Unit 2: Introduction to Python

Python Programming — DI02032011

Milav Dabgar

Lecturer, Dept of ICT
GP Palanpur

April 2, 2026

Unit 2 — Table of Contents

Syllabus Coverage

Weightage: ≈14% | Hours: 6

2.1 Introduction to Python

Features & Applications

What is Python?

Definition

Python is a **high-level, interpreted, general-purpose** programming language known for its simple syntax and readability. Created by **Guido van Rossum** (released 1991).

Quick Facts

Paradigm	Object-Oriented, Functional, Procedural
Typing	Dynamically typed
Extension	`.py`
Interpreter	CPython (default), PyPy, Jython
Version	Python 3.x (current)
License	Open-source (PSF License)

Features of Python (Su'25, W'25 — 3M)

Key Features of Python

Simple & Readable	English-like syntax, beginner-friendly
Interpreted	Executes line-by-line, no compilation needed
Dynamically Typed	No need to declare variable types
Object-Oriented	Supports classes, objects, inheritance
Large Library	Rich standard library (math, os, re, etc.)
Platform-Independent	Runs on Windows, Linux, macOS
Free & Open-Source	Available at python.org for free
High-Level	Handles memory management automatically

Applications of Python (Su'25 OR, W'25 OR — 3M)

Where is Python Used?

Web Development	Django, Flask frameworks
Data Science	NumPy, Pandas, Matplotlib
Machine Learning	TensorFlow, PyTorch, scikit-learn
Automation/Scripting	Automate repetitive tasks
Scientific Computing	SciPy for research and analysis
Game Development	Pygame library
Desktop Applications	Tkinter, PyQt
Cybersecurity	Penetration testing, scripting

2.2 Program Structure, Keywords

Identifiers & Variables

Basic Python Program Structure

Structure of a Python Program

```
# This is a comment
# 1. Import statements (optional)
import math

# 2. Variable declarations
name = "GTU Student"
marks = 85

# 3. Main logic
print("Name:", name)
print("Marks:", marks)
print("Square root:", math.sqrt(marks))
```

Python Keywords

Definition

Keywords are **reserved words** with predefined meanings. They cannot be used as variable names.

Python Keywords (Selected)

`False`	`True`	`None`	`and`	`or`	`not`
`if`	`elif`	`else`	`for`	`while`	`break`
`continue`	`def`	`return`	`class`	`import`	`from`
`in`	`is`	`lambda`	`pass`	`try`	`except`
`global`	`del`	`with`	`as`	`raise`	`yield`

Identifiers & Variables (W'23 OR — 3M)

Identifier

Name given to a variable, function, class or object.

Rules for Identifiers / Variables

1. Must start with a **letter** (a–z, A–Z) or **underscore** _
2. Can contain letters, digits (0–9) and underscores
3. **Cannot** start with a digit
4. **Case-sensitive**: `age` ≠ `Age` ≠ `AGE`
5. Cannot be a keyword (`for`, `if`, etc.)
6. No spaces or special characters allowed

Examples

Valid: `name`, `_count`, `total_marks`, `x1`
Invalid: `2name`, `for`, `my-var`, `hello world`

2.3 Data Types

int, float, str, bool, complex

Python Data Types (W'23, W'24, Su'25, W'25 — 3/7M)

Built-in Data Types

Type	Example	Description
int	`x = 10`	Whole numbers, no decimal
float	`y = 3.14`	Decimal/real numbers
str	`s = "Hello"`	Text / sequence of characters
bool	`b = True`	True or False values
complex	`c = 3+4j`	Real + imaginary part
list	`l = [1,2,3]`	Ordered, mutable collection
tuple	`t = (1,2,3)`	Ordered, immutable collection
dict	`d = {"a":1}`	Key-value pairs
set	`s = {1,2,3}`	Unordered, unique elements

Identifying Data Types

```
x = 10              # int
y = 3.14            # float
name = "GTU"        # str
flag = True         # bool
c = 2 + 3j          # complex

print(type(x))    # <class 'int'>
print(type(y))    # <class 'float'>
print(type(name))# <class 'str'>
print(type(flag))# <class 'bool'>
```

Type Conversion (Curriculum — 2.4)

Implicit (Automatic)

Python converts type automatically when safe.

```
x = 5       # int
y = 2.0     # float
z = x + y   # 7.0 (float)
print(type(z))
# <class 'float'>
```

