Hewlett Packard Enterprise

HPE Digital Learner Python Content Pack

HPE Content Pack number	CP026
Content Pack length	20 Hours
Content Pack category	Category 1
Learn more	**View now**

In this course, you will learn basic to advanced programming with Python, including Python for data science, developing artificial intelligence and machine learning solutions with Python.

Why HPE Education Services?

- IDC MarketScape leader 5 years running for IT education and training*
- Recognized by IDC for leading with global coverage, unmatched technical expertise, and targeted education consulting services*
- Key partnerships with industry leaders OpenStack®, VMware®, Linux®, Microsoft®, ITIL, PMI, CSA, and SUSE
- Complete continuum of training delivery options—self-paced eLearning, custom education consulting, traditional classroom, video on-demand instruction, live virtual instructor-led with hands-on lab, dedicated onsite training
- Simplified purchase option with HPE Training Credits

Audience

Software developers, IT analysts, data scientists

Content Pack objectives

This Content Pack provides the information necessary to:

- Identify the basics and philosophy of Python
- Set up Python
- Write a basic Python program with built-in data types, loops and conditionals
- Describe Python documentation and file handling
- Explain Python iteration and iterables and exception handling
- Describe the Django and TurboGears frameworks for developing web application
- Apply basic data science fundamentals to Python

*Realize Technology Value with Training, IDC Infographic 2037, Sponsored by HPE, October 2017

Detailed Content Pack outline

Python Fundamentals

Python: The Basics

Python has a unique culture and community which values its core philosophy, expressed as a series of aphorisms, and is available with a few key presses from any Python installation. In this course, you will learn the basics of Python and its philosophy, setting up Python and writing a basic program with built-in data types, loops and conditionals.

- Describe the features of the Python programming language and how and where it is used
- Describe the philosophy of Python
- Recognize reasons to choose one version of Python over the other
- Install Python 3 on Windows
- Install Python 3 on Mac OS X and Linux
- Evaluate the major IDEs available for Python
- Use whitespace to lay out a Python program into functional code blocks
- Recognize the Python REPL – read, evaluate, print loop
- Create and execute a "Hello World" application with Python
- Get and manipulate user input from the command line with Python
- Create a module and import a module in Python
- Use the int data type in Python and recognize its characteristics
- Use the float data type in Python and recognize its characteristics
- Perform basic math functions, such as addition, subtraction, multiplication and division, and use the math module
- Use the bool data type in Python and recognize its characteristics
- Describe sequence types and use the str type in Python
- Use the bytes type in Python
- Use the bytearray type in Python
- Use the list type in Python
- Use the tuple type in Python
- Use slicing on sequence types in Python
- Use the range function and work with range objects in Python
- Use the set type in Python and describe its characteristics
- Use the dict type in Python and describe its properties
- Construct a while loop in Python
- Construct a for loop in Python
- Use the if statement in Python to control program flow
- Write a Python program to reverse user input

Python: Classes and Modules

Python is a lot more than a scripting language and can be used to create OOP applications using classes or using a functional paradigm. This course covers some of the ways Python programs can execute. You will learn about building your own custom functions and classes, and documentation and file handling.

- Create and import a module at the Python REPL
- Define a function in Python
- Describe the difference in operation between Python scripts, programs and modules
- Run a module as a script using the __name__ == __main__ syntax in Python
- Create a main function that takes command line arguments in Python
- Describe the relationship between classes and types in Python
- Create a class definition and describe the structure in Python
- Write a class initializer method in Python
- Write and use class instance methods in Python
- Write and use static methods in Python
- Use inheritance and describe the semantics in Python
- Describe class properties in Python
- Describe how inheritance affects properties in Python
- Write a class that implements operator overloading in Python
- Write docstrings in Python
- Write comments in Python
- Describe best practice for documenting Python code as set out in PEP 8
- Read text files in Python
- Write data in Python
- Write large files in Python
- Read binary data in Python
- Write binary data in Python
- Write a Python class to represent a vector

Python: Iterations and Exceptions

Iterations and exceptions are nuanced in Python and it is important to fully understand how they work in order to produce quality software. In this course, you will learn about comprehensions, a powerful, concise syntax for creating iterable objects. The course also covers iteration and iterables, and exception handling.

- Create a list comprehension in Python
- Create a nested comprehension in Python
- Use the zip() function in a generator in Python
- Create a set comprehension in Python
- Create a dictionary comprehension in Python
- Describe the function of iter(), next() and StopIteration() in Python iteration
- Use the map() function in an iteration in Python
- Use the filter() function in an iteration in Python
- Use functools.reduce() to iterate over an iterable
- Implement a custom iterable class in Python
- Implement an iterable using consecutive integer indexing in Python
- Implement an iterable using the extended iter() function
- Create a simple generator in Python
- Create a lazy generator in Python and understand its characteristics
- Create a recursive generator in Python
- Write a basic exception handler in Python to catch all exceptions
- Write an exception handler in Python to catch a specific error and recognize the reason why catching all errors is bad practice
- Describe the inheritance hierarchy of exceptions in Python and how to catch multiple exception types using a base type
- Raise an exception using a payload and retrieve a payload when handling an error
- Create a custom exception class in Python
- Access and manipulate traceback objects for an exception in Python
- Use assertions in a Python program
- Use implicit and explicit chaining of exceptions in Python
- Create an iterable data type that handles exceptions in Python

Python: Web Application Development

Frameworks provide a way to create powerful web applications in Python. In this course, you will learn about the Django and TurboGears frameworks for developing web applications.

