98-381
Introduction to Programming Using Python

Exam number: 98-381

Exam title: **Introduction to Programming Using Python**

Publish date:

GUID:

Language(s) this exam will be available in: ENU, JPN, CHT, CHS, ESM, PTB, FRA, DEU, KOR

Audience (IT professionals, Developers, Information workers, etc.): Beginning programming students

Technology: Python

Credit type (example: MCSA)**: MTA**

Exam provider (VUE, Certiport, or both)**:** **Both**

Exam Design

Audience Profile
Candidates for this exam should be able to recognize and write syntactically correct Python code, recognize data types supported by Python, and be able to recognize and write Python code that will logically solve a given problem.

Candidates are expected to have had, at minimum, instruction and/or hands-on experience of approximately 100 hours with the Python programming language, be familiar with its features and capabilities, and understand how to write, debug, and maintain well-formed, well documented Python code.

Language version: Python 3.6

Prerequisite skills for this exam:
Core Algebra (Algebra I) (typical US 9th/10th grade level)

Skills measured

Perform Operations using Data Types and Operators

Evaluate an expression to identify the data type Python will assign to each variable
Data types include str, int, float, and bool
Convert between and work with data types
Type casting; constructing data structures; indexing and slicing operations
Determine the sequence of execution based on operator precedence
Assignment; Comparison; Logical; Arithmetic; Identity (is); Containment (in)
Select the appropriate operator to achieve the intended result
Assignment; Comparison; Logical; Arithmetic; Identity (is); Containment (in)

Control Flow with Decisions and Loops

Construct and analyze code segments that use branching statements
if; elif; else; nested and compound conditionals
Construct and analyze code segments that perform iteration
while; for; break; continue; pass; nested loops and loops that include compound conditionals

Perform Input and Output Operations

Construct and analyze code segments that perform file input and output operations
open; close; read; write; append; check existence; delete; with statement
Construct and analyze code segments that perform console input and output operations
Read input from console; print formatted text; use of command line arguments

Document and Structure Code

Document code segments using comments and documentation strings
Use of indentation and white space; comments and documentation strings; pydoc
Construct and analyze code segments that include function definitions
Call signatures; default values; return; def; pass

Perform Troubleshooting and Error Handling

Analyze, detect, and fix code segments that have errors
Syntax errors; logic errors; runtime errors
Analyze and construct code segments that handle exceptions
Try; except; else; finally; raise

Perform Operations Using Modules and Tools

Perform basic operations using built-in modules
math; datetime; io; sys; os; os.path; random
Solve complex computing problems by using built-in modules
math; datetime; random