Explicit (Manual)

Use `int()`, `float()`, `str()` etc.

```
a = "123"
b = int(a)    # 123
c = float(a) # 123.0
d = str(456) # "456"
e = bool(0)   # False
e = bool(5)   # True
```

Python Operators — Overview (W'24, Su'24 — 3/7M)

Types of Operators

Type	Examples
Arithmetic	`+`, `-`, `*`, `/`, `//`, `%`, `**`
Comparison	`==`, `!=`, `>`, `<`, `>=`, `<=`
Logical	`and`, `or`, `not`
Assignment	`=`, `+=`, `-=`, `*=`, `/=`
Bitwise	`&`, `|`, `^`, `~`, `<<`, `>>`
Membership	`in`, `not in`
Identity	`is`, `is not`

2.4 Operators in Detail

Arithmetic, Comparison, Logical, Assignment, Membership

Arithmetic Operators (Su'23 OR, Su'24 OR — 4M)

Arithmetic Operators with Examples

Op	Name	Example	Result
+	Addition	5 + 3	8
-	Subtraction	5 - 3	2
*	Multiplication	5 * 3	15
/	Division	7 / 2	3.5
//	Floor Division	7 // 2	3
%	Modulus	7 % 2	1
**	Exponentiation	2 ** 3	8

Comparison Operators (W'23 — 7M; W'25 OR — 7M)

Comparison (Relational) Operators

Op	Meaning	Example	Result
==	Equal to	5 == 5	True
!=	Not equal to	5 != 3	True
>	Greater than	5 > 3	True
<	Less than	3 < 5	True
>=	Greater or equal	5 >= 5	True
<=	Less or equal	3 <= 5	True

```
x, y = 10, 20
print(x > y)    # False
print(x == 10) # True
print(x != y)   # True
```

Logical Operators (Su'24 — 7M)

Logical Operators — and, or, not

Op	Meaning	Example	Result
`and`	Both True	`T and F`	False
`or`	Either True	`T or F`	True
`not`	Negation	`not T`	False

```
x, y = 10, 20
print(x > 5 and y > 5)    # True
print(x > 15 or y > 15)   # True
print(not(x > 5))          # False
```

Truth Table (and)

T and T = T T and F = F F and T = F F and F = F

Assignment Operators (W'24 OR — 3M; Su'23 — 7M)

Assignment Operators

Op	Equivalent to	Example	After (x=10)
=	Assign value	`x = 10`	x = 10
+=	x = x + val	`x += 5`	x = 15
-=	x = x - val	`x -= 3`	x = 7
*=	x = x * val	`x *= 2`	x = 20
/=	x = x / val	`x /= 2`	x = 5.0
//=	x = x // val	`x //= 3`	x = 3
%=	x = x % val	`x %= 3`	x = 1
**=	x = x ** val	`x **= 2`	x = 100

Membership Operators (W'24 — 3M)

Definition

Membership operators test whether a value exists in a sequence (string, list, tuple, set, dict).

in and not in

Op	Description	Returns
`in`	Returns True if value is in sequence	True / False
`not in`	Returns True if value not in sequence	True / False

```
fruits = ["apple","mango","banana"]
print("mango" in fruits)      # True
print("grape" not in fruits) # True
name = "Python"
print("Py" in name)           # True
```

Program: Average of Three Numbers (Su'23 — 4M)

Read three numbers and find average

```
# Program to find average of 3 numbers
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
c = float(input("Enter third number: "))

average = (a + b + c) / 3
print("Average =", average)
```

Sample Output

```
Enter first number: 10
Enter second number: 20
Enter third number: 30
Average = 20.0
```

Program: Celsius to Fahrenheit (W'23 — 7M)

Temperature Conversion

```
# Celsius to Fahrenheit conversion
celsius = float(input("Enter temperature in Celsius: "))

fahrenheit = (celsius * 9/5) + 32

print("Temperature in Fahrenheit =", fahrenheit)
```

Formula & Sample

$$F = \frac{C \times 9}{5} + 32$$

```
Enter temperature in Celsius: 100
Temperature in Fahrenheit = 212.0
```

Program: Simple & Compound Interest (Su'24 OR — 7M)

