Python

Nilesh Devdas

Course Introduction

ABOUT THE COURSE

Pre-Requisites

- Knowledge of basic programming
- Knowledge of logic
- Knowledge of Basic IDE/Editor
- Database Knowledge

System Setup

- Windows/Linux/Mac any operating System
- Python (To be installed)
- Python Editors
- MySQL for Database Connectivity

Course Outline

- An Overview of Python
- What is Python?
- Interpreted languages
- Advantages and disadvantages
- Downloading and installing
- Which version of Python
- Where to find documentation
- The python environment
- Structure of a Python script
- Using the interpreter interactively
- Running standalone scripts under Unix and Windows
- Getting Started
- Using variables
- String types: normal, raw and Unicode
- String operators and expressions
- Math operators and expressions
- Writing to the screen
- Command line parameters
- Reading from the keyboard
- Flow Control
- About flow control
- Indenting is significant
- The if and elif statements
- while loops
- Using lists
- Using the for statement
- The range() function
- Array types
- List operations
- list methods
- Strings are special kinds of lists
- Tuples
- Sets
- Dictionaries
- Working with Files
- Text file I/O overview
- Opening a text file
- Reading text files
- Raw (binary) data
- Using the pickle module
- Writing to a text file3
- Dictionaries and Sets
- Dictionary overview
- Creating dictionaries
- Dictionary functions
- Fetching keys or values
- Testing for existence of elements
- Deleting elements
- Functions
- Syntax of function definition
- Formal parameters
- Global versus local variables
- Passing parameters and returning values
- Sorting
- The sorted() function
- Alternate keys
- Multiple keys
- Lambda functions

Python Course Outline

- Errors and Exception Handling
- Dealing with syntax errors
- Exceptions
- Handling exceptions with try/except
- Cleaning up with finally
- Modules and Packages
- What is a module?
- The import statement
- Function aliases
- Packages
- Regular Expressions
- RE Objects
- Pattern matching
- Parsing data
- Subexpressions
- Complex substitutions
- RE tips and tricks
- Highlights of the Standard Library
- Working with the operating system
- Grabbing web pages
- Sending email
- Using glob for filename wildcards
- math and random
- Accessing dates and times with datetime
- Working with compressed files4
- An Introduction to Python Classes
- About o-o programming
- Defining classes
- Constructors
- Instance methods
- Instance data
- Class methods and data
- Destructors
- Data Frame Manipulation
- Data Acquisition
- Indexing, Filtering
- Sorting & Summarizing
- Descriptive Statistics
- Combining and Merging Data Frames
- Discretization and Binning
- String Manipulation
- Projects
- Default Modeling using Logistic Regression in Python
- Credit Risk Analytics using SVM in Python
- Intrusion Detection using Decision Trees & Ensemble Learning in Python
- Data Structures in Python
- Intro to Numpy Arrays
- Creating ndarrays
- Indexing
- Data Processing using Arrays
- File Input and Output
- Getting Started with Pandas
- Other Predictive Modelling Tools
- Intro to Machine Learning
- Random Forests
- Sklearn Library and Statsmodels

Introduce Yourself

- Name
- Experience
- Exposure to programming languages and Object Orientation

Breaks

- 15 Min Tea break in the first half
- 45 Mins Lunch Break
- 15 Mins Tea break in the second half

Agenda – Day1

- An Overview of Python
- What is Python?
- Interpreted languages
- Advantages and disadvantages
- Downloading and installing
- Which version of Python
- Where to find documentation
- The python environment
- Structure of a Python script
- Using the interpreter interactively
- Running standalone scripts under Unix and Windows
- Getting Started
- Using variables
- String types: normal, raw and Unicode
- String operators and expressions
- Math operators and expressions
- Writing to the screen
- Command line parameters
- Reading from the keyboard
- Flow Control
- About flow control
- Indenting is significant
- The if and elif statements
- while loops
- Using lists
- Using the for statement
- The range() function
- Array types
- List operations
- list methods
- Strings are special kinds of lists
- Tuples
- Sets
- Dictionaries
- Working with Files
- Text file I/O overview
- Opening a text file
- Reading text files
- Raw (binary) data

Introduction to Python

What is python

What is Python

- Python is programming language
- Python is interpreted programming language
- Introduced invented by Guido van Rossum
  - **Also an Benevolent dictator for life** (**BDFL**)
- Developed as an open source software
- Non Profit organization manages the software
- Python is dynamically typed language
  - No type checking of the code prior to running it unlike java
  - This is also known as duck
  - The idea is that it doesn't actually *matter* what type my data is - just whether or not I can do what I want with it.

