

# PYTHON 3 QUICK REFERENCE

## Basics

### Variables

```
name = "Alice" # str
age = 20       # int
gpa = 3.85    # float
active = True  # bool
```

### Data Types

<b>str</b>	Text: "hello"
<b>int</b>	Integer: 42
<b>float</b>	Decimal: 3.14
<b>bool</b>	True / False
<b>list</b>	Ordered, mutable: [1, 2, 3]
<b>tuple</b>	Ordered, immutable: (1, 2)
<b>dict</b>	Key-value: {"a": 1}
<b>set</b>	Unique items: {1, 2, 3}

### Arithmetic

<b>+ - *</b>	Add, subtract, multiply
<b>/</b>	Division (float): 7/2 → 3.5
<b>//</b>	Floor division: 7//2 → 3
<b>%</b>	Modulo: 7%2 → 1
<b>**</b>	Power: 2**3 → 8

### Type Conversion

```
int("42") # 42
float("3.14") # 3.14
str(100) # "100"
list("abc") # ['a', 'b', 'c']
```

### User Input

```
name = input("Your name? ")
age = int(input("Age? "))
```

## Strings

### Creating Strings

```
s1 = 'single quotes'
s2 = "double quotes"
s3 = """triple quotes
for multiline"""
```

### f-Strings (Python 3.6+)

```
name = "Alice"
f"Hello, {name}!" # Hello, Alice!
f"{2 + 3}" # 5
f"{3.14159:.2f}" # 3.14
f"{1000:,"} # 1,000
```

### String Slicing

```
s = "Python"
# Index: 0 1 2 3 4 5
s[0] # 'P'
s[-1] # 'n'
s[2:5] # 'tho'
s[:2] # 'Py'
s[2:] # 'thon'
s[::-1] # 'nohtyP' (reverse)
```

### String Methods

<b>len(s)</b>	Length of string
<b>s.upper()</b>	UPPERCASE
<b>s.lower()</b>	lowercase
<b>s.strip()</b>	Remove leading/trailing whitespace
<b>s.split(",")</b>	Split into list
<b>",".join(lst)</b>	Join list into string
<b>s.replace(a, b)</b>	Replace a with b
<b>s.find("x")</b>	Index of first match (-1 if none)
<b>s.startswith(x)</b>	Check prefix → bool
<b>s.endswith(x)</b>	Check suffix → bool
<b>s.count(x)</b>	Count occurrences
<b>"x" in s</b>	Contains check → bool

## Lists

### Creating & Accessing

```
fruits = ["apple", "banana", "cherry"]
fruits[0] # "apple"
fruits[-1] # "cherry"
fruits[1:3] # ["banana", "cherry"]
```

### List Comprehension

```
squares = [x**2 for x in range(5)]
# [0, 1, 4, 9, 16]
evens = [x for x in range(10) if x%2==0]
# [0, 2, 4, 6, 8]
```

### List Methods

<b>lst.append(x)</b>	Add to end
<b>lst.extend(lst2)</b>	Add all from lst2
<b>lst.insert(i, x)</b>	Insert at index i
<b>lst.pop()</b>	Remove & return last
<b>lst.pop(i)</b>	Remove & return at i
<b>lst.remove(x)</b>	Remove first x
<b>del lst[i]</b>	Delete by index

<b>lst.sort()</b>	Sort in place
<b>sorted(lst)</b>	Return sorted copy
<b>lst.reverse()</b>	Reverse in place
<b>len(lst)</b>	Number of items
<b>x in lst</b>	Membership check
<b>lst.index(x)</b>	First index of x
<b>lst.count(x)</b>	Count of x

## Tuples & Sets

### Tuples (Immutable)

```
point = (3, 4)
x, y = point # unpacking
point[0] # 3 (read-only)
```

### Sets (Unique Items)

```
s = {1, 2, 3}
s.add(4); s.remove(1)
a & b # intersection
a | b # union
a - b # difference
```

## Dictionaries

### Creating & Accessing

```
student = {"name": "Alice", "age": 20}
student["name"] # "Alice"
student.get("gpa", 0) # 0 (default)
student["gpa"] = 3.85 # add/update
```