SI and CI Program

```
p = float(input("Principal: "))
r = float(input("Rate (%): "))
t = float(input("Time (years): "))

si = (p * r * t) / 100
ci = p * ((1 + r/100) ** t) - p

print("Simple Interest =", round(si, 2))
print("Compound Interest =", round(ci, 2))
```

Program: Total & Average Marks (W'24 — 7M)

Calculate total and average of 4 subject marks

```
s1 = float(input("Subject 1 marks: "))
s2 = float(input("Subject 2 marks: "))
s3 = float(input("Subject 3 marks: "))
s4 = float(input("Subject 4 marks: "))

total   = s1 + s2 + s3 + s4
average = total / 4

print("Total Marks   =", total)
print("Average Marks =", average)
```

Program: Square & Cube (W'24 OR — 7M)

Find square and cube of a number

```
num = float(input("Enter a number: "))

square = num ** 2
cube   = num ** 3

print("Square of", num, "=", square)
print("Cube of",   num, "=", cube)
```

Sample Output

```
Enter a number: 5
Square of 5.0 = 25.0
Cube of 5.0 = 125.0
```

Output Prediction — Operators (Su'23 OR, Su'24 OR — 4M)

Predict the Output

```
x = 10
y = 3
print(x * y)    # ?
print(x ** y)   # ?
print(x // y)   # ?
print(x % y)    # ?
print(x > y)    # ?
```

Answers

`x * y = 30` `x ** y = 1000` `x // y = 3` `x % y = 1` `x > y = True`

Comparison Operators — Full Program (Su'23 — 7M)

List all comparison operators with examples

```
a, b = 15, 10
print("a == b :", a == b)   # False
print("a != b :", a != b)   # True
print("a  > b :", a >  b)   # True
print("a  < b :", a <  b)   # False
print("a >= b :", a >= b)   # True
print("a <= b :", a <= b)   # False
```

Logical Operators — Full Program (Su'24 — 7M)

Demonstrate all logical operators

```
x, y = 5, 10

# and operator
print(x > 3 and y > 3)   # True
print(x > 3 and y > 15)  # False

# or operator
print(x > 3 or y > 15)   # True
print(x > 8 or y > 15)   # False

# not operator
print(not(x > 3))          # False
print(not(x > 8))          # True
```

Assignment Operators — Program (Su'23 — 7M)

Demonstrate three assignment operators

```
x = 10
print("Initial x =", x)    # 10

x += 5
print("After x+=5:", x)    # 15

x -= 3
print("After x-=3:", x)    # 12

x *= 2
print("After x*=2:", x)    # 24

x //= 4
print("After x//=4:", x)   # 6
```

2.5 Programs: Data Types

3-mark, 4-mark and 7-mark solutions

Data Types Program (W'24 — 4M)

List data types in Python and explain any three

```python
# int type
age = 25
print("age =", age, "| type:", type(age))

# float type
price = 99.99
print("price =", price, "| type:", type(price))

# str type
name = "GTU"
print("name =", name, "| type:", type(name))

# bool type
flag = True
print("flag =", flag, "| type:", type(flag))
```

All Data Types — 7M Program (Su'25, W'25 — 7M)

List different data types with examples

```
x   = 10                # int
y   = 3.14              # float
s   = "Hello Python"    # str
b   = True              # bool
c   = 2 + 3j            # complex
lst = [1, 2, 3]         # list
tup = (4, 5, 6)         # tuple
dct = {"a": 1}          # dict
st  = {7, 8, 9}         # set

for v in [x,y,s,b,c,lst,tup,dct,st]:
    print(v, "->", type(v).__name__)
```

Types of Operators — 7M (Su'25 OR, W'25 — 7M)

Types of Operators in Python

Arithmetic	+, -, *, /, //, %, **
Comparison	==, !=, >, <, >=, <=
Logical	and, or, not
Assignment	=, +=, -=, *=, /=, //=, %=
Membership	in, not in
Identity	is, is not
Bitwise	&, |, ^, ~, «, »

Two types demonstrated

```
# Arithmetic + Relational
x, y = 12, 5
print(x + y, x // y, x % y) # 17 2 2
print(x > y, x == y)              # True False
```

input() and print() Functions

input() Function

Reads input from user as **string**.

```
name = input("Enter name: ")
age  = int(input("Age: "))
```

print() Function

Displays output to the screen.

```
print("Hello")          # basic
print("Age:", age)      # multi
print(3 + 4)            # expr
print("A", end=" ")     # end
print("B")              # A B
```

