

2

Chapter 2

🧠 Python Notes – Chapter 2: Variables, Data Types, Operators, and Input

📌 1) **variables.py** – Simple Variables & Arithmetic

📜 Code:

```
a = 12 # Integer value assigned to variable 'a'  
b = 829 # Integer value assigned to variable 'b'  
  
print(b / a) # Division operator returns float (even if both operands are integers)
```



Explanation:

- Variables in Python can store values like numbers, text, etc.
- Python is **dynamically typed**, meaning you don't need to declare the data type.
- `/` performs **float division** (e.g., `10/3` = `3.333...`)
- For **integer division**, use `//` (e.g., `10//3` = `3`)

📌 2) database.py – Common Data Types

Code:

```
a = 8723 # Integer  
b = 78076.3496346 # Float (decimal number)  
c = True # Boolean: Can only be True or False  
d = "Prathamesh" # String (text)  
e = None # Special null value in Python  
  
print(d is e) # Checks whether both variables refer to the same object (identity)
```



Explanation:

- **int** – Whole numbers
- **float** – Decimal numbers
- **bool** – `True` or `False`
- **str** – Text
- **NoneType** – Represents **no value / empty**

🔍 `is` operator: Checks **object identity**, not just value

Example: `a = None` , `b = None` , then `a is b` → `True`

📌 3) rules.py – Variable Naming Rules

⚠ Rules to Remember:

- Can't start with a **number** (e.g., `1value` is invalid)
- Can't start with **special characters** (`@` , `#` , `&` , etc.)
- Can't have **spaces** (`my var = 5` is invalid)

✓ Valid examples:

```
name = "Prathamesh"  
my_var2 = 10  
age_17 = True
```

 Use **snake_case** for variables (`first_name` , `user_age`)

4) `operators.py` – Operators in Python

Types of Operators:

Type	Example	Description
Arithmetic	<code>+, -, *, /, %, //, **</code>	Math operations
Assignment	<code>=, +=, -=</code>	Assign/update variable values
Comparison	<code>==, !=, <, >, <=, >=</code>	Return Boolean
Logical	<code>and, or, not</code>	Combine Boolean values

Arithmetic Operators:

```
a = 12
b = -23
c = a + b # Addition: 12 + (-23)
print(c)
```

Formulas:

-  Addition
- Subtraction
- Multiplication
-  Division (float)
-  Floor Division (returns int)
-  Modulus (remainder)
-  Exponentiation (e.g., `2**3 = 8`)

Assignment Operators:

```
d = 73
d += 7 # Same as d = d + 7
print(d)
```

Other examples:

- `x -= 1` → Decrement
- `x *= 2` → Multiply and store
- `x /= 2` → Divide and store

✓ Comparison Operators:

```
e = 5 < 3    # False  
f = 5 == 4    # False  
g = 3 >= 3   # True
```

```
print(e)  
print(f)  
print(g)
```

📌 Always returns **True/False**

✓ Logical Operators:

```
# OR Operator  
print("True or False is ", True or False)
```

```
# AND Operator  
print("True and False is ", True and False)
```

```
# NOT Operator  
print("not(True) =", not(True))
```

12 34 Truth Tables:

A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

`not(True)` → `False`

`not(False)` → `True`

✓ `not` only needs one operand (unary operator)

12
34

Top 10 Logical Operators – Formula Sheet

Operator	Name	Formula
AND	Logical AND	$A * B$
OR	Logical OR	$A + B$
NOT	Logical NOT	$A \bar{+}$ or $\neg A$
NAND	Not AND	$(A * B) \bar{+}$
NOR	Not OR	$(A + B) \bar{*}$
XOR	Exclusive OR	$A \bar{*} B + AB \bar{+}$
XNOR	Exclusive NOR (Equality)	$AB + A \bar{*} B \bar{+}$
IMPLICATION	A implies B ($A \rightarrow B$)	$A \bar{+} B$
REVERSE IMPL.	B implies A ($B \rightarrow A$)	$B \bar{+} A$
BICONDITIONAL	A if and only if B (\equiv)	$AB + A \bar{*} B \bar{+}$

5) `type.py` – Type Checking and Type Casting

Code:

```
a = 34.334
b = type(a) # Returns <class 'float'>
print(b)
```

```
c = "86.6"
d = float(c) # Converts string to float
e = type(d)
print(d)
print(e)
```

Explanation:

- `type()` tells you what data type a variable is.
- Type Casting:
 - `int("5")` → 5
 - `float("6.7")` → 6.7
 - `str(88)` → "88"

 **Be careful:** `int("4.5")` will throw an error. You need to convert to float first.

6) `input.py` – Taking User Input

Code:

```
a = int(input("Enter Number 1: "))
b = int(input("Enter Number 2: "))
print("The sum of these numbers is:", a + b)
```

Explanation:

- `input()` takes input from the user in **string format**.
- Wrapping with `int()` or `float()` converts it to a number.

