🚀 Python Basics - Identifiers, Variables & Data Types

Features of Python 🐍

1. **Easy to Learn & Readable** – Simple syntax like English.
2. **Interpreted Language** – No compilation needed, executes line by line.
3. **Dynamically Typed** – No need to declare variable types.
4. **Object-Oriented** – Supports classes and objects.
5. **Extensive Libraries** – Rich set of built-in modules (NumPy, Pandas, etc.).
6. **Platform Independent** – Runs on Windows, macOS, Linux.
7. **Memory Management** – Automatic garbage collection.
8. **Multi-Paradigm** – Supports procedural, functional, and OOP styles.
9. **Scalability** – Used in web dev, AI, ML, data science, and more!

🆔 Identifiers

Identifiers are the names used in Python to define functions, modules, methods, classes, and variables.

✅ Rules for Identifiers:

- Must start with a letter (A-Z or a-z) or an underscore (_).
- Followed by letters, digits (0-9), or underscores.
- Cannot be a reserved keyword.
- Case-sensitive (e.g., **name** and **Name** are different).
- Cannot contain special characters like @, #, $, %, **etc**.

💡 Example:

```
my_variable = 10  # ✅Valid identifier
_variable = 20    # ✅Valid identifier
2variable = 30    # ❌ Invalid identifier (Cannot start with a number)
class = 40        # ❌ Invalid identifier (Reserved keyword)
```

🔢 Variables

Variables are containers that hold data values in Python.

📌 Declaring Variables:

```
x = 5           # Integer
name = "Alice" # String
is_valid = True # Boolean
```

> 📝 **Note:** Python is dynamically typed, meaning you don't need to declare the type explicitly.

📊 Data Types

Data types define the type of value a variable can store.

Data Type	Example	Description
Integer	`10, 100, -5`	Whole numbers
Float	`10.6, 695.3`	Decimal numbers
Boolean	`True, False`	Represents True/False values
Byte	`b'128'`	Sequence of bytes
ByteArray	`bytearray(b'124')`	Mutable byte sequence
None	`None`	Represents a null value

Data Type	Example	Description
Complex	`10+20j`	Complex numbers (real & imaginary)
List	`[1, 12, 23, 3, 5]`	Ordered, mutable collection
Tuple	`(1, 2, 3, 6, 4, 7)`	Ordered, immutable collection
Set	`{1, 2, 5, 47, 3}`	Unordered, unique values
Dictionary	`{'a': 1, 'b': 2}`	Key-value pairs
String	`"abcdef"`	Sequence of characters
FrozenSet	`frozenset({1,2,3,5})`	Immutable set
Range	`range(1, 10)`	Sequence of numbers

🔍 Checking Data Types

Use the

`type()`

function to check the type of a variable.

📌 Example:

```
data = {1, 2, 3, 4, 5, 6}  # Set
data = {'course': "Python"}  # Dictionary
data = "Python"  # String
data = 10 + 16j  # Complex Number

print(data)
print(type(data))
```

🎯 **Summary:**

- Identifiers follow naming rules and cannot use special characters.
- Variables store data and Python dynamically assigns types.
- Different data types exist in Python, each with a specific purpose.
- Use
  ```
  type()
  ```
  to check the data type of a variable.