Introduction to Python

- ❑ In python everything is an object
- ❑ A python program is written with any text editor
- ❑ “Python” or “CPython” is written in C/C++
- ❑ - Version 2.7 came out in mid-2010
- ❑ - Version 3.1.2 came out in early 2010
- ❑ “Jython” is written in Java for the JVM
- ❑ “IronPython” is written in C# for the .Net environment

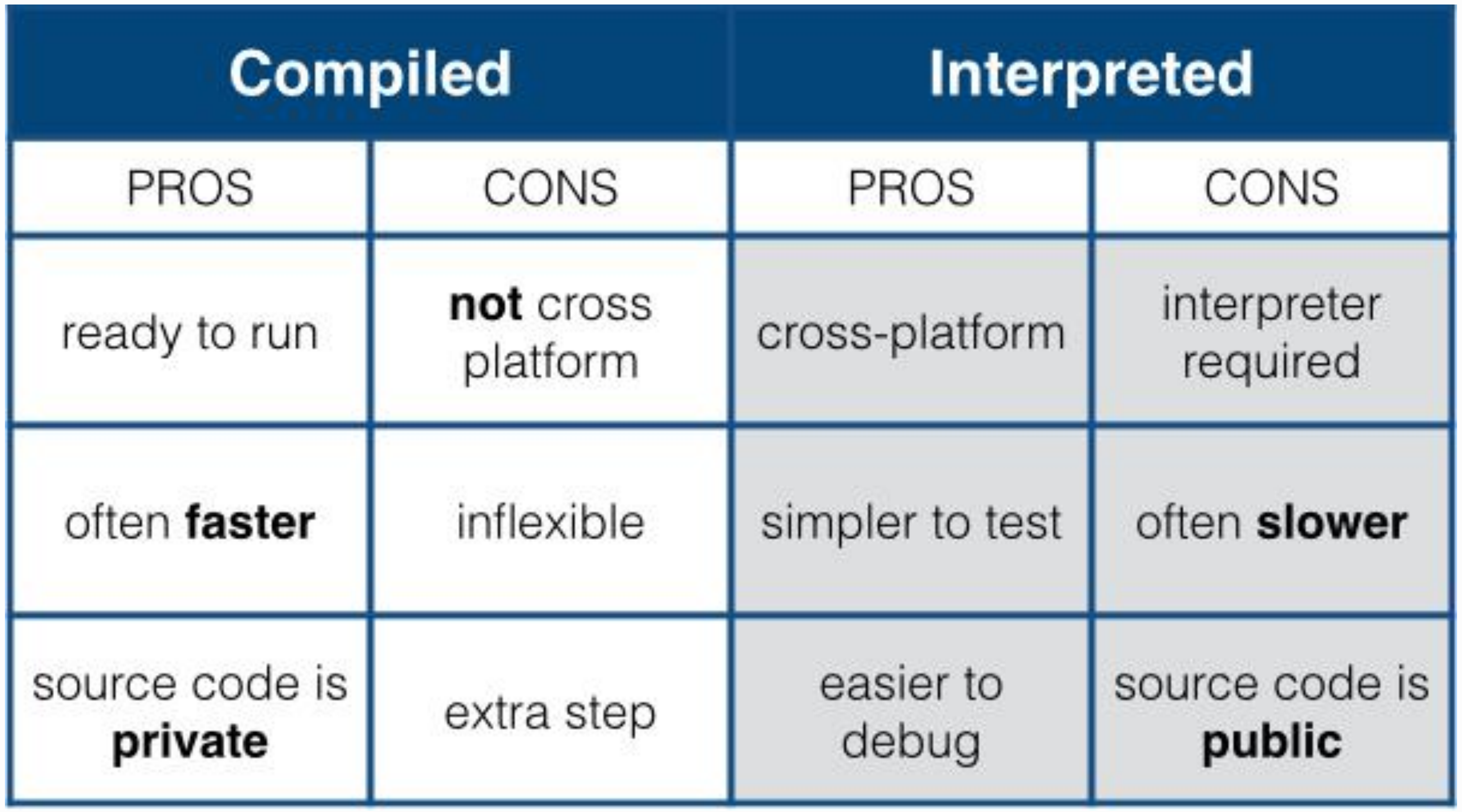
Where can we use python ?

