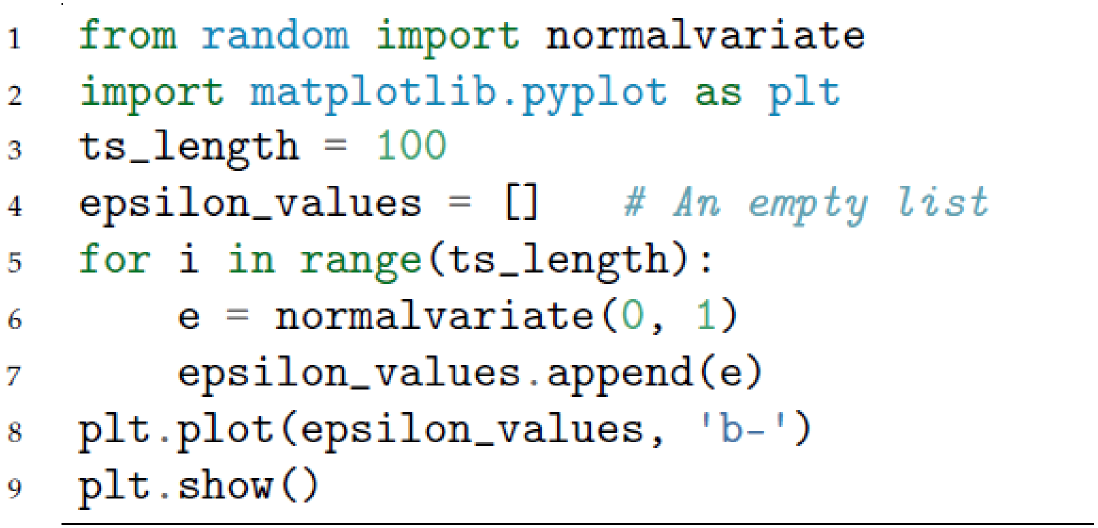
```
% python helloworld.py
Hello World
```

Python Environment

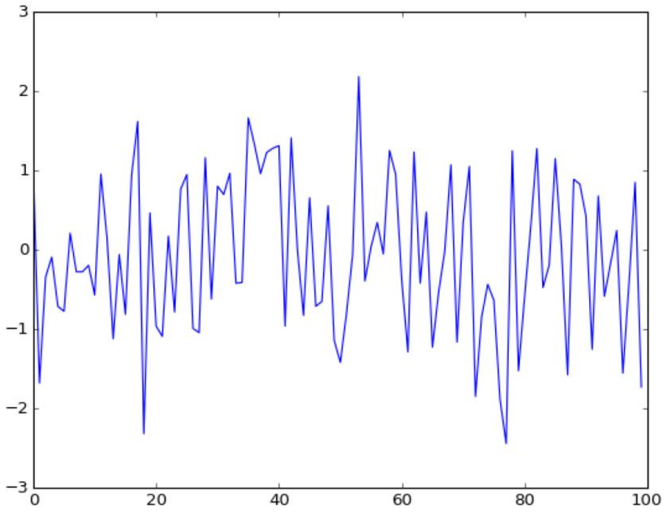
- Anaconda
  - One of the free Python distributions that contains the core Python and the most popular scientific libraries
  - Python 2.7 or Python 3.5 is available currently
  - Package update
    - conda update anaconda
- iPython
  - Jupyter notebook: browser-based interface to Python
  - .ipynb file
- Spyder
  - IDE for Python
- Others
  - PyCharm, Wing IDE, PyDev (for Eclipse), Python tools for VS, …

