# QAPYTH3

Python 3 Programming Instructors guide

**QAPYTH3 V3**

[Instructor’s Pack……… 2](#_Toc259514148)

[Course goals……………… 3](#_Toc259514149)

[Delegate prerequisites 3](#_Toc259514153)

[Course Format and Approach 3](#_Toc259514154)

[Timing…………………... 4](#_Toc259514155)

[0. Common introduction 5](#_Toc259514157)

[1. Introduction to Python 5](#_Toc259514158)

[2. Fundamental Variables 6](#_Toc259514159)

[3. Flow Control…………….. 6](#_Toc259514160)

[4. String Handling……………. 7](#_Toc259514161)

5. Collections…………………………………………………………………………….7

[6. Regular Expressions 8](#_Toc259514163)

[7. Data Storage and File Handling 8](#_Toc259514164)

[8. Functions…………… 9](#_Toc259514165)

[9. Advanced Collections 9](#_Toc259514166)

[10. Modules and Packages 10](#_Toc259514167)

[11. Classes and OOP……………………………… 10](#_Toc259514168)

[12. Error Handling and Exceptions 11](#_Toc259514169)

[13. Multitasking………………. 12](#_Toc259514170)

The Course Consultants are:

Donald Cameron QA [donald.cameron@qa.com](mailto:donald.cameron@qa.com)

Andy Smith QA [andy.smith@qa.com](mailto:andy.smith@qa.com)

Introduction

This document is intended to provide help to the existing and new lecturers of the “Python 3 Programming” course. The course layout, philosophy and structure will be discussed. This, together with the timing, potential difficulties and comments should help any lecturer in achieving good level of satisfaction when teaching this course.

Please email comments and suggestions concerning this Instructor's Guide to the course consultant.

This course is **not** an introduction to programming. It is targeted at people that are already experienced script designers and/or programmers.

Course-level information

## Instructor’s Pack

Instructors should be provided with the following items:

* QAPYTH3 Course Outline
* QAPYTH3 Exercise Guides
* QAPYTH3 Presentations
* QAPYTH3 Scripts&Files and Challenge Solutions
* QAPYTH3 Instructor Guide (this document)

If any of the items in the list above is missing, take it up with your normal contact with QA.

Errors detected since the printed edition of these manuals are mentioned in the pages which follow, and will be corrected in the next release. If you find any more please report them to the course consultant.

## Course goals

The main purpose of this course is to prepare the delegates for using the Python 3 programming language to write scripts.

## VM setup

during the 4-day virtual delivery, delegates will be given access to either LoD machines or have GTMPC access to virtual machines.

The latest QAPYTH3 image includes the following:

* Pycharm Community edition
* Python 3.10
* All updated labs and files needed
* Instructor machine will have all presentations, files, challenges and solutions needed in C:\instructor

## Delegate prerequisites

Required: experience of either of UNIX or Windows environments. The delegates should be familiar with any high-level language – which does not include HTML or SQL. The course spends minimal time on first principles and delegates that don’t have a programming past struggle.

## Course Format and Approach

The course format follows the standard, with a title slide, content, then finishing on a summary slide.

Additional slides often follow the summary, these cover obscure aspects, or some common pitfalls. I suggest you mention these to delegates and use them as back-up to answer questions.

The Appendices do not necessarily follow this format, and represent additional material which may be referred to, or even taught if there is sufficient time and interest. There are no labs for material in the Appendices.

The practical sessions can have a lot of time allocated. Currently the course is a four-day event, with plenty of time to explore areas of interest to the delegates.

During classes that run longer, particularly in bootcamp or bespoke deliveries, there is a file of challenges which should be used as exploratory time at the end of any day. Delegates should be encouraged to attempt these challenges throughout the course.

## Timing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time | Day 1 | Day 2 | Day 3 | Day 4 |
| 9:00 |  | Review | Review | Review |
| 9:15 |  | 5 – Collections | 8 - Functions | 11 - Practical |
| 9:30 | 0 - Introduction |  |  |  |
| 9:45 |  |  |  |  |
| 10:00 | 1 – Intro to Python |  | 8 - Practical | 12 – Error Handling and Exceptions |
| 10:15 |  | 5 – Practical |  |  |
| 10:30 |  |  |  |  |
| 10:45 | 1 – Practical |  | 9 - Advanced Collections | 12 - Practical |
| 11:00 |  |  |  |  |
| 11:15 | 2 – Fundamental Variables | 6 – Regular Expressions |  |  |
| 11:30 |  |  | 9 - Practical |  |
| 11:45 |  |  |  | 13 - Testing |
| 12:00 | 2 – Practical |  |  |  |
| 12:15 |  |  |  |  |
|  | ***Lunch*** | ***12:30 - 13:30*** |  |  |
| 13:30 | 3 – Flow Control | 6- Practical | 10 – Modules and Packages |  |
| 13:45 |  |  |  | 13 - Practical |
| 14:00 |  |  |  |  |
| 14:15 | 3 - Practical |  |  |  |
| 14:30 |  | 7 – Data Storage and File Handling | 10 - Practical |  |
| 14:45 |  |  |  | 14 -Multitasking |
| 15:00 | 4 – String Handling |  |  |  |
| 15:15 |  |  | 11 – Classes and OOP |  |
| 15:30 |  | 7 - Practical |  | 14 - Practical |
| 15:45 |  |  |  |  |
| 16:00 | 4 – Practical |  |  |  |
| 16:15 |  |  |  |  |
| 16:30 |  |  | 11 - Practical | Conclusions and evaluations |
| 16:45 |  |  |  |  |

# Chapter Summary

## 0. Common introduction

You may wish to display the course contents, and briefly walk through the subjects covered. You may also wish to introduce the course materials, and the three manuals. Now is a good time for personal introductions, and to check prerequisites. It helps to get the delegates to state the operating system they use. This allows pitching the discussions for the rest of the course. For example, if there is nobody with Linux or UNIX, you can virtually skip mentioning it.

