Data Management Pipeline Test
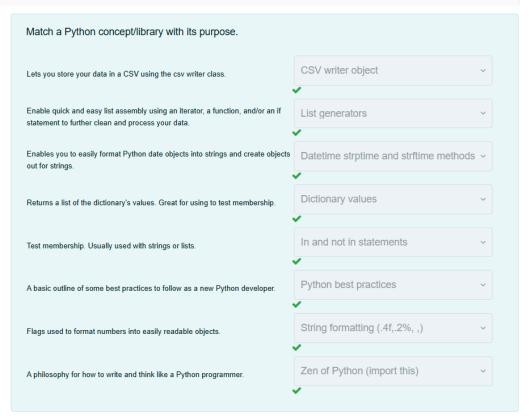
Started on	Wednesday, 20 November 2024, 1:54 PM
State	Finished
Completed on	Wednesday, 20 November 2024, 2:10 PM
Time taken	16 mins 41 secs
Marks	2.00/2.00
Grade	**10.00** out of 10.00 (**100%**)

Question 1

Correct

Mark 1.00 out of 1.00

⚑ Flag question

Match a Python concept/library with its purpose.

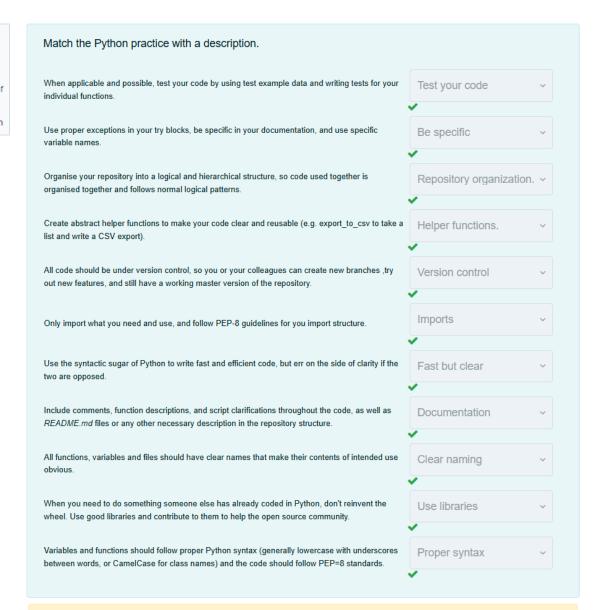|	
Lets you store your data in a CSV using the csv writer class.	CSV writer object ✔
Enable quick and easy list assembly using an iterator, a function, and/or an if statement to further clean and process your data.	List generators ✔
Enables you to easily format Python date objects into strings and create objects out for strings.	Datetime strptime and strftime methods ✔
Returns a list of the dictionary's values. Great for using to test membership.	Dictionary values ✔
Test membership. Usually used with strings or lists.	In and not in statements ✔
A basic outline of some best practices to follow as a new Python developer.	Python best practices ✔
Flags used to format numbers into easily readable objects.	String formatting (.4f,.2%, ,) ✔
A philosophy for how to write and think like a Python programmer.	Zen of Python (import this) ✔

Your answer is correct.

The correct answer is: Lets you store your data in a CSV using the csv writer class. → CSV writer object, Enable quick and easy list assembly using an iterator, a function, and/or an if statement to further clean and process your data. → List generators, Enables you to easily format Python date objects into strings and create objects out for strings. → Datetime strptime and strftime methods, Returns a list of the dictionary's values. Great for using to test membership. → Dictionary values, Test membership. Usually used with strings or lists. → In and not in statements, A basic outline of some best practices to follow as a new Python developer. → Python best practices, Flags used to format numbers into easily readable objects. → String formatting (.4f,.2%, ,), A philosophy for how to write and think like a Python programmer. → Zen of Python (import this)

Match the Python practice with a description.

When applicable and possible, test your code by using test example data and writing tests for your individual functions.

Test your code ⌄

✓

Use proper exceptions in your try blocks, be specific in your documentation, and use specific variable names.

Be specific ⌄

✓

Organise your repository into a logical and hierarchical structure, so code used together is organised together and follows normal logical patterns.

Repository organization. ⌄

✓

Create abstract helper functions to make your code clear and reusable (e.g. export_to_csv to take a list and write a CSV export).

Helper functions. ⌄

✓

All code should be under version control, so you or your colleagues can create new branches ,try out new features, and still have a working master version of the repository.

Version control ⌄

✓

Only import what you need and use, and follow PEP-8 guidelines for you import structure.

Imports ⌄

✓

Use the syntactic sugar of Python to write fast and efficient code, but err on the side of clarity if the two are opposed.

Fast but clear ⌄

✓

Include comments, function descriptions, and script clarifications throughout the code, as well as *README.md* files or any other necessary description in the repository structure.

Documentation ⌄

✓

All functions, variables and files should have clear names that make their contents of intended use obvious.

Clear naming ⌄

✓

When you need to do something someone else has already coded in Python, don't reinvent the wheel. Use good libraries and contribute to them to help the open source community.

Use libraries ⌄

✓

Variables and functions should follow proper Python syntax (generally lowercase with underscores between words, or CamelCase for class names) and the code should follow PEP=8 standards.

Proper syntax ⌄

✓

Your answer is correct.

The correct answer is: When applicable and possible, test your code by using test example data and writing tests for your individual functions. → Test your code, Use proper exceptions in your try blocks, be specific in your documentation, and use specific variable names. → Be specific, Organise your repository into a logical and hierarchical structure, so code used together is organised together and follows normal logical patterns. → Repository organization., Create abstract helper functions to make your code clear and reusable (e.g. export_to_csv to take a list and write a CSV export). → Helper functions., All code should be under version control, so you or your colleagues can create new branches ,try out new features, and still have a working master version of the repository. → Version control, Only import what you need and use, and follow PEP-8 guidelines for you import structure. → Imports, Use the syntactic sugar of Python to write fast and efficient code, but err on the side of clarity if the two are opposed. → Fast but clear, Include comments, function descriptions, and script clarifications throughout the code, as well as *README.md* files or any other necessary description in the repository structure. → Documentation, All functions, variables and files should have clear names that make their contents of intended use obvious. → Clear naming, When you need to do something someone else has already coded in Python, don't reinvent the wheel. Use good libraries and contribute to them to help the open source community. → Use libraries, Variables and functions should follow proper Python syntax (generally lowercase with underscores between words, or CamelCase for class names) and the code should follow PEP=8 standards. → Proper syntax

✔ Done

Re-attempt quiz

Grading method: Highest grade

Summary of your previous attempts

Attempt	State	Marks / 2.00	Grade / 10.00	Review
1	Finished Submitted Wednesday, 20 November 2024, 2:10 PM	2.00	10.00	Review

Highest grade: 10.00 / 10.00.