Introductory Example

- ❖ Plotting a White Noise Process



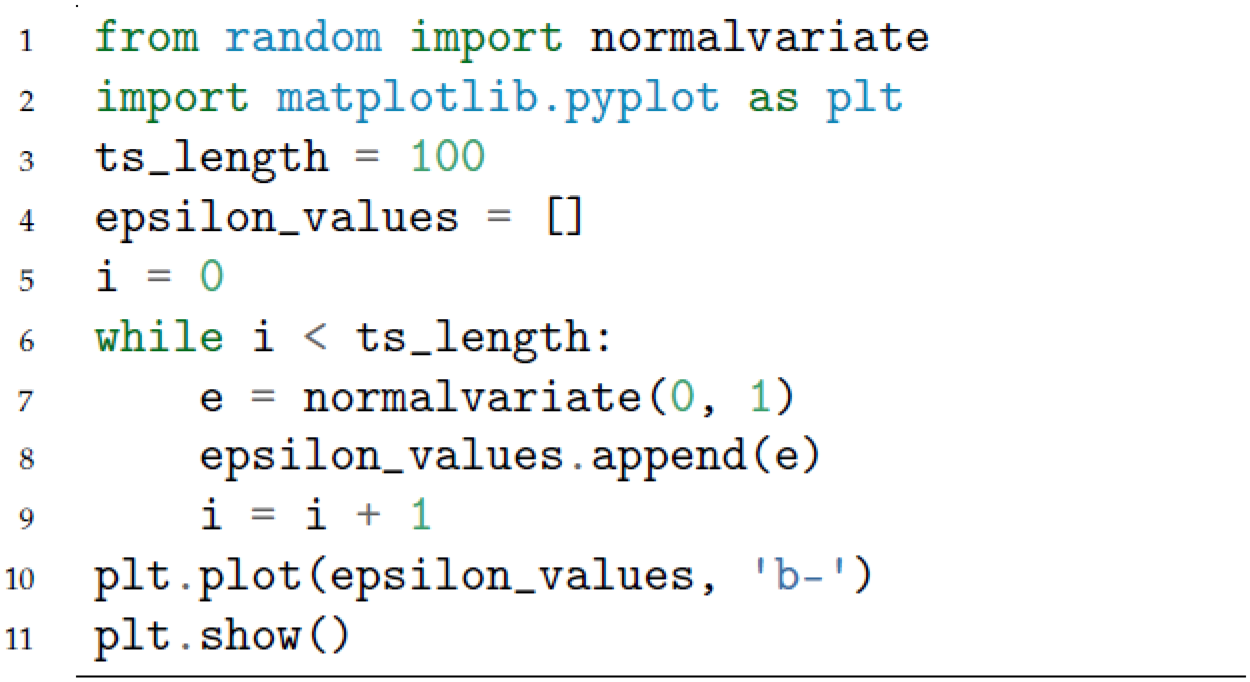
```python
from random import normalvariate
import matplotlib.pyplot as plt
ts_length = 100
epsilon_values = []    # An empty list
for i in range(ts_length):
    e = normalvariate(0, 1)
    epsilon_values.append(e)
plt.plot(epsilon_values, 'b-')
plt.show()
```



