**• Indentation, comments, and naming conventions in Python.**

Python places significant emphasis on code readability, and this is reflected in its conventions for indentation, comments, and naming.

Indentation:

* Python uses indentation to define blocks of code, unlike other languages that use braces or keywords.
* The standard convention, as defined by PEP 8 (Python Enhancement Proposal 8), is to use four spaces per indentation level.
* Consistency is crucial; mixing tabs and spaces for indentation within the same file is highly discouraged and can lead to IndentationError.

Comments:

* **Single-line comments**

begin with a hash symbol (#) and extend to the end of the line. They are used to explain specific lines or small sections of code.

* **Block comments**

are typically used to explain larger sections of code or provide context for a function or class. While not strictly enforced by Python's syntax, they are conventionally achieved by using multiple single-line comments, each starting with #.

* **Docstrings**

(documentation strings) are multi-line strings enclosed in triple quotes ("""Docstring content""" or '''Docstring content'''). They are used to document modules, classes, functions, and methods, providing information about their purpose, arguments, and return values.

Naming Conventions:

PEP 8 outlines standard naming conventions to improve code consistency and readability:

* **Modules:** Use short, all-lowercase names, potentially with underscores (e.g., my\_module.py).
* **Packages:** Similar to modules, but generally without underscores (e.g., my\_package).
* **Classes:** Use CapWords (CamelCase) convention (e.g., MyClass).
* **Functions and Methods:** Use lowercase with words separated by underscores (e.g., my\_function, calculate\_total).
* **Variables:** Similar to functions, use lowercase with underscores (e.g., my\_variable, total\_count).
* **Constants:** Use all uppercase with words separated by underscores (e.g., MAX\_VALUE, PI).
* **Private members:** Precede with a single leading underscore (e.g., \_private\_variable).
* **Name mangling:** Precede with two leading underscores to invoke name mangling (e.g., \_\_mangled\_name).
* **Special methods/attributes:** Use two leading and two trailing underscores (e.g., \_\_init\_\_, \_\_str\_\_).