 Without `int()`, `input()` gives a **string**, so:

```
print("1" + "2") # Output: 12 (concatenation)
```

 But with `int()`:

```
print(1 + 2) # Output: 3 (math)
```

Practice Problems

Problem 1: Add two numbers

Code:

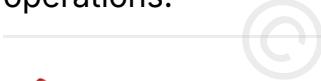
```
a = 28  
b = 38748.28373  
print(a+b)
```



Notes:

- `a` is an integer, `b` is a float.
- Python automatically converts the result to float if any operand is float.
- **Concept Used:** Basic arithmetic using `+` operator.

Learning: Python handles type conversion **automatically** during arithmetic operations.



📌 Problem 2: Find Remainder

Question Guess: Write a program to find the remainder when one number is divided by another.

Code:

```
a = int(input("Enter number 1 :"))  
b = int(input("Enter number 2 :"))  
print(a % b)
```



Notes:

- `%` operator is called the **modulus** operator.
- It returns the **remainder** after division.
- Example: `10 % 3 = 1`



📌 Problem 3: Check Data Type from Input



Question Guess: Write a program to check the data type of user input.

Code:

```
a = input("Enter Anything : ")  
b = type(a)  
print(b)
```

```
# One-liner version  
c = type(input("Enter Anything : "))  
print(c)
```



Notes:

- `input()` always returns data in **string** (`str`) **format**, even if you type a number.
- `type()` function is used to check the data type.

Learning: Always **type-cast** input when needed (e.g. to `int()` or `float()`).



Problem 4: Compare Two Numbers

Question Guess: Check if the second number is greater than the first number.

Code:

```
a = int(input("Enter 1st number : "))  
b = int(input("Enter 2nd number : "))  
c = b > a  
print("The denotion that the second number is greater than first number is",  
c)
```



Notes:

- `>` is a **comparison operator**.
- The result (`c`) is a **boolean value** (`True` or `False`).



Problem 5: Average of Two Numbers

Question Guess: Write a program to find the average of two numbers entered by the user.

Code:

```
a = int(input("Enter first number : "))  
b = int(input("Enter second number : "))  
c = (a + b) / 2  
print(c)
```



Notes:

- Use `()` to group the addition before dividing.
- The average is calculated using the formula: $\text{Average} = \frac{2a+b}{2}$
$$\text{Average} = \frac{(a+b)}{2}$$



Problem 6: Square of a Number



Question Guess: Write a program to find the square of a number.

Code:

```
a = int(input("Enter the number: "))
print("The square of number is:", a * a)
print("The square of number is:", a ** 2)
```



Notes:

- `a * a` is the usual multiplication.
- `a ** 2` is the **exponentiation operator** (i.e. a raised to the power of 2).
- ~~`X a ^ 2`~~ is **NOT** squaring — it's bitwise XOR.



Learning:

- `*` is for multiplication
- `**` is for power
- `^` is for bitwise XOR (used in logic)



Chapter 2 Summary

Topic	Key Concepts Learned
Variables	Can hold int, float, str, bool, None; dynamic typing in Python
Rules for Naming	Cannot start with number, special characters, or space
Operators	Arithmetic, Assignment, Comparison, Logical; all behave like in math
Logical Operators	AND (<code>*</code>), OR (<code>+</code>), NOT (<code>~</code>); plus advanced ones like XOR, NAND, etc.

Topic	Key Concepts Learned
Data Types	Use <code>type()</code> to check; <code>input()</code> returns string by default
Type Conversion	Convert with <code>int()</code> , <code>float()</code> , <code>str()</code> as needed
User Input	Always string by default, cast it if math is required
Common Problems	Add, average, remainder, compare, square, and check data types
Exponentiation	Use <code>**</code> for powers instead of <code>^</code> (which is bitwise XOR)
Practical Notes	Logic building, input handling, type awareness, and operator usage are crucial