

My Python Course Notes

Structured Revision for Every Lesson

Siddharth Patel
HTW Berlin

May 28, 2025

Contents

1	Lesson 1: Print Function – Full Usage Guide	3
2	Lesson 2: Input Function – Full Usage Guide	4
3	Lesson 3: Math Operators – Full Usage Guide	5
4	Lesson 4: Strings – Full Usage Guide	6

1 Lesson 1: Print Function – Full Usage Guide

```
1 # PRINT FUNCTION – FULL USAGE GUIDE
2
3 # Basic Syntax:
4 # print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
5
6 # Parameters:
7 # *objects → One or more objects to be printed (comma-separated).
8 # sep      → String inserted between objects. Default is ' ' (space).
9 # end      → String appended after the last object. Default is '\n' (new line).
10 # file     → A file-like object (stream); default is sys.stdout.
11 # flush    → If True, forcibly flush the stream. Default is False.
12
13 # -----
14
15 # 1. Basic print
16 print("Hello, World!") # Hello, World!
17
18 # 2. Printing multiple objects
19 print("Hello", "Python", 3) # Hello Python 3
20
21 # 3. Using 'sep' to change separator
22 print("2025", "05", "27", sep="-") # 2025-05-27
23
24 # 4. Using 'end' to avoid new line
25 print("Loading", end="...") # Loading...
26
27 # 5. Using custom separator and end together
28 print("Name", "Age", sep=": ", end=" years\n") # Name: Age years
29
30 # 6. Printing to a file
31 with open("output.txt", "w") as f:
32     print("Saving this line to a file.", file=f)
33
34 # 7. Forcing flush (useful in loops/real-time output)
35 import time
36 for i in range(3):
37     print(i, end=" ", flush=True)
38     time.sleep(0.5) # Output appears immediately
39
40 # 8. Printing escape characters
41 print("Line1\nLine2") # New line
42 print("Tabbed\tSpace") # Tab space
43 print("He said \"hello\"") # Quotes inside string
44
45 # 9. Printing with formatted strings (f-strings)
46 name = "Siddhart"
47 age = 21
48 print(f"Hello, my name is {name} and I am {age} years old.")
49
50 # 10. Using print with unpacking
51 nums = [1, 2, 3, 4]
52 print(*nums) # 1 2 3 4
53 print(*nums, sep=", ") # 1, 2, 3, 4
54
55 # 11. Printing Unicode/emojis (note: removed for LaTeX safety)
56 print("Python is fun")
```

Additional Functions Used in This Lesson

Referenced Functions – Syntax and Output Type

Function	Syntax	Return / Output Type
with open()	with open("file.txt", "w") as f:	File object
print(..., file=f)	print("text", file=f)	Writes to file, returns None
range()	range(3) or range(start, stop, step)	Range object (iterable)
time.sleep()	time.sleep(seconds)	None (pauses execution)

2 Lesson 2: Input Function – Full Usage Guide

```

1 # INPUT FUNCTION – FULL USAGE GUIDE
2
3 # Basic Syntax:
4 # input(prompt='')
5
6 # Parameters:
7 # prompt → A string, written to standard output without a trailing newline,
8 #          to ask the user for input. Default is an empty string ''.
9 # Returns → A string entered by the user (always str type).
10 # Notes → Always returns a string. You need to convert it using int(), float(), etc. if needed.
11
12 # -----
13
14 # 1. Basic usage with no prompt
15 user_input = input()
16 print("You entered:", user_input)
17
18 # 2. Input with a prompt
19 name = input("Enter your name: ")
20 print("Hello,", name)
21
22 # 3. Converting input to integer
23 age = int(input("Enter your age: "))
24 print("You will be", age + 1, "next year.")
25
26 # 4. Converting input to float
27 height = float(input("Enter your height in meters: "))
28 print("Your height in cm is", height * 100)
29
30 # 5. Reading multiple values (as strings)
31 x, y = input("Enter two words separated by space: ").split()
32 print("Word 1:", x)
33 print("Word 2:", y)
34
35 # 6. Reading and converting multiple values to int
36 a, b = map(int, input("Enter two integers: ").split())
37 print("Sum =", a + b)
38
39 # 7. Reading many values into a list of ints
40 numbers = list(map(int, input("Enter multiple numbers: ").split()))
41 print("You entered:", numbers)

```

```

42
43 # 8. Handling invalid input using try/except
44 try:
45     salary = float(input("Enter your monthly salary: "))
46     print("Yearly salary:", salary * 12)
47 except ValueError:
48     print("Invalid input! Please enter a number.")
49
50

```

Referenced Functions – Syntax and Output Type

Function / State-ment	Syntax	Return / Output Type
.split()	string.split() string.split("delimiter")	or List of strings
map()	map(function, iterable)	Map object (can be converted to list)
list()	list(iterable)	List object
try / except	try: code except ErrorType: fallback	Flow control – no return value; handles runtime errors