- Describe the key features of the Django framework
- Install and configure the Django framework
- Create a Django project
- Configure the Django web server
- Create a sample Django app
- Incorporate views and templates in an app
- Use Django to include data in a Python web application
- Utilize forms in a Python web application
- Describe the key features of the TurboGears framework
- Install and configure the TurboGears framework
- Incorporate TurboGears templates into a Python web app
- Incorporate TurboGears views into a Python web app
- Create and use a controller in a Python web app
- Describe rendering and how it is used in TurboGears
- Use TurboGears to include data in a Python web application
- Use RESTful URLs in TurboGears
- Describe the key features of Flask
- Create a basic Flask application
- Incorporate a template into a Flask app
- Work with web forms in a Flask project
- Connect to and retrieve data using a Flask app
- Use Django to create a view for a Python web application

Python: web2py and Test-Driven Development

The web2py framework lets you build scalable, secure and portable web applications. Testing provides a way to mitigate bugs and errors before the release of Python applications. In this course, you will learn about the web2py framework and the testing frameworks included in Python and their use.

- Describe the key features of the web2py framework
- Explore the core concepts of web2py
- Demonstrate the use of Views in a web2py application
- Describe how to use the database abstraction layer in a web application
- Incorporate forms and form validation into a web2py application
- Send email with web2py
- Implement authentication and authorization in a web2py application
- Create and run some basic web2py services
- Provide an overview of testing in Python
- Describe the key features of the doctest module
- Embed a basic doctest in a docstring
- Use doctest directives and handle raised exceptions in a doctest test
- Provide an overview of unit testing in Python
- Install Python Mocker and use it in a unit test
- Use the unittest framework to run tests on Python code
- Create and use a unittest test fixture
- Use the subtest() context manager to distinguish unittest test iterations
- Use nose to run unit tests on Python application code
- Perform unit testing of a project with pytest
- Describe the key features of the Robot framework
- Perform web automation testing using Robot with the Selenium 2 test library
- Use unittest module to write and run tests on Python code

Python: Data Science Fundamentals

Python is a high level programming language that has code readability and simplicity as its primary design goals. Coupled with a few key APIs, it also becomes a very powerful data analysis tool. This course will cover basic data science fundamentals and apply them to Python.

- Demonstrate how to set up and use Anaconda for Python
- Describe the key features of Jupyter as well as how to install it
- Work with the Notebook server and dashboard
- Detail the key characteristics of Numpy, including how to install and use
- Create an example that utilizes NumPy arrays
- Detail the key characteristics and how to install pandas
- Perform basic data manipulation using pandas
- Create a data visualization using matplotlib
- Use scikit-learn to perform data normalization
- Perform supervised learning by using the scikit-learn library to perform optical recognition of hand-written digits
- Install and use the ArcGIS Python API in a Python app
- Use NLTK and Python to tokenize words and sentences
- Analyze an ego network using Python and Networkx
- Perform web scraping using BeautifulSoup 4 in Python
- Install and configure PySpark for Python
- Perform basic data manipulation using pandas

Mastering DevOps with Python

Python Fundamentals

The Python programming language, its syntax and standard libraries have undergone some changes between versions 2 and 3. In this course, you will learn about the key fundamental concepts and features of Python version 3.

- Recall syntax considerations for data types, numbers, and operations
- Work with strings and print formatting
- Declare functions, accept arguments and return values from functions
- Work with tuples and arrays in Python
- Work with dictionaries and lists in Python
- Use loops and branching in Python
- Convert dates and times between formats
- Implement a module and include it in another source file
- Work with file input and output operations
- Create and instantiate a class
- Define and initialize class variables and methods
- Use default arguments to implement function overloading
- Define and instantiate a derived class
- Overload an operator in a class
- Read a file and load it into a Python class

Python Data Processing

Working with databases, regular expressions and XML data are common tasks for DevOps in Python. This course demonstrates how to perform these common tasks.

- Create a database and a user in MySQL from Python
- Build a table and perform CRUD operations from Python
- Perform a transactional query with commit in Python
- Handle errors returned from database queries in Python
- Work with regular expression search syntax using the standard RE module
- Perform a search and replace using various regular expression modifiers
- Parse XML data with SAX using the standard XML module
- Use the DOM with the standard XML module to parse XML data
- Read data from XML and save it in a MySQL database

Python Applications

In this course, you will learn about web programming and GUI programming in Python.

- Activate a virtual environment and install web programming prerequisites with pip
- Create a web application project using Flask
- Define route mappings in a web application
- Create a web application template using the Jinja2 language
- Parse HTTP headers in a web application
- Respond to various HTTP methods in a web application
- Use cookies in a web application
- Run a web application through a Web Server Gateway Interface (WSGI) using the Apache web server
- Create a Tkinter window with some common widgets
- Modify Tkinter widgets using common attributes
- Use Tkinter geometry managers to organize widgets
- Start a WSGI service and process an HTTP request

Learn more at

www.hpe.com/ww/digitallearner
www.hpe.com/ww/digitallearner-contentpack

Follow us:

CP026 A.00, January 2019

Hewlett Packard Enterprise