Python Programming

Duration: 28 hours

Prerequisites: Prior scripting experience or knowledge of fundamental programming concepts.

Course Description: This hands on Python programming course shows how to rapidly develop and maintain effective Python programs. The course includes thorough coverage of Python syntax, built in data types and control constructs. The course takes a practical approach to creating and organizing Python programs using functions, packages, modules and classes as part of Python's object-oriented paradigm. Attendees will use regular expressions to rapidly process data captured from users and from the file system.

Attendees will learn how to use Python to create scripts that manipulate data, automate tasks, perform error handling and store and retrieve data by using relational databases. Students will be able to create Python scripts that assist with system administration.

Comprehensive hands on exercises are integrated throughout to reinforce learning and develop real competency.

Students Will Learn

- Understanding Python
- Writing Expressions in Python
- Working With Numbers
- Working with Strings
- Building Structured Data with Lists and Tuples
- Building Structured Data with Dictionaries and Sets
- Controlling the Flow of a Python Program
- Creating Modular Code with Functions
- Input / Output
- Object-Oriented Programming with Python
- Using Modules in Python Programs
- Exception Handling
- Using Regular Expressions in Python
- Database Programming with Python

Overview

Introduction to Python 3

- Origin and Goals of Python
- Overview of Python Features
- Getting and Installing Python
- Accessing Python Documentation: Python Enhancement Proposals (PEP)
- Python's Strengths
- Using Python with Other Programming Languages

Language Fundamentals

- Python's Lexical Analyzer

Using Python

- Executing Python Programs from the Command Line
- Python Command Line Options
- Environment Variables that Influence Python
- Creating Python GUI Applications
  - Standalone vs. Web-Enabled Interfaces
- The Python Standard Library

Flow Control Constructs

- `if/elif/else` Statements

- Using Whitespace to Structure Programs
- Identifiers and Keywords
- Python's Execution Model
  - Naming Objects and Binding
- Python's Data Model
  - Immutable and Mutable Objects
  - Values
  - Types
- Creating and Using Variables

Expressions

- Unary and Binary Arithmetic Operations
- Comparison and Boolean Operations
- Conditional Expressions
- Lambda Expressions
- Order of Operations and Operator Evaluation
- Expression Lists
- Assignment Operations

Using the String Object

- Using ASCII and Unicode Strings
- Manipulating Strings with String Methods
- Using the `format()` Function to Format Strings
- Using Escape Sequences
- Working with Raw Strings

Arrays, Collections and Dictionaries

- Sequenced Data Structures
  - Arrays
  - Collections
  - Dictionaries
- Creating and Accessing Lists
- Manipulating Lists
- Creating and Accessing Tuples
- Understanding the Differences Between Lists and Tuples
- Using Dictionaries to Create Data Records
- Manipulating Dictionaries Using Dictionary Methods
- Creating Sets
- Performing Set Operations
  - Union

- Creating Loops with `while` and `for`
- Understanding Iterators
- Returning Values with `return` Statements
- Loop Modification with `break` and `continue`
- Returning Generator Iterators with the `yield` Statement
- Retrieving Iterators with `next()`

Exception Handling

- Types of Python Exceptions
- Handling Exceptions with `try/except/finally`
- Triggering Exceptions with `raise`
- Defining New Exception Types
- Implementing Exception Handling in Functions, Methods and Classes
- Working with the Regular Expression Error Exception

Organizing Code

- Defining Functions
- Calling Functions
- Creating Anonymous Functions
- Altering Function Functionality with Decorator Functions
- Creating Classes with the `class` Statement
- Creating Objects as Class Instances
- Using Preexisting Classes as the Basis of a New Class
- Using Modules to Group Related Functions, Classes and Variables
- Locating and Importing Modules
- Using Packages to Group Modules Together

Working with Arguments

- Passing Arguments to Functions by Reference and by Value
- Defining Functions with Required Arguments
- Defining Functions with Default Arguments
- Defining Flexible Functions that Take Variable Length Arguments

Regular Expressions

- Regular Expression Syntax
- Using Regular Expressions in Python
- Altering Regular Expression Processing with Regular Expression Modifiers
- Using Regular Expression Operators
- Scanning Through Strings Using the `search()` and

    - Intersect
    - Difference
- Differences Between Sets and Dictionaries
- Using Generators to Return Iterators

Object Oriented Programming Concepts

- The Object Oriented Programming Paradigm
- Encapsulating Information
- Classes vs. Instances of Objects
- Built-in Class Attributes
- Implementing Class Inheritance
- Using Objects in Code

Data Management

- Embedding SQLite Databases in Applications
- Best Practices for Data Management
- Storing Data in Local Databases
- Discussing and Understanding the DB API
- Understanding and Using Common SQL Statements
- Connecting to a SQLite Database
- Using Cursors to interact with Data from a Database
- Implementing Error Handling with Database Connections

`match()` Methods
- Creating Reusable Patterns by Using the `compile()` Method

I/O Handling

- Sending Output to STDOUT Using the `print()` Method
- Reading Input with the `input()` Method
- Creating File Objects with the `open()` Method
- Controlling File Access Modes
- Working with File Object Attributes
- Closing File Objects with the `close()` Method
- Reading and Writing to File Objects with `read()` and `write()`
- Using File Processing Functions from the OS Module

Related Bootcamp

Track	Duration	Price
Python Programmer	49 hours	$1,790

Contact Us

Address: 6 Hemlock Drive Chelmsford, MA 01824

Phone: 978.250.4983

Mon - Thur: 9 am - 5 pm EST
Fri: 9 am - 4 pm EST

E-mail: info@developer-bootcamp.com

Copyright© 2025 Developer Bootcamp