Python

Day2: **Advanced Rules for Variables in Python**

1. **Naming Rules:**
   * Can contain letters (a–z, A–Z), digits (0–9), and underscores \_.
   * Cannot start with a digit.
   * Cannot be a Python **keyword** or built-in function name.
   * Case-sensitive (age ≠ Age).( Python, C / C++,Java, JavaScript)but not case sensitive(SQL, BASIC)
2. **Dynamic Typing:**
   * Python variables **don’t require type declaration**.
   * The type is determined automatically based on the value.

x = 10 # int

x = "Ram" # str (type changes dynamically)

1. **Multiple Assignment:**
   * Python allows assigning multiple variables in one line:

a, b, c = 1, 2, 3

**Valid Examples:**

student\_name = "Ram"

\_score = 95

totalMarks123 = 100

1. **Invalid Examples:**

1st\_name = "Ram" # starts with digit ❌

for = 10 # keyword ❌

1. **Memory Reference:**
   * Variables in Python are **references to objects** in memory.
   * Changing the value of a variable points it to a new object.
2. **Global & Local Scope:**
   * Variables can be **global** (accessible anywhere) or **local** (inside a function only).
   * Sure Ram! Here’s a simple explanation of **why we use different naming cases** in Python and programming:

**🔹 Why We Use Different Naming Cases**

1. **Readability** ✅

* Naming conventions make code **easy to read and understand**.
* Example: student\_name = "Ram" # easy to read (snake\_case)

1. **Consistency** 🎯

* Using a standard case across a project keeps code **consistent**.
* Easier for **team collaboration**.

1. **Indicates Purpose / Type** 🔍

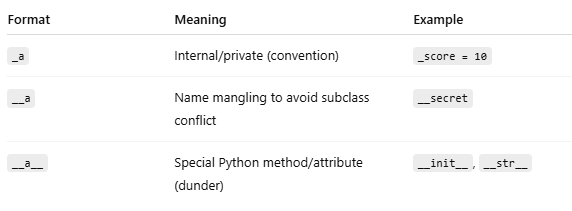
* Different cases help programmers **know what the variable represents**:
  + snake\_case → regular variable or function
  + PascalCase → class name
  + UPPER\_CASE → constant value

1. **Avoids Errors** ⚠️

* Helps avoid naming conflicts and mistakes in large programs.

✅ **Summary Table:**

| **Case** | **Use in Python** | **Example** |
| --- | --- | --- |
| Snake Case (All lowercase letters, words separated by \_) | Variables, functions | student\_name |
| Camel Case(First word lowercase, following words start with uppercase, no spaces) | Rare in Python, common in JS/Java(Variables and functions in some other languages like Java, JavaScript) | studentName |
| Pascal Case(Every word starts with uppercase, no spaces) | Classes | StudentName |
| Kebab Case(not common in Python, more in URLs/filenames) | URLs, filenames | student-name |
| Upper Case(All letters uppercase, words separated by \_) | Constants | PI, MAX\_VALUE |



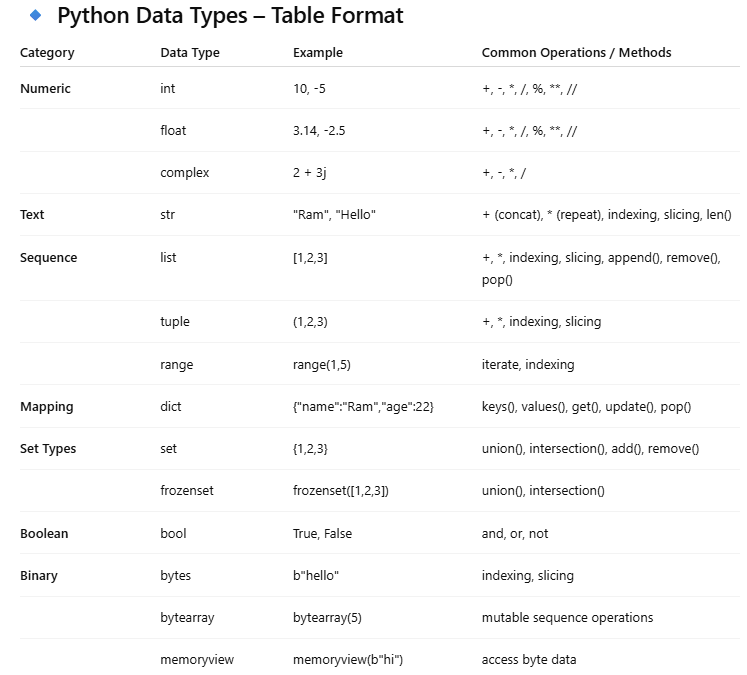
Paxadi ramro snaga details ma xa

Mainly 36 reserver word/keyword

Code : Import keyword

Print(keyword.kwlist)

['False', 'None', 'True', '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']



**🔹 Number Systems in Python**

Python supports **different number systems** for integers.

| **System** | **Prefix in Python** | **Example** | **Base** |
| --- | --- | --- | --- |
| **Binary** | 0b or 0B | 0b1010 | 2 |
| **Decimal** | No prefix | 10 | 10 |
| **Octal** | 0o or 0O | 0o12 | 8 |
| **Hexadecimal** | 0x or 0X | 0xA | 16 |

**🔹 Examples in Python**

# Binary

x = 0b1010

print(x) # 10 (decimal)

# Decimal

y = 10

print(y) # 10

# Octal

z = 0o12

print(z) # 10 (decimal)

# Hexadecimal

h = 0xA

print(h) # 10 (decimal)

**🔹 Key Points**

1. Python **automatically converts** to decimal when doing calculations.
2. Use **bin(), oct(), hex()** to convert decimal numbers to other bases:

n = 10

print(bin(n)) # 0b1010

print(oct(n)) # 0o12

print(hex(n)) # 0xa

**🔹 id() in Python**

**Definition:**

* The **id() function** in Python returns the **unique identity (address) of an object** in memory.
* Every object in Python has a **unique ID**, which is its **memory location**.