Practical Introduction to Python Training

Duration: **3 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

Overview

82% of learners would recommend this course.

- **Learn to build and run Python 3 applications from the ground up.**
- **Designed for programmers developing new Python 3 projects or migrating existing Python 2 applications.**
- **Develop practical programming skills through hands-on exercises in real coding environments.**

Python 3 Programming Training is a technical course designed to introduce participants to the **Python 3 programming language** and develop the skills needed to create applications using its core and advanced features.

The course combines **instructor-led presentations** with **hands-on programming exercises**. Practical activities are delivered in a **Microsoft Windows environment**, although the programming techniques covered are equally applicable to other platforms, including **Linux**.

Throughout the programme, participants explore **Python 3 syntax**, variables, data types, control flow, functions and collections. The course also covers **file handling**, **regular expressions**, **modules**, **packages** and **Object-Oriented Programming (OOP)**.

More advanced topics include **error and exception handling**, **multitasking**, **multiprocessing** and the **Python Standard Library**.

Participants also examine the **differences between Python 2 and Python 3**, gaining an understanding of considerations involved in moving existing applications to Python 3.

Prerequisites

Previous Python experience is not required. However, participants should already be familiar with at least **one programming language**.

Experience in the following areas would be beneficial:

- Another scripting language, such as **Perl or PHP**
- Previous exposure to **Object-Oriented Programming (OOP)**

Practical Introduction to Python Training

Duration: **3 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

Who Should Attend

This course is suitable for:

- **Programmers starting new software projects with Python 3**
- **Developers migrating existing applications from Python 2 to Python 3**
- Software professionals with experience in other programming languages who want to develop their **Python 3 programming skills**

What You Will Learn

By completing the **Python 3 Programming Training**, participants will develop the knowledge and practical programming skills needed to create applications using Python 3.

By the end of the course, learners will be able to:

- Use the **Python interactive interpreter** to write and execute Python 3 programs.
- Understand the fundamental **language elements of Python 3**.
- Make effective use of Python's extensive **libraries and modules**.
- Recognise the key **differences between Python 2 and Python 3**.
- Identify simple and complex **variable types** and select suitable data structures.
- Use **Python 3 operators and built-in functions**.
- Understand and apply **procedural control flow** in Python 3.
- Implement **file input/output operations**, including persistent data objects.
- Create well-structured and efficient code using **functions and modules**.
- Apply **Object-Oriented Programming techniques** in Python 3.
- Build reliable applications with **error handling and reporting**.
- **Execute and control other programs** from Python.
- Use **multiprocessing** to benefit from multi-core processors.

This course helps participants strengthen their **Python 3 programming capabilities**, transfer existing programming knowledge into the Python environment and prepare to develop new applications through **practical coding exercises**.

Practical Introduction to Python Training

Duration: **3 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

Outline

Chapter 1: Introduction to Python 3

This chapter introduces the **Python programming language**, its key characteristics and the different ways Python programs can be executed.

- **What is Python?**
- **What is Python 3?**
- Reasons for using Python
- **Performance limitations**
- The **Python community**
- Running Python interactively
- **Python scripts**
- Python help facilities
- **Anatomy of a Python script**
- **Modules**
- **Functions and built-in functions**

Chapter 2: Fundamental Variables

This chapter examines Python variables, objects and the fundamental data types used in Python programs.

- Python as an **Object-Oriented language**
- **Python variables**
- Variable naming conventions
- **Type-specific methods**
- Operators and data types
- **Augmented assignments**
- Python data types
- **Type conversion**
- Introduction to **lists**
- Introduction to **tuples**

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Introduction to **dictionaries**

Chapter 3: Flow Control

Participants learn how to manage program execution using conditional statements, loops and logical expressions.

- **Python conditionals**
- **Indentation**
- Understanding truth values
- **Boolean and logical operators**
- **Chained comparisons**
- Sequence and collection tests
- Object types
- Introduction to **exception handling**
- **while loops**
- Loop control statements
- **for loops**
- Using **enumerate()**
- Counting for loops
- Processing multiple lists with **zip()**
- **Conditional expressions**
- **Unconditional flow control**

Chapter 4: String Handling

This chapter explores the creation, formatting and manipulation of text data in Python 3.

- **Python 3 strings**
- The **print() function**
- Preparing and processing strings
- **String concatenation**
- Working with quotation marks
- **String methods**

BILGINÇ IT Academy

info@bilginc.com

bilginc.com

- String tests
- **String formatting**
- Additional formatting techniques
- **String slicing**
- **split() and join() methods**

Chapter 5: Collections

This chapter focuses on Python's core collection types and the operations available for managing them.