Type Safety

Always convert `input()` when numeric:
`x = int(input("..."))` or `x = float(input("..."))`

2.6 GTU PYQ Practice

Su'23 – W'25

3-Mark PYQ Questions

Try These (3 marks)

1. List the features of Python. *[Su'25, W'25]*
2. List the applications of Python. *[Su'25 OR, W'25 OR]*
3. Describe data types in Python with examples. *[W'23]*
4. List out rules for defining variables in Python. *[W'23 OR]*
5. Discuss membership operator of Python with example. *[W'24]*
6. Write a short note on assignment operator in Python. *[W'24 OR]*

4-Mark PYQ Questions

Try These (4 marks)

1. List various data types in Python and explain any three with example. *[W'24]*
2. Develop Python code to read three numbers and find their average. *[Su'23]*
3. Determine the output: `x=10, y=3; x*y, x**y, x//y, x%y, x>y.` *[Su'23 OR, Su'24 OR]*

7-Mark PYQ Questions

Try These (7 marks)

1. List assignment operators; build code to demonstrate any three. *[Su'23]*
2. Develop code to convert temperature from Celsius to Fahrenheit. *[W'23]*
3. List all comparison operators and explain each with Python code. *[W'23 OR]*
4. List all logical operators and explain each with Python code. *[Su'24]*
5. Calculate SI and CI. *[Su'24 OR]*
6. Calculate total and average marks of 4 subject marks. *[W'24]*
7. Find square and cube of a given number. *[W'24 OR]*
8. List different data types and explain all with examples. *[Su'25, W'25]*

Bitwise Operators (W'25 OR — 3M)

Bitwise Operators in Python

Op	Name	Example (a=10, b=6)	Result
&	AND	`10 & 6`	2 (1010 & 0110 = 0010)
|	OR	`10 | 6`	14 (1010 | 0110 = 1110)
^	XOR	`10 ^ 6`	12 (1010 ^ 0110 = 1100)
~	NOT	`~10`	-11 (inverts bits)
<<	Left Shift	`10 << 1`	20 (shift bits left)
>>	Right Shift	`10 >> 1`	5 (shift bits right)

Note

Bitwise operators work on integer binary representations. 10 in binary = 1010; 6 = 0110.

Identity Operators (W'25, Su'25 OR — 3M)

Identity Operators

Operator	Meaning	Returns True if
`is`	Identity check	Both point to **same object** in memory
`is not`	Not Identity	Both point to **different objects**

```
a = [1, 2, 3]
b = [1, 2, 3]
c = a
print(a is c)      # True  (same object)
print(a is b)      # False (different objects)
print(a == b)      # True  (same value)
print(a is not b)  # True
```

print() Function — Formatting (W'25 OR — 3M)

```python
# sep parameter: separator between values
print("A", "B", "C", sep="-")     # A-B-C
print(1, 2, 3, sep=", ")          # 1, 2, 3

# end parameter: what to print at end
print("Hello", end=" ")
print("World")                    # Hello World

# f-string formatting (Python 3.6+)
name = "Raj"
marks = 95.5
print(f"Name: {name}, Marks: {marks:.1f}")
# Name: Raj, Marks: 95.5

# format() method
print("Value = {:.2f}".format(3.14159))  # 3.14
```

Escape Sequences (Su'23 OR — 3M)

Common Escape Sequences

Sequence	Meaning	Example Output
\n	Newline	Moves to next line
\t	Tab	Horizontal tab space
\\	Backslash	Prints \
\"	Double quote	Prints "
\'	Single quote	Prints '
\r	Carriage return	Returns cursor to line start

```
print("Line1\nLine2")      # on two lines
print("Name\tMarks")       # Name    Marks
print("C:\\Python")        # C:\Python
print("He said \"Hi\"")    # He said "Hi"
```

Multiple Assignment & Operator Precedence (W'24 — 3M)

```
# Multiple assignment
a = b = c = 10          # all get 10
x, y, z = 1, 2, 3      # unpacking
a, b = b, a             # swap (Pythonic)

# Augmented assignment
x = 5
x += 3    # x = 8
x -= 2    # x = 6
x *= 4    # x = 24
x //= 5   # x = 4
x **= 2   # x = 16
```

Operator Precedence (High to Low)