### Dict Comprehension

```
sq = {x: x**2 for x in range(5)}
# {0:0, 1:1, 2:4, 3:9, 4:16}
```

### Iterating

```
for k, v in student.items():
    print(f"{k}: {v}")
```

### Dict Methods

<b>d.keys()</b>	All keys
<b>d.values()</b>	All values
<b>d.items()</b>	All (key, value) pairs
<b>d.get(k, default)</b>	Get with fallback
<b>d.update(d2)</b>	Merge d2 into d
<b>d.pop(k)</b>	Remove & return value
<b>del d[k]</b>	Delete key
<b>"k" in d</b>	Key exists? → bool
<b>len(d)</b>	Number of entries

# PYTHON 3 QUICK REFERENCE (continued)

## Control Flow

### if / elif / else

```
if score >= 90:
    grade = "A"
elif score >= 80:
    grade = "B"
else:
    grade = "C"
```

### Ternary Operator

```
status = "pass" if score >= 60 else "fail"
```

## Loops

### for Loop

```
for fruit in ["apple", "banana"]:
    print(fruit)
```

### range()

```
range(5)          # 0, 1, 2, 3, 4
range(2, 5)       # 2, 3, 4
range(0, 10, 2)   # 0, 2, 4, 6, 8
```

### while Loop

```
while count < 10:
    count += 1
```

### enumerate() & zip()

```
for i, val in enumerate(["a", "b"]):
    print(i, val) # 0 a, 1 b

for a, b in zip([1, 2], ["x", "y"]):
    print(a, b) # 1 x, 2 y
```

### break & continue

```
for x in range(10):
    if x == 5: break # stop loop
    if x % 2 == 0: continue # skip
```

## Functions

### Defining & Calling

```
def greet(name, greeting="Hi"):
    return f"{greeting}, {name}!"

greet("Alice") # "Hi, Alice!"
greet("Bob", "Hello") # "Hello, Bob!"
```

### Multiple Return Values

```
def min_max(lst):
    return min(lst), max(lst)

lo, hi = min_max([3, 1, 4, 1, 5])
```

### \*args & \*\*kwargs

```
def total(*args): # args is a tuple
    return sum(args)
total(1, 2, 3) # 6
```

```
def info(**kwargs): # kwargs is a dict
    print(kwargs)
```

### Lambda Functions

```
square = lambda x: x**2
square(5) # 25
sorted(lst, key=lambda x: x["age"])
```

## Classes

```
class Dog:
    def __init__(self, name, breed):
        self.name = name
        self.breed = breed

    def bark(self):
        return f"{self.name} says Woof!"
```

```
dog = Dog("Rex", "Lab")
dog.bark() # "Rex says Woof!"
```

### Inheritance

```
class Puppy(Dog):
    def __init__(self, name, breed, toy):
        super().__init__(name, breed)
        self.toy = toy
```

## Error Handling

```
try:
    result = 10 / 0
except ZeroDivisionError as e:
    print(f"Error: {e}")
finally:
    print("Always runs")
```

## File I/O

### Reading Files

```
with open("data.txt") as f:
    content = f.read() # full text

with open("data.txt") as f:
    for line in f: # line by line
        print(line.strip())
```

### Writing Files

```
with open("out.txt", "w") as f:
    f.write("Hello\n")
```

"r" = read "w" = write (overwrite) "a" = append

## CSV

```
import csv

with open("data.csv") as f:
    reader = csv.DictReader(f)
    for row in reader:
        print(row["name"])

with open("out.csv", "w", newline="") as f:
    writer = csv.writer(f)
    writer.writerow(["name", "age"])
```

## JSON

```
import json

data = json.loads('{ "name": "Alice" }') # parse
text = json.dumps(data) # serialize

with open("data.json") as f:
    data = json.load(f) # read file
with open("out.json", "w") as f:
    json.dump(data, f, indent=2) # write file
```

## HTTP Requests

```
import requests

# GET
r = requests.get("https://api.example.com/data")
r.status_code # 200
data = r.json() # parse JSON

# POST
r = requests.post(url, json={"key": "val"})
```

## pandas Basics

```
import pandas as pd
df = pd.read_csv("data.csv")
df.head() # first 5 rows
df.shape # (rows, cols)
df["name"] # single column
df[df["age"] > 20] # filter rows
```

## Useful Built-ins

<code>print()</code>	Output to console
<code>len()</code>	Length / count
<code>type()</code>	Type of object
<code>range()</code>	Sequence of numbers
<code>enumerate()</code>	Index + value pairs
<code>zip()</code>	Pair items from iterables
<code>sorted()</code>	Return sorted copy
<code>sum()</code> <code>min()</code> <code>max()</code>	Aggregate functions

## Modules

```
import math
from math import sqrt, pi
import pandas as pd # alias
```