Pythion 50 Repetithon

Complete Code Structure Documentation

CoreCode Programming Academy - Python Masterclass

Project Overview

This repository is a comprehensive Python programming masterclass called "Pythion 50 Repetithon", organized into 85 sequential learning sessions (SESSION_006 through SESSION-092). It covers fundamentals through advanced topics including OOP, functional programming, closures, GUI development, and more.

Project Statistics

Total Files	213+
Python Files	162
PDF Documents	15
PowerPoint Presentations	2
Text Documentation Files	20+
Session Directories	72
Tkinter GUI Examples	11

1. Top-Level Directory Layout

```
Pythion_50_Repetithon/
|-- GUI_USING_TKINTER/        11 Tkinter GUI example scripts
|-- OPEN_FOR_ALL_WEEK/        Course materials & demonstrations
|-- SESSION_006/              Early session (basics)
|-- SESSION_007/              Early session (operators)
|-- SESSION-008/ ... SESSION-092/  70 session directories
|-- _BLANK_FOLDERS/           Archived empty session folders
|-- ___All_Errors.txt         Log of files that failed to download
```

2. GUI_USING_TKINTER/

Contains 11 Python scripts demonstrating progressive Tkinter GUI programming, from basic widgets to a complete multi-frame admission form application.

- `01_widget_demo.py` - Basic Tkinter widget demonstration (Label, Entry, Button)
- `02_padx_pady_demo.py` - Layout management with padding and spacing
- `03_textvariable_demo.py` - Text variable binding with StringVar/IntVar
- `04_feet_to_meter.py` - Unit conversion GUI with real-time calculation
- `05_compute_gravitational_gui.py` - Physics gravitational force calculator GUI
- `06_variables_type.py` - Tkinter variable type examples (BooleanVar, etc.)
- `07_check_box_demo.py` - Checkbox (Checkbutton) widget examples
- `08_combo_box_demo.py` - Dropdown / Combobox widget examples
- `09_radio_button.py` - Radio button group selection examples
- `10_cpa_admission_form.py` - Complex multi-frame admission form with 5 courses, radio buttons, checkboxes, and registration logic
- `11_menu_demo.py` - Menu bar and dropdown menu items

3. OPEN_FOR_ALL_WEEK/

The largest directory (~40 MB). Contains course materials, instructor resources, demonstration code, and the Python installer for students.

3a. DEMONSTRATION_CODE/

Advanced Python demonstration scripts used in live sessions.

- `abc_implementation.py` - Abstract base class implementation using a custom metaclass named 'interface'; demonstrates abstract method enforcement and inheritance
- `logger.py` - Function decorator that logs calls, arguments, return values, and exceptions to a file
- `test_logger.py` - Test suite for the logger decorator with multiple functions including recursive calls
- `GUI_USING_TKINTER/` - Duplicate copy of the top-level Tkinter examples

3b. DETAILED_COURSE_CONTENT/

- `MASTERCLASS_IN_PYTHON_COURSE_CONTENT.pdf` - Complete curriculum PDF covering all 50+ sessions

3c. WEEK_1_MATERIAL/

Foundation PDF documents for the first week of the course.

- `1_Python_Installation_Guide.pdf` - Step-by-step Python setup instructions
- `2_PythonFileStructure.pdf` - Python file organization concepts
- `3_AssignmentStatementScenarios.pdf` - Assignment statement use cases
- `4_AssignmentStatementAlgorithm.pdf` - Algorithm details for assignment
- `5_DataTypes.pdf` - Data type fundamentals (int, float, str, etc.)

3d. IDLE_DOWNLOAD/

`1_Python_Installation_Guide.pdf` - Installation guide (duplicate)

`python-3.13.6-amd64.exe` - Windows Python 3.13.6 installer (28.8 MB)

3e. Root Files in OPEN_FOR_ALL_WEEK/

`CPA_PRESENTATION.pptx` - CoreCode Programming Academy presentation (5.3 MB)

`Goals_and_definite_chief_aim.pptx` - Goal setting presentation (45 KB)

`PROFILE - YOGESHWAR SHUKLA.pdf` - Instructor profile document (5.6 MB)

`The_Definite_Chief_Aim.png` - Goal visualization image (89 KB)

4. Session Directories (SESSION_006 - SESSION-092)

85 sequential learning sessions forming the core of the course. Each session directory contains Python scripts, IDLE logs, and/or PDF materials for that session's topic. Below is a detailed breakdown organized by learning phase.

