MTA: Introduction to Programming Using Python – Skills Measured

NOTE: The bullets that appear below each of the skills measured are intended to illustrate how we are assessing that skill. This list is not definitive or exhaustive.

NOTE: In most cases, exams do NOT cover preview features, and some features will only be added to an exam when they are GA (General Availability).

Exam 98-381: Introduction to Programming Using Python

Perform Operations using Data Types and Operators (20-25%)

Evaluate an expression to identify the data type Python will assign to each variable

- identify str, int, float, and bool data types

Perform data and data type operations

- convert from one data type to another type; construct data structures; perform 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 (25-30%)

Construct and analyze code segments that use branching statements

- if; elif; else; nested and compound conditional expressions

Construct and analyze code segments that perform iteration

- while; for; break; continue; pass; nested loops and loops that include compound conditional expressions

Perform Input and Output Operations (20-25%)

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 (15-20%)

Document code segments using comments and documentation strings

- use indentation, white space, comments, and documentation strings; generate documentation by using pydoc

Construct and analyze code segments that include function definitions

- call signatures; default values; return; def; pass

Perform Troubleshooting and Error Handling (5-10%)

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 (1-5%)

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