Document Generated: 07/05/2024

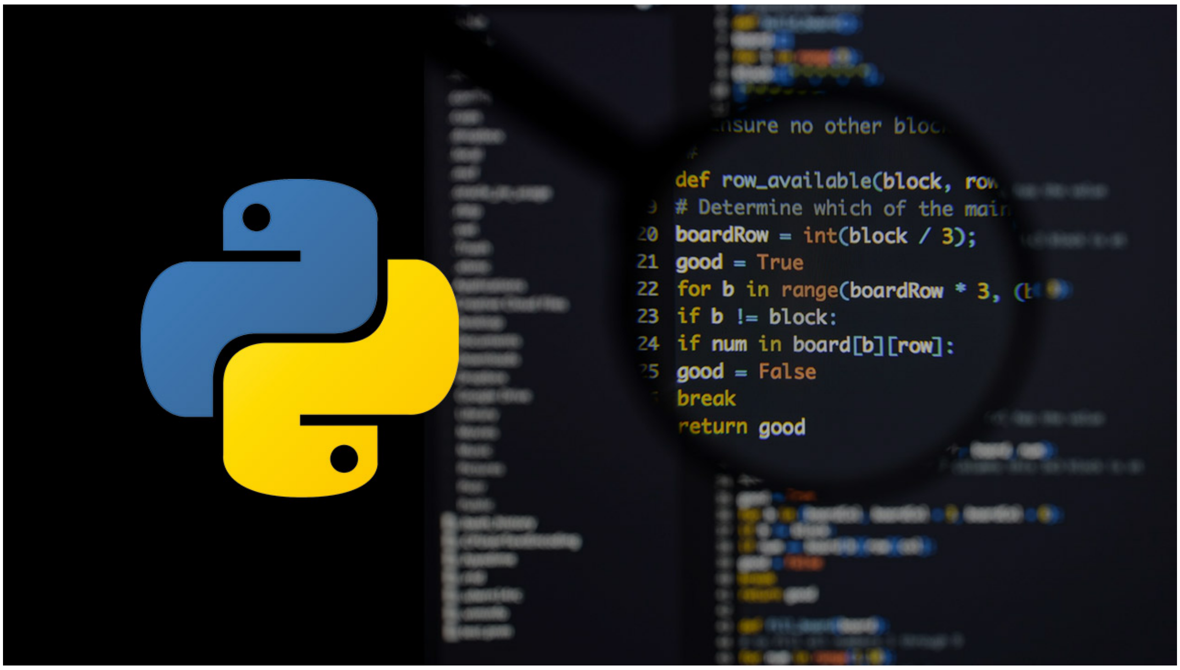
Learning Style: Virtual Classroom

Provider:

Difficulty: Advanced

Course Duration: 5 Days

Mastering Python Programming (with Next-Level Topics) (TTPS4820)



nsure no other bloc
def row_available(block, row
# Determine which of the mair
boardRow = int(block / 3);
good = True
for b in range(boardRow * 3, (b
if b != block:
if num in board[b][row]:
good = False
break
return good

About this course:

Geared for experienced users, **Mastering Python Programming** is an **introductory and beyond-level** practical, hands-on Python training course that leads the student from the basics of writing and running Python scripts to more advanced features such as file operations, regular expressions, working with binary data, and using the extensive functionality of Python modules. Extra emphasis is placed on features unique to Python, such as tuples, array slices, and output formatting.

This comprehensive, practical course provides an in-depth exploration of working with the programming language, not an academic overview of syntax and grammar. Students will immediately be able to use Python to complete tasks in the real world.

Course Objective:

Throughout the course, students will be led through a series of progressively advanced topics, where each topic consists of lecture, group discussion, comprehensive hands-on lab exercises, and lab review. This course is “skills-centric”, designed to train attendees in essential Python and web development skills, coupling the most current, effective techniques with best practices.