** > ~+/- (unary) > * / // % > + - > << >> > & ^ | > == != < > <= >= > not > and > or

input() Function & Type Conversion (W'25 — 3M)

```python
# input() always returns a string
name = input("Enter name: ")      # string
age  = int(input("Enter age: ")) # integer
gpa  = float(input("GPA: "))      # float

# Type conversion functions
print(int("42"))        # 42
print(float("3.14"))    # 3.14
print(str(100))         # '100'
print(bool(0))          # False
print(bool(5))          # True

# Multiple inputs on one line
a, b = map(int, input("a b: ").split())
```

Program: Area of Circle & Triangle (W'25, Su'24 OR — 4M)

```python
import math

# Area of Circle
r = float(input("Enter radius: "))
area = math.pi * r * r
print(f"Area of circle = {area:.2f}")

# Area of Triangle (Heron's Formula)
a = float(input("Side a: "))
b = float(input("Side b: "))
c = float(input("Side c: "))
s = (a + b + c) / 2
area = math.sqrt(s*(s-a)*(s-b)*(s-c))
print(f"Area of triangle = {area:.2f}")
```

Program: BMI Calculator (W'25 OR — 4M)

```
# BMI = weight / (height^2)
weight = float(input("Weight (kg): "))
height = float(input("Height (m): "))

bmi = weight / (height ** 2)
print(f"BMI = {bmi:.2f}")

if bmi < 18.5:
    print("Underweight")
elif bmi < 25:
    print("Normal Weight")
elif bmi < 30:
    print("Overweight")
else:
    print("Obese")
```

Program: Hypotenuse & Power Without ** (Su'25 — 4M)

```python
import math

# Hypotenuse of right triangle
a = float(input("Base: "))
b = float(input("Height: "))
hyp = math.sqrt(a**2 + b**2)
print(f"Hypotenuse = {hyp:.2f}")

# Power without ** operator
base = int(input("Base: "))
exp  = int(input("Exponent: "))
result = 1
for i in range(exp):
    result *= base
print(f"{base}^{exp} = {result}")
```

Number Systems & Conversions (W'25 OR, Su'25 OR — 3M)

Literals

System	Prefix	Example
Binary	`0b` or `0B`	`0b1010` = 10
Octal	`0o` or `0O`	`0o12` = 10
Hexadecimal	`0x` or `0X`	`0xA` = 10

```
# Conversion functions
print(bin(10))    # 0b1010
print(oct(10))    # 0o12
print(hex(10))    # 0xa
print(int('1010', 2))   # binary to int = 10
print(int('12', 8))     # octal to int  = 10
print(int('A', 16))     # hex to int    = 10
```

Comments in Python (Su'23 — 3M)

Types of Comments

Type	Syntax
Single-line comment	`# This is a comment`
Multi-line string (docstring)	`'''...'''` or `"""..."""`

```
# Single-line comment
x = 10    # inline comment

'''
This is a
multi-line comment
(docstring)
'''

def greet():
    """This function prints a greeting."""
    print("Hello!")
```

Output Prediction Practice (Su'24, W'24 — 4M)

```
# Predict output
a = 7
b = 3
print(a // b)      # ?
print(a % b)       # ?
print(a ** b)      # ?
print(a > b and b > 0)   # ?
print(not (a == 7))      # ?
```

Answers

a // b = 2
a % b = 1
a ** b = 343
a>b and b>0 = True
not (a==7) = False

Remember:
// = floor division
% = remainder
** = power

Program: Distance, Volume, Perimeter (Su'25 — 4M)

```python
# Distance using speed and time
speed = float(input("Speed (km/h): "))
time  = float(input("Time (h): "))
dist  = speed * time
print(f"Distance = {dist:.2f} km")

# Volume of cylinder
import math
r = float(input("Radius: "))
h = float(input("Height: "))
vol = math.pi * r * r * h
print(f"Volume = {vol:.2f}")

# Perimeter of rectangle
l = float(input("Length: "))
w = float(input("Width: "))
print(f"Perimeter = {2*(l+w):.2f}")
```

String Basics (W'23 OR, Su'25 — 3M)

```
s = "Python"
print(len(s))          # 6
print(s[0])            # P    (indexing)
print(s[-1])           # n    (last char)
print(s[0:3])          # Pyt  (slicing)
print(s * 2)           # PythonPython
print("Py" in s)       # True

# String is immutable
# s[0] = "p"           # Error!
name = input("Name: ")
print("Hello, " + name + "!")   # concatenation
```