Phase 1: Fundamentals (Sessions 006-021)

Session	Key Files	Description
SESSION_006	`BASICS.txt`	Fundamental concepts: class vs algorithm, built-in types and functions
SESSION_007	`ARITHMETIC-OPERATORS-IDLE-LOG.py`, `ASCII-CODES-IDLE-LOG.txt`	Basic arithmetic operators and ASCII character codes
SESSION-008	`01-input-data.py`	Basic input/output operations and type conversion (int, float, str)
SESSION-009	`01-first-gui-app.py`	Introduction to GUI programming with Tkinter
SESSION-010	`01-def-statement-demo.py`	Function definition basics using the def statement
SESSION-011	`3 IDLE log files`	Interactive function demonstrations in the IDLE environment
SESSION-012	`Session logs`	Continued IDLE session practice
SESSION-015	`Control flow scripts`	Control flow combined with function definitions
SESSION-016	`Control flow scripts`	Continued control flow practice
SESSION-017	`insertion_sort.py + boolean scripts`	Boolean operations and the insertion sort algorithm
SESSION-018	`01-if-statement.py`, `02-if-else-statement.py`, `03-if-elif-else-statement.py`	Branching statements: if, if-else, if-elif-else patterns
SESSION-020	`GUI button demos`	Tkinter button widget demonstrations
SESSION-021	`Toggle functionality scripts`	GUI toggle button and event handling

Phase 2: Core Programming (Sessions 023-050)

Session	Key Files	Description
SESSION-023	`4 files: global vars, GUI squares`	Global variables and GUI square-drawing applications
SESSION-024	`01-class-demo.py`	Class basics: defining a simple Date class with attributes
SESSION-025 to 030	`OOP progression scripts`	Object-oriented programming: classes, methods, constructors, self
SESSION-032 to 045	`Data structure scripts`	Data structure and collection handling (lists, tuples, dicts, sets)
SESSION-048	`abc.py, pqr.py, xyz.py, show_pwd.py`	Small utility scripts and module import examples
SESSION-050	`01-create-and-write-on-file.py`, `03-file-copy.py`, `04-commandline-arguments.py`	File I/O: creating, writing, copying files; command-line arguments with sys.argv

Phase 3: Functional Programming (Sessions 051-077)

Session	Key Files	Description
SESSION-051/052	`6 files each`	File handling and text processing techniques

Session	Key Files	Description
SESSION-059/060	`SET-IDLE-LOG.py`	Set operations: union, intersection, difference, symmetric difference
SESSION-063	`LIST-GENERAL-SYNTAX.py`, `LIST-IDLE-LOG.py`, `VALIDITY-OF-OPERATOR.py`	List operations: indexing, slicing, operators, methods
SESSION-064 to 072	`Multiple scripts`	Advanced data structures and introduction to functional programming
SESSION-073	`LAMBDA-IDLE-LOG.py`, `power_lambda_map.py + PDFs`	Lambda expressions and map() function with lambda
SESSION-074	`01-write-demo.py + IDLE-FILTER-LOG.pdf`	Filter operations: filter() with lambda and named functions
SESSION-076	`List-Comprehension-General-Format-And-Some-Examples.py + PDF`	List comprehensions: syntax, single-variable forms (V1 to V4)
SESSION-077	`Advanced-List-Comprehension.pdf`	Advanced list comprehensions with multiple variables and conditions

Phase 4: Advanced Functions & Scope (Sessions 079-088)

Session	Key Files	Description
SESSION-079	`Nested-Def-Statement/01-def-under-def.py, Language-Topics-Covered-So-Far.txt`	Nested function definitions, scope management, and curriculum checklist
SESSION-080	`01-nested-def-statement-practice.py, 02-symbol-table.py`	Symbol tables and nested def statement practice
SESSION-081	`Scope scripts`	Python scope rules introduction
SESSION-082	`03-LEGB-Scope-Rule.py, 04-UnboundLocalError.py, 05-NameError-FreeVariable.py`	LEGB scope rule: Local, Enclosing, Global, Built-in; UnboundLocalError and NameError demos
SESSION-084	`04-Solution...global-statement.py, 07-nonlocal-demo.py`	Global and nonlocal statements for modifying enclosing scope variables
SESSION-085/086	`01-Returning-Function-As-Data-Object.py, 02-FunctionFactory.py`	Higher-order functions: returning functions, closures, function factories
SESSION-087	`PARAMETER-PASSING-GURU-KILLI.py + PDFs`	Parameter passing mechanisms: positional, keyword, *args, **kwargs

