Center for Teaching, Research and Learning
Research Support Group
American University, Washington, D.C.
Hurst Hall 203
rsg@american.edu
(202) 885-3862

Introduction to Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. It is used for its readability and productivity.

WORKSHOP OBJECTIVE:

This workshop is designed to give a basic understanding of Python, including object types, Python operations, Python module, basic looping, function, and control flows.

By the end of this workshop you will:

- Know how to install Python on your own computer
- Understand the interface and language of Python
- Be familiar with Python object types, operations, and modules
- Create loops and functions
- Load and use Python packages (libraries)
- Use Python both as a powerful calculator and to write software that accept inputs and produce conditional outputs.

NOTE: Many of our examples are from the Python website's tutorial, http://docs.python.org/py3k/contents.html. We often include references to the relevant chapter section, which you could consult to get greater depth on that topic.

I. Installation and starting Python

Python is distributed free from the website http://www.python.org/. There are distributions for Windows, Mac, Unix, and Linux operating systems. Use http://python.org/download to find these distribution files for easy installation. The latest version of Python is version 3.1.

Once installed, you can find the Python3.1 folder in "Programs" of the Windows "start" menu. For computers on campus, look for the "Math and Stats Applications" folder. To open Python, select "IDLE (Python GUI)". A new window titled "Python Shell" will appear.

The program IDLE does two things. First, it allows you to create, run, and save more comprehensive programs in Python. Secondly, and our focus of this introduction, is that it also acts as an interpreter. This means instead of having to compile code, you can simply type code into the Python Shell after the >>> prompt, and IDLE will execute that code. This makes developing in Python easy, because you can test bits of code immediately to see if they work properly.

There are also many other Python distributors that provide more powerful and user friendly Python shells, including **canopy** (https://www.enthought.com/products/canopy/) and **winPython (**https://winpython.github.io/). Canopy is free to use for all AU students. Students just need to register an online account with their AU email address in order to install. In addition, **Jupyter Notebook** is a python notebook that integrates python module into standard notebook for easy presentation. The advantage of using 3rd party Python distributors is that many common Python modules (packages) have already included in the installer so users don't need to install them separately.

Also, please be aware of the different versions of Python. There are some differences between Python 3.x and Python 2.x. As a result, some old modules might not be able to run on Python3.x and vice versa. Many prior Python users still prefer Python 2.x but most of the new users start with Python 3.x. Which version to use is your choice and it will not make too much difference.

II. Object Types in Python:

Python is an object-oriented computer language, meaning everything is stored in the computer as an object. Understanding different object types is essential for understanding how Python works.

1. Numbers:
    a. Integers: 1
    b. Floating numbers 1.2
2. Strings: Characters: "I am a King"

3. List: [1, 2, 3, 'A', 'A']
   List is the most flexible format. It takes numbers, strings and most python types. Also, it allows duplicates. In addition, elements in list is ordered and can be changed.

4. Dictionary: {"a": [1,2,3], "b": [2,3,4]}
   Dictionary is good for grouping elements. In this example, each group of element starts with their group indicator "a" and "b". This makes dictionary a convenient way to construct data frame in python.

5. Set: {1,2,3}
   Elements in set are not ordered thus they cannot be access by position. Also, set doesn't allow duplicates. This property makes set an easy way to remove duplicates in dataset.

6. Tuple (1,2,3)

Elements in tuple is ordered thus can be accessed by position. However, the elements and positions are locked so they cannot be changed.

Every piece of information is stored as an object in Python so users can assign them to a “name”
A = [1,2,3]

Accessing list-like objects by position:

```
A = [1,2,3,4,5]

A[0]       #position start with 0 not 1

A[1:3]     # from position two to four

A[-3, -1]  # from the end
```

III. Operations

Expression operators:

+, -, *, /, >,<, **, &, etc

Built-in mathematical functions:

pow, *abs*, *round*, *int*, *bin*, etc

```
Pow(2,3)

abs(-1)

round(3.3333, 2)

int(1.2)

bin(8)
```

Boolean operators:

and(&) *or*(|) *not*(!)

```
a = [i for i in [1,2,3,4] if i >1 and i<4 and i!=3]
```

IV. Looping with Python

1. *for* loop
go through a list like object

Example:

```
for i in [1,2,3]:

     Print(i)
```

2. *while* loop
Run statement repeatedly until certain condition is reached.

Example:

```
a = 1

while a <10:

    a = a+1

    print(a)
```

Attention: please be careful with the syntax and format. (indentation, colon). Python will not run if there are errors.

V. Control Flow

Choose from multiple groups of statements based on conditions
if \ elif \ else: separate statement based on condition
continue \ break: control loops

```
for i in [1,2,3,4,5,6,7,8]:
    if i ==1:
        print(i)
   elif I ==2:
         print(i*2)
   elif i ==3:
         continue
   elif ==4:
         print(i*10)
   else:
         break
```

VI. Function

What is function? It accepts inputs, does something with the inputs, and then it produces outputs!

To define a function in Python:

```
def function( x ):
     return x**2
def loop ( x ):  ####take a list
   for i in x:
       print i**3
```

VII. Pandas Basics

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the **Python** programming language.
It takes common data files such as csv and excel. The following examples show how to load data into python using *pandas*.

```
import pandas as pd       ## import pandas

pd.__version__            ## check you pandas version

df = pd.read_csv(“data file path”)   ##load csv data

df1 = pd.read_excel(“data file path” , sheetname = “name”  ##load excel sheet
```

Note: Many other options can be used when loading csv or excel format data. Please refer to python documentations for details. Documentations can be easily accessed by using help() commend. For example: help(pd.read_csv).

VIII. Python Packages (Library)

There are very limited functions built in Python. However, there are many libraries (package, module) available for users to import. Popular packages include numpy (for math), scipy (for statistics), pandas (for panel data), matplotlib (for plotting) and statsmodels (for regression).

Import package:

```
import numpy

import numpy as np

form numpy import array
```

Once a package is imported, it can be used in Python.

Some Examples:

```
name = input("What's your name? ")
print("Nice to meet you " + name + "!")
age = input("Your age? ")
print("So, you are already " + str(age) + " years old, " + name + "!")
do = input("What can I do for you?")
if len(do)>15:
         print('That is too much! Sorry I cannot help!')
else:
        print('That is easy you should do it on your own.')
do1 = input("What else?")
if len(do1)>15:
        print('That is too much! Sorry I cannot help!')
else:
        print('That is easy you should do it on your own.')
```

```
a = [1,2,3]
b = [4,5,6]
a+b = [1,2,3,4,5,6]
c = np.array(a) + np.arry(b)
c = [5,7,9]
d = {“a” : a, “b”:b}
pd.DataFram(d)
Out[51]:
   a  b
0  1  4
1  2  5
2  3  6
```

```
import numpy as np
import statsmodels.api as sm
obs = 100
X = np.random.random((nobs, 2))
X = sm.add_constant(X)
beta = [1, .1, .5]
e = np.random.random(obs)
y = np.dot(X, beta) + e
results = sm.OLS(y, X)
results = results.fit()
results.summary()
```

Tips:

There is a large Python user community so it is relatively easy to find materials and solution to your Python problems.

Also use help( ) to get the documentation for functions and packages.

“Python help, example”

help(abs)

Alternatively, you can read online documentation (http://docs.python.org/) or use the “module docs” to find the descriptions of the function.

For a full list of our other research workshops, go to http://www.american.edu/ctrl/rsgevents.cfm

One-on-one assistance with Python is also available in the CTRL lab. Please email rsg@american.edu to set up an appointment. For more information, go to http://www.american.edu/ctrl/lab.cfm