

Beginning Python

A Two-Day Intensive Course

FA-0117 · Two-Day

Software Development

Introduction

Are you ready to kickstart your journey into the world of programming? Join us for our two-day intensive "Beginning Python" course, designed to equip you with in-demand skills that will elevate your career prospects. Our expert trainers, with extensive industry experience, have crafted a comprehensive curriculum that covers Python's core concepts and applications. By participating in this course, you will not only gain a solid foundation in Python but also learn practical techniques used by professionals in the field.

In today's competitive job market, having a strong understanding of Python can open up doors to numerous opportunities across various industries such as data analytics, web development, automation, and artificial intelligence. Our course is specifically tailored to help you skill up and become job-ready. Through hands-on exercises and real-world examples, you will learn how to write efficient Python code and solve complex problems. Don't miss this chance to invest in your future and boost your employability by mastering Python with our experienced trainers. Register for the "Beginning Python" course today, and get ready to embark on an exciting and rewarding career path!

Prerequisites

The prerequisites for the Beginning Python course are minimal, making it suitable for participants with little or no prior programming experience. However, having a basic understanding of the following concepts will be helpful:

1. Familiarity with computer usage: Participants should be comfortable using a computer, navigating the operating system, and installing software.
2. Basic problem-solving skills: A fundamental understanding of logical thinking and problem-solving techniques will be beneficial when learning programming concepts.
3. Fundamental knowledge of mathematics: A basic understanding of arithmetic operations and mathematical concepts, such as variables and functions, can aid in grasping programming concepts more effectively.

4. General understanding of programming concepts (optional): Although not mandatory, having a general idea of programming languages, algorithms, and data structures can be helpful in learning Python more efficiently.
5. Motivation and willingness to learn: A positive attitude and an eagerness to learn new skills are essential for participants to make the most of this course.

Learning Outcome

Upon completion of the Beginning Python course, participants can expect to have achieved the following learning outcomes:

1. Understand the significance and applications of Python in various industries.
2. Set up and configure a Python development environment, including installing Python and using an Integrated Development Environment (IDE).
3. Demonstrate proficiency in Python syntax, indentation, and basic programming constructs.
4. Utilize variables, data types, and operators to perform operations and create Python programs.
5. Implement control structures, such as conditional statements and loops, to control the flow of a program.
6. Design and utilize functions to modularize and reuse code effectively.
7. Master the use of Python data structures, including lists, tuples, dictionaries, and sets, to store and manipulate data.
8. Perform file handling operations, such as opening, closing, reading, and writing to files.
9. Implement error handling using exceptions, try-except blocks, and custom exceptions to create robust and error-free programs.
10. Understand the concept of modules and packages, explore built-in modules, and create custom modules.
11. Leverage Python community resources and popular libraries for further learning and skill enhancement.
12. Apply the learned concepts to develop practical, real-world Python applications.
13. Demonstrate a solid foundation in Python programming, increasing employability and competitiveness in the job market.

Training outline

1. Day 1:

1. Introduction (1 hour)
 - 1.1. Welcome and course overview

- 1.2. Python history and significance
- 1.3. Python use cases and applications
- 2. Setting up the environment (1 hour)
 - 2.1. Installing Python
 - 2.2. Introduction to Integrated Development Environments (IDEs)
 - 2.3. Configuring your IDE (e.g., PyCharm, Visual Studio Code)
- 3. Python basics (2 hours)
 - 3.1. Syntax and indentation
 - 3.2. Variables and data types (strings, integers, floats, booleans)
 - 3.3. Basic operators (arithmetic, assignment, comparison, logical)
 - 3.4. Input and output functions (input, print)
 - 3.5. Exercise: Simple calculator program
- 4. Control structures (1.5 hours)
 - 4.1. Conditional statements (if, elif, else)
 - 4.2. Looping structures (while, for)
 - 4.3. Loop control statements (break, continue)
 - 4.4. Exercise: Guess the number game
- 5. Functions (1.5 hours)
 - 5.1. Defining and calling functions
 - 5.2. Parameters and arguments
 - 5.3. Return values
 - 5.4. Exercise: Creating a function for temperature conversion

2. Day 2:

- 6. Data structures (2 hours)
 - 6.1. Lists (creation, indexing, slicing, methods)
 - 6.2. Tuples (creation, indexing, immutability)
 - 6.3. Dictionaries (creation, keys, values, methods)
 - 6.4. Sets (creation, methods, set operations)

6.5. Exercise: Phone book application

7. File handling (1 hour)

7.1. Opening and closing files (open, close)

7.2. Reading and writing to files (read, write, append)

7.3. File modes and exceptions

7.4. Exercise: Simple file-based note-taking application

8. Error handling (1 hour)

8.1. Understanding exceptions

8.2. Try and except blocks

8.3. Raising exceptions (raise, assert)

8.4. Finally block and custom exceptions

8.5. Exercise: Adding error handling to previous exercises

9. Modules and packages (1 hour)

9.1. Introduction to modules and importing

9.2. Exploring built-in modules (random, datetime, math)

9.3. Creating and using custom modules

9.4. Introduction to packages and installing via pip

9.5. Exercise: Creating a custom module for a utility function

10. Wrap-up and next steps (1 hour)

10.1. Summary of key concepts

10.2. Python community and resources

10.3. Introduction to popular libraries (NumPy, Pandas, Requests)

10.4. Suggestions for further learning and practice

10.5. Q&A and closing remarks