1. Variables and Data Types

In Python, variables are containers for storing data values. Unlike some other languages, Python has no command for declaring a variable type; it is determined automatically when you assign a value.

Common Data Types

Data Type	Keyword	Description	Example
Integer	int	Whole numbers (positive or negative)	10, -5, 0
Float	float	Numbers with decimals	10.5, 3.14
String	str	Text enclosed in quotes	"Hello", 'A'
Boolean	bool	Represents logic	True, False

Code Example: Assignment

```
# Correct Variable Assignment
student_name = "Alex"       # String
student_age = 17            # Integer
average_score = 85.5        # Float
has_passed = True           # Boolean

# Printing the type to see what Python thinks it is
print(type(student_name))   # Output: <class 'str'>
print(type(average_score))  # Output: <class 'float'>
```

NOTE

- DO: Use snake_case for variable names (e.g., total_score instead of TotalScore).
- DON'T: Start variable names with a number (e.g., 1st_student will cause an error).
- CAUTION: Python is Case Sensitive. Age and age are treated as two completely different variables.

2. Operators

Operators allow us to perform operations on variables and values.

A. Mathematical Operators

Operator	Name	Description	Example Code	Result
+	Addition	Adds values	5 + 3	8
-	Subtraction	Subtracts values	5 - 3	2
*	Multiplication	Multiplies values	5 * 3	15
/	Division	Divides (result is always float)	10 / 2	5.0
//	Floor Division	Divides and removes decimal	10 // 3	3
%	Modulus	Returns the remainder	10 % 3	1

B. Comparison & Logical Operators

Operator Type	Symbol	Meaning	Example	Result
Comparison	==	Equal to	5 == 5	True
	!=	Not equal to	5 != 3	True
	>	Greater than	5 > 2	True
	<	Less than	5 < 10	True
Logical	and	Returns True if both are true	True and False	False
	or	Returns True if one is true	True or False	True
	not	Reverses the result	not True	False

Code Example: Operators

```
x = 10
y = 3

# Math
print(x // y)  # Output: 3 (Because 3 goes into 10 three times)
print(x % y)   # Output: 1 (The remainder of 10 divided by 3)

# Logic
is_adult = True
has_ticket = False

# Both must be true to enter
if is_adult and has_ticket:
    print("Enter")
else:
    print("Stop") # This will print because has_ticket is False
```

3. Decision Making (If, Elif, Else)

This allows the program to take different paths based on conditions.

Code Example

```
score = 85


if score >= 90:
    print("Grade: A")
elif score >= 80:
    print("Grade: B")   # This block will run
elif score >= 70:
    print("Grade: C")
else:
    print("Grade: Fail")
```

NOTE

- DO: Remember the Indent. Python relies on spacing (tab or 4 spaces) to know what code belongs inside the if statement.
- DON'T: Forget the colon : at the end of the if line.
- CAUTION: Do not confuse assignment (=) with comparison (==). if x = 5 is an error. if x == 5 is correct.

4. Loops (Iteration)

Loops allow you to repeat a block of code multiple times.

A. For Loop (Count-Controlled)

Used when you know exactly how many times you want to loop.

```
# 'range(5)' generates numbers 0, 1, 2, 3, 4
for i in range(5):
    print("Iteration:", i)
```

B. While Loop (Condition-Controlled)

Used when you want to loop until a condition becomes false.

```
count = 0
while count < 3:
    print("Count is:", count)
    count = count + 1  # IMPORTANT: Increment the counter
```

NOTE

- DO: Use for loops when working with ranges or lists.
- CAUTION: In a while loop, if you forget to change the condition (like count = count + 1), you will create an Infinite Loop that crashes the program.
- NOTE: range(1, 5) includes 1 but excludes 5. It prints: 1, 2, 3, 4.

5. Functions

Functions are reusable blocks of code.

Function Type	Description	Example Usage
Void (Non-Return)	Performs an action (like printing) but gives nothing back to the main code.	Displaying a menu or a welcome message.
Return Type	Performs a calculation and sends the result back to be used later.	Calculating tax, adding scores, converting units.

Code Example

Python

```python
# 1. Non-Return Function (Void)
def greet_user(name):
    print(f"Hello, {name}!")
    # Just prints to screen, doesn't save any data

# 2. Return Function
def calculate_square(number):
    result = number * number
    return result
    # Sends the value back

# Main Program
greet_user("Sarah")          # Calls the void function

my_number = calculate_square(4) # Calls the return function and saves
answer
print(my_number)             # Output: 16
```

NOTE

- DO: Define functions at the top of your script before you call them.
- DON'T: Put any code *after* the return line inside a function; it will never run (unreachable code).
- CAUTION: Variables created inside a function (local variables) cannot be used outside that function.