3 Lesson 3: Math Operators – Full Usage Guide

```

1 # MATH OPERATORS – FULL USAGE GUIDE
2
3 # Basic Syntax:
4 # <operand1> <operator> <operand2>
5
6 # Operators:
7 # + Addition      → a + b
8 # - Subtraction   → a - b
9 # * Multiplication → a * b
10 # / Division      → a / b
11 # // Floor Division → a // b
12 # % Modulus (Remainder) → a % b
13 # ** Exponentiation → a ** b
14
15 # -----
16
17 # 1. Addition
18 print("1 + 1 =", 1 + 1)
19
20 # 2. Subtraction
21 print("2 - 3 =", 2 - 3)
22
23 # 3. Multiplication
24 print("4 * 5 =", 4 * 5)
25
26 # 4. Division (always returns float)

```

```
27 print("6 / 3 =", 6 / 3)
28
29 # 5. Floor Division (truncates decimals)
30 print("7 // 2 =", 7 // 2)
31
32 # 6. Rounded division result using round()
33 number1 = 1.85
34 number2 = 1.35
35 number3 = 1.5
36 print(f"{number1} rounded is:", round(number1)) # 2
37 print(f"{number2} rounded is:", round(number2)) # 1
38 print(f"{number3} rounded is:", round(number3)) # 2
39
40 # 7. Exponentiation
41 print("3 ** 3 =", 3 ** 3) # 27
42
43 # 8. Modulus (Remainder)
44 print("20 / 6 =", 20 / 6) # Division
45 print("20 % 6 =", 20 % 6) # Remainder (2)
46
47 # 9. Operator Precedence in Python:
48 # 1. ()
49 # 2. **
50 # 3. * and /
51 # 4. + and -
52 # Evaluated left to right within same level
```

4 Lesson 4: Strings – Full Usage Guide

```
1 # STRINGS – FULL USAGE GUIDE
2
3 # Basic Explanation:
4 # A string is a sequence of characters enclosed in single ( ' ') or double ( " ") quotes.
5 # Strings are immutable in Python.
6
7 # -----
8 # 1. Creating Strings
9 name = 'math' # single-quoted string
10 subject = "math" # double-quoted string
11
12 # 2. String Addition and Printing
13 print("math" + "works") # mathworks
14 print("math", "works") # math works
15
16 # 3. String Multiplication
17 string1 = "hello"
18 string2 = "world"
19 number = 5
20
21 print(string1, string2) # hello world
22 print(string1 + string2) # helloworld
23 print(string1 * number) # hellohellohellohellohello
24
25 # 4. Invalid Concatenation Example
26 # print(string1 + number) # TypeError: can only concatenate str (not "int")
27
28
29
```

```
30
31 # STRING METHODS - TOP 10 DEFINITIONS
32
33 text = "hello WORLD"
34
35 # 5. capitalize()
36 # Returns string with first character uppercased, rest lowercased.
37 print(text.capitalize()) # Hello world
38
39 # 6. lower()
40 # Converts all characters to lowercase.
41 print(text.lower())      # hello world
42
43 # 7. title()
44 # Capitalizes first letter of each word.
45 print(text.title())      # Hello World
46
47 # 8. casefold()
48 # Aggressive lowercase, suitable for comparisons.
49 text2 = "Straße"
50 print(text2.casefold())  # strasse
51
52 # 9. upper()
53 # Converts all characters to uppercase.
54 print(text.upper())      # HELLO WORLD
55
56 # 10. count()
57 # Counts how many times a substring appears.
58 print(text.count("l"))   # 3
59 print(text.count("l", 3, 6)) # 1
60
61 # 11. find()
62 # Finds index of substring, or -1 if not found.
63 print(text.find("WORLD")) # 6
64 print(text.find("not_here")) # -1
65
66 # 12. replace()
67 # Replaces substring with another.
68 print(text.replace("WORLD", "Python")) # hello Python
69 print(text.replace("l", "X", 2))      # heXXo WORLD
70
71 # 13. swapcase()
72 # Swaps uppercase to lowercase and vice versa.
73 print("Hello World".swapcase()) # hELLO wORLD
74
75 # 14. join()
76 # Joins elements of iterable with separator.
77 words = ["hello", "world"]
78 print("-".join(words))          # hello-world
```

Referenced Methods – Syntax and Output Type

Method / Function	Syntax	Return / Output Type
<code>.capitalize()</code>	<code>str.capitalize()</code>	str
<code>.lower()</code>	<code>str.lower()</code>	str
<code>.title()</code>	<code>str.title()</code>	str
<code>.casefold()</code>	<code>str.casefold()</code>	str
<code>.upper()</code>	<code>str.upper()</code>	str
<code>.count()</code>	<code>str.count(substring, start, end)</code>	int
<code>.find()</code>	<code>str.find(substring, start, end)</code>	int
<code>.replace()</code>	<code>str.replace(old, new, count)</code>	str
<code>.swapcase()</code>	<code>str.swapcase()</code>	str
<code>.join()</code>	<code>"separator".join(iterable)</code>	str