Worked: All Operator Types Program (W'25, Su'25 — 7M)

```python
a, b = 15, 4
# Arithmetic
print(a+b, a-b, a*b, a/b, a//b, a%b, a**b)
# 19 11 60 3.75 3 3 50625

# Comparison
print(a>b, a<b, a==b, a!=b)    # T F F T

# Logical
print(a>5 and b>0)   # True
print(a>20 or b>2)   # True
print(not(a>5))      # False

# Assignment (augmented)
x = 10; x += 5; print(x)   # 15

# Membership
print(15 in [10,15,20])     # True
print("t" in "Python")      # True
```

Chained Comparison & Ternary Expression (W'25 OR — 3M)

```python
# Chained comparison (Pythonic)
x = 15
print(10 < x < 20)    # True
print(1 <= x <= 100) # True

# Ternary (conditional) expression
a, b = 10, 20
max_val = a if a > b else b
print("Max:", max_val)    # Max: 20

# type() and isinstance()
n = 42
print(type(n))              # <class 'int'>
print(isinstance(n, int))   # True
print(isinstance(n, float))# False
```

Quick Revision: Data Types & Variables

Data Type	Example	type()	Mutable
int	`42, -5`	`int`	No
float	`3.14`	`float`	No
str	`"Hi"`	`str`	No
bool	`True`	`bool`	No
list	`[1,2]`	`list`	**Yes**
tuple	`(1,2)`	`tuple`	No
dict	`{k:v}`	`dict`	**Yes**
set	`{1,2}`	`set`	**Yes**

Variable Rules

- Start with letter or _
- Case-sensitive: `Name` $\neq$ `name`
- No spaces or special chars (except _)
- Cannot use Python keywords (`if`, `for`, `while`…)

Worked: Largest of Three Numbers (Su'24, W'23 — 4M)

```
# Method 1: if-elif-else
a = int(input("a: "))
b = int(input("b: "))
c = int(input("c: "))

if a >= b and a >= c:
    print("Largest:", a)
elif b >= c:
    print("Largest:", b)
else:
    print("Largest:", c)

# Method 2: using max()
print("Largest:", max(a, b, c))
```

Worked: Multiple Python Short Programs (W'25 — 4M)

```python
# Swap without temp (Pythonic)
a, b = 10, 20
a, b = b, a
print(f"a={a}, b={b}")  # a=20, b=10

# Count even numbers in list
nums = [1,2,3,4,5,6,7,8,9,10]
even = sum(1 for n in nums if n%2==0)
print("Even count:", even)   # 5

# Print ASCII value of a character
ch = input("Character: ")
print(f"ASCII of {ch} = {ord(ch)}")
print(f"Char of 65 = {chr(65)}")

# Check if number is single digit
n = int(input("Number: "))
if 0 <= n <= 9:
    print("Single digit")
else:
    print("Multiple digits")
```

Unit 2 — Rapid Revision Flash Sheet

Topic	Key Fact
Python features	Interpreted, dynamic, OOP, open-source
Program structure	`import`, variables, statements, functions
Identifiers	Start with letter/_; case-sensitive
Data types	int, float, str, bool, complex, list, tuple, dict, set
Explicit conversion	`int()`, `float()`, `str()`, `bool()`
Arithmetic ops	`+ - * / // % **` (7 operators)
Comparison ops	`== != > < >= <=` (return bool)
Logical ops	`and or not`
Assignment ops	`= += -= *= /= //= %= **=`
Membership ops	`in`, `not in`
Identity ops	`is`, `is not` (check object identity)
Bitwise ops	`& | ^ ~ << >>`
`input()`	Always returns **string**; use `int()` to convert
`print()`	`sep=`, `end=`, f-strings

Unit 2 — Summary

Key Concepts

Features	Simple, interpreted, OOP, open-source, cross-platform
Data Types	int, float, str, bool, complex, list, tuple, dict, set
Identifiers	Start with letter/_, case-sensitive, no keywords
Operators	Arithmetic, Comparison, Logical, Assignment, Membership
Type Conv.	int(), float(), str(), bool() for explicit conversion
I/O	`input()` returns str; `print()` displays output

Exam Weightage (Unit 2 — 14%)

3-mark × 1–2 | 4-mark × 1 | 7-mark × 1–2