Introductory Example

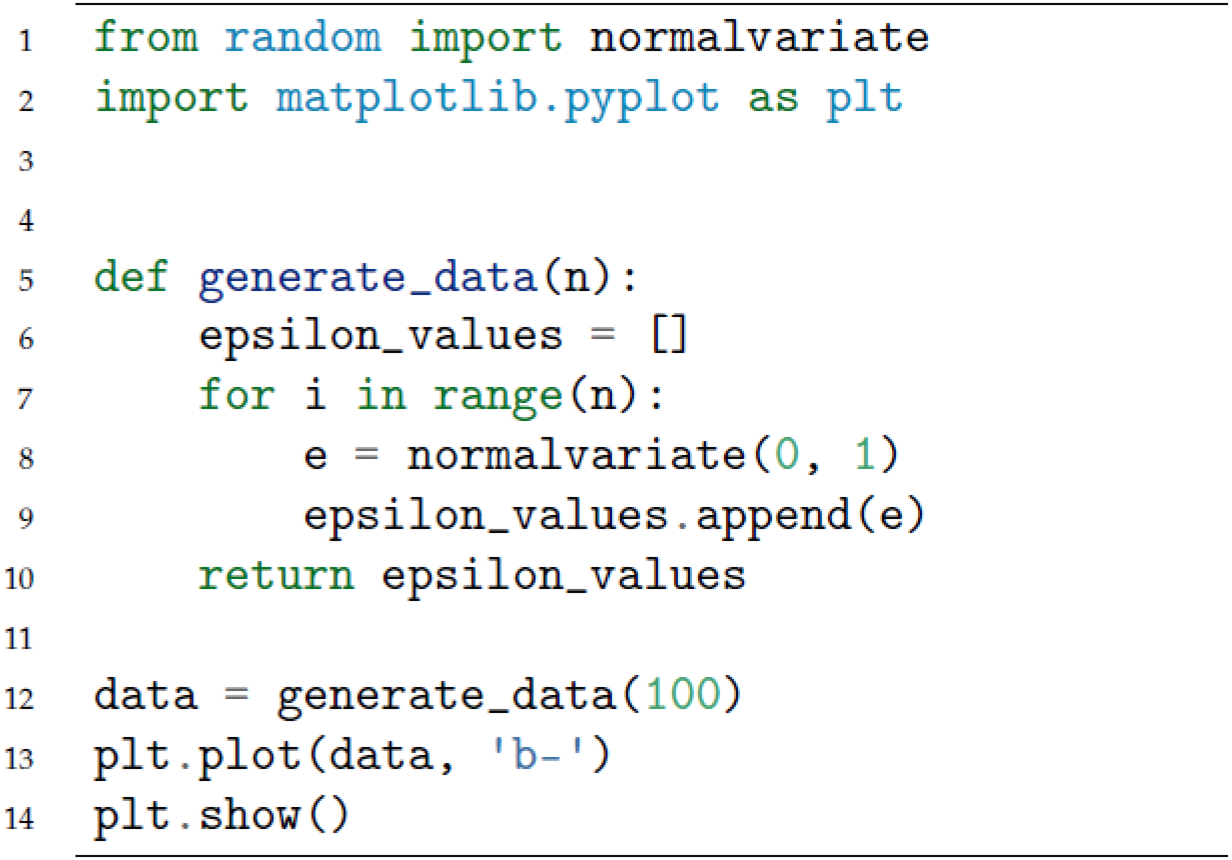
- In the codes,
  - Import from external libraries
    - Modules
    - Packages
  - List variable
  - Iteration
    - Code Blocks and Indentation
  - Display
    - Print on a screen
    - Drawing graph
    - File output
- Not in the codes,
  - Comments

While Loop



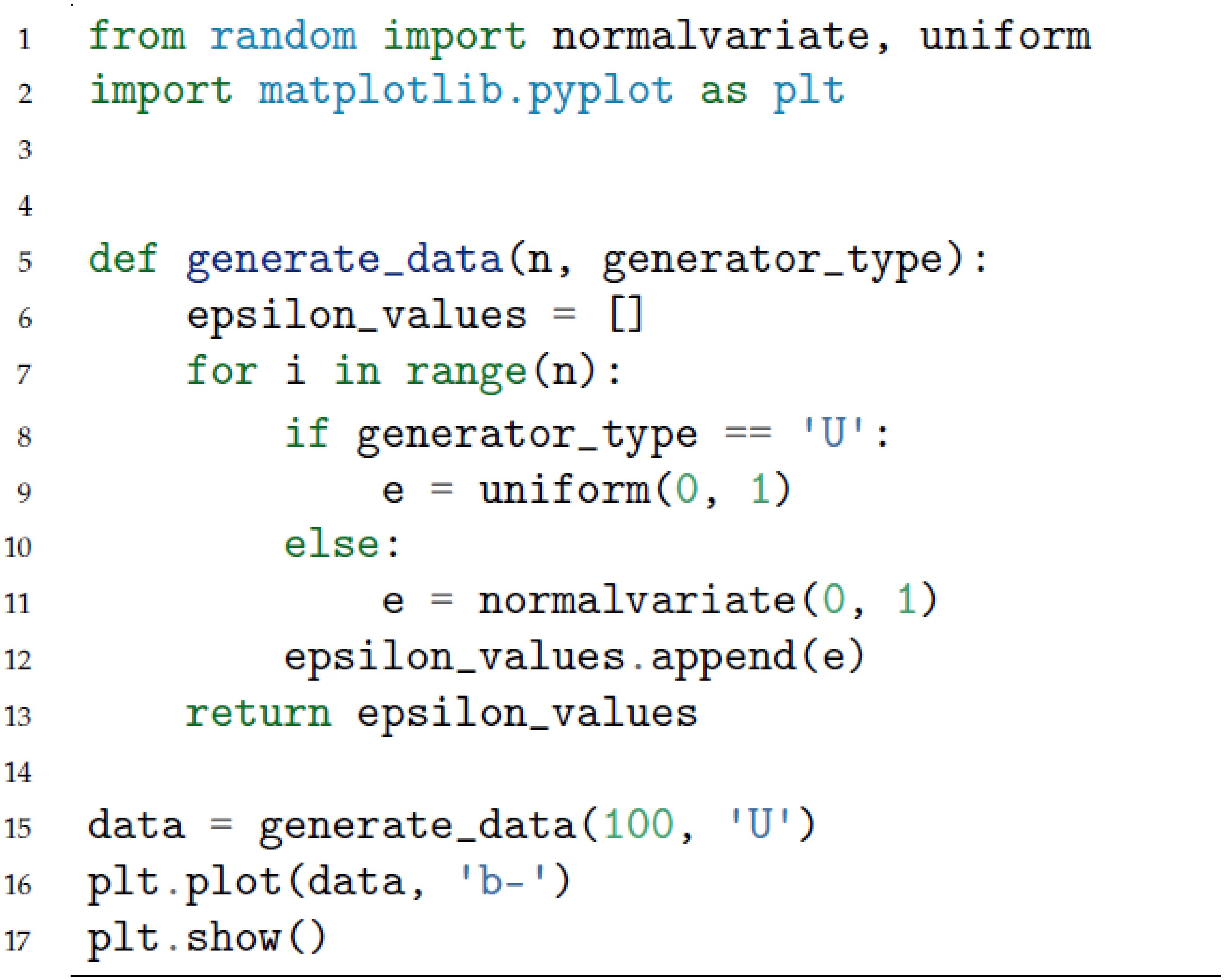
```python
from random import normalvariate
import matplotlib.pyplot as plt
ts_length = 100
epsilon_values = []
i = 0
while i < ts_length:
    e = normalvariate(0, 1)
    epsilon_values.append(e)
    i = i + 1
plt.plot(epsilon_values, 'b-')
plt.show()
```

