Tokens in Python

Lecture 10: Keywords, Variables, Data Types, & Operators
Understanding the fundamental building blocks of Python data handling.

Agenda

1. Keywords and Identifiers
2. Variables and Assignment
3. Primitive Data Types
4. Arithmetic Operators
5. Relational (Comparison) Operators
6. Logical and Assignment Operators

Python Keywords

Keywords are reserved words that have special meaning to the interpreter.
They cannot be used as variable names, function names, or any other identifier.
Examples: ‘if‘, ‘else‘, ‘while‘, ‘for‘, ‘def‘, ‘class‘, ‘import‘, ‘True‘, ‘False‘, ‘None‘.
Most are lowercase, except ‘True‘, ‘False‘, and ‘None‘.
To see the list:

```
import keyword
print(keyword.kwlist)
```

Identifiers and Naming Rules

Identifiers are the names given to variables, classes, methods, etc.
Rules:
1. Can contain letters (a-z, A-Z), digits (0-9), and underscores ‘_‘.
2. Cannot start with a digit (‘1variable‘ is invalid).
3. Cannot be a keyword.
4. Case-sensitive (‘Var‘ and ‘var‘ are different).

Best Practice (PEP 8):
- Variables and functions: ‘snake_case‘
- Classes: ‘CamelCase‘

Variables and Assignment

Variables act as containers for storing data values.
Python has no command for declaring a variable; it is created the moment you assign a value.

```
age = 20          # Integer assignment
name = 'Alice'    # String assignment

# Multiple Assignment
x, y, z = 1, 2, 3
a = b = c = 10
```

Variables are actually references (pointers) to objects in memory.

Core Data Types

Data types define the kind of data a variable holds.
Numeric Types:
- ‘int‘: Integers (e.g., ‘10‘, ‘-5‘)
- ‘float‘: Floating-point numbers (e.g., ‘3.14‘, ‘-0.001‘)
- ‘complex‘: Complex numbers (e.g., ‘3 + 4j‘)
Sequence Types:
- ‘str‘: Strings, sequence of characters (e.g., ‘’Hello’‘)
Boolean:
- ‘bool‘: Represents ‘True‘ or ‘False‘

Checking Data Types

Use the built-in ‘type()‘ function to find out the type of a variable.

```
x = 10
print(type(x))  # <class 'int'>

y = 3.14
print(type(y))  # <class 'float'>

z = 'Hello'
print(type(z))  # <class 'str'>

is_active = True
print(type(is_active))  # <class 'bool'>
```

Arithmetic Operators

Used with numeric values to perform common mathematical operations.

- ‘+‘ Addition
- ‘-‘ Subtraction
- ‘*‘ Multiplication
- ‘/‘ Division (always returns float)
- ‘//‘ Floor Division (discards remainder)
- ‘%‘ Modulus (returns remainder)
- ‘**‘ Exponentiation (power)

```
print(10 / 3)    # 3.3333333333333335
print(10 // 3)   # 3
```

Relational (Comparison) Operators

Used to compare two values. They return a Boolean (‘True‘ or ‘False‘).

- ‘==‘ Equal to
- ‘!=‘ Not equal to
- ‘¿‘ Greater than
- ‘¡‘ Less than
- ‘¿=‘ Greater than or equal to
- ‘¡=‘ Less than or equal to

```
x = 5
y = 10
print(x == y)   # False
print(x < y)    # True
```

Logical Operators

Used to combine conditional statements.
- ‘and‘: Returns True if both statements are true.
- ‘or‘: Returns True if one of the statements is true.
- ‘not‘: Reverse the result, returns False if the result is true.

```
x = 5
print(x > 3 and x < 10)  # True
print(not(x > 3))        # False
```

Short-circuit evaluation is used: if the first operand in ‘and‘ is False, the second is not evaluated.

Assignment Operators

Used to assign values to variables.
Include compound assignment operators for shorthand operations.
- ‘=‘ Assign
- ‘+=‘ Add and assign (‘x += 3‘ is ‘x = x + 3‘)
- ‘-=‘ Subtract and assign
- ‘*=‘ Multiply and assign
- ‘/=‘ Divide and assign

```
count = 0
count += 1  # count is now 1
```

Note: Python does not have ‘++‘ or ‘–‘ increment/decrement operators.

Summary

Keywords are reserved; identifiers must follow naming rules.
Variables are dynamically created on assignment.
Basic types include ‘int‘, ‘float‘, ‘str‘, and ‘bool‘.
We use Arithmetic, Relational, Logical, and Assignment operators to manipulate data.