Students will explore:

- Running Python Scripts
- Getting Started
- Flow Control
- Sequence Data
- Defining Functions
- Working with Files
- Dictionaries and Sets
- Errors and Exception Handling
- Using Modules
- Regular Expressions
- Highlights of the Standard Library
- Introduction to Python Classes
- Real-life programming
- Special data types
- Network services
- Dates and times

Working within in an engaging, hands-on learning environment, guided by our expert Python practitioner, students will learn to:

- Create working Python scripts following best practices
- Use python data types appropriately
- Read and write files with both text and binary data
- Search and replace text with regular expressions
- Get familiar with the standard library and its work-saving modules
- Use lesser-known but powerful Python data types
- Create "real-world", professional Python applications
- Work with dates, times, and calendars
- Know when to use collections such as lists, dictionaries, and sets
- Understand Pythonic features such as comprehensions and iterators
- Write robust code using exception handling

Audience:

This course is appropriate for advanced users, system administrators, and website administrators who want to use Python to support their server installations, as well as anyone else who wants to automate or simplify common tasks with the use of Python scripts.

Prerequisite:

Students should already have a working, user-level knowledge of Unix/Linux, Mac, or Windows. While not required, basic skills with at least one other programming language will be helpful.

Course Outline:

Module 1: An Overview of Python

- What is python?
- An overview of Python
- What is python?
- Python Timeline
- Advantages/Disadvantages of Python
- Getting help with pydoc

Module 2: The Python Environment

- Starting Python
- Using the interpreter
- Running a Python script
- Python scripts on Unix/Windows
- Editors and IDEs

Module 3: Getting Started

- Using variables
- Builtin functions
- Strings
- Numbers
- Converting among types
- Writing to the screen
- Command line parameters

Module 4: Flow Control

- About flow control
- White space
- Conditional expressions
- Relational and Boolean operators
- While loops
- Alternate loop exits

Module 5: Sequences

- About sequences
- Lists and list methods
- Tuples
- Indexing and slicing
- Iterating through a sequence
- Sequence functions, keywords, and operators
- List Comprehensions
- Generator Expressions
- Nested sequences

Module 6: Working with files

- File Overview
- Opening a text file
- Reading a text file
- Writing to a text file
- Reading and writing raw (binary) data
- Converting binary data with struct

Module 7: Dictionaries and Sets

- About dictionaries
- Creating dictionaries
- Iterating through a dictionary
- About sets
- Creating sets
- Working with sets

Module 8: Functions

- Defining functions
- Parameters
- Global and local scope
- Nested functions
- Returning values

Module 9: Sorting

- The sorted() function
- Alternate keys
- Lambda functions
- Sorting collections
- Using operator.itemgetter()
- Reverse sorting

Module 10: Errors and Exception Handling

- Syntax errors

- Exceptions
- Using try/catch/else/finally
- Handling multiple exceptions
- Ignoring exceptions

Module 11: Modules and Packages

- The import statement
- Module search path
- Creating Modules
- Using packages
- Function and Module aliases

Module 12: Classes

- About o-o programming
- Defining classes
- Constructors
- Methods
- Instance data
- Properties
- Class methods and data

Module 13: Regular Expressions

- RE syntax overview
- RE Objects
- Searching and matching
- Compilation flags
- Groups and special groups
- Replacing text
- Splitting strings

Module 14: The standard library

- The sys module
- Launching external programs
- Math functions
- Random numbers
- The string module
- Reading CSV data

Module 15: Dates and times

- Working with dates and times
- Translating timestamps
- Parsing dates from text
- Formatting dates
- Calendar data

Module 16: Working with the file system

- Paths, directories, and filenames
- Checking for existence
- Permissions and other file attributes
- Walking directory trees
- Creating filters with file input
- Using shutil for file operations

Module 17: Advanced data handling

- Defaultdict and Counter
- Prettyprinting data structures
- Compressed archives (zip, gzip, tar, etc.)
- Persistent data

Module 18: Network services

- Grabbing web content
- Sending email
- Using SSH for remote access
- Using FTP

Module 19: Writing real-life applications

- Parsing command-line options
- Detecting the current platform
- Trapping signals
- Implementing logging
- Python Timeline
- Advantages/Disadvantages of Python
- Getting help with pydoc

Credly Badge:

QUICKSTART
Course Name:
Mastering Python Programming (with Next-Level Topics) (TTPS4820)
APPLICATION DEVELOPMENT

Display your Completion Badge And Get The Recognition You Deserve.

Add a completion and readiness badge to your Linkedin profile, Facebook page, or Twitter account to validate your professional and technical expertise. With badges issued and validated by Credly, you can:

- Let anyone verify your completion and achievement by clicking on the badge
- Display your hard work and validate your expertise
- Display each badge's details about specific

skills you developed.

Badges are issued by QuickStart and verified through Credly.

Find Out More or See List Of Badges