Module 2: Variables, Data Types, and Operators

2.1 Variables and Assignment

A **variable** is a name that refers to a value stored in the computer's memory. The process of creating a variable and linking it to a value is called **assignment**.

Defining Variables (No explicit declaration needed)

In Python, you don't need to specify the variable's type before using it (unlike C++ or Java). You simply assign a value using the single equals sign (=).

Example Program:

Python
```
# Simple variable assignment
age = 30
name = "Alice"
is_student = False
pi_value = 3.14159

# The variables are created and assigned values instantly.
print(f"Name: {name}, Age: {age}")
```

Variable Naming Rules and Conventions (e.g., PEP 8)

- **Rules (Must Follow):**
  1. Can only contain letters (a-z, A-Z), digits (0-9), and the underscore (_).
  2. Must start with a letter or an underscore. Cannot start with a digit.
  3. Variable names are **case-sensitive** (e.g., Age is different from age).
  4. Cannot be a reserved Python keyword (e.g., if, for, while, print).
- **Conventions (PEP 8 - Best Practice):**
  1. Use **snake_case** (all lowercase, words separated by underscores).
     - *Good:* total_sales, user_name
     - *Bad:* TotalSales, totalsales
  2. Choose meaningful, descriptive names.
     - *Good:* price_per_unit
     - *Bad:* ppu (unless commonly understood abbreviation)
  3. A single leading underscore (_variable) suggests an internal or private use (a convention, not strictly enforced).

Dynamic Typing (Type is checked at runtime)

Python is **dynamically typed**, meaning the type of a variable is determined by the value it currently holds, and a variable can change its type during program execution.

Example Program:

Python
```
x = 10       # x is an integer (int)
```

```
print(f"Type of x: {type(x)}")

x = "Hello"   # Now x is a string (str)
print(f"Type of x: {type(x)}")
```

The id() and type() functions

- **type(variable):** Returns the type of the object the variable refers to.
- **id(variable):** Returns the identity (memory address) of the object. For two variables to be identical (using is), they must have the same id().

Example Program:

Python
```
number = 42
name = "Python"

print(f"The value is: {number}, its type is: {type(number)}, and its memory ID is: {id(number)}")
print(f"The value is: {name}, its type is: {type(name)}, and its memory ID is: {id(name)}")

a = 10
b = 10
print(f"ID of a: {id(a)}")
print(f"ID of b: {id(b)}")
# Since 10 is an immutable object, a and b often point to the same memory location
```

2.2 Built-in Data Types (Primitives)

Numeric Types: int, float, complex

- **int (Integer):** Whole numbers (positive, negative, or zero) without a decimal point.
- **float (Floating Point):** Numbers with a decimal point.
- **complex (Complex Number):** Numbers with a real and an imaginary part, written as $a + bj$.

Example Program:

Python
```
integer_num = 100
float_num = 100.0  # Even though it's 100, the decimal point makes it a float
large_num = 9876543210
pi = 3.14159265
complex_num = 3 + 4j

print(f"Type of {integer_num}: {type(integer_num)}")
print(f"Type of {float_num}: {type(float_num)}")
print(f"Type of {complex_num}: {type(complex_num)}")
```

Boolean Type: bool (True, False)

The Boolean type represents truth values. It has only two possible values: **True** and **False** (note the capitalization). These are crucial for control flow.

Example Program:

Python
```
is_active = True
is_logged_in = False

print(f"Is active: {is_active}, Type: {type(is_active)}")

# Booleans can be treated as numbers (True is 1, False is 0)
result = is_active + is_logged_in  # 1 + 0 = 1
print(f"Result of arithmetic: {result}")
```

Type Conversion (Type Casting): int(), float(), str(), etc.

Python provides built-in functions to convert (cast) values from one type to another.

Function	**Purpose**
int(x)	Converts x to an integer.
float(x)	Converts x to a floating-point number.
str(x)	Converts x to a string.
bool(x)	Converts x to a boolean.

Example Program:

Python
```
num_str = "123"
decimal_num = 7.99
is_data = 1

# String to Integer
int_val = int(num_str)  # 123
print(f"String '{num_str}' as int: {int_val}")

# Float to Integer (truncates the decimal part)
int_from_float = int(decimal_num)  # 7
print(f"Float {decimal_num} as int: {int_from_float}")

# Integer/Float to String
str_val = str(int_val) + " dollars"
print(f"String + int: {str_val}")

# Integer to Boolean (0 is False, any non-zero number is True)
bool_val = bool(is_data)  # True
print(f"Integer {is_data} as bool: {bool_val}")
```