- Python is general purpose programming language
- Can be used for
  - Web Applications
  - Desktop Applications
  - Workflows
  - Complex Mathematics
- Python works on different platforms
  - Windows
  - Mac
  - Linux
  - Raspberry PI

Why python ?

- ❑ Python provides interfaces to all major databases
- ❑ Python provides both structural as well as OOP
- ❑ Dynamically typed
- ❑ Supports GUI Programming
- ❑ Supports Web Programming
- ❑ Supports automatic garbage collection
- ❑ Can be easily integrated with c, C++ , java

Compiled v/s Interpreted



Compiled		Interpreted	
PROS	CONS	PROS	CONS
ready to run	**not** cross platform	cross-platform	interpreter required
often **faster**	inflexible	simpler to test	often **slower**
source code is **private**	extra step	easier to debug	source code is **public**

Advantages and Disadvantages

- **Advantages**
  - **Extensive Support Libraries**
  - **Integration Feature**
    - **Call C++, Java**
  - **Excellent IDE**
  - **Easy to Learn**
  - **Faster productivity**
- **Disadvantages**
  - Weak in mobile computing
  - Interpreted and hence large program may run slow
  - Dynamically typed may have design limitations
  - Database access layer is underdeveloped as compared to odbc and jdbc

Downloading and installing

- https://www.python.org/downloads/
  - All versions of python are available to be downloaded

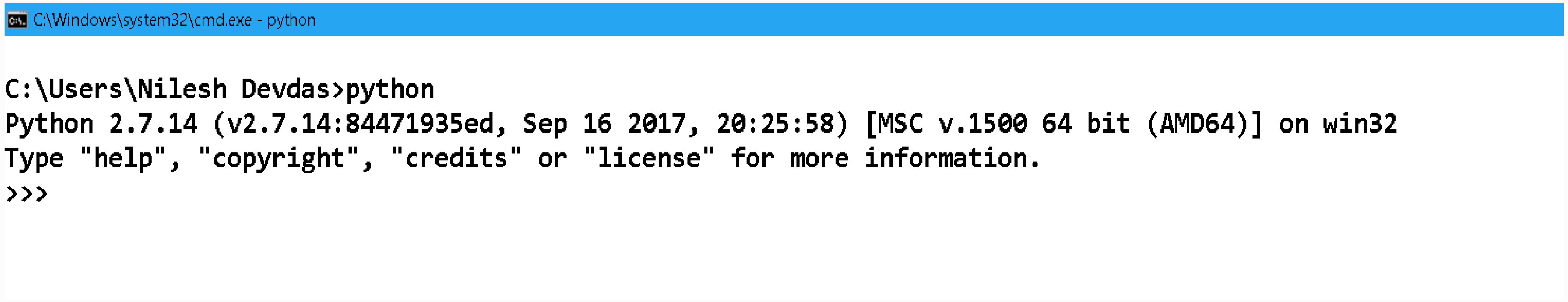
- Which Version of Python
  - 2.x or 3.x

Python 2.x v/s 3.x

- Python 3.0 was released in 2008.
- The final 2.x version 2.7 release came out in mid-2010, with a statement of extended support for this end-of-life release.
- The 2.x branch will see no new major releases after that.
- 3.x is under active development and has already seen over five years of stable releases

