

Referência Rápida de Python 3

Básico ao pandas, requests, csv, json

Básico

Variáveis

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

Tipos de Dados

```
str Texto: "hello"
int Inteiro: 42
float Decimal: 3.14
bool True / False
list Ordenada, mutável: [1, 2, 3]
tuple Ordenada, imutável: (1, 2)
dict Chave-valor: {"a": 1}
set Itens únicos: {1, 2, 3}
```

Aritmética

```
+ - * Adição, subtração, multiplicação
/ Divisão (float): 7/2 → 3.5
// Divisão inteira: 7//2 → 3
% Módulo: 7%2 → 1
** Potência: 2**3 → 8
```

Conversão de Tipo

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

Entrada do Usuário

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

Strings

Criando 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
```

Fatiamento de Strings

```
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)
```

Métodos de String

len(s)	Comprimento da string
s.upper()	MAIÚSCULAS
s.lower()	minúsculas
s.strip()	Remover espaços no início/fim
s.split(",")	Dividir em lista
",".join(lst)	Unir lista em string
s.replace(a, b)	Substituir a por b
s.find("x")	Índice da primeira ocorrência (-1 se não encontrar)
s.startswith(x)	Verificar prefixo → bool
s.endswith(x)	Verificar sufixo → bool
s.count(x)	Contar ocorrências
"x" in s	Verificar se contém → bool

Listas

Criar & Acessar

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

Compreensão de Lista

```
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]
```

Métodos de Lista

lst.append(x)	Adicionar ao final
lst.extend(lst2)	Adicionar todos de lst2
lst.insert(i, x)	Inserir no índice i
lst.pop()	Remover e retornar o último
lst.pop(i)	Remover e retornar no índice i
lst.remove(x)	Remover primeiro x
del lst[i]	Deletar pelo índice
lst.sort()	Ordenar in-place
sorted(lst)	Retornar cópia ordenada
lst.reverse()	Inverter in-place
len(lst)	Número de itens
x in lst	Verificar membro
lst.index(x)	Primeiro índice de x
lst.count(x)	Contagem de x

Tuplas & Conjuntos

Tuplas (Imutáveis)

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

Conjuntos (Itens Únicos)

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

Dicionários

Criar & Acessar

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

Compreensão de Dict

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

Iterando

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

Métodos de Dict

d.keys()	Todas as chaves
d.values()	Todos os valores
d.items()	Todos os pares (chave, valor)
d.get(k, default)	Obter com fallback
d.update(d2)	Mesclar d2 em d
d.pop(k)	Remover e retornar valor
del d[k]	Deletar chave
"k" in d	Chave existe? → bool
len(d)	Número de entradas

Controle de Fluxo

if / elif / else

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

Operador Ternário

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

Laços

Laço 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
```

Laço 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
```

Referência Rápida de Python 3

Funções

Definir & Chamar

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

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

Múltiplos Retornos

```
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)
```

Funções 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!"
```

Herança

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

Tratamento de Erros

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

Arquivos

Lendo Arquivos

```
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())
```

Escrevendo Arquivos

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

"r" = leitura "w" = escrita (sobrescreve) "a" = append

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
```

Requisições 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"})
```

Básico 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
```

Built-ins Úteis

print()	Imprimir no console
len()	Comprimento / contagem
type()	Tipo do objeto
range()	Sequência de números
enumerate()	Pares de índice + valor
zip()	Emparelhar itens de iteráveis
sorted()	Retornar cópia ordenada
sum() min() max()	Funções de agregação

Módulos

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