800.674.3550
Info@AppliedTechAc.com
AppliedTechnologyAcademy.com

PCAP Certified Associate in Python Programming Training

Python Institute Authorized Training · Applied Technology Academy

PCAP™ – Certified Associate Python Programmer certification focuses on the Object-Oriented Programming approach to Python, and shows that the individual is familiar with the more advanced aspects of programming, including the essentials of OOP, the essentials of modules and packages, the exception handling mechanism in OOP, advanced operations on strings, list comprehensions, lambdas, generators, closures, and file processing.

LEVEL	DURATION	EXPERIENCE	AVERAGE SALARY	LABS
Intermediate	**2 Days**	**2 years; Python**	**$90,000**	**Yes**

Course Overview

Becoming PCAP certified ensures that the individual is fully acquainted with all the primary means provided by Python 3 to enable her/him to start her/his own studies, and to open a path to the developer's career.

Course Outline

- **Module 1: Control and Evaluations**
  - Objectives covered by the module
  - basic concepts: interpreting and the interpreter, compilation and the compiler, language elements, lexis, syntax and semantics, Python keywords, instructions, indenting
  - literals: Boolean, integer, floating-point numbers, scientific notation, strings
  - operators: unary and binary, priorities and binding
  - numeric operators: ** * / % // + –
  - bitwise operators: ~ & ^ | << >>
  - string operators: * +
  - Boolean operators: not and or
  - relational operators (== != > >= < <=), building complex Boolean expressions
  - assignments and shortcut operators
  - accuracy of floating-point numbers
  - basic input and output: input(), print(), int(), float(), str() functions
  - formatting print() output with end= and sep= arguments
  - conditional statements: if, if-else, if-elif, if-elif-else
  - the pass instruction
  - simple lists: constructing vectors, indexing and slicing, the len() function
  - simple strings: constructing, assigning, indexing, slicing comparing, immutability

800.674.3550
Info@AppliedTechAc.com
AppliedTechnologyAcademy.com

  - building loops: while, for, range(), in, iterating through sequences
  - expanding loops: while-else, for-else, nesting loops and conditional statements
  - controlling loop execution: break, continue

- **Module 2: Data Aggregates**
  - Objectives covered by the module
  - strings in detail: ASCII, UNICODE, UTF-8, immutability, escaping using the \ character, quotes and apostrophes inside strings, multiline strings, copying vs. cloning, advanced slicing, string vs. string, string vs. non-string, basic string methods (upper(), lower(), isxxx(), capitalize(), split(), join(), etc.) and functions (len(), chr(), ord()), escape characters
  - lists in detail: indexing, slicing, basic methods (append(), insert(), index()) and functions (len(), sorted(), etc.), del instruction, iterating lists with the for loop, initializing, in and not in operators, list comprehension, copying and cloning
  - lists in lists: matrices and cubes
  - tuples: indexing, slicing, building, immutability
  - tuples vs. lists: similarities and differences, lists inside tuples and tuples inside lists
  - dictionaries: building, indexing, adding and removing keys, iterating through dictionaries as well as their keys and values, checking key existence, keys(), items() and values() methods
- **Module 3: Functions and Modules**
  - Objectives covered by the module
  - defining and invoking your own functions and generators
  - return and yield keywords, returning results, the None keyword, recursion
  - parameters vs. arguments, positional keyword and mixed argument passing, default parameter values
  - converting generator objects into lists using the list() function
  - name scopes, name hiding (shadowing), the global keyword
  - lambda functions, defining and using
  - map(), filter(), reduce(), reversed(), sorted() functions and the sort() method
  - the if operator
  - import directives, qualifying entities with module names, initializing modules
  - writing and using modules, the __name__ variable
  - pyc file creation and usage
  - constructing and distributing packages, packages vs. directories, the role of the __init__.py file
  - hiding module entities
  - Python hashbangs, using multiline strings as module documentation
- **Module 4: Classes, Objects, and Exceptions**
  - Objectives covered by the module
  - defining your own classes, superclasses, subclasses, inheritance, searching for missing class components, creating objects

800.674.3550
Info@AppliedTechAc.com
AppliedTechnologyAcademy.com

- class attributes: class variables and instance variables, defining, adding and removing attributes, explicit constructor invocation
- class methods: defining and using, the self parameter meaning and usage
- inheritance and overriding, finding class/object components
- single inheritance vs. multiple inheritance
- name mangling
- invoking methods, passing and using the self argument/parameter
- the __init__ method
- the role of the __str__ method
- introspection: __dict__, __name__, __module__, __bases__ properties, examining class/object structure
- writing and using constructors
- hasattr(), type(), issubclass(), isinstance(), super() functions
- using predefined exceptions and defining your own ones
- the try-except-else-finally block, the raise statement, the except-as variant
- exceptions hierarchy, assigning more than one exception to one except branch
- adding your own exceptions to an existing hierarchy
- assertions
- the anatomy of an exception object
- input/output basics: opening files with the open() function, stream objects, binary vs. text files, newline character translation, reading and writing files, bytearray objects
- read(), readinto(), readline(), write(), close() methods

Intended Audience

This course is designed for students who have a foundational competence in Python and want to continue developing important skills for programming efficiently in the language.

This course is also designed to assist students in preparing for the Python Institute's PCAP™ – Certified Associated in Python Programming (PCAP-31-03) credential.

Prerequisites

To ensure your success in this course, you should have knowledge of Python basics, including syntax, data types, data structures, conditional statements, loops, functions, and basic exception handling. You should also be fairly comfortable writing simple Python code.

You can obtain this level of skills and knowledge by taking the Certified Entry-Level Python Programmer (PCEP™) course from Logical Operations.

Follow-On Courses

- Python for Network Automation
- Python for Reverse Engineers
- Threat Hunting with Python
- Python for Data Sciences

Ready to enroll? Call 800.674.3550 or email Info@AppliedTechAc.com — private team cohorts, GSA MAS purchasing, military credentialing funding, VA GI Bill and VR&E and student financing available.