Development Environment

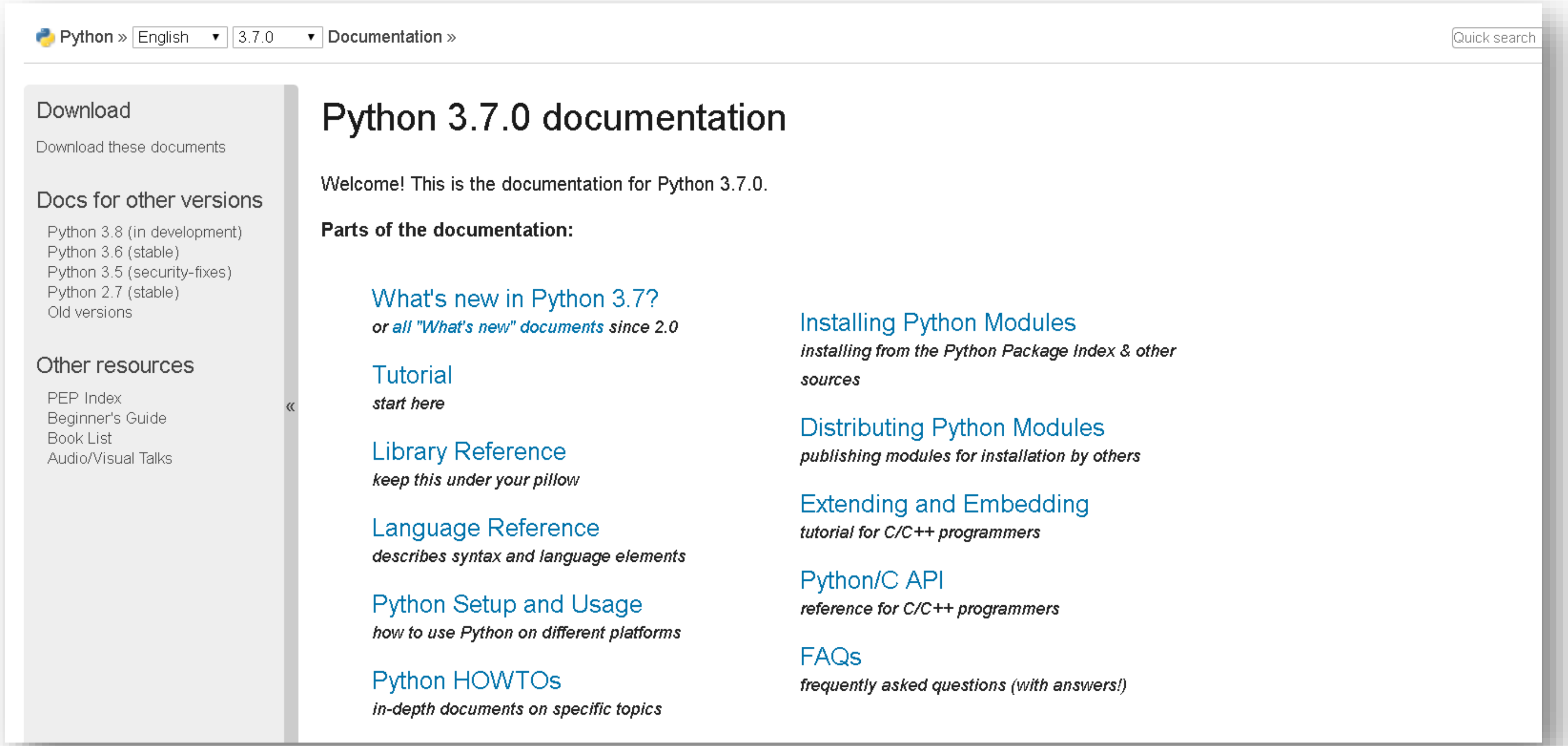
- Python Runtime
- Any Text editor
- Python can be developed with interactive Shell
  - You can type it in the running environment



```
C:\Users\Nilesh Devdas>python
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:25:58) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Where to find documentation

- https://docs.python.org/3/



Python » English ▾ 3.7.0 ▾ Documentation »

Quick search

Download

Download these documents

Docs for other versions

Python 3.8 (in development)
Python 3.6 (stable)
Python 3.5 (security-fixes)
Python 2.7 (stable)
Old versions

Other resources

PEP Index
Beginner's Guide
Book List
Audio/Visual Talks

Python 3.7.0 documentation

Welcome! This is the documentation for Python 3.7.0.

