

Référence rapide Python 3

Bases, pandas, requests, csv, json

Bases

Variables

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

Types de données

```
str Texte: "hello"
int Entier: 42
float Décimal: 3.14
bool True / False
list Ordonné, mutable: [1, 2, 3]
tuple Ordonné, immuable: (1, 2)
dict Clé-valeur: {"a": 1}
set Éléments uniques: {1, 2, 3}
```

Arithmétique

```
+ - * Addition, soustraction, multiplication
/ Division (float): 7/2 → 3.5
// Division entière: 7//2 → 3
% Modulo: 7%2 → 1
** Puissance: 2**3 → 8
```

Conversion de types

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

Saisie utilisateur

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

Chaînes de caractères

Créer des chaînes

```
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
```

Découpage de chaîne

```
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)
```

Méthodes de chaîne

```
len(s) Longueur de la chaîne
s.upper() MAJUSCULES
s.lower() minuscules
s.strip() Supprimer espaces en début/fin
s.split(",") Découper en liste
", ".join(lst) Joindre une liste en chaîne
s.replace(a, b) Remplacer a par b
s.find("x") Index de la première occurrence (-1 si absent)
s.startswith(x) Vérifier le préfixe → bool
s.endswith(x) Vérifier le suffixe → bool
s.count(x) Compter les occurrences
"x" in s Vérifier la présence → bool
```

Listes

Créer & Accéder

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

Compréhension de liste

```
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]
```

Méthodes de liste

```
lst.append(x) Ajouter en fin
lst.extend(lst2) Ajouter tous les éléments de lst2
lst.insert(i, x) Insérer à l'index i
lst.pop() Retirer & retourner le dernier
lst.pop(i) Retirer & retourner à l'index i
lst.remove(x) Supprimer la première occurrence de x
del lst[i] Supprimer par index
lst.sort() Trier en place
sorted(lst) Retourner une copie triée
lst.reverse() Inverser en place
len(lst) Nombre d'éléments
x in lst Vérifier la présence
lst.index(x) Premier index de x
lst.count(x) Nombre d'occurrences de x
```

Tuples & Ensembles

Tuples (Immuables)

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

Ensembles (Éléments uniques)

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

Dictionnaires

Créer & Accéder

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

Compréhension de dictionnaire

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

Itération

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

Méthodes de dictionnaire

```
d.keys() Toutes les clés
d.values() Toutes les valeurs
d.items() Toutes les paires (clé, valeur)
d.get(k, default) Obtenir avec valeur par défaut
d.update(d2) Fusionner d2 dans d
d.pop(k) Supprimer & retourner la valeur
del d[k] Supprimer une clé
"key" in d La clé existe ? → bool
len(d) Nombre d'entrées
```

Structures de contrôle

if / elif / else

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

Opérateur ternaire

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

Boucles

Boucle for

```
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
```

Boucle while

```
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
```

Référence rapide Python 3

Fonctions

Définir & Appeler

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

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

Valeurs de retour multiples

```
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)
```

Fonctions lambda

```
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!"
```

Héritage

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

Gestion des erreurs

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

Fichiers

Lire des fichiers

```
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())
```

Écrire des fichiers

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

"r" = lecture "w" = écriture (écrase) "a" = ajout

refmint.com

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
```

Requêtes HTTP

```
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"})
```

Bases de pandas

```
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
```

Fonctions intégrées utiles

print()	Afficher dans la console
len()	Longueur / nombre
type()	Type d'un objet
range()	Séquence de nombres
enumerate()	Paires index + valeur
zip()	Associer des éléments d'itérables
sorted()	Retourner une copie triée
sum() min() max()	Fonctions d'agrégation

Modules

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