## Introduction to Python

## Content

What is Python?  
Why Python?  
Zen of Python

Python Interactive Sessions

IPython and Jupyter  
Python Scripts

Python help

Example Python scripts

Modules

Functions and built-ins

#### **Labs**

The purpose of these labs is to get familiar with the environment.

Please ensure you point out to delegates the labs directory, which, as well as solutions, contains some code templates and test data. The notes directory contains all the code from the slides.

Don't spend too long on these labs, make sure everyone can at least create and run a script. Encourage fast-trackers to browse the documentation.

## 2. Fundamental Variables

**Content**

Python Objects

Python Variables

Type specific methods

Augmented assignments

Python types

Python lists

Python tuples

Python dictionaries

Python operators

Python reserved words

**Labs**

## The purpose of these labs is to experiment with some of the basic variable types within Python and to use some of their operations

## 3. Flow Control

**Content**

Python conditionals

What is truth?

Boolean and logical operators

Chained comparisons

Sequence and collection tests

Object types

While loops

For loops

Conditional expressions

Unconditional closedown

**Labs.**

The purpose of these labs is to use the flow control structures of Python and to gain familiarity in coding based on indentation! That does take a little practice. We’ll also be using a couple of modules from the Python standard library.

## 4. String Handling

**Content**

Python 3 Strings

The print function

Cooking strings

String concatenation

“Quotes”

String methods

String tests

String formatting

Literal string interpolation

Slicing a string

String methods – split and join

**Labs**

The purpose of this lab is to consolidate string manipulation in Python. This includes further practise at general Python constructs, such as loops.

### 5. Collections

**Content**

Python types – reminder

Generic built-in functions

Useful tuple operations

Python lists

Tuple and list slicing

Adding items to a list

Removing items from a list

Sorting

List methods

Sets

Set operators

Python dictionaries

Dictionary methods

View objects

**Labs**

The purpose of these labs is to understand the use and syntax of containers in Python 3. We’ll also compare different ways to access a list.

6. Regular Expressions

**Content**

Python Regular Expressions

Elementary extended RE meta-characters

Regular expression objects

Regular expression substitution

Matching alternatives

Anchors

Class shortcuts

Repeat quantifiers

Parenthesis groups

Back-references

**Labs**

The purpose of this lab is to become familiar with some of Python's regular expression tools and continue practise with Python syntax and string handling.

7. Data Storage and File Handling

**Content**

File objects

Reading files

Writing files

Standard streams

File tricks

Random access

Pickles

Shelves

Compression

**Labs**

The purpose of this lab is to use some of the Python 3 file handling methods, as well as the pickle and gzip modules.

## 8. Functions

**Content**

Python functions

Function parameters

Variadic functions

Assigning default values

Named parameters

Annotations

Returning objects

Variables in functions

Nested functions

Function documentation

Lambda functions

**Labs**

The purpose of this lab is to write your own user-written functions

## 9. Advanced Collections

**Content**

Advanced functions – filter

List comprehensions

Sets and dictionary comprehensions

Lazy lists

Generators

Copying collections

**Labs**

The purpose of this lab is to extend the use of functions from the previous chapter and to use some advanced techniques to streamline code

## 10. Modules and Packages

**Content**

What are modules and packages?

How does python find a module?

Multiple source files

Importing a module

Importing names

Directories as a package

Writing a module

Module documentation

The ‘main’ trick

**Labs**

## The purpose of these labs is to write and call our own modules. The files created in this lab will be used in future labs, and it may be pertinent to provide learners with a working copy of a file at the end of the lab if they have not completed it.

## 11. Classes and OOP

**Content**

Using objects

Duck typing

Python OO

Simple classes

Defining classes

Defining methods

Constructing an object

Special methods

Operator overloading

Properties and decorators

inheritance

**Labs**

The purpose of this lab is to build a simple class using Python's object orientation. The first exercise will take you through the process of creating a simple file class step-by step. Most of the code is provided for you. The second exercise uses inheritance and is a little more challenging – you will have to figure out some of the code yourself. Subsequent exercises are optional and are suitable only for experienced programmers.

## 12. Error Handling and Exceptions

**Content**

Writing to stderr

Exception handling

Exception syntax

Exception arguments

The finally block

Order of execution

Python 3 exception hierarchy

Assert

Raise

The raise statement

Raising our own exceptions

Getting tracebacks

Labs

The purpose of this lab is to embed exception handling in a module environment. The lab is built on the exercise completed after chapter 10 and will need to start with a working copy of those modules 13. Testing

Content

What is Software Testing

Types of testing

Manual Vs Automatic

Docstrings

Docstring testing

Automated testing

Unit testing

The assert statement

Test runners

Test scripts

Pytest

Labs

The purpose of this lab is to test functions with both doctest and unittest

14. Multitasking and AsyncIO

**Content**

Family life

Creating a process in Python

Using the subprocess module

Subprocess.run

Capturing output of Popen

Basic threads in Python

Using the Multiprocessing module

Queue objects

Why Async code?

Async language

Basic async

Synchronous Vs asynchronous

Waiting for external data

Labs

The purpose of this lab is to run external programs, in this case other Python scripts using a variety of methods.