Real Time College

Course: Programming Python

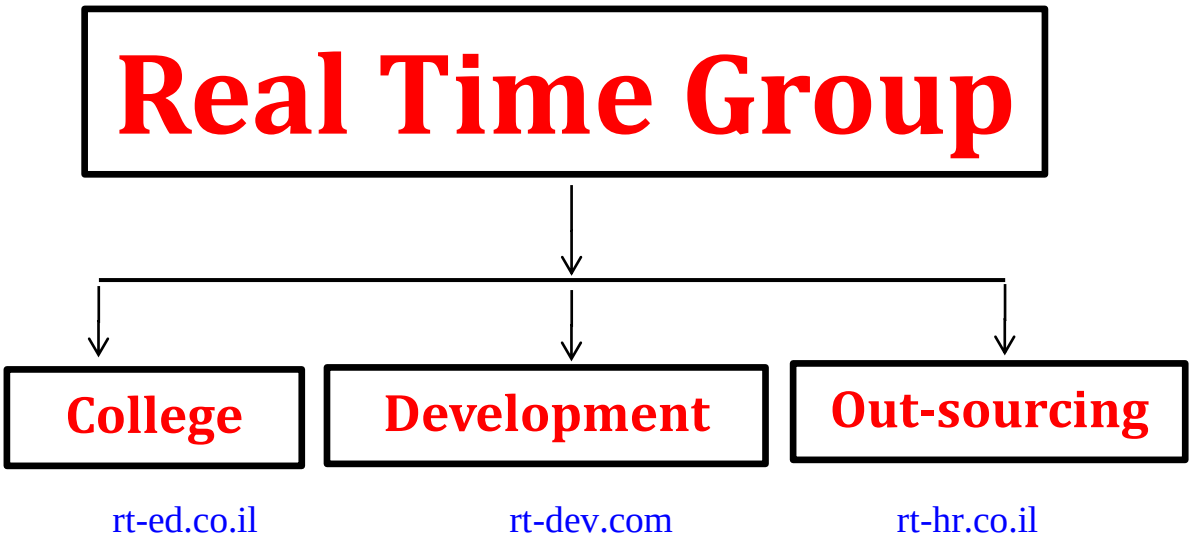
Duration: 90 Hours

Hands-On-Training: 60%

Real Time Group is a multi-disciplinary dynamic and innovative Real-Time O.S. and Embedded Software Solutions Center, established in 2007.

Providing Bare-Metal and Embedded Linux solutions, professional services and consulting, end-to-end flexible system infrastructure, outsourcing, integration and training services for Hardware, Software and RT-OS \ Embedded Systems.

The company is divided into the following three Divisions:



Training Division:

Professional Training Services for Hardware, Software, RT-OS and Embedded systems industries.

We provide the knowledge and experience needed to enable professional engineers to Develop, Integrate and QA Hardware, Software and Networking Projects.

In order to insure experience, all courses are practical – hands-on-training.
The latest Development, QA and Automation equipment which are adopted by the industry are used.

All students are supplied with Development-Boards for home-work and course projects.

Course Overview:

This Python programming course targets anyone who wishes to develop Python based applications, focusing on Python 3, aiming to provide you with the knowledge and experience needed in order to program object oriented real industry applications.

The course is suited for deferent fields in the industry such as Mathematical Probability and Statistics, Algorithms, Desktop and Mobile applications, Automation and Mechanics, etc.

The course is divided into Three main parts:

1**. Python Basics**
Starts by introducing the basic concepts of programming, development tools, environment, debugging tools and essentials of python programming: data types, creating variables, input and output, decision making and repetition, iterators, list comprehension and functions, object oriented programming, inheritance, exception handling and using data structures ,Files and Directories Using and Building Modules and more.

2. **Advanced Python**
The second part of the course covers some more advanced topics such as:
Multi-Threading, Managing SQL Data Base, Socket Programming, Interacting with GUI, Design Patterns etc.
Designed for experienced users with knowledge in software development in OOP.

3. **Python Web Development with Django** (**Based on Time constraints)
Django is the most popular and mature Python web development framework.
During the course you'll get an intro to Django, how to use it in order to easily build smart and efficient Web apps with less code.

The course includes dozens of class and home hands-on exercises and practices.

Who should attend:

- Suitable for delegates with existing experience of the Windows \ Linux operating systems .
- Anyone whom needs to develop Python based Scripts\Web applications.
- Hardware and software engineers who need to quickly test their as a development platform.

Prerequisite:

- A basic level of computer literacy is expected, using a PC running Windows.
- No prior knowledge or experience in software development is needed.

Notes:

** The second & third parts are based on the time available and some subjects it might not be included.

** The Curriculum can be modified based on client needs. Companies may choose only the desired topics from the Syllabus.

** Companies which implement Python on an embedded system, an ARM Evaluation card will be provided for each student throughout the course - Used for class hands-on-training and homework exercises.