- Review of Python data types
- Useful **tuple operations**
- **Python lists**
- Tuple and list slicing
- **Extended iterable unpacking**
- Adding items to lists
- Removing list items by position
- Removing list items by content
- **Sorting**
- List methods
- **Sets**
- Working effectively with sets
- **Set operators**
- **Python dictionaries**
- Dictionary values
- Removing dictionary items
- Dictionary methods
- **View objects**

Chapter 6: Regular Expressions

Participants explore how **regular expressions** can be used for pattern matching, searching and text

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

replacement.

- **Python regular expressions**
- Basic extended regular expression metacharacters
- **Regular expression objects**
- Regular expression substitution
- Regular expression splitting
- **Matching alternatives**
- **Anchors**
- Character class shortcuts
- **Flags**
- Repeat quantifiers
- **Quantifiers**
- Parentheses and groups
- **Back-references**
- **Global matches**

Chapter 7: Data Storage and File Handling

This chapter introduces file input/output operations, persistent data storage and basic database access.

- **File objects**
- Reading files into Python
- Advanced reading techniques
- Filter programs using the **fileinput module**
- **Binary mode**
- Writing data to files
- **Standard streams**
- Additional file-handling techniques
- **Random access**
- Persistent storage using **pickle**

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Pickle protocols
- Data storage with **shelve**
- **Compression**
- Overview of database interfaces
- **Using SQLite from Python**

Chapter 8: Functions

This chapter examines how to define functions, manage parameters and organise reusable code.

- **Python functions**
- Function parameters
- **Variadic functions**
- Default parameter values
- **Named or keyword parameters**
- Enforcing keyword-only parameters
- Returning objects from functions
- **Variables within functions**
- **Nested functions**
- Variables in nested functions
- **Function documentation**
- **Lambda functions**
- Using lambda as a sorting key
- **Lambda expressions in re.sub()**

Chapter 9: Advanced Collections

Participants learn advanced techniques for working with Python collections and managing data efficiently.

- Advanced list operations using **filter()**
- **List comprehensions**
- **Set and dictionary comprehensions**
- **Lazy lists**

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- **Generators**
- Generator objects, **next()** and coroutines
- Using comprehensions as generators
- **Collection copying challenges**
- Copying collections using slicing
- **Deep copying with deepcopy()**

Chapter 10: Modules and Packages

This chapter focuses on organising Python programs across multiple files and creating reusable modules and packages.

- **Modules and packages**
- Working with multiple source files
- How Python locates modules
- **Importing modules**
- Importing names from modules
- Using directories as packages
- **Writing modules**
- Module documentation
- **Testing modules**
- The **Python debugger**
- The **Python profiler**
- Distributing libraries with **distutils**

Chapter 11: Introduction to Classes and OOP

This chapter introduces **Object-Oriented Programming** concepts and class-based application development in Python.

- **Classes and OOP**
- Object-Oriented terminology
- **Object-Oriented Programming principles**
- Using objects

BILGINÇ IT Academy

info@bilginc.com

bilginc.com

- **Defining classes**
- **Defining methods**
- Constructing objects
- **Special methods**
- Operator overloading special methods
- **Properties**
- **Properties and decorators**
- **Class methods**
- **Inheritance**
- Inheritance terminology

Chapter 12: Error Handling and Exceptions

Participants learn how to detect, handle and report errors in Python applications.

- Writing output to **stderr**
- Managing warnings
- **Exception handling**
- Exception syntax
- Handling multiple exceptions
- Exception arguments
- The **finally block**
- Order of execution
- **Python exception hierarchy**
- A common exception-handling mistake
- The **raise statement**
- Creating custom exceptions
- The **assert statement**

Chapter 13: Multitasking

This chapter explores how Python can execute and manage other processes, work with threads and use multiprocessing capabilities.

BILGINÇ IT Academy

info@bilginc.com

bilginc.com

- **Process families and relationships**
- Creating processes from Python
- Examples of older process interfaces
- **Waiting for child processes**
- The **subprocess module**
- The **subprocess.Popen class**
- Running a basic process
- Capturing process output
- **Passing data through pipes**
- **Processes and threads**
- Basic threading in Python
- **Threading synchronisation objects**
- Challenges associated with threads
- The **multiprocessing module**
- **Queue objects**

Chapter 14: The Python Standard Library

The final chapter introduces a range of standard Python modules and explores their use in different programming scenarios.

- **The Python Standard Library**
- Example of **converting Python 2 scripts to Python 3**
- The **Pretty Printer** utility
- Operating system interfaces: **os and related modules**
- System-specific attributes: **sys**
- **Signal handling with signal**
- Converting signals into exceptions
- **Configuration files**
- The **configparser module**
- The **datetime module** and related utilities

Practical Introduction to Python Training

Duration: **3 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

- The **platform module**
- External function interfaces using **ctypes**
- The **socket module**
- **__future__**
- Other standard library modules