

Python 3 快速参考

基础到 pandas、requests、csv、json

基础

变量

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

数据类型

```
str 文本: "hello"
int 整数: 42
float 小数: 3.14
bool True/False
list 有序可变: [1, 2, 3]
tuple 有序不可变: (1, 2)
dict 键值对: {"a": 1}
set 唯一元素集合: {1, 2, 3}
```

算术运算

```
+ - * 加、减、乘
/ 除法 (浮点): 7/2→3.5
// 整除: 7//2→3
% 取模: 7%2→1
** 幂运算: 2**3→8
```

类型转换

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

用户输入

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

字符串

创建字符串

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

f-String (Python 3.6+)

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

字符串切片

```
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)
```

字符串方法

```
len(s) 字符串长度
s.upper() 转大写
s.lower() 转小写
s.strip() 去除首尾空白
s.split(",") 分割为列表
",".join(lst) 列表合并为字符串
s.replace(a, b) 将 a 替换为 b
s.find("x") 首个匹配的索引 (未找到返回 -1)
s.startswith(x) 检查前缀 → bool
s.endswith(x) 检查后缀 → bool
s.count(x) 统计出现次数
"x" in s 包含检查 → bool
```

列表

创建与访问

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

列表推导式

```
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]
```

列表方法

```
lst.append(x) 追加到末尾
lst.extend(lst2) 追加 lst2 的所有元素
lst.insert(i, x) 在索引 i 处插入
lst.pop() 移除并返回最后一个元素
lst.pop(i) 移除并返回索引 i 处的元素
lst.remove(x) 移除第一个 x
del lst[i] 按索引删除
lst.sort() 原地排序
sorted(lst) 返回排序副本
lst.reverse() 原地反转
len(lst) 元素数量
x in lst 成员检查
lst.index(x) x 的第一个索引
lst.count(x) x 的出现次数
```

元组与集合

元组 (不可变)

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

集合 (唯一元素)

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

字典

创建与访问

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

字典推导式

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

遍历

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

字典方法

```
d.keys() 所有键
d.values() 所有值
d.items() 所有 (键, 值) 对
d.get(k, default) 获取值, 带默认值
d.update(d2) 将 d2 合并到 d
d.pop(k) 移除并返回值
del d[k] 删除键
"k" in d 键是否存在 → bool
len(d) 条目数量
```

控制流

if/elif/else

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

三元表达式

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

循环

for 循环

```
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 循环

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

enumerate() 与 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
```

Python 3 快速参考

break 与 continue

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

函数

定义与调用

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

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

多返回值

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

*args 与 **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 函数

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

类

```
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!"
```

继承

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

异常处理

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

文件读写

读取文件

```
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())
```

写入文件

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

"r" = 读 "w" = 写 (覆盖) "a" = 追加

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 请求

```
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 基础

```
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
```

常用内置函数

print()	输出到控制台
len()	长度 / 数量
type()	对象类型
range()	数字序列
enumerate()	索引 + 值对
zip()	配对多个可迭代对象
sorted()	返回排序副本
sum() min() max()	聚合函数

模块

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