User-defined Functions



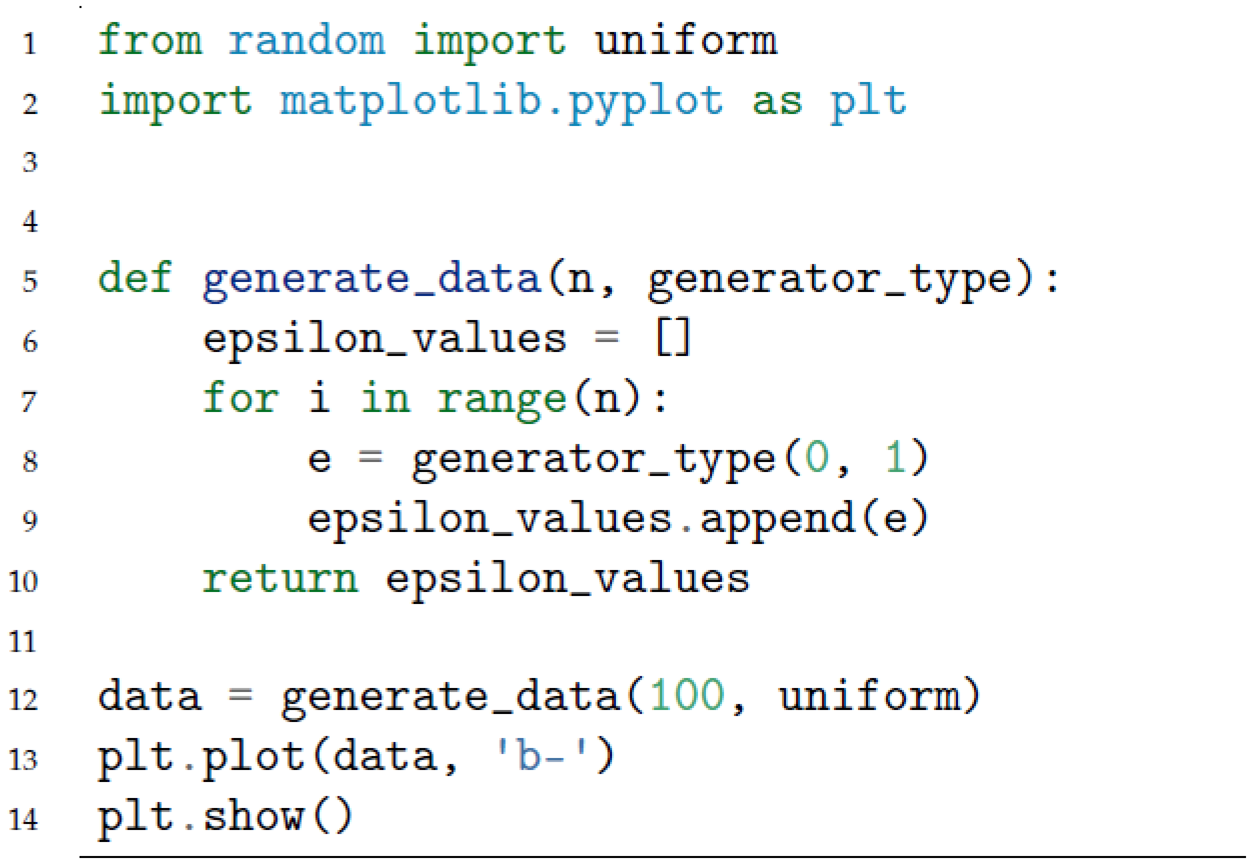
```python
from random import normalvariate
import matplotlib.pyplot as plt


def generate_data(n):
    epsilon_values = []
    for i in range(n):
        e = normalvariate(0, 1)
        epsilon_values.append(e)
    return epsilon_values

data = generate_data(100)
plt.plot(data, 'b-')
plt.show()
```

