Bài 1: Hàm open(), read(), write(), close()

1. Mở file với open()

f = open(tên\_file, chế\_độ\_mở, encoding='utf-8')

| **Chế độ** | **Ý nghĩa** |
| --- | --- |
| "r" | Mở file để đọc (mặc định) |
| "w" | Mở để ghi (xóa nội dung cũ nếu file tồn tại) |
| "a" | Mở để ghi thêm vào cuối file |
| "rb" | Đọc file nhị phân (ví dụ ảnh, video) |
| "wb" | Ghi file nhị phân |

1. Đọc nội dung file với read()

f = open("example.txt", "r", encoding="utf-8")

content = f.read()

print(content)

f.close()

**Một số hàm đọc khác:**

* f.readline() – đọc 1 dòng
* f.readlines() – đọc tất cả các dòng, trả về list

1. Ghi vào file với write()

f = open("output.txt", "w", encoding="utf-8")

f.write("Xin chào, Oanh!\n")

f.write("Chào mừng bạn đến với khóa học Python.")

f.close()

=> Nếu file **chưa tồn tại**, Python sẽ **tự tạo mới**.

1. Luôn nhớ đóng file với close

Nếu bạn không gọi f.close(), có thể dẫn đến **rò rỉ bộ nhớ** hoặc **file bị khóa không ghi được**.

1. Sử dụng with open() thay cho open() và close()

Khi dùng with open(...), Python sẽ **tự động đóng file cho bạn**, dù có lỗi xảy ra trong khối lệnh hay không.  
# Ghi file

with open("hello.txt", "w", encoding="utf-8") as f:

f.write("Xin chào Oanh!\n")

f.write("Học Python làm việc với file thật dễ!")

# Đọc file

with open("hello.txt", "r", encoding="utf-8") as f:

content = f.read()

print(content)

* + Tránh rò rỉ tài nguyên nếu chương trình bị lỗi giữa chừng

1. Làm việc với file CSV trong Python

File csv (Comma-Separated Values - giá trị phân cách bởi dấu phẩy)

Mỗi dòng là một bản ghi (record), mỗi cột là một trường dữ liệu (field)

**Đọc file csv bằng module csv:**

import csv

with open("students.csv", "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

print(row)

Output:

['Name', 'Age', 'Score']

['Oanh', '22', '9.5']

['Minh', '21', '8.8']

['An', '20', '9.0']

**Ghi file CSV bằng csv.writer():** (newline='' để tránh ghi dòng trắng thừa trong Windows)

import csv

with open("students\_out.csv", "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerow(["Name", "Age", "Score"])

writer.writerow(["Oanh", 22, 9.5

writer.writerow(["Minh", 21, 8.8])

1. Ghi nhiều dòng vào file csv:

import csv

data = [

["Name", "Age", "Hobby"],

["Oanh", 22, "Coding"],

["Linh", 21, "Music"],

["Nam", 23, "Photography"]

]

with open("friends\_out.csv", "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.**writerows**(data) # 🔥 Ghi nhiều dòng một lúc

1. JSON

JSON (JavaScript Object Notation) là định dạng lưu trữ dữ liệu có cấu trúc, gần giống Python dict

{

"Name": "Oanh",

"Age": 22,

"Hobbies": ["Coding", "Music"]

}

1. Làm việc với file JSON:

**Ghi dữ liệu vào file .json:**

import json

data = {

"Name": "Oanh",

"Age": 22,

"Hobbies": ["Coding", "Music"]

}

with open("user.json", "w", encoding="utf-8") as f:

**json.dump**(data, f, ensure\_ascii=False, indent=4)

* json.dump(data, file) – ghi data vào file.
* ensure\_ascii=False để giữ tiếng Việt.
* indent=4 để format đẹp, dễ đọc.

**Đọc dữ liệu từ file JSON:**

with open("user.json", "r", encoding="utf-8") as f:

user\_data = **json.load(f)**

print(user\_data)

print("Tên:", user\_data["Name"])

print("Sở thích:", user\_data["Hobbies"])

1. Làm việc với chuỗi JSON:

**Chuyển chuỗi JSON → Python object**

import json

json\_string = '{"Name": "Oanh", "Age": 22}' # Đây là chuỗi JSON (dạng text)

data = **json.loads(json\_string)** # Chuyển sang dict Python

print(data) # {'Name': 'Oanh', 'Age': 22}

print(data["Name"]) # Oanh

**Chuyển Python object → chuỗi JSON**

data = {

"Name": "Oanh",

"Age": 22,

"Skills": ["Python", "ML"]

}

json\_str = **json.dumps(data, indent=2)**

print(json\_str)

1. Làm việc với file .env

File .env thường dùng để lưu biến môi trường:

Thường dùng để chứa API key, secret key, config riêng tư

AWS\_ACCESS\_KEY=abc123xyz

AWS\_SECRET\_KEY=def456secret

ENV=dev

1. Dùng thư viện python-dotenv để đọc .env

from dotenv import load\_dotenv

import os

# Tải biến từ file .env

load\_dotenv()

# Truy cập biến bằng os.getenv

access\_key = os.getenv("AWS\_ACCESS\_KEY")

secret\_key = os.getenv("AWS\_SECRET\_KEY")

print("ACCESS KEY:", access\_key)

print("SECRET KEY:", secret\_key)

Lưu ý bảo mật:

File .env không nên đẩy lên GitHub

Hãy thêm .env vào file .gitignore

1. Làm việc với file yaml:

.yaml là gì?

Yaml (Yet Another Markup Language) – định dạng dễ đọc, dùng để lưu config, giống JSON nhưng dễ đọc hơn.

model:

name: "resnet50"

learning\_rate: 0.001

batch\_size: 32

aws:

access\_key: "abc123"

secret\_key: "def456"

1. Dùng thư viện PyYAML để đọc file .yaml

(pip install pyyaml)

**import yaml**

with open("config.yaml", "r", encoding="utf-8") as f:

config = yaml.safe\_load(f)

print("Model name:", config["model"]["name"])

print("LR:", config["model"]["learning\_rate"])

print("AWS key:", config["aws"]["access\_key"])

Về việc tạo, kích hoạt, thoát và xóa môi trường ảo:

A screenshot of a computer

AI-generated content may be incorrect.

1. Làm việc với file .yaml (pip install pyyaml)

**Tạo file train\_config.yaml**

training:

epochs: 10

lr: 0.01

model: "resnet18"

**Cách tạo bằng python: (tạo 1 dictionary, bên trong gồm key là “training”, value là 1 dictionary khác, gọi là nested dictionary)**

import yaml

config\_data = {

"training": {

"epochs": 10,

"lr": 0.01,

"model": "resnet18"

}

}

# Ghi file YAML

with open("train\_config.yaml", "w", encoding="utf-8") as f:

yaml.dump(config\_data, f)

**Đọc file .yaml:**

import yaml

with open("train\_config.yaml", "r", encoding="utf-8") as f:

content = yaml.**safe\_load(f)**

# Lấy thông tin

epochs = content["training"]["epochs"]

lr = content["training"]["lr"]

model = content["training"]["model"]

print(f"Huấn luyện model {model} với {epochs} epochs, lr = {lr}")