Python 3 빠른 참조

기초부터 pandas, requests, csv, json까지

기본

변수

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

데이터 타입

str	텍스트: **"hello"**
int	정수: **42**
float	소수: **3.14**
bool	**True** / **False**
list	순서 있고 변경 가능: **[1, 2, 3]**
tuple	순서 있고 변경 불가: **(1, 2)**
dict	키-값: **{"a": 1}**
set	고유 항목: **{1, 2, 3}**

산술

+ - *	더하기, 빼기, 곱하기
/	나눗셈 (float): 7/2 → 3.5
//	정수 나눗셈: 7//2 → 3
%	나머지: 7%2 → 1
******	거듭제곱: 2**3 → 8

타입 변환

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

사용자 입력

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

문자열

문자열 생성

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

f-문자열 (Python 3.6+)

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

문자열 슬라이싱

```
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)
```

문자열 메서드

len(s)	문자열 길이
s.upper()	대문자
s.lower()	소문자
s.strip()	앞뒤 공백 제거
s.split(",")	리스트로 분리
",".join(lst)	리스트를 문자열로 합치기
s.replace(a, b)	a를 b로 치환
s.find("x")	첫 번째 일치 인덱스 (-1은 없음)
s.startswith(x)	접두사 확인 → bool
s.endswith(x)	접미사 확인 → bool
s.count(x)	출현 횟수
"x" in s	포함 여부 → bool

리스트

생성 및 접근

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

리스트 컴프리헨션

```
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]
```

리스트 메서드

lst.append(x)	끝에 추가
lst.extend(lst2)	lst2의 모든 항목 추가
lst.insert(i, x)	인덱스 i에 삽입
lst.pop()	마지막 항목 제거 후 반환
lst.pop(i)	인덱스 i 항목 제거 후 반환
lst.remove(x)	첫 번째 x 제거
del lst[i]	인덱스로 삭제
lst.sort()	제자리 정렬
sorted(lst)	정렬된 복사본 반환
lst.reverse()	제자리 역순
len(lst)	항목 수
x in lst	포함 여부 확인
lst.index(x)	x의 첫 번째 인덱스
lst.count(x)	x의 개수

튜플 및 집합

튜플 (변경 불가)

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

집합 (고유 항목)

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

딕셔너리

생성 및 접근

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

딕셔너리 컴프리헨션

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

반복

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

딕셔너리 메서드

d.keys()	모든 키
d.values()	모든 값
d.items()	모든 (키, 값) 쌍
d.get(k, default)	기본값과 함께 가져오기
d.update(d2)	d2를 d에 병합
d.pop(k)	키 제거 후 값 반환
del d[k]	키 삭제
"k" in d	키 존재 여부 → bool
len(d)	항목 수

제어 흐름

if / elif / else

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

삼항 연산자

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

반복문

for 반복문

```
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 반복문

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

enumerate() 및 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 빠른 참조

break 및 continue

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

함수

정의 및 호출

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

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

다중 반환값

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

*args 및 **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)
```

람다 함수

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

클래스

```
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!"
```

상속

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

오류 처리

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

파일 I/O

파일 읽기

```
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())
```

파일 쓰기

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

"r" = 읽기 "w" = 쓰기 (덮어씀) "a" = 추가

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 요청

```
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 기본

```
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
```

유용한 내장 함수

print()	콘솔 출력
len()	길이 / 개수
type()	객체 타입
range()	숫자 시퀀스
enumerate()	인덱스 + 값 쌍
zip()	이터러블 항목 쌍 만들기
sorted()	정렬된 복사본 반환
sum() min() max()	집계 함수

모듈

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