Day 2

Python

Keyword

Keywords in Python are reserved words that can not be used as a variable name, function name, or any other identifier.

Using code to print all the keywords in Python:

```
import keyword

# printing all keywords at once using "kwlist()"
print("The list of keywords is : ")
print(keyword.kwlist)
```

Output:

```
The list of keywords are:
['False', 'None', 'True',"__peg_parser__ 'and', 'as', 'assert', 'async', 'await', 'break',
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if',
'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

Identifiers:

Identifier is a user-defined name given to a variable, function, class, module, etc. The identifier is a combination of character digits and an underscore. They are case-sensitive i.e., **'num' and 'Num' and 'NUM'** are three different identifiers in python. It is a good programming practice to give meaningful names to identifiers to make the code understandable.

We can also use the Python string isidentifier() method to check whether a string is a valid identifier or not.

Rules for Naming Python Identifiers

- It cannot be a reserved python keyword.
- It should not contain white space.
- It can be a combination of A-Z, a-z, 0-9, or underscore.
- It should start with an alphabet character or an underscore (_).

- It should not contain any special character other than an underscore (_).

Examples

Valid identifiers:

- name1
- _name1
- _1_name
- name_1

Invalid Identifiers

- !var1
- 1name
- 1_name
- name#1
- name 1

Data Types in Python

Fundamental Data Types

Python has several basic data types that are built into the language. With these types, we can represent numeric values, text and binary data, and Boolean values in our code. So, these data types are the basic building blocks of most Python programs and projects.

The different fundamental data types are;

Numeric types - int, float, and complex
str data type
bytes and bytearray for storing bytes
Boolean values with bool data type

Visualising all fundamental data types and their classes and operations that can be performed on them:

Source Code:

```python
# Integer (int)
a = 10
print("Integer:")
print("Value:", a, "| Type:", type(a))

# Float (float)
b = 3.14
print("\nFloat:")
print("Value:", b, "| Type:", type(b))

# String (str)
c = "Hello"
print("\nString:")
print("Value:", c, "| Type:", type(c))

# Boolean (bool)
d = True
print("\nBoolean:")
print("Value:", d, "| Type:", type(d))

# Operations between them:

# Arithmetic Operations
print("\nArithmetic Operations:")
print(f"Addition (int + float): {a} + {b} = {a + b}")  # int + float
print(f"Multiplication (int * float): {a} * {b} = {a * b}")  # int * float
print(f"Division (float / int): {b} / {a} = {b / a}")  # float / int

# String Operations
print("\nString Operations:")
print(f"Concatenation: {c} + ' World' = {c + ' World'}")  # string concatenation
print(f"Repetition: {c} * 3 = {c * 3}")  # string repetition

# Boolean Operations
print("\nBoolean Operations:")
print(f"Boolean AND: {d} and False = {d and False}")  # boolean AND
print(f"Boolean OR: {d} or False = {d or False}")  # boolean OR
print(f"Boolean NOT: not {d} = {not d}")  # boolean NOT

# Combined Operations
```

```
print("\nCombined Operations:")
print(f"String and Boolean: {c + ' is ' + str(d)}")  # Concatenate string with boolean (after converting boolean to string)
print(f"String and Arithmetic: {c + ' ' + str(a + b)}")  # Concatenate string with the result of arithmetic operation
```

Output:

```
Integer:
Value: 10 | Type: <class 'int'>

Float:
Value: 3.14 | Type: <class 'float'>

String:
Value: Hello | Type: <class 'str'>

Boolean:
Value: True | Type: <class 'bool'>

Arithmetic Operations:
Addition (int + float): 10 + 3.14 = 13.14
Multiplication (int * float): 10 * 3.14 = 31.400000000000002
Division (float / int): 3.14 / 10 = 0.314

String Operations:
Concatenation: Hello + ' World' = Hello World
Repetition: Hello * 3 = HelloHelloHello

Boolean Operations:
Boolean AND: True and False = False
Boolean OR: True or False = True
Boolean NOT: not True = False

Combined Operations:
String and Boolean: Hello is True
String and Arithmetic: Hello 13.14
```