Parts of the documentation:

What's new in Python 3.7?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

Distributing Python Modules
publishing modules for installation by others

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

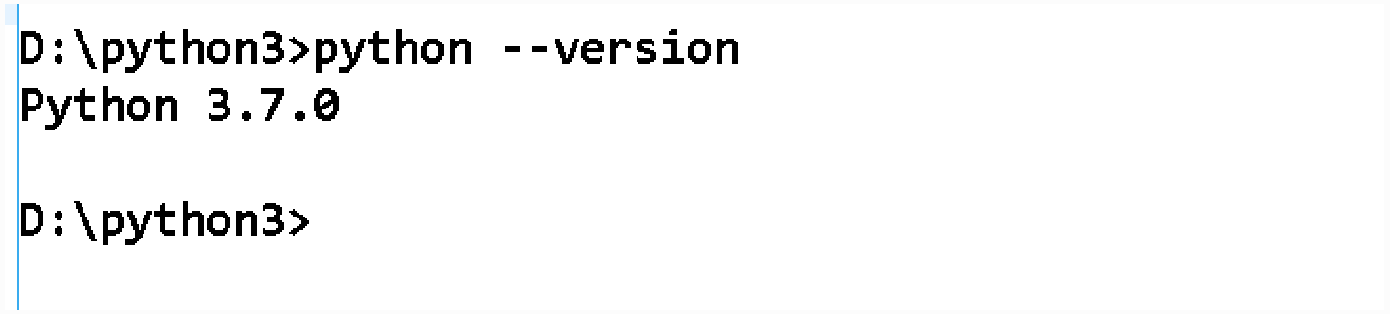
FAQs
frequently asked questions (with answers!)

REPL

- What is REPL ?
- Short for Read, Eval, Print and Loop.
  - **Read:** take user input.
  - **Eval:** evaluate the input.
  - **Print:** shows the output to the user.
  - **Loop:** repeat.
- We can type all kinds of input in the interactive shell:
- You can run a python program from the terminal or a python ide

The python environment

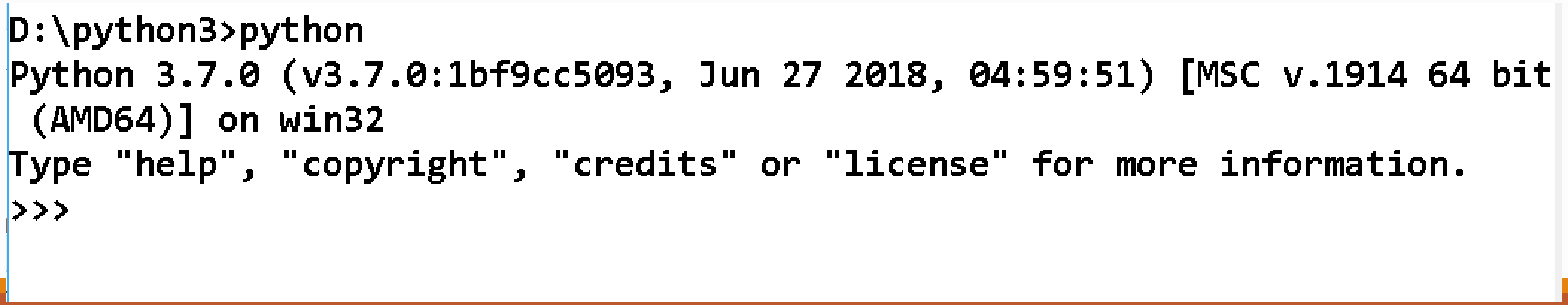
- Python environment
  - Requires python to be in path
- Version
  - python --version



