“Advanced Python Programming for Everybody”

Instructor: Ernest Bonat, Ph.D.
Senior Software Engineer
Senior Data Scientist
ebonat@15itresources.com
Cell: 503.730.4556

About the Instructor

1. Currently working for 15 IT Resources as a Consulting Software Engineer and Data Scientist

2. Teaches Computer Science and Business Statistics classes online

3. Organized the Hillsboro Python Machine Learning meetup

(https://www.meetup.com/Hillsboro-Python-Machine-Learning-Meetup/)

4. Worked for Intel as Senior Software Engineer from 2015 to 2016

Module 1 “Object-Oriented Programming in Python”

GitHub: https://github.com/ebonat/intel_module_1

Class Agenda

- Prerequisites
- Requirements
- Duration
- Cost – FREE!
- Objective – To learn Python best programing practices and become a better Python Software Engineer!
- Modules

What you will DO

- Study, read and practice every week
- Follow Python standard software development guides
- Always try to find the best possible solution of your programming tasks (careful with Google)
- Try to use the latest Python developed libraries and framework if possible (first look at Python Documentation and then Google)
- Participate in program code review with experienced Software Engineers – **very important!**
- Participate in Python meetups meeting (once a month) and conferences (watch conferences videos on YouTube)

- Ask yourself the following question: If someone looks at my developed Python code, can they say that I'm a good Python Software Engineer?

Main issues with Python Developers everywhere today:

- Very old Top-Down programming style
- Use of Procedure Programming ("the Pythonic way") instead of Object-Oriented Programming (OOP) design and implementation
- No error handling implementation

- Use Python old code from previous versions
- Hardcode default numerical and string parameters including Machine Learning models hyperparameter values
- Code comments isn't provided at all, especially "dostring" comments for functions procedures and class objects
- Not following the Python naming and conversion standards provided in Style Guide for Python Code "PEP 8" (https://www.python.org/dev/peps/pep-0008/)
- Programs unit tests are not implemented at all

- Developer program documentation isn’t provided

- Don’t try to simulate the same programming language code from another in Python. Every programming language is different!

Why Object-Oriented Programming (OOP)?

1. OOP provides a clear modular structure for programs which makes it good for defining abstract datatypes where implementation details are hidden and the unit has a clearly defined interface

2. OOP makes it easy to maintain and modify existing code as new objects can be created with small differences to existing ones

3. OOP provides a good framework for code libraries where supplied software components can be easily adapted and modified by the programmer. This is particularly useful for developing graphical user interfaces

Python Main Program Definition

```
import libraries, files, classes

def function1():

def function2():


def main():
   some code....
   variable = function1()


if __name__ == '__main__':
```

main()

Python Function Definition

```
def function_name(parameters)
   try:
      some code...
   except:
      error handing...
   finally:
      objects clean-up…
   return
```

Exercise 1

Design a function that format and return a decimal number with specific number of digits (1, 2, 3, 4 or 5)

Python Class Object Definition

```
import libraries
class ClassName(object):
    define consts and fields

    def __init__(self):
        some code…


     def function_name(self, parameters)
        some code…
```

Exercise 2

Base on the files functions.py and utility_functions.py developer the following files to get the full name of any person.

- name_base_class
- name_derived_class
- name_program