Phase 5: Object-Oriented Programming (Sessions 090-092)

Session	Key Files	Description
SESSION-090	`03-Quadrilateral-Version-I.py, GENERAL-SYNTAX.txt`	Quadrilateral class Version I: class definition with constructor, attributes for 4 sides and 4 angles
SESSION-091	`Quadrilateral full implementation (2 files)`	Quadrilateral class complete implementation with methods
SESSION-092	`01-Triangle.py, 02-Vector3D.py`	Geometric shape classes: Triangle with perimeter/area; Vector3D with add, subtract, normalize, dot/cross product

5. _BLANK_FOLDERS/

Archive of session folders that were initially empty or contained files that could not be downloaded.

- __SESSION-040/ - Originally contained 02-while-else-demo.py (download failed)
- __SESSION-041/ - Originally contained while-statement-repeatithon.py (download failed)
- __SESSION-043/ - Originally contained class-statement-repeatithon.py (download failed)
- __SESSION-049/ - Originally contained abc.txt (download failed)

6. ___All_Errors.txt

A log file at the repository root tracking files that failed to download during initial setup. Lists 4 files across sessions 040, 041, 043, and 049 that could not be retrieved.

7. Curriculum Progression Summary

The course follows a carefully structured learning path. Below is the topic progression as documented in the repository.

Fundamentals (Sessions 006-021)

- Python execution model and memory management
- Assignment statements and the object model
- Data types: bool, int, float, str, list, tuple, dict, set
- Branching: if / elif / else
- Looping: while, for, break, continue, pass

Procedural Programming (Sessions 022-050)

- Function definition with def statement
- Parameter passing (positional, keyword, defaults)
- Type annotations and return values
- Exception handling basics
- File I/O and command-line arguments

Functional Programming (Sessions 051-077)

- Lambda expressions
- Built-in higher-order functions: map(), filter(), reduce()
- List comprehensions (single and multi-variable)
- Set and dictionary operations

Advanced Functions (Sessions 079-088)

- Nested function definitions
- Scope and LEGB rule (Local, Enclosing, Global, Built-in)
- global and nonlocal statements
- Higher-order functions and closures
- Function factories and implicit state saving

Object-Oriented Programming (Sessions 090-092)

- Class definitions with __init__ constructor
- Instance attributes and the self parameter
- Object and class namespaces
- Geometric shape implementations (Quadrilateral, Triangle, Vector3D)

Remaining Topics (Not Yet Covered)

- Operator overloading
- Inheritance and polymorphism
- Advanced exception handling
- Iterators and generators
- Context managers and decorators
- Metaclasses
- Threads, regex, databases, ML/DS applications

8. Key Code Highlights

Notable scripts in the repository that demonstrate important concepts:

`OPEN_FOR_ALL_WEEK/DEMONSTRATION_CODE/abc_implementation.py`
Implements abstract base classes using a custom metaclass named 'interface'. Demonstrates how Python enforces abstract methods through metaclass __call__ and how inheritance of abstract methods works.

`OPEN_FOR_ALL_WEEK/DEMONSTRATION_CODE/logger.py`
A production-style function decorator that logs every function call with arguments, return values, and exception details to a log file. Shows decorator pattern and exception handling.

`GUI_USING_TKINTER/10_cpa_admission_form.py`
A complex Tkinter application with multiple frames, radio buttons for 5 courses, checkboxes, and a complete registration workflow. Demonstrates real-world GUI application structure.

`SESSION-092/02-Vector3D.py`
A 3D vector class with mathematical operations: addition, subtraction, normalization, dot product, and cross product. Demonstrates operator concepts and mathematical programming with classes.

`SESSION-082/03-LEGB-Scope-Rule.py`
Comprehensive demonstration of Python's LEGB scope resolution rule with examples showing Local, Enclosing, Global, and Built-in scope lookups.

`SESSION-085/02-FunctionFactory.py`
Demonstrates closures and function factories - functions that create and return other functions with captured state from the enclosing scope.