Programming Python

1. Python Basics

1. **Introduction to Python**
   a. Why Python?
   b. Performance using python
   c. Running Python interactively
   d. Using Python scripts
   e. Python help
   f. Lunching the Python interpreter
   g. Python Versions .
   h. Launching the Python Command Prompt
   i. Commenting Python
   j. Launching Python programs
   k. Integrated Development Environments.
   l. python community
   m. Python Modules
   n. Python Functions

2. **Variables and Data Types:**
   a. Is Python Object Oriented ?
   b. Python Syntax
   c. Python Object Types (int, float, str, bool etc)
   d. Python variables
      i. Variable names
      ii. Type specific methods
      iii. Operators and types
      iv. Python types
      v. Numeric types
      vi. Switching types
   e. Python Numbers

3. **Advanced data types:**
   a. Lists methods:
      i. List usage
      ii. Adding List Elements
      iii. Mutability
      iv. Methods

   b. String methods:
      i. basic string operations
      ii. Indexing and slicing strings
      iii. String Formatting
      iv. Combining and Separating Strings
      v. Regular Expressions .

   c. Dictionaries:
      i. making a dictionary
      ii. Basic operations
      iii. Dictionary details .

   d. Tuples
      i. Why Use Tuples?
      ii. Sequence Unpacking
      iii. Methods .

4. **Python Statements**
   a. Assignment
   b. Expressions/Calls
   c. if-else-elif, nested conditions
   d. Chained comparisons
   e. Sequence and collection tests
   f. Conditional statements
   g. while Loops
   h. for Loops
   i. using ranges
   j. using lists
   k. nested loops
   l. break and continue Statements
   m. Mathematical \ Logical operators,
   n. Using the return value
   o. Recursions

5. **Python String Handling**
   a. printing function
   b. Concatenating strings
   c. “Simple String Manipulation in Python
   d. Format Strings
   e. Flexible Pattern Matching with Regular Expressions”
   f. String methods
   g. String tests
   h. Slicing a string
   i. split and join

6. **Defining and Using Functions:**
   a. Python functions
   b. Function parameters
   c. Arguments
   d. Assigning default values to parameters
   e. Built-in functions
   f. User-defined functions
   g. Keyword parameters
   h. Returning values from a function
   i. Local Variables in functions
   j. Global variables and the global statement
   k. Nested functions
   l. Anonymous (lambda) Functions

7. **Collections**
   a. Useful tuple operations
   b. Python lists
   c. Tuple & list slicing
   d. Adding items to a list
   e. Removing items by position
   f. Removing list items by content
   g. Sorting
   h. Miscellaneous list methods
   i. List methods
   j. Sets
   k. Exploiting sets
   l. Set operators
   m. Python dictionaries

8. **Modules**
   a. What are modules?
   b. How does Python find a module?
   c. Directories as packages
   d. Using modules
      i. Importing a module
      ii. Importing names
   e. Writing a module
   f. Multiple source files
   g. Module documentation

9. **Debugging in python: pdb**
   a. The Python Debugger
   b. Debugger Commands
   c. Break points
   d. Step over / Step into / Step out / Resume
   e. Variables / Watches

10. **File management with Python**
   a. Files and Streams
   b. Creating New files
   c. Reading Files
   d. Writing to files
   e. Seeking \ Iterating Through Files
   f. Filter programs – file-input module
   g. Standard streams
   h. Random access
   i. Python Pickle Protocols
   j. Compression in Python

11. **Error & Exception Handling**
   a. Runtime Errors
   b. Catching Exceptions:
      i. try and except
      ii. Exception syntax
   c. try…except…else…finally
   d. Exception Class Hierarchy
   e. Exception arguments
   f. Controlling warnings
   g. The finally block
   h. Raising Exceptions: raise
   i. assert

2. Advanced Python

12. **Classes and Objects:**
    a. Object-Oriented Programming
    b. Using objects
    c. A simple class
    d. Defining classes
    e. Constructing an object
    f. Class attributes,
    g. Class methods,
    h. Class pointers and instances
    i. Class Interfaces,
    j. Implementations and Polymorphism.
    k. Class Inheritance and Abstract Classes.
    l. New-style classes
    m. Properties and decorators

13. **The Python Standard Library:**
    a. The Standard Library
    b. Operating System interfaces
    c. System specific attributes
    d. The socket module
    e. Other modules

14. **Socket Networking**
    a. Client/Server Programming.
    b. Creating a new socket in python
    c. Receiving and transmitting data through the socket

15. **Working with SQL in Python**
    a. Connecting to a Database
    b. Using SQL to Query a Database
    c. Pulling Data from Data Base
    d. Inserting data to Data Base

16. **Multithreading In Python**
    a. Creating Threads in python
    b. Killing Threads
    c. Threading modules
    d. Family life
    e. Creating a process from Python
    f. Waiting for a child process
    g. The subprocess.Popen class
    h. Running a basic process
    i. Passing data through a pipe
    j. Difference between Processes and threads
    k. Synchronizing objects in threading
        - usage of Locks for Synchronization
    l. Using the multiprocessing module
    m. Queuing objects

17. **GUI - Graphic User Interface:**
    a. Widgets & Events
    b. Check Buttons, Radio Buttons, Menus, ...
    c. C Code With Python Extension

3. Introduction to Web Development with Django

(Based on Time constraints)**

18. **Introduction to Django**
    a. What is Django?
    b. Django and Python
    c. Django' s take on MVC: Model, View and Template
    d. DRY programming: Don't Repeat Yourself
    e. How to get and install Django