Conditions



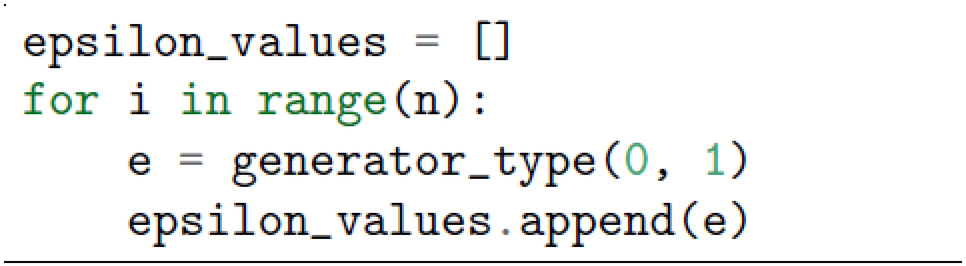
```python
from random import normalvariate, uniform
import matplotlib.pyplot as plt


def generate_data(n, generator_type):
    epsilon_values = []
    for i in range(n):
        if generator_type == 'U':
            e = uniform(0, 1)
        else:
            e = normalvariate(0, 1)
        epsilon_values.append(e)
    return epsilon_values

data = generate_data(100, 'U')
plt.plot(data, 'b-')
plt.show()
```

Passing a Generator as a Function

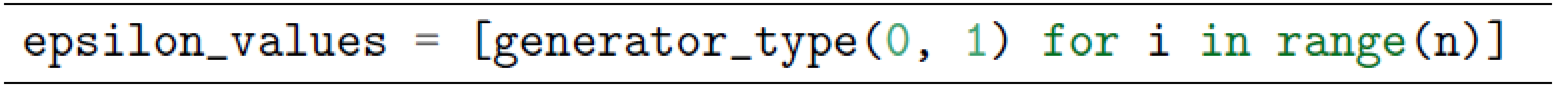


```python
from random import uniform
import matplotlib.pyplot as plt


def generate_data(n, generator_type):
    epsilon_values = []
    for i in range(n):
        e = generator_type(0, 1)
        epsilon_values.append(e)
    return epsilon_values

data = generate_data(100, uniform)
plt.plot(data, 'b-')
plt.show()
```

List comprehensions



```
epsilon_values = []
for i in range(n):
    e = generator_type(0, 1)
    epsilon_values.append(e)
```



```
epsilon_values = [generator_type(0, 1) for i in range(n)]
```

Q & A