

# Referencia Rápida de Python 3

Fundamentos, pandas, requests, csv, json

## Fundamentos

### Variables

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

### Tipos de Datos

```
str Texto: "hello"
int Entero: 42
float Decimal: 3.14
bool True / False
list Ordenado, mutable: [1, 2, 3]
tuple Ordenado, inmutable: (1, 2)
dict Clave-valor: {"a": 1}
set Elementos únicos: {1, 2, 3}
```

### Aritmética

```
+ - * Suma, resta, multiplicación
/ División (float): 7/2 → 3.5
// División entera: 7//2 → 3
% Módulo: 7%2 → 1
** Potencia: 2**3 → 8
```

### Conversión de Tipos

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

### Entrada del Usuario

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

## Cadenas

### Crear Cadenas

```
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
```

### Slicing de Cadenas

```
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 Cadenas

```
len(s) Longitud de la cadena
s.upper() MAYÚSCULAS
s.lower() minúsculas
s.strip() Eliminar espacios al inicio y final
s.split(",") Dividir en lista
",".join(lst) Unir lista en cadena
s.replace(a, b) Reemplazar a con b
s.find("x") Índice de la primera coincidencia (-1 si no existe)
s.startswith(x) Verificar prefijo → bool
s.endswith(x) Verificar sufijo → bool
s.count(x) Contar ocurrencias
"x" in s Verificar contenido → bool
```

## Listas

### Crear y Acceder

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

### Comprensión de Listas

```
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) Agregar al final
lst.extend(lst2) Agregar todos de lst2
lst.insert(i, x) Insertar en el índice i
lst.pop() Eliminar y retornar el último
lst.pop(i) Eliminar y retornar en i
lst.remove(x) Eliminar primer x
del lst[i] Eliminar por índice
lst.sort() Ordenar en sitio
sorted(lst) Retornar copia ordenada
lst.reverse() Invertir en sitio
len(lst) Número de elementos
x in lst Verificar membresía
lst.index(x) Primer índice de x
lst.count(x) Conteo de x
```

## Tuplas y Conjuntos

### Tuplas (Inmutables)

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

### Conjuntos (Elementos Únicos)

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

## Diccionarios

### Crear y Acceder

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

## Comprensión de Dict

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

### Iteración

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

### Métodos de Dict

```
d.keys() Todas las claves
d.values() Todos los valores
d.items() Todos los pares (clave, valor)
d.get(k, default) Obtener con valor predeterminado
d.update(d2) Combinar d2 en d
d.pop(k) Eliminar y retornar valor
del d[k] Eliminar clave
"k" in d ¿La clave existe? → bool
len(d) Número de entradas
```

## Flujo de Control

### if / elif / else

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

### Operador Ternario

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

## Bucles

### Bucle 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
```

### Bucle while

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

### enumerate() y 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 y continue

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

# Referencia Rápida de Python 3

## Funciones

### Definir y Llamar

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

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

### Múltiples Valores de Retorno

```
def min_max(lst):
    return min(lst), max(lst)
lo, hi = min_max([3, 1, 4, 1, 5])
```

### \*args y \*\*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)
```

### Funciones Lambda

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

## Clases

```
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!"
```

### Herencia

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

## Manejo de Errores

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

## E/S de Archivos

### Leer Archivos

```
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())
```

### Escribir Archivos

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

"r" = leer "w" = escribir (sobreescribir) "a" = agregar

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
```

## Solicitudes 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"})
```

## pandas Básico

```
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
```

## Funciones Integradas Útiles

<b>print()</b>	Salida a consola
<b>len()</b>	Longitud / conteo
<b>type()</b>	Tipo de objeto
<b>range()</b>	Secuencia de números
<b>enumerate()</b>	Pares de índice + valor
<b>zip()</b>	Emparejar elementos de iterables
<b>sorted()</b>	Retornar copia ordenada
<b>sum()</b> <b>min()</b> <b>max()</b>	Funciones de agregación

## Módulos

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