```
D:\python3>python --version
Python 3.7.0

D:\python3>
```

- Python REPL
  - You can enter the python REPL by just typing python



```
D:\python3>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit
 (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Python

Python IDE

- PyCharm
- Spyder IDE
- Wing IDE
- Sublime Text
- VIM
- IDLE
- PYDEV

Structure of a Python program

Python Coding

- Comment
  - #   single line comment
  - “”” multi line comment
- Python files have extension .py
- print is to write output to screen
- Input is to take input from screen

Multi Line Statement

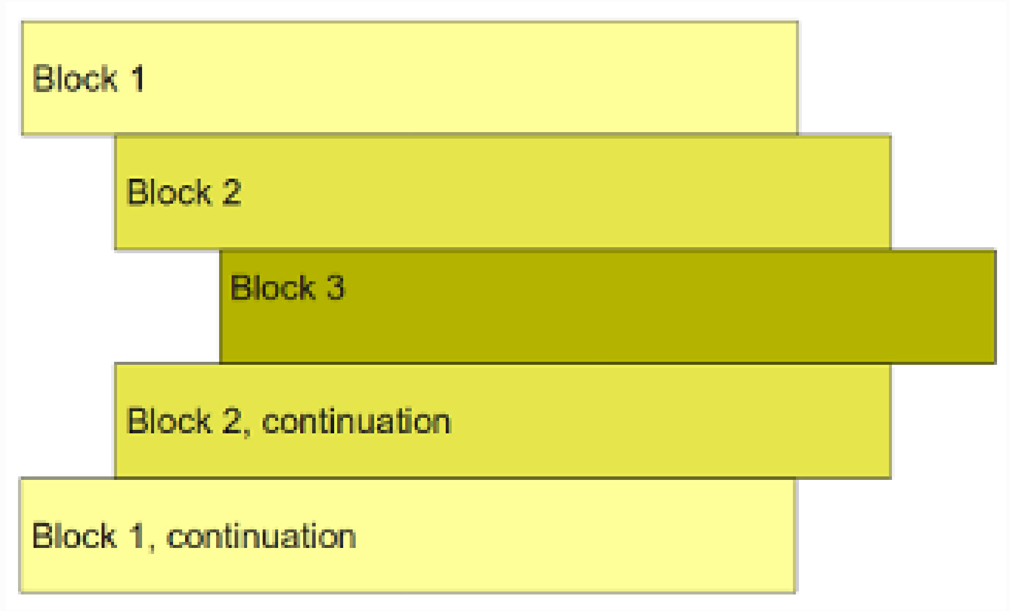
- Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue.
- For example – Items within [] , {} , () does not need to denote continuation character
- users = [‘Monday’, ‘Tuesday’ ,
- ‘Wednesday’]
- For Multiple statements in the same line you can use ‘;’ as the separator

Command Line arguments

- Python can accept command line arguments
- The command line arguments passed from the command line can be accessed with sys.argv
- The argv is a list of parameter and start with the program name passed
- Command line arguments

Python Indentation

- Indentation is used in python to delimit blocks the number of space is variable
- Block or compound statements should be terminated with a colon
- The semicolon is an optional statement



Python Reserved Words

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Variables

- Python is dynamically typed you need to declare variable types
- The declaration happens automatically when you assign the variable
- Variables can change type by simply assigning a new value
- Allows you to assign values to several variable simultaneously

Python Data Types

- Numbers are immutable objects in python
- The built in data types are
  - Integer (int)
  - Floating point number
  - Complex Number (Not much used in python)

Python Strings

- ❑ Python strings are immutable objects and cannot change their values
- ❑ You can update a string by assigning a variable to another string
- ❑ Python does not support character type
- ❑ Both single quote and double quote denote string
- ❑ String indexes start at 0 and

String formatting

- Python uses C-style string formatting to create new, formatted strings.
- The "%" operator is used to format a set of variables enclosed in a "tuple" (a fixed size list), together with a format string, which contains normal text together with "argument specifiers", special symbols like "%s" and "%d".
- String formatting can also be done using {}
  - Can use manual formatting by use of numbers
  - Can use auto formatting using positional placements

