TÀI LIỆU HƯỚNG DẪN SỬ DỤNG

PYTHON CHO NGƯỜI MỚI BẮT ĐẦU

Dành cho người chưa từng lập trình, muốn cài Python 3.14 và tự viết, chạy, sửa chương trình nhỏ từng bước

Using Python – a bilingual guide

Cài đặt · trình thông dịch · chạy tệp .py · biến môi trường

Biến · chuỗi · danh sách · hàm · lớp · tệp · lỗi · gỡ lỗi

pip · môi trường ảo · biểu thức chính quy · argparse · Python 3.14

Song ngữ Việt – Anh · Vietnamese – English

Phiên bản 1.0 – Tháng 11 năm 2026

https://tailieuso.vin

MỤC LỤC

CONTENTS

Chương 1. Cài Python và chạy chương trình đầu tiên – *Chapter 1. Installing Python and Running Your First Program* 5

1.1. Chọn phiên bản Python và những điểm mới của 3.14 – *1.1. Choosing a Python Version and What Is New in 3.14* 5

1.2. Cài Python trên Windows, macOS và Linux – *1.2. Installing Python on Windows, macOS and Linux* 8

1.3. Gõ lệnh trực tiếp trong chế độ tương tác – *1.3. Typing Commands in Interactive Mode* 12

1.4. Viết tệp .py và chạy từ dòng lệnh – *1.4. Writing a .py File and Running It from the Command Line* 16

1.5. Tuỳ chọn dòng lệnh và biến môi trường của Python – *1.5. Command-Line Options and Environment Variables* 20

1.6. Tìm trợ giúp: help(), tài liệu chính thức và hỏi đáp chung – *1.6. Getting Help: help(), Official Docs and General FAQ* 24

1.7. Thực hành: chương trình chào và tính tuổi – *1.7. Practice: A Greeting and Age Calculator* 28

1.8. Bài thực hành có hướng dẫn – *1.8. Guided practice exercises* 31

1.9. Câu hỏi trắc nghiệm – *1.9. Multiple-choice questions* 33

Chương 2. Python như chiếc máy tính: số, chữ và danh sách – *Chapter 2. Python as a Calculator: Numbers, Text and Lists* 35

2.1. Số nguyên, số thập phân và các phép tính – *2.1. Integers, Decimals and Arithmetic* 35

2.2. Biến và phép gán – *2.2. Variables and Assignment* 38

2.3. Chuỗi ký tự: dấu nháy, ký tự thoát và chuỗi thô – *2.3. Strings: Quotes, Escape Sequences and Raw Strings* 42

2.4. Nối, lặp, đánh chỉ số và cắt lát chuỗi – *2.4. Joining, Repeating, Indexing and Slicing Strings* 47

2.5. Danh sách: gom nhiều giá trị vào một chỗ – *2.5. Lists: Keeping Several Values Together* 51

2.6. Chương trình có nhiều bước: while, print và dãy Fibonacci – *2.6. Multi-Step Programs: while, print and Fibonacci* 54

2.7. Thực hành: máy tính tiền điện bậc thang – *2.7. Practice: A Tiered Electricity Bill Calculator* 58

2.8. Bài thực hành có hướng dẫn – *2.8. Guided practice exercises* 63

2.9. Câu hỏi trắc nghiệm – *2.9. Multiple-choice questions* 65

Chương 3. Rẽ nhánh và vòng lặp: để chương trình tự quyết định – *Chapter 3. Branching and Loops: Letting Programs Decide* 67

3.1. Câu lệnh if, elif, else và phép so sánh – *3.1. if, elif, else and Comparisons* 67

3.2. Vòng lặp for duyệt qua từng phần tử – *3.2. for Loops: Visiting Each Item* 72

3.3. Hàm range() để lặp một số lần định trước – *3.3. range() for a Fixed Number of Repeats* 76
3.4. break, continue và else của vòng lặp – *3.4. break, continue and Loop else* 79
3.5. pass và match: chỗ trống và so khớp mẫu – *3.5. pass and match: Placeholders and Pattern Matching* 83
3.6. Thụt lề, dòng lệnh dài và chú thích – *3.6. Indentation, Long Lines and Comments* 87
3.7. Thực hành: đoán số và trò chơi nhỏ trong dòng lệnh – *3.7. Practice: A Number-Guessing Game* 92
3.8. Bài thực hành có hướng dẫn – *3.8. Guided practice exercises* 95
3.9. Câu hỏi trắc nghiệm – *3.9. Multiple-choice questions* 98

Chương 4. Hàm: đóng gói việc lặp đi lặp lại – *Chapter 4. Functions: Packaging Repeated Work* **100**
4.1. Định nghĩa và gọi hàm đầu tiên – *4.1. Defining and Calling Your First Function* 100
4.2. Giá trị mặc định và tham số đặt tên – *4.2. Default Values and Keyword Arguments* 103
4.3. Tham số đặc biệt: dấu / và dấu * – *4.3. Special Parameters: the / and * Markers* 106
4.4. Danh sách tham số tuỳ ý và giải nén khi gọi – *4.4. Arbitrary Argument Lists and Unpacking* 109
4.5. Lambda, chuỗi mô tả và chú thích kiểu – *4.5. Lambda, Docstrings and Annotations* 113
4.6. Biến cục bộ, biến toàn cục và lỗi UnboundLocalError – *4.6. Local and Global Variables and UnboundLocalError* 117
4.7. Phong cách viết mã và thực hành hàm – *4.7. Coding Style and Function Practice* 121
4.8. Bài thực hành có hướng dẫn – *4.8. Guided practice exercises* 125
4.9. Câu hỏi trắc nghiệm – *4.9. Multiple-choice questions* 127

Chương 5. Cấu trúc dữ liệu: danh sách, bộ, tập hợp và từ điển – *Chapter 5. Data Structures: Lists, Tuples, Sets and Dictionaries* **129**
5.1. Các phương thức của danh sách; dùng như ngăn xếp và hàng đợi – *5.1. List Methods; Lists as Stacks and Queues* 129
5.2. Biểu thức tạo danh sách gọn (list comprehension) – *5.2. List Comprehensions* 132
5.3. Bộ (tuple), câu lệnh del và giải nén dãy – *5.3. Tuples, del and Sequence Unpacking* 135
5.4. Tập hợp (set): loại trùng và so sánh nhóm – *5.4. Sets: Removing Duplicates and Comparing Groups* 139
5.5. Từ điển (dict): tra cứu theo khoá – *5.5. Dictionaries: Lookup by Key* 143
5.6. Kỹ thuật duyệt và so sánh các dãy – *5.6. Looping Techniques and Comparing Sequences* 146
5.7. Thực hành: sổ danh bạ và thống kê điểm – *5.7. Practice: A Phone Book and Score Statistics* 150
5.8. Bài thực hành có hướng dẫn – *5.8. Guided practice exercises* 153
5.9. Câu hỏi trắc nghiệm – *5.9. Multiple-choice questions* 155

Chương 6. Module, tệp và định dạng dữ liệu – *Chapter 6. Modules, Files and Data Formats* **156**

6.1. Dùng và tự viết module với import – *6.1. Using and Writing Modules with import* 156
6.2. Gói (package) và nơi Python tìm module – *6.2. Packages and Where Python Looks for Modules* 160
6.3. Đọc và ghi tệp văn bản – *6.3. Reading and Writing Text Files* 164
6.4. Lưu và nạp dữ liệu bằng JSON – *6.4. Saving and Loading Data with JSON* 167
6.5. Định dạng chuỗi: f-string, format và chuỗi mẫu t-string – *6.5. String Formatting: f-strings, format() and t-strings* 171
6.6. Làm quen thư viện chuẩn: os, sys, math, random, datetime – *6.6. A Tour of the Standard Library: os, sys, math, random, datetime* 175
6.7. Thực hành: dọn danh sách tệp và ghi báo cáo – *6.7. Practice: Listing Files and Writing a Report* 180
6.8. Bài thực hành có hướng dẫn – *6.8. Guided practice exercises* 183
6.9. Câu hỏi trắc nghiệm – *6.9. Multiple-choice questions* 185

Chương 7. Lỗi, ngoại lệ và cách gỡ lỗi – *Chapter 7. Errors, Exceptions and Debugging* 187

7.1. Đọc thông báo lỗi cú pháp và lỗi lúc chạy – *7.1. Reading Syntax Errors and Runtime Errors* 187
7.2. try và except: bắt lỗi để chương trình không sập – *7.2. try and except: Catching Errors* 192
7.3. else, finally, raise và chuỗi ngoại lệ – *7.3. else, finally, raise and Exception Chaining* 196
7.4. Nhóm ngoại lệ và except* – *7.4. Exception Groups and except** 200
7.5. Gỡ lỗi: print, pdb và gắn vào tiến trình đang chạy – *7.5. Debugging: print, pdb and Attaching to a Running Process* 204
7.6. Công cụ kiểm tra mã và đóng gói thành chương trình chạy riêng – *7.6. Code Checking Tools and Stand-Alone Programs* 208
7.7. Thực hành: sửa chương trình đọc điểm bị lỗi – *7.7. Practice: Fixing a Broken Score-Reading Program* 211
7.8. Bài thực hành có hướng dẫn – *7.8. Guided practice exercises* 215
7.9. Câu hỏi trắc nghiệm – *7.9. Multiple-choice questions* 216

Chương 8. Cài thư viện ngoài bằng pip và môi trường ảo – *Chapter 8. Installing Third-Party Packages with pip and Virtual Environments* 218

8.1. Gói, pip, PyPI: các khái niệm cần biết trước khi cài – *8.1. Packages, pip and PyPI: Key Terms Before Installing* 218
8.2. Cài, nâng cấp và gỡ gói bằng pip – *8.2. Installing, Upgrading and Removing Packages with pip* 222
8.3. Tạo và dùng môi trường ảo cho từng dự án – *8.3. Creating and Using a Virtual Environment for Each Project* 226
8.4. Cài gói cho riêng người dùng và chạy nhiều bản Python song song – *8.4. Per-User Installs and Multiple Python Versions Side by Side* 230
8.5. Gói khoa học và gói có phần mở rộng nhị phân – *8.5. Scientific Packages and Binary Extensions* 233

8.6. Lỗi cài đặt thường gặp và cách xử lý – *8.6. Common Installation Problems and Fixes* 237
8.7. Thực hành: dựng dự án có môi trường ảo riêng – *8.7. Practice: Setting Up a Project with Its Own Virtual Environment* 241
8.8. Bài thực hành có hướng dẫn – *8.8. Guided practice exercises* 245
8.9. Câu hỏi trắc nghiệm – *8.9. Multiple-choice questions* 247

Chương 9. Lớp và đối tượng: tự tạo kiểu dữ liệu của mình – *Chapter 9. Classes and Objects: Building Your Own Data Types* 249
9.1. Phạm vi và không gian tên – *9.1. Scopes and Namespaces* 249
9.2. Định nghĩa lớp và tạo đối tượng đầu tiên – *9.2. Defining a Class and Creating the First Object* 253
9.3. Thuộc tính của lớp, của đối tượng và phương thức – *9.3. Class Attributes, Instance Attributes and Methods* 256
9.4. Kế thừa và nhiều lớp cha – *9.4. Inheritance and Multiple Base Classes* 260
9.5. Biến riêng theo quy ước, lớp dữ liệu đơn giản và lưu ý thực tế – *9.5. Private Naming Conventions, Simple Data Classes and Practical Notes* 263
9.6. Trình lặp (iterator) và generator – *9.6. Iterators and Generators* 267
9.7. Thực hành: lớp sổ chi tiêu cá nhân – *9.7. Practice: A Personal Expense Ledger Class* 270
9.8. Bài thực hành có hướng dẫn – *9.8. Guided practice exercises* 274
9.9. Câu hỏi trắc nghiệm – *9.9. Multiple-choice questions* 276

Chương 10. Xử lý văn bản bằng biểu thức chính quy và chương trình dòng lệnh argparse – *Chapter 10. Text Processing with Regular Expressions and Command-Line Programs with argparse* 278
10.1. Biểu thức chính quy: mẫu ký tự và lặp ký tự – *10.1. Regular Expressions: Character Patterns and Repetition* 278
10.2. Dùng module re: compile, match, search, findall – *10.2. Using the re Module: compile, match, search, findall* 282
10.3. Chuỗi thô và vấn đề dấu gạch chéo ngược – *10.3. Raw Strings and the Backslash Problem* 286
10.4. Nhóm, nhóm đặt tên và kiểm tra trước (lookahead) – *10.4. Groups, Named Groups and Lookahead* 289
10.5. Tách chuỗi, tìm và thay thế – *10.5. Splitting Strings, Search and Replace* 292
10.6. Bẫy thường gặp của regex – *10.6. Common Regex Pitfalls* 296
10.7. Viết chương trình dòng lệnh với argparse – *10.7. Writing Command-Line Programs with argparse* 299
10.8. Bài thực hành có hướng dẫn – *10.8. Guided practice exercises* 303
10.9. Câu hỏi trắc nghiệm – *10.9. Multiple-choice questions* 305

Phụ lục A. Đáp án trắc nghiệm – *Appendix A. Answers to the multiple-choice questions* 307

Phụ lục B. Bảng thuật ngữ Việt – Anh – *Appendix B. Vietnamese–English glossary*. 315

Phụ lục C. Tài liệu tham khảo – *Appendix C. References* 322

Chương 1. Cài Python và chạy chương trình đầu tiên – *Chapter 1. Installing Python and Running Your First Program*

Mục tiêu chương – *Chapter objectives*

- Cài Python và kiểm tra bản cài bằng lệnh xem phiên bản trong Terminal hoặc Command Prompt
  Install Python and check the installation with the version command in Terminal or Command Prompt
- Gõ lệnh trong chế độ tương tác, viết tệp .py và chạy bằng dòng lệnh, kể cả với các tuỳ chọn như -V và -c
  Type commands in interactive mode, write a .py file and run it from the command line, including options such as -V and -c
- Tra cứu cách dùng một hàm bằng help() và nhận ra điểm mới của Python 3.14 như chuỗi mẫu (template strings)
  Look up how a function works with help() and recognise what is new in Python 3.14, such as template strings

1.1. Chọn phiên bản Python và những điểm mới của 3.14 – *1.1. Choosing a Python Version and What Is New in 3.14*

Python là ngôn ngữ lập trình: bạn viết các dòng chỉ dẫn bằng chữ gần với tiếng Anh thường, máy tính làm theo. Nó giống công thức nấu ăn, mỗi dòng là một việc cần làm. Python miễn phí và chạy trên Windows, macOS lẫn Linux.

Python is a programming language: you write instructions in text that looks close to plain English, and the computer follows them. Think of a recipe where each line is one thing to do. Python is free and runs on Windows, macOS and Linux.

Python dùng để làm gì, và vì sao chọn nó khi mới học – *What Python is used for, and why it suits beginners*

Người ta dùng Python để tự động hoá việc lặp lại (đổi tên hàng trăm tệp, gộp bảng tính), phân tích dữ liệu, làm trang web, viết công cụ nhỏ cho công việc hằng ngày. Nó cũng là ngôn ngữ phổ biến trong giảng dạy.

People use Python to automate repetitive jobs (renaming hundreds of files, merging spreadsheets), analyse data, build websites and write small tools for daily work. It is also a common choice in teaching.

Với người mới, có ba lý do. Mã Python ngắn và dễ đọc, nên bạn tập trung vào ý tưởng thay vì ký hiệu rườm rà. Bạn có thể gõ một dòng và thấy kết quả ngay, rất tiện để thử. Cộng đồng lớn, nên gần như câu hỏi nào bạn gặp cũng đã có người trả lời.

For a newcomer there are three reasons. Python code is short and readable, so you focus on ideas instead of clutter. You can type one line and see the result at once, which is handy for experimenting. The community is large, so almost any question you hit has already been answered.

Cách đọc số phiên bản – *How to read a version number*

Số phiên bản Python gồm ba phần nối bằng dấu chấm, ví dụ 3.14.0. Số đầu (3) là dòng chính. Số giữa (14) là bản chính phụ: mỗi năm Python ra một bản như vậy, mang thêm tính năng mới. Số cuối (0) là bản vá, chỉ sửa lỗi.

A Python version number has three parts joined by dots, such as 3.14.0. The first (3) is the major line. The middle one (14) is the feature release: Python ships one each year, adding new features. The last (0) is the patch number, which only fixes bugs.

Khi nói "Python 3.14", người ta chỉ hai số đầu. Mọi bản 3.14.x đều dùng được như nhau với người mới, nên bạn chỉ cần chọn đúng 3.14 rồi lấy số vá mới nhất.

When people say "Python 3.14", they mean the first two numbers. Every 3.14.x release works the same for a beginner, so just pick 3.14 and take the newest patch number.

Đôi khi bạn thấy đuôi như 3.14.0rc3. Chữ rc là release candidate, tức bản ứng viên phát hành. Đó là bản thử cuối trước khi chốt chính thức, rc3 là lần thử thứ ba. Dùng để thử nghiệm thì được, còn học hay làm việc thật thì lấy bản không có đuôi rc.

Sometimes you see a suffix like 3.14.0rc3. The letters rc stand for release candidate: a final trial build before the official release, and rc3 is the third trial. It is fine for testing, but for learning or real work take a version with no rc suffix.

Mỗi phần của số phiên bản là một số nguyên riêng, không phải số thập phân. Vì vậy 3.14 mới hơn 3.9, do 14 lớn hơn 9. Hãy so từng phần từ trái sang phải.

Each part of a version number is a separate whole number, not a decimal. So 3.14 is newer than 3.9, because 14 is greater than 9. Compare the parts one by one from left to right.

> **VÍ DỤ / EXAMPLE –** Bạn thấy hai bản 3.14.2 và 3.14.1 trên trang tải. Số đầu và số giữa giống nhau (3 và 14). Đến số cuối, 2 lớn hơn 1, nên 3.14.2 là bản vá mới hơn. Bạn lấy 3.14.2.
> *You see two builds, 3.14.2 and 3.14.1, on the download page. The first and middle numbers match (3 and 14). At the last number, 2 is greater than 1, so 3.14.2 is the newer patch. Take 3.14.2.*

Chọn bản nào và kiểm tra máy bạn đang có bản nào – *Which version to pick, and how to check what you have*

1. Mở dòng lệnh: Terminal trên macOS hoặc Linux, Command Prompt hoặc PowerShell trên Windows.
   Open a command line: Terminal on macOS or Linux, Command Prompt or PowerShell on Windows.
2. Gõ lệnh sau rồi nhấn Enter (macOS và Linux thường dùng python3 thay cho python):
   Type the following and press Enter (macOS and Linux usually use python3 instead of python):

```
python --version
```

1. Đọc dòng kết quả. Nếu thấy Python 3.14.x, bạn đã có bản mới nhất của dòng này.
   Read the result. If you see Python 3.14.x, you already have the latest of this line.
2. Nếu thấy số nhỏ hơn, chẳng hạn Python 3.12.4, hoặc máy báo không tìm thấy lệnh, hãy cài bản 3.14 theo hướng dẫn ở mục sau.

If you see a lower number such as Python 3.12.4, or the system says the command is not found, install 3.14 using the next section's instructions.

> **VÍ DỤ / EXAMPLE –** Bạn gõ python3 --version trên máy Mac và nhận Python 3.13.1. Máy này dùng được cho phần lớn bài học, vì các lệnh cơ bản giống nhau giữa các bản 3.x. Nhưng để các màn hình bạn thấy khớp với tài liệu này, hãy cài 3.14.
> *You type python3 --version on a Mac and get Python 3.13.1. That machine works for most lessons, since basic commands are the same across 3.x releases. But to make your screen match this guide, install 3.14.*

Thử dòng lệnh tương tác đầu tiên – *Trying the interactive prompt for the first time*

Trình gõ lệnh tương tác (REPL, viết tắt của read-eval-print loop) là nơi bạn gõ một dòng, Python chạy ngay và in kết quả, rồi chờ dòng tiếp theo. Giống nhắn tin qua lại với máy.

The interactive prompt (REPL, short for read-eval-print loop) is where you type one line, Python runs it and prints the result, then waits for the next line. It works like a chat with the machine.

1. Trong cửa sổ dòng lệnh, gõ python (hoặc python3 trên macOS và Linux) rồi nhấn Enter. Bạn sẽ thấy vài dòng giới thiệu và dấu >>> ở cuối.
   In the command window, type python (or python3 on macOS and Linux) and press Enter. You will see a few intro lines and a >>> mark at the end.
2. Sau dấu >>>, gõ phép tính 2 + 3 rồi nhấn Enter. Python trả về 5 ngay dưới dòng vừa gõ.
   After the >>> mark, type the sum 2 + 3 and press Enter. Python returns 5 right under the line you typed.
3. Gõ tiếp print("Xin chào") rồi nhấn Enter. Lệnh print bảo Python in chữ trong ngoặc ra màn hình, nên bạn thấy Xin chào.
   Next type print("Xin chào") and press Enter. The print command tells Python to show the text in the brackets, so you see Xin chào.
4. Gõ exit() rồi nhấn Enter để thoát về dòng lệnh thường.
   Type exit() and press Enter to return to the normal command line.

Điều người mới sẽ thấy ở 3.14 – *What a beginner will notice in 3.14*

Thay đổi dễ thấy nhất là trình gõ lệnh tương tác (gõ python rồi nhấn Enter, bạn thấy dấu >>>) giờ tô màu cú pháp ngay khi bạn gõ. Từ khoá, chuỗi chữ và số có màu riêng, nên lỗi gõ sai dễ lộ hơn. Tính năng này bật sẵn. Nó tắt nếu bạn đặt biến môi trường `PYTHON_BASIC_REPL` hoặc dùng biến tắt màu của hệ thống.

The most visible change is that the interactive prompt (type python and press Enter, and you see >>>) now colours syntax as you type. Keywords, strings and numbers get their own colours, so typos stand out. It is on by default. It switches off if you set the environment variable `PYTHON_BASIC_REPL` or use your system's colour-disabling variables.

Thông báo lỗi cũng được chăm cho thân thiện và chính xác hơn, nên đọc dòng cuối của báo lỗi là biết nên sửa gì. Cách đọc lỗi nằm ở Chương 7.

Error messages have also been polished to be friendlier and more accurate, so the last line of an error usually tells you what to fix. Reading errors is covered in Chapter 7.

Bản này còn có vài tính năng sâu: chế độ free-threaded (chạy nhiều luồng song song hơn), nhiều trình thông dịch trong thư viện chuẩn, chuỗi mẫu và nén Zstandard. Bạn chưa cần đến, cứ bỏ qua.

The release also has some deeper features: free-threaded mode (running threads in parallel more freely), multiple interpreters in the standard library, template strings and Zstandard compression. You do not need them yet, so skip them.

Bản free-threaded chạy chậm hơn bản thường một chút với chương trình đơn giản, và một số thư viện bên thứ ba chưa sẵn sàng cho nó. Đó là lý do nó là lựa chọn cài thêm, không phải mặc định.

The free-threaded build runs a little slower than the regular one on simple programs, and some third-party libraries are not ready for it yet. That is why it is an optional extra install, not the default.

LỖI HAY GẶP / COMMON MISTAKE – Cài nhầm bản rc hoặc bản free-threaded vì thấy chúng ở trang tải. Bản free-threaded là lựa chọn riêng, không cài mặc định; hãy dùng bản thường, không đuôi rc.

Installing an rc build or a free-threaded build because it appears on the download page. The free-threaded build is a separate option that is not installed by default; use the regular build with no rc suffix.

LỖI HAY GẶP / COMMON MISTAKE – Không thấy màu trong trình gõ lệnh rồi tưởng máy hỏng. Hãy kiểm tra bạn đang dùng bản 3.14 và biến `PYTHON_BASIC_REPL` chưa được đặt.

Seeing no colour at the prompt and assuming something is broken. Check that you are on 3.14 and that `PYTHON_BASIC_REPL` is not set.

LỖI HAY GẶP / COMMON MISTAKE – Đọc 3.14 như số thập phân rồi cho rằng nó cũ hơn 3.9. Hãy coi 14 và 9 là hai số nguyên và so chúng với nhau.

Reading 3.14 as a decimal and deciding it is older than 3.9. Treat 14 and 9 as two whole numbers and compare them.

MẸO / TIP – Học từ sách hoặc video cũ vẫn ổn, miễn là chúng dùng Python 3. Chỉ tránh tài liệu dùng Python 2, vì lệnh in ra màn hình và nhiều chi tiết khác đã khác. Ví dụ, Python 2 viết print "chào" không cần ngoặc, còn Python 3 bắt buộc có ngoặc: print("chào").

Older books and videos are fine as long as they use Python 3. Only avoid material written for Python 2, where printing to the screen and many other details differ. For example, Python 2 writes print "hello" without brackets, while Python 3 requires them: print("hello").

1.2. Cài Python trên Windows, macOS và Linux – *1.2. Installing Python on Windows, macOS and Linux*

Cài Python là đặt chương trình thông dịch lên máy. Thông dịch là đọc từng dòng lệnh bạn viết rồi làm theo. Bạn chỉ cần cài một lần cho mỗi máy, sau đó dùng cho mọi chương trình về sau. Mỗi hệ điều hành có cách cài hơi khác nhau, nhưng đều kết thúc bằng một phép thử: gõ lệnh hỏi phiên bản.

Installing Python puts the interpreter on your computer. The interpreter reads the commands you write, line by line, and carries them out. You install it once per machine and reuse it for every program afterwards. Each operating system has its own route, but all of them end with the same test: a command that asks for the version.

Trước khi bắt đầu, bạn cần biết PATH (đường dẫn tìm lệnh). Hãy hình dung PATH như cuốn danh bạ của máy. Khi bạn gõ python, máy lật danh bạ để tìm chương trình tên đó nằm ở thư mục nào. Python chưa có trong danh bạ thì máy báo không tìm thấy, dù chương trình đã nằm sẵn trên ổ đĩa.

First, you need to know PATH, the list of folders where the computer looks for commands. Think of it as the machine's phone book. When you type python, the computer looks up that name to find which folder holds the program. If Python is not in the book, the computer says it cannot find it, even though the program sits on your disk.

Windows – *Windows*

1. Mở trình duyệt, vào trang python.org, chọn mục Downloads. Trang tự gợi ý bản mới nhất cho Windows, bạn bấm nút tải đó. Chỉ tải từ trang chính thức này.
   Open your browser, go to python.org and choose Downloads. The page suggests the latest Windows version automatically. Click that download button, and download only from this official site.
2. Mở tệp vừa tải (thường nằm trong thư mục Downloads) bằng cách bấm đúp. Nếu Windows hỏi có cho phép thay đổi máy không, bấm Yes.
   Double-click the downloaded file, usually in your Downloads folder. If Windows asks whether to allow changes to your device, click Yes.
3. Ở cửa sổ đầu tiên, tìm ô có dòng chữ về việc thêm python.exe vào PATH và tích vào đó. Đây là ô dễ bị bỏ sót nhất. Giao diện bộ cài có thể khác chút giữa các phiên bản, nhưng ô này luôn nằm ở phần dưới.
   In the first window, find the checkbox about adding python.exe to PATH and tick it. This is the box people miss most often. The installer layout varies slightly between versions, but this box sits near the bottom.
4. Bấm Install Now và đợi thanh tiến trình chạy hết. Khi thấy thông báo thành công, bấm Close.
   Click Install Now and wait for the progress bar to finish. When you see the success message, click Close.
5. Bấm phím Windows, gõ cmd rồi nhấn Enter để mở cửa sổ dòng lệnh. Cửa sổ này là nơi bạn gõ lệnh bằng chữ thay vì bấm chuột.
   Press the Windows key, type cmd and press Enter to open a command window. This is where you give instructions by typing text instead of clicking.
6. Gõ lệnh sau rồi nhấn Enter:
   Type the following command and press Enter:

```
python --version
```

1. Đọc dòng máy in ra. Nếu thấy chữ Python kèm một dãy số, việc cài đã xong. Nếu thấy thông báo lỗi, chuyển xuống phần chỗ hay sai ở cuối mục.
   Read the line the computer prints. If you see the word Python followed by numbers, the

installation is done. If you see an error message, go to the common mistakes at the end of this section.

macOS – *macOS*

1. Vào python.org, chọn Downloads và tải bộ cài cho macOS. Tệp tải về có đuôi .pkg.
   Go to python.org, choose Downloads and get the macOS installer. The file ends in .pkg.

Trang tải còn ghi kèm checksum của từng tệp. Checksum là một chuỗi ký tự đặc trưng của tệp, giống số seri trên hộp hàng: nếu tệp bị hỏng hoặc bị sửa, chuỗi này sẽ khác. Người mới không bắt buộc kiểm tra, nhưng cần biết để hiểu vì sao trang có dòng đó.
The download page also lists a checksum for each file. A checksum is a fingerprint string for the file, like a serial number on a product box: if the file is damaged or altered, the string changes. Beginners are not required to check it, but it explains why that line is on the page.

1. Bấm đúp tệp .pkg. Ứng dụng Installer của macOS mở ra, bạn bấm Continue qua từng cửa sổ, đồng ý điều khoản, rồi bấm Install. Máy có thể hỏi mật khẩu đăng nhập của bạn.
   Double-click the .pkg file. The standard macOS Installer opens. Click Continue through each window, accept the terms, then click Install. The Mac may ask for your login password.
2. Nhấn Command + Space, gõ Terminal rồi nhấn Enter. Sau đó gõ lệnh sau (trên macOS, tên lệnh là python3, có số 3):
   Press Command + Space, type Terminal and press Enter. Then type this command (on macOS the command name is python3, with the 3):

```
python3 --version
```

Linux – *Linux*

Nhiều bản Linux đã cài sẵn Python, nên bạn hãy kiểm tra trước. Mở ứng dụng Terminal và gõ python3 --version. Nếu máy in ra số phiên bản thì bạn đã có thể dùng ngay. Nếu báo không tìm thấy hoặc phiên bản quá cũ, hãy cài bằng trình quản lý gói của hệ thống, tức là công cụ cài phần mềm có sẵn của bản Linux đó.
Many Linux systems ship with Python, so check first. Open the Terminal app and type python3 --version. If it prints a version number, you can start right away. If it says the command is not found, or the version is too old, install it with the system's package manager, the built-in tool your Linux uses to install software.

1. Trên Ubuntu hoặc Debian, gõ lệnh sau và nhập mật khẩu khi được hỏi:
   On Ubuntu or Debian, type this command and enter your password when asked:

```
sudo apt install python3
```

1. Trên Fedora, gõ:
   On Fedora, type:

```
sudo dnf install python3
```

1. Gõ lại python3 --version để kiểm tra.
   Run python3 --version again to check.

Đọc kết quả kiểm tra – *Reading the check result*

VÍ DỤ / EXAMPLE – Bạn gõ python --version trên Windows và máy in ra dòng Python 3.14.0. Dòng này nghĩa là máy đã tìm thấy Python trong danh bạ PATH và đây là bản 3.14.0. Số của bạn có thể khác nếu bạn tải bản mới hơn, và vậy vẫn đúng. Chỉ cần dòng bắt đầu bằng chữ Python và một dãy số.
You type python --version on Windows and the machine prints Python 3.14.0. That line means the computer found Python through PATH and the version is 3.14.0. Your number may differ if you downloaded a newer release, and that is fine. All you need is a line starting with the word Python followed by numbers.

Chạy thử một dòng lệnh đầu tiên – *Running your first line of code*

Biết Python đã cài chưa đủ. Bạn nên cho nó làm một việc nhỏ để chắc rằng nó chạy được. Cách nhanh nhất là đưa cho Python một phép tính ngay trên dòng lệnh.
Knowing Python is installed is not enough. Give it one small job to confirm it really runs. The quickest way is to hand it a calculation straight from the command line.

1. Mở cửa sổ dòng lệnh (cmd trên Windows, Terminal trên macOS và Linux).
   Open the command window (cmd on Windows, Terminal on macOS and Linux).
2. Gõ lệnh sau rồi nhấn Enter. Trên macOS và Linux, đổi chữ python thành python3. Hai dấu nháy kép bao quanh phần lệnh cần giữ nguyên.
   Type the following command and press Enter. On macOS and Linux, change python to python3. Keep the two double quotes around the code.

```
python -c "print(2 + 3)"
```

1. Đọc kết quả. Python tính 2 + 3 rồi in ra số 5 trên một dòng riêng. Thấy số 5 nghĩa là máy đọc và chạy đúng lệnh của bạn.
   Read the result. Python works out 2 + 3 and prints 5 on a line of its own. Seeing 5 means the machine read and ran your instruction correctly.

LỖI HAY GẶP / COMMON MISTAKE – Windows báo không tìm thấy lệnh python, hoặc tự mở cửa hàng Microsoft Store. Nguyên nhân thường gặp là bạn quên tích ô thêm vào PATH. Cách sửa: chạy lại bộ cài, chọn Modify hoặc cài lại và tích ô đó. Sau đó đóng cửa sổ dòng lệnh, mở cái mới rồi thử lại, vì cửa sổ cũ vẫn giữ danh bạ cũ.
Windows says the python command is not found, or opens the Microsoft Store. The usual cause is forgetting to tick the PATH box. Fix: run the installer again, choose Modify or reinstall, and tick that box. Then close the command window, open a new one and retry, because the old window keeps the old phone book.

LỖI HAY GẶP / COMMON MISTAKE – Trên macOS và Linux, gõ python thay vì python3 có thể báo lỗi. Hai hệ này dùng tên python3, nên hãy gõ đúng tên đó.
On macOS and Linux, typing python instead of python3 may fail. These systems use the name python3, so type that one.

LỖI HAY GẶP / COMMON MISTAKE – Trên Ubuntu hoặc Debian, lệnh cài báo không tìm thấy gói. Danh sách phần mềm của máy có thể đã cũ. Gõ sudo apt update để làm mới danh sách, rồi chạy lại lệnh cài.

On Ubuntu or Debian, the install command says it cannot find the package. The machine's software list may be out of date. Type sudo apt update to refresh it, then run the install command again.

LỖI HAY GẶP / COMMON MISTAKE – Vừa cài xong mà Terminal trên macOS vẫn không nhận lệnh. Hãy đóng Terminal, mở lại cửa sổ mới rồi gõ lại. Cửa sổ mở trước lúc cài chưa biết có Python.
Right after installing, the macOS Terminal still does not recognise the command. Close Terminal, open a fresh window and type it again. A window opened before the install does not know about Python.

MẸO / TIP – Trên Windows còn có lệnh py --version. Nếu python không chạy mà py chạy được, Python đã được cài đúng, chỉ là PATH chưa có. Bạn vẫn dùng py để làm các bài sau trong lúc sửa PATH.
Windows also has the command py --version. If python fails but py works, Python is installed correctly and only PATH is missing. You can keep using py for the later exercises while you fix PATH.

GHI CHÚ / NOTE – Ba việc cần nhớ: tải từ python.org, tích ô thêm vào PATH trên Windows, và kiểm tra bằng lệnh hỏi phiên bản (python trên Windows, python3 trên macOS và Linux).
Three things to remember: download from python.org, tick the PATH box on Windows, and check with the version command (python on Windows, python3 on macOS and Linux).

1.3. Gõ lệnh trực tiếp trong chế độ tương tác – *1.3. Typing Commands in Interactive Mode*

Chế độ tương tác giống như nói chuyện qua lại với Python: bạn gõ một câu lệnh, bấm Enter, Python trả lời ngay rồi chờ câu tiếp theo. Chương trình đứng ra đọc và chạy từng câu lệnh ấy gọi là trình thông dịch (interpreter). Chế độ này hợp để thử nhanh một phép tính hay kiểm tra một ý nghĩ trước khi viết thành tệp.
Interactive mode is a back-and-forth conversation with Python: you type one statement, press Enter, and Python answers right away, then waits for the next one. The program that reads and runs each statement is called the interpreter. It suits quick experiments, such as a calculation or a small idea you want to test before saving it in a file.

Mở trình thông dịch – *Starting the interpreter*

1. Trên Windows, mở Command Prompt hoặc PowerShell (gõ tên chúng vào ô tìm kiếm của menu Start). Trên macOS hoặc Linux, mở ứng dụng Terminal.
   On Windows, open Command Prompt or PowerShell (type its name into the Start menu search). On macOS or Linux, open the Terminal app.
2. Gõ lệnh sau rồi bấm Enter. Trên macOS và Linux, thường phải gõ python3 thay cho python. Trên Windows, nếu python không chạy, thử py.
   Type this command and press Enter. On macOS and Linux you usually need python3 instead of python. On Windows, if python does not work, try py.

```
python
```

1. Python in ra vài dòng chào, gồm số phiên bản (ví dụ 3.14) và thông tin bản quyền. Dòng cuối là dấu nhắc (prompt), tức ký hiệu báo máy đang chờ bạn gõ: >>>
   Python prints a short welcome message with the version number (for example 3.14) and a copyright notice. The last line is the prompt, the symbol telling you the interpreter is waiting for input: >>>

Thấy >>> nghĩa là bạn đang ở trong Python. Khi dấu này chưa xuất hiện, bạn vẫn ở màn hình dòng lệnh của hệ điều hành, và lệnh Python gõ vào đó sẽ báo lỗi.

Seeing >>> means you are inside Python. Until it appears, you are still in your operating system's command line, and Python statements typed there will fail.

Phép tính đầu tiên – *Your first calculation*

Sau dấu nhắc, gõ một phép tính rồi bấm Enter. Đừng gõ chính ba dấu >>>, vì máy đã in sẵn. Bạn chỉ gõ phần đứng sau nó.

After the prompt, type a calculation and press Enter. Do not type the three >>> signs yourself, since the interpreter prints them. You type only what comes after.

```
>>> 2 + 3
5
```

Dòng không có >>> là kết quả Python trả về. Dùng như máy tính bỏ túi thì đủ các phép sau (chương 2 sẽ nói kỹ từng phép):

The line without >>> is the answer from Python. As a pocket calculator it handles these operations (chapter 2 goes through each one in detail):

```
>>> 12500 * 4
50000
>>> 7 / 2
3.5
>>> 7 // 2
3
>>> 2 ** 10
1024
```

Dấu * là nhân, / là chia (cho kết quả thập phân), // là chia lấy phần nguyên, ** là luỹ thừa. Ví dụ, 7 chia 2 được 3,5 nhưng phần nguyên chỉ là 3.

*The sign * means multiply, / divides (giving a decimal result), // divides and keeps only the whole part, and ** raises to a power. For instance, 7 divided by 2 is 3.5, but its whole part is just 3.*

Còn một phép nữa hay dùng: % cho số dư của phép chia. Chia 7 cho 2 được 3, dư 1, nên 7 % 2 ra 1.

One more operation you will use often: % gives the remainder of a division. Dividing 7 by 2 gives 3 with 1 left over, so 7 % 2 returns 1.

Ví dụ làm mẫu: chia tiền mua đồ – *Worked example: splitting a shopping bill*

Bạn mua 3 cuốn vở giá 12500 đồng một cuốn và 2 cây bút giá 8000 đồng một cây. Ba người cùng chia tiền. Hãy để Python tính.

You buy 3 notebooks at 12500 dong each and 2 pens at 8000 dong each. Three people split the bill. Let Python do the work.

VÍ DỤ / EXAMPLE – Bước 1: gõ giá vở nhân số lượng, 3 * 12500 ra 37500. Bước 2: gõ 2 * 8000 ra 16000. Bước 3: cộng hai kết quả, 37500 + 16000 ra 53500. Bước 4: chia ba lấy phần nguyên, 53500 // 3 ra 17833. Bước 5: kiểm số dư, 53500 % 3 ra 1. Mỗi người trả 17833 đồng, còn dư 1 đồng, vì 17833 × 3 = 53499.

*Step 1: type the notebook price times the quantity, 3 * 12500 gives 37500. Step 2: type 2 * 8000, which gives 16000. Step 3: add the two results, 37500 + 16000 gives 53500. Step 4: divide by three and keep the whole part, 53500 // 3 gives 17833. Step 5: check the remainder, 53500 % 3 gives 1. Each person pays 17833 dong with 1 dong left over, because 17833 × 3 = 53499.*

```
>>> 3 * 12500 + 2 * 8000
53500
>>> 53500 // 3
17833
>>> 53500 % 3
1
```

Python nhân trước rồi mới cộng, giống toán ở trường. Muốn đổi thứ tự thì dùng ngoặc tròn, ví dụ (3 + 2) * 10 ra 50.

*Python multiplies before it adds, just like school maths. To change the order, use round brackets: (3 + 2) * 10 gives 50.*

Chữ cũng tính được – *Text works too*

Chữ đặt trong dấu nháy kép gọi là chuỗi (string). Bạn nối hai chuỗi bằng dấu +. Python in kết quả kèm dấu nháy đơn để bạn biết đó là chuỗi.

Text inside double quotes is called a string. You join two strings with +. Python prints the result with single quotes so you can tell it is a string.

```
>>> "Xin chào" + " bạn"
'Xin chào bạn'
```

Muốn ghi chú cho mình, gõ # rồi viết tiếp. Python bỏ qua mọi thứ từ # đến hết dòng. Ví dụ 5 * 6 # nhân ra 30.

*To leave a note for yourself, type # and keep writing. Python ignores everything from # to the end of the line. For example, 5 * 6 # multiply gives 30.*

Dòng tiếp nối ... – *Continuation lines: ...*

Một số câu lệnh gồm nhiều dòng, ví dụ câu lệnh if (nếu điều kiện đúng thì làm việc gì đó). Sau dòng đầu, Python đổi dấu nhắc thành ... để báo rằng nó chờ phần còn lại. Dòng trong thân lệnh phải thụt vào, ở đây là bốn dấu cách.

Some statements span several lines, such as an if statement (do something only when a condition holds). After the first line, Python changes the prompt to ... to say it is waiting for the rest. Lines inside the body must be indented, here by four spaces.

```
>>> tuoi = 20
>>> if tuoi >= 18:
...     print("Đủ tuổi")
...
Đủ tuổi
```

Dòng ... cuối cùng bỏ trống: bạn bấm Enter một lần nữa để báo đã gõ xong. Chỉ khi đó Python mới chạy cả khối.

The last ... line is empty: you press Enter once more to say you have finished. Only then does Python run the whole block.

Để ý dòng tuoi = 20: lệnh gán giá trị không in gì ra. Python chỉ in kết quả khi dòng bạn gõ là một biểu thức, tức thứ có thể tính ra giá trị. Trong tệp .py, kết quả như vậy không tự hiện, bạn phải dùng print().

Notice the line tuoi = 20: an assignment prints nothing. Python echoes a result only when the line is an expression, meaning something that can be evaluated to a value. In a .py file such results do not appear on their own, so you need print().

Xem lại lệnh cũ và huỷ dòng đang gõ – *Recalling old commands and cancelling a line*

- Bấm phím mũi tên lên để gọi lại lệnh vừa gõ, bấm tiếp để lùi xa hơn. Mũi tên xuống đi ngược lại. Sửa rồi bấm Enter để chạy lại.
  Press the up arrow to bring back the last command, and keep pressing to go further back. The down arrow goes the other way. Edit the line and press Enter to run it again.
- Gõ dở mà muốn bỏ, bấm Ctrl+C. Python huỷ dòng đó và trở về dấu >>>.
  If you want to abandon a half-typed line, press Ctrl+C. Python cancels it and returns to >>>.
- Đang ở dấu ... mà gõ nhầm cả khối, Ctrl+C cũng huỷ khối đó và đưa bạn về >>>.
  If you are at ... and the whole block is wrong, Ctrl+C cancels it too and takes you back to >>>.

Đọc phản hồi khi gõ sai – *Reading the response to a mistake*

Gõ sai không làm hỏng gì cả. Python in thông báo lỗi rồi quay lại dấu >>> cho bạn thử lại. Hai lỗi thường gặp ở người mới:

A mistake breaks nothing. Python prints an error message and returns to >>> so you can try again. Two errors beginners meet often:

```
>>> 10 / 0
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
>>> prin(1)
NameError: name 'prin' is not defined. Did you mean: 'print'?
```

Đọc dòng cuối cùng trước: nó nêu loại lỗi và lý do. Ở trên, một lỗi là chia cho 0, lỗi kia là gõ thiếu chữ t của print và Python còn gợi ý tên đúng.

Read the last line first: it names the kind of error and the reason. Above, one error is a division by zero, and the other is print missing its final t, where Python even suggests the right name.

Thoát khỏi Python – *Leaving Python*

1. Tại dấu >>>, gõ exit() rồi bấm Enter.
   At the >>> prompt, type exit() and press Enter.
2. Nếu không được, thử quit(), hoặc bấm Ctrl+D (macOS, Linux) hay Ctrl+Z rồi Enter (Windows).
   If that fails, try quit(), or press Ctrl+D (macOS, Linux) or Ctrl+Z then Enter (Windows).

Bản Python mới còn cho gõ exit hoặc quit không cần cặp ngoặc, nhưng gõ exit() thì bản nào cũng chạy.

Newer Python versions also accept exit or quit without parentheses, but exit() works on every version.

LỖI HAY GẶP / COMMON MISTAKE – Gõ python ngay khi đang ở dấu >>>, hoặc gõ lệnh Python như print("a") khi chưa mở Python, đều báo lỗi. Hãy nhìn đầu dòng: có >>> thì mới gõ lệnh Python.

Typing python while already at >>>, or typing a Python statement such as print("a") before starting Python, both produce errors. Check the start of the line: only type Python statements after >>>.

LỖI HAY GẶP / COMMON MISTAKE – Quên thụt lề ở dòng sau ..., hoặc quên bấm Enter thêm lần cuối, nên khối lệnh báo lỗi hoặc không chạy.

Forgetting the indent after ..., or forgetting the final extra Enter, makes the block fail or never run.

LỖI HAY GẶP / COMMON MISTAKE – Gõ dấu nháy cong (“ ”) do chép từ trang web hoặc Word. Python chỉ hiểu nháy thẳng " hoặc '. Nếu thấy SyntaxError ở dòng có chuỗi, hãy gõ lại dấu nháy bằng tay.

Typing curly quotes (“ ”) copied from a web page or a word processor. Python only understands straight quotes, " or '. If you see a SyntaxError on a line with a string, retype the quotes by hand.

GHI CHÚ / NOTE – Trên Windows, nếu máy báo không nhận ra lệnh python, thường là Python chưa được thêm vào đường dẫn tìm lệnh lúc cài. Thử gõ py trước khi cài lại.

On Windows, if the system does not recognise the python command, Python usually was not added to the command search path during installation. Try py before reinstalling.

1.4. Viết tệp .py và chạy từ dòng lệnh – *1.4. Writing a .py File and Running It from the Command Line*

Chế độ tương tác ở mục trước rất tiện để thử nhanh, nhưng đóng cửa sổ là mất hết. Muốn giữ lại chương trình và chạy lại bất cứ lúc nào, bạn lưu các lệnh vào một tệp văn bản có đuôi .py rồi nhờ Python chạy tệp đó.

The interactive mode from the previous topic is handy for quick tests, but everything is gone once you close the window. To keep a program and run it again whenever you like, save the commands in a text file ending in .py and ask Python to run that file.

Hình dung như viết công thức nấu ăn ra giấy. Nói miệng thì quên, còn viết ra thì lần nào cũng làm lại đúng y như vậy.

Think of writing a recipe on paper. If you only say it aloud you forget it, but a written recipe can be followed the same way every time.

Chuẩn bị – *What you need*

Bạn cần Python đã cài xong và một trình soạn thảo văn bản thuần, ví dụ Notepad trên Windows. Đừng dùng Word: nó thêm định dạng mà Python không hiểu.

You need Python installed and a plain-text editor, such as Notepad on Windows. Avoid Word, because it adds formatting that Python cannot read.

Các bước tạo và chạy tệp – *Creating and running the file*

1. Tạo một thư mục riêng cho bài tập, ví dụ thư mục baitap trong Documents. Việc này giúp bạn nhớ tệp nằm đâu.
   Create a folder for your exercises, for example baitap inside Documents. This makes it easy to remember where your files are.
2. Mở trình soạn thảo và gõ đúng các dòng của ví dụ chao.py ở phần sau. Dòng thụt vào dùng bốn dấu cách.
   Open the editor and type the lines of the chao.py example in the later part exactly. Indented lines use four spaces.
3. Chọn Save (Lưu), đặt tên chao.py, lưu vào thư mục baitap. Ở ô mã hoá (Encoding), chọn UTF-8.
   Choose Save, name the file chao.py, and save it in the baitap folder. In the Encoding box, choose UTF-8.
4. Mở cửa sổ dòng lệnh, tức khung chữ nơi bạn gõ lệnh cho máy thay vì bấm chuột. Cách mở xem bảng sau.
   Open a command-line window, the text box where you type instructions to the computer instead of clicking. The table that follows shows how.
5. Gõ cd rồi dấu cách và đường dẫn thư mục để đi vào đúng chỗ, ví dụ cd Documents/baitap. Trên Windows có thể dùng dấu gạch chéo ