

PYTHON 3 PROGRAMMING

This test is designed to measure a programmer's knowledge of core components of the Python language through version 3. The Secure Evaluation Mode and Remote versions of this test will contain a minimum of 54 questions and will require a typical time of 64 minutes. The Secure Interview Mode and QwikChek versions will provide as many questions as the test taker can answer within an approximate 40 minute time limit. The CoreChek version will contain 27 questions and will require a typical time of 32 minutes.

Following is a description of each sub-topic on the test:

Python Dictionaries assesses understanding of the ability to define and use dictionary (mapping) types in a Python 3 program, including setting values, using default values, and iterations. **Core**

Python File I/O evaluates knowledge of external file interaction using Python 3, including reading, writing, opening, and closing files and sockets. **Core**

Python Functions tests Python 3 function definition syntax, parameter passing, and returning results. Lambda expressions are also covered. **Core**

Python Lists and Tuples demonstrates knowledge of the Python 3 list and tuple sequence aggregate types, including accessing values, simple functions, and common usage including sorting and searching. **Core**

Python Modules and Imports tests understanding of how to use external modules in Python 3, how modules are defined, and how to use modules to enforce data hiding practices.

Python Regular Expression Usage measures the definition and application of Python 3 regular expressions to locate, extract, and change text in context.

Python Classes tests knowledge of object-oriented programming using Python 3 and object-oriented features like inheritance and polymorphism as well as subclassing, static methods, and private attributes. **Core**

Python Control Flow determines knowledge of the Python flow control statements (i.e., while, if, for, break, range, pass, etc.) and what instructions are executed or evaluated depending on the value of the contextual data or the resulting expressions. **Core**

Python Exception Handling evaluates skills needed to guarantee that errors reported during the execution of a Python 3 program are properly processed to ensure proper application cleanup while maintaining useful error reporting. **Core**

Python Comprehensions tests knowledge of how to derive new lists, dictionaries, sets, and iterators using Python 3's comprehension syntax, as well as the use of functional tools like `map` and `filter` to build more expressive comprehensions. **Core**

Python Strings evaluates knowledge of how to manipulate Python 3 strings, including slicing operations and Unicode encoding. **Core**

Python Concurrency assesses knowledge of Python 3's facilities for concurrent code execution, including inter-thread communication and safety, such as are found in the `threading`, `multiprocessing`, and `asyncio` modules.

Python Generators and Coroutines analyzes knowledge of generators and coroutine function syntax, how control flow differs from normal functions, and their application as iterators and in asynchronous programming.

Python Inheritance determines ability to define a class that inherits all the methods and properties from another class within Python 3 code. This is part of object-oriented programming in Python

Python Precedence and Associativity measures understanding of the order in which an expression is evaluated that has multiple operators of the same precedence in Python 3 code.

PYTHON 3 PROGRAMMING

Python Formatting and Decorators tests knowledge of dynamically altering the functionality of a function, method, or class without having to directly use subclasses or change the Python 3 source code of the function being decorated.

Python Encapsulation assesses ability to hide an object from view outside of the object's definition within Python 3 code. This is part of implementing object-oriented programming (OOP) in Python.

Python Functional Programming determines knowledge of and using functional programming patterns in Python 3 code. Includes avoiding state changes as much as possible and writing functions that take and return instances representing objects in an application.

Core indicates the sub-set of sub-topics offered in the CoreChek assessment version.