COURSE GLOSSARY

Writing Efficient Python Code

%timeit: An IPython line or cell magic that runs the given code many times and reports statistical timing information (mean and standard deviation), providing a robust estimate of execution time

Boolean indexing: Selecting elements from an array by passing a boolean mask (an array of True/False values), which returns only the elements where the mask is True and enables concise filtering expressions

Broadcasting (vectorization): NumPy's ability to apply arithmetic and other operations elementwise across whole arrays (or between arrays of compatible shapes) without explicit Python loops, yielding large performance gains

Code profiling: The practice of measuring how long and how often parts of a program execute (and/or how much memory they use) in order to identify performance bottlenecks for optimization

collections.Counter: A dict-like class in the collections module that counts hashable items in an iterable and returns a mapping of item→count, providing a fast, concise alternative to manual counting loops

DataFrame.apply: A pandas method that applies a function along an axis (rows or columns) of a DataFrame, acting like a vectorized map and often providing clearer and faster alternatives to explicit row-by-row Python loops

Efficient code: Code that achieves its intended result with minimal runtime latency and prudent use of system resources such as memory, balancing speed and resource overhead for the task at hand

Iterative prompt refinement: The cyclical process of testing a prompt, analyzing model outputs, and modifying the prompt or examples to progressively improve performance and reliability

Function: A named block of code that performs an action or computation and optionally returns a value, taking inputs as arguments

IPython magic commands: Special convenience commands available in IPython/Jupyter that start with % (line magics) or %% (cell magics) and provide shortcuts for tasks like timing, debugging, and profiling

Latency: The time delay between initiating code execution and receiving its result, often used as a measure of code responsiveness and performance

line_profiler (%lprun): A third-party profiling tool (invoked via the %lprun magic) that reports execution time per source-line inside functions so you can see which lines consume the most runtime

List comprehension: A concise, expressive Python construct for creating lists by applying an expression to each item in an iterable (optionally with a condition), often replacing explicit loops with clearer and faster code

map: A built-in Python function that applies a provided function to each item of an iterable and returns an iterator of the results, enabling concise element-wise transformations often without explicit loops

Membership testing: The operation of checking whether an item belongs to a container (e.g., using the in operator), which is much faster on sets and dicts (average O(1)) than on lists or tuples (O(n))

memory_profiler (%mprun): A third-party tool (invoked via the %mprun magic) that reports line-by-line memory usage of functions by querying the OS, useful for locating memory allocation hotspots (note: profiled functions often must be defined in files, not the interactive session)

NumPy array: A homogeneous, fixed-type multidimensional array provided by NumPy that stores numeric data compactly and enables fast, vectorized numerical operations

Overhead: Extra computational cost (time, memory, or other resources) required by a program or data structure beyond the minimal cost of performing the core task

pandas DataFrame: A two-dimensional, labeled tabular data structure in pandas with rows and columns that supports heterogeneous column types and a rich set of vectorized operations for data analysis

Python Standard Library: The collection of modules and packages that ship with Python and provide commonly needed functionality (e.g., itertools, collections, sys) so you can solve many problems without external dependencies

Pythonic: Code that follows Python's idioms and best practices—readable, concise, and expressive—typically resulting in clearer and often more efficient implementations

range: A built-in Python function that produces a sequence-like range object of integers defined by start, stop (exclusive), and optional step parameters, commonly used for indexed iteration

set (data type and operations): An unordered Python collection of unique, hashable items that supports efficient operations like union, intersection, difference, and symmetric difference for comparing membership and relationships between collections

sys.getsizeof: A function in Python's sys module that returns the size in bytes of a single Python object as reported by the interpreter, useful for quick checks of object memory footprint

Zen of Python: A collection of guiding aphorisms for Python design and style (e.g., "Simple is better than complex") that inform Pythonic coding practices and readability