Python course (40 hours)

This workshop is designed for persons who wish to learn the Python programming language, and want to use its prowess to develop general- purpose applications. No prior knowledge of Python is assumed. This is a hands-on class. It is comprised of lectures combined with practicing programs on a personal computer. The study will include Python best practices, standards and the ‘Pythonic’ ways of coding. In addition to the standard features of the language, an overview of what exists as code libraries will be given, and some important libraries will be explored further.

Objectives

* Upon completion the students should be able write general purpose programs in the Python programming language.
* Use some important modules such as os, sys, time, random etc.
* Use error recovery techniques.
* Utilize all the built-in data structures of Python (lists, tuples, dictionaries, sets, strings etc.)
* Write object-oriented programs.
* Be familiar with many built-in functions, and be able to use dir/help to search for unexplored options.
* Create their own modules, classes and methods.
* Extend built-in classes, use inheritance, data encapsulation and polymorphism.
* Create regular expressions and use them.
* Have enough exposure to Python GUI programming to trigger code.
* Be able to use graphics/animation and event-driven Python code.
* Use IDE’s of IDLE and Pycharm, for debugging and running applications.
* If time permits, learn about parallel processing, data visualization, writing games and network communication software in Python.

Prerequisites

The audience should be familiar with programming using at least one programming language, preferably also with object-oriented paradigm and principles (data encapsulation, data abstraction, inheritance and polymorphism).

The following sections describe the content of the course

Introduction

* History of the language.
* Advantages and disadvantages vs. other languages.
* Python versions 2/3
* Installing Python and its IDE’s IDLE , Pycharm
* Using Python’s help features
* The nature of interpreted language
* PEP8 conventions
* Program structure and well known standards
* Dynamic binding
* Python keywords

Work environments

* Command interpreter
* IDLE
* Pycharm and debugging options
* Customizing the IDE

Basic features of the language

* Code blocks / Indentations
* Program flow control
* Functions and methods
* Data conversions
* All types of comments
* Doc-strings and their usage
* String and list slicing
* Printing and formatting
* Logical operators
* Relational operators
* Conditional statements (if)
* Loops

Data containers

* Numeric Data Types
* String Data Type
* Boolean
* Lists
* Tuples
* sets
* Dictionaries

Advanced features

* File manipulations
* Lambda functions
* Navigation of the file system / directories
* Advanced sorting
* Iterators

Introduction to Important Modules

* Random
* os
* sys
* time
* NumPy
* SciPy
* Pygame
* PIL

Error recovery techniques

* Asserts
* Try/except blocks
* Exception classes and objects
* Pycharm debugger

File processing (Input/Output)

* Text vs. binary files
* Read, write, append
* Using files as iterators
* Use os and shutil modules to process files
* Simulate database with files with shelve module
* Unicode encoding for text files
* Low level file operations, seek, remove, copy etc.
* Using glob module for robust file processing

Regular expression usage

* Introduction
* Simple character matches
* Special / meta characters
* Character classes
* Quantifiers
* Grouping
* Greedy matching
* Specific python re methods
* Substitutions (replace)
* Compiling re
* Objects of re
* Findall method

Object oriented Python model

* The python class model
* Constructors
* Class variables vs. Instance variables
* Class methods
* Inheritance / Multiple inheritance
* Data hiding
* Polymorphism
* The pickle module

GUI – using Tkinter

* Basics of Python user interface
* Organizing controls
* Associating functionality to controls
* Text boxes, buttons
* Event driven code

Graphics programming

* Graphic windows and shapes
* Manipulate images
* Creating animation
* Color and sound control
* Using Pygame to create games
* Using other graphic tools

NumPy and SciPy

* Numpy arrays
* Multi-dimensional arrays
* Array calculations
* Matrix manipulation
* Use Matplotlib for data visualization
* Bar plots
* Line plots
* Linear algebra