Basic Structure of a Python Program

Lecture 9
Understanding how Python code is organized and formatted.

Agenda

1. Anatomy of a Python Script
2. The Role of Indentation
3. Writing Comments
4. Statements and Multi-line Statements
5. Importing Modules
6. The ‘__main__‘ block

Anatomy of a Python Script

A typical Python program consists of:
1. **Imports:** Bringing in external modules/libraries.
2. **Global Variables:** Variables accessible throughout the file.
3. **Function/Class Definitions:** Reusable blocks of code.
4. **Main Execution Block:** The entry point of the script.

Python executes files from top to bottom.

Indentation: Python's Defining Feature

Most languages (C, Java, JavaScript) use curly braces ‘{}‘ to define blocks of code.
Python uses **whitespace (indentation)**.
- Defines scope for loops, functions, and conditional statements.
- Standard convention is **4 spaces** per indentation level.
- Do not mix spaces and tabs! (Causes ‘TabError‘).

Indentation Example

Correct:

```
if 5 > 2:
    print('Five is greater than two.')
    print('Still inside the if block.')
print('Outside the if block.')
```

Incorrect (IndentationError):

```
if 5 > 2:
print('This will cause an error')
```

Comments in Python

Comments are ignored by the interpreter.
Used to explain code, make it readable, and prevent execution during testing.
Single-line comments: Start with a hash ‘#‘.

```
# This is a comment
x = 5  # Inline comment
```

Multi-line Comments and Docstrings

Python does not have a specific syntax for multi-line comments.
You can use ‘#‘ on multiple lines, or use multi-line strings (“”’‘ or “”””‘) that are not assigned to a variable.
Docstrings (Documentation Strings):

```
def my_function():
    """
    This function does nothing yet.
    Used to document functions/classes.
    """
    pass
```

Python Statements

A statement is an instruction that the Python interpreter can execute.
- Example: ‘x = 5‘ is an assignment statement.
Usually, one statement per line. No semicolon ‘;‘ is needed at the end.
You *can* use semicolons to put multiple statements on one line (but it is unpythonic).

```
x = 5; y = 10; print(x + y)  # Discouraged
```

Multi-line Statements

To split a long statement over multiple lines, use the backslash ‘\{}‘ as a line continuation character.

```
total = 1 + 2 + 3 + \
        4 + 5 + 6
```

Alternatively, line continuation is implied inside parentheses ‘()‘, brackets ‘[]‘, and braces ‘{}‘.

```
total = (1 + 2 + 3 +
         4 + 5 + 6) # Preferred method
```

Importing Modules

Python is extended through modules.
Use the ‘import‘ keyword to bring in external code.
Usually placed at the very top of the file.

```
import math
print(math.sqrt(16))

from datetime import date
print(date.today())
```

The __main__ Block

How do we prevent code from running when a file is imported elsewhere?
Use the ‘if __name__ == "__main__":‘ check.

```
def calculate():
    return 10 * 5

if __name__ == "__main__":
    # This only runs if this script is executed directly
    # It will NOT run if this file is imported
    result = calculate()
    print(result)
```

Summary

Python relies on **indentation** to define blocks, not braces.
Use ‘#‘ for comments and triple quotes ‘”””“ for docstrings.
Statements don’t need semicolons.
Use parentheses for long multi-line statements.
The ‘if __name__ == ’__main__’‘ pattern is standard for scripts.