|  |
| --- |
| **Course OutlineModule 1: Perform Operations Using Data Types and Operators**This module explains how to use Python operators and data types to achieve a specified result.**Lessons**   * Assign data types to variables * Perform data and data type operations * Perform Arithmetic, Comparison and Logical Operations * Review   **Lab : Perform Operations Using Data Types and Operators**   * Assign data types to variables * Perform data and data type operations * Perform Arithmetic, Comparison and Logical Operations   After completing this module, students will be able to:   * Assign data types to variables * Manage data and data type operations * Perform Arithmetic, Comparison and Logical Operations   **Module 2: Control Flow with Decisions and Loops**This module explains how to use Control Flow and Looping operations in Python.**Lessons**   * Construct and analyze code segments that use branching statements * Construct and analyze code segments that perform iterations * Review   **Lab : Control Flow with Decisions and Loops**   * Using branching operations * Using iteration operations   After completing this module, students will be able to:   * Create branching operations * Create iteration operations   **Module 3: Perform Input and Output Operations**This module explains how to construct input and output operations using files or from the console.**Lessons**   * Create Python code segments that perform file input and output operations * Create Python code segments that perform console input and output operations * Review   **Lab : Perform Input and Output Operations**   * Perform input and output operations using files * Perform input and output operations from the console   After completing this module, students will be able to:   * Use files for input and output operations * Use the console to perform input and output operations   **Module 4: Document and Structure Code**This module explains how to structure and document well-written Python code.**Lessons**   * Construct and analyze code segments * Document code segments using comments and documentation strings * Review   **Lab : Document and Structure Code**   * Construct and Analyze Code Segments * Document Code Segments   After completing this module, students will be able to:   * Create code segments * Document code segments   **Module 5: Perform Troubleshooting and Error Handling**This module explains how to perform troubleshooting and error handling operations in Python.**Lessons**   * Analyze, Detect and Fix code segments that have errors * Analyze and construct code segments that handle exceptions * Review   **Lab : Perform Troubleshooting and Error Handling**   * Analyze, Detect and Fix code segments that have errors * Analyze and Construct code segments that handle exceptions   After completing this module, students will be able to:   * Detect and Fix errors in code * Create error handling code   **Module 6: Perform Operations Using Modules and Tools**This module explains how to use built-in modules.**Lessons**   * Use Built-In Modules to perform basic operations * Use Built-In Modules to perform complex operations * Review   **Lab : Perform Operations Using Modules and Tools**   * Use Built-In Modules to perform basic operations * Use Built-In Modules to perform complex operations   After completing this module, students will be able to:   * Use Built-In modules to perform operating system, date and mathematical operations |
| **Additional Reading**  To help you prepare for this class, review the following resources:   * Microsoft test objectives for Exam 98-381 |
| **Course OutlineModule 1: Perform Operations Using Data Types and OperatorsThis module explains how to use Python operators and data types to achieve a specified result.Lessons**   * **Assign data types to variables** * **Perform data and data type operations** * **Perform Arithmetic, Comparison and Logical Operations** * **Review**   **Lab : Perform Operations Using Data Types and Operators**   * **Assign data types to variables** * **Perform data and data type operations** * **Perform Arithmetic, Comparison and Logical Operations**   **After completing this module, students will be able to:**   * **Assign data types to variables** * **Manage data and data type operations** * **Perform Arithmetic, Comparison and Logical Operations**   **Module 2: Control Flow with Decisions and LoopsThis module explains how to use Control Flow and Looping operations in Python.Lessons**   * **Construct and analyze code segments that use branching statements** * **Construct and analyze code segments that perform iterations** * **Review**   **Lab : Control Flow with Decisions and Loops**   * **Using branching operations** * **Using iteration operations**   **After completing this module, students will be able to:**   * **Create branching operations** * **Create iteration operations**   **Module 3: Perform Input and Output OperationsThis module explains how to construct input and output operations using files or from the console.Lessons**   * **Create Python code segments that perform file input and output operations** * **Create Python code segments that perform console input and output operations** * **Review**   **Lab : Perform Input and Output Operations**   * **Perform input and output operations using files** * **Perform input and output operations from the console**   **After completing this module, students will be able to:**   * **Use files for input and output operations** * **Use the console to perform input and output operations**   **Module 4: Document and Structure CodeThis module explains how to structure and document well-written Python code.Lessons**   * **Construct and analyze code segments** * **Document code segments using comments and documentation strings** * **Review**   **Lab : Document and Structure Code**   * **Construct and Analyze Code Segments** * **Document Code Segments**   **After completing this module, students will be able to:**   * **Create code segments** * **Document code segments**   **Module 5: Perform Troubleshooting and Error HandlingThis module explains how to perform troubleshooting and error handling operations in Python.Lessons**   * **Analyze, Detect and Fix code segments that have errors** * **Analyze and construct code segments that handle exceptions** * **Review**   **Lab : Perform Troubleshooting and Error Handling**   * **Analyze, Detect and Fix code segments that have errors** * **Analyze and Construct code segments that handle exceptions**   **After completing this module, students will be able to:**   * **Detect and Fix errors in code** * **Create error handling code**   **Module 6: Perform Operations Using Modules and ToolsThis module explains how to use built-in modules.Lessons**   * **Use Built-In Modules to perform basic operations** * **Use Built-In Modules to perform complex operations** * **Review**   **Lab : Perform Operations Using Modules and Tools**   * **Use Built-In Modules to perform basic operations** * **Use Built-In Modules to perform complex operations**   **After completing this module, students will be able to:**   * **Use Built-In modules to perform operating system, date and mathematical operations** |
| **Additional Reading**  **To help you prepare for this class, review the following resources:**   * **Microsoft test objectives for Exam 98-381**   [Perform Operations using Data Types and Operators (20-25%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-1)   * 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%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-2)   * 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%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-3)   * 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%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-4)   * 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%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-5)   * 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%)](https://www.microsoft.com/en-us/learning/exam-98-381.aspx#syllabus-6)   * 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 |