Python Operators

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Python Collections (Arrays)

- Python support 4 types of collection data in python programming language
  - List (Mutable Ordered Collections which allows duplicate members)
  - Tuple (Immutable Collection Ordered Collection allows duplicate Members)
  - Set (Unordered and unindexed, No duplicate members allowed)
  - Dictionary (Unordered , mutable and indexed , Duplicate indexes will be overwritten latest value honored)
  -

Functions

Function

- Functions are a convenient way to divide your code into useful blocks, allowing us to order our code, make it more readable,
- Functions can be used to reuse the functionality and save time.
- Also functions are a key way to define interfaces so programmers can share their code.
- Defining a Function in python

```
def   funcName(param1,,,,paramn):
            function Body
            function Body
```

- Function parameters can have value by default
- Default values to function Parameter will allow you to define values if not passed and invoke based on default values

Documenting a Function

- Function document can be writing using
  - def functionName(fnParams …. N)

    “Documentations xxxx Goes Here”

    Body of function

- Printing the document function
  - functionName.__doc__ will print the documentation on the REPL

Introduction to Classes

Classes

- Objects are an encapsulation of variables and functions into a single entity.
- Objects get their variables and functions from classes.
- Classes are essentially a template to create your objects.

Creating Classes

class *name*:

    "*documentation*"

    statements

-or-

class *name*(*base1*, *base2*, ...):

    ...

Most, *statements* are method definitions:

    def *name*(self, *arg1*, *arg2*, ...):

        ...

May also be *class variable* assignments

Multiple Inheritance in Python

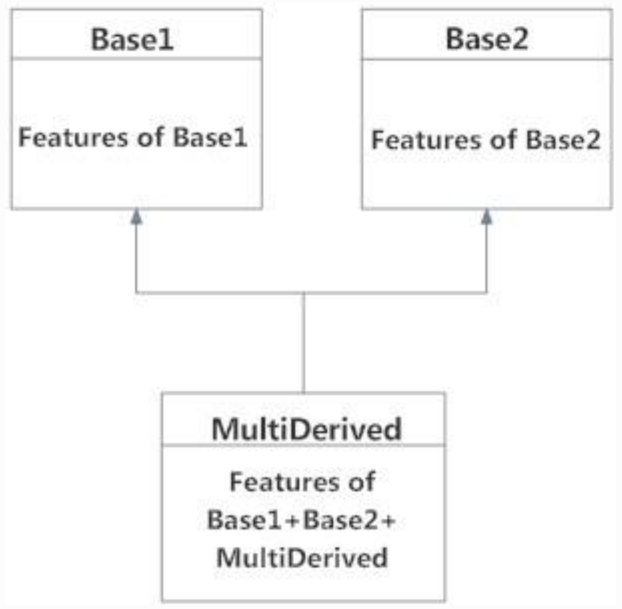
- Like C++, a class can be derived from more than one base classes in Python. This is called multiple inheritance.

- **Example**

```
class Base1:
      pass

class Base2:
      pass

class MultiDerived(Base1, Base2):
      pass
```



Multilevel Inheritance in Python

- we can also inherit form a derived class. This is called multilevel inheritance. It can be of any depth in Python.
- In multilevel inheritance, features of the base class and the derived class is inherited into the new derived class.

Constructors and Destructors

Python Constructors are created when the object is created

__init__(self)   Is used to define a constructor

Use () to create an object

Python Destructors are called when the object is deleted

__del__(self) is used to define a destructor

User del keyword to delete an object

MRO (Method Resolution Order)

- Every class in Python is derived from the class object. It is the most base type in Python.
- So technically, all other class, either built-in or user-defines, are derived classes and all objects are instances of object class.
- In the multiple inheritance scenario, any specified attribute is searched first in the current class. If not found, the search continues into parent classes in depth-first, left-right fashion without searching same class twice.
- __mro__ can be used to find the Method Resolution Order