# Python-learning-Day1

1. Comments in Python

Single-line comments: Use the # symbol. Everything after # on that line is ignored by the interpreter.

# This is a single-line comment

print("Hello, World!") # This is an inline comment

Multi-line comments: Technically, Python doesn’t have multi-line comment syntax, but you can use triple quotes (''' or """) as a workaround.

"""

This is a multi-line comment

that spans several lines.

"""

print("Multi-line comments above!")

3. Keywords in Python

Definition: Reserved words that have special meaning in Python. They are part of the syntax and cannot be used as identifiers (variable names).

Examples: if, else, elif, while, for, def, return, True, False, import, class, try, except, finally, with, as, break, continue, pass, global, lambda, nonlocal, yield, etc.

if True:

print("This is a keyword example.")

Note: You can view all keywords using the keyword module:

import keyword

print(keyword.kwlist)

3.1 Identifiers in Python

Definition: Names used to identify variables, functions, classes, modules, etc.

Rules:

It can contain letters (A–Z, a–z), digits (0–9), and underscores (\_).

It cannot start with a digit.

It cannot be a Python keyword.

Case-sensitive (Variable and variable are different).

name = "John" # valid identifier

\_age = 25 # valid identifier

1st\_number = 100 # invalid identifier (cannot start with a digit)

Here's an overview of the requested Python concepts:

1. Comments in Python

Single-line comments: Use the # symbol. Everything after # on that line is ignored by the interpreter.

python

Copy

Edit

# This is a single-line comment

print("Hello, World!") # This is an inline comment

Multi-line comments: Technically, Python doesn’t have multi-line comment syntax, but you can use triple quotes (''' or """) as a workaround.

python

Copy

Edit

"""

This is a multi-line comment

that spans several lines.

"""

print("Multi-line comments above!")

2. Keywords in Python

Definition: Reserved words that have special meaning in Python. They are part of the syntax and cannot be used as identifiers (variable names).

Examples: if, else, elif, while, for, def, return, True, False, import, class, try, except, finally, with, as, break, continue, pass, global, lambda, nonlocal, yield, etc.

python

Copy

Edit

if True:

print("This is a keyword example.")

Note: You can view all keywords using the keyword module:

python

Copy

Edit

import keyword

print(keyword.kwlist)

3. Identifiers in Python

Definition: Names used to identify variables, functions, classes, modules, etc.

Rules:

Can contain letters (A–Z, a–z), digits (0–9), and underscores (\_).

Cannot start with a digit.

Cannot be a Python keyword.

Case-sensitive (Variable and variable are different).

python

Copy

Edit

name = "John" # valid identifier

\_age = 25 # valid identifier

1st\_number = 100 # invalid identifier (cannot start with a digit)

4. Operators in Python

a. Arithmetic Operators

Used for mathematical operations:

+ (Addition)

- (Subtraction)

\* (Multiplication)

/ (Division)

// (Floor Division)

% (Modulus - remainder)

\*\* (Exponentiation)

a = 10

b = 3

print(a + b) # 13

print(a - b) # 7

print(a \* b) # 30

print(a / b) # 3.333...

print(a // b) # 3

print(a % b) # 1

print(a \*\* b) # 1000

b. Comparison Operators

Used to compare values, returning 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

print(x > y) # False

print(x < y) # True

print(x >= 5) # True

print(y <= 10) # True

Here's an overview of the requested Python concepts:

1. Comments in Python

Single-line comments: Use the # symbol. Everything after # on that line is ignored by the interpreter.

python

Copy

Edit

# This is a single-line comment

print("Hello, World!") # This is an inline comment

Multi-line comments: Technically, Python doesn’t have multi-line comment syntax, but you can use triple quotes (''' or """) as a workaround.

python

Copy

Edit

"""

This is a multi-line comment

that spans several lines.

"""

print("Multi-line comments above!")

2. Keywords in Python

Definition: Reserved words that have special meaning in Python. They are part of the syntax and cannot be used as identifiers (variable names).

Examples: if, else, elif, while, for, def, return, True, False, import, class, try, except, finally, with, as, break, continue, pass, global, lambda, nonlocal, yield, etc.

python

Copy

Edit

if True:

print("This is a keyword example.")

Note: You can view all keywords using the keyword module:

python

Copy

Edit

import keyword

print(keyword.kwlist)

3. Identifiers in Python

Definition: Names used to identify variables, functions, classes, modules, etc.

Rules:

Can contain letters (A–Z, a–z), digits (0–9), and underscores (\_).

Cannot start with a digit.

Cannot be a Python keyword.

Case-sensitive (Variable and variable are different).

python

Copy

Edit

name = "John" # valid identifier

\_age = 25 # valid identifier

1st\_number = 100 # invalid identifier (cannot start with a digit)

4. Operators in Python

a. Arithmetic Operators

Used for mathematical operations:

+ (Addition)

- (Subtraction)

\* (Multiplication)

/ (Division)

// (Floor Division)

% (Modulus - remainder)

\*\* (Exponentiation)

python

Copy

Edit

a = 10

b = 3

print(a + b) # 13

print(a - b) # 7

print(a \* b) # 30

print(a / b) # 3.333...

print(a // b) # 3

print(a % b) # 1

print(a \*\* b) # 1000

b. Comparison Operators

Used to compare values, returning True or False:

== (Equal to)

!= (Not equal to)

> (Greater than)

< (Less than)

>= (Greater than or equal to)

<= (Less than or equal to)

python

Copy

Edit

x = 5

y = 10

print(x == y) # False

print(x != y) # True

print(x > y) # False

print(x < y) # True

print(x >= 5) # True

print(y <= 10) # True

c. Assignment Operators

Used to assign values to variables:

= (Simple assignment)

+= (Add and assign)

-= (Subtract and assign)

\*= (Multiply and assign)

/= (Divide and assign)

//= (Floor divide and assign)

%= (Modulus and assign)

\*\*= (Exponentiate and assign)

a = 5 # Assign 5 to a

a += 3 # Equivalent to a = a + 3 (a becomes 8)

a \*= 2 # Equivalent to a = a \* 2 (a becomes 16)

a -= 4 # Equivalent to a = a - 4 (a becomes 12)

a /= 3 # Equivalent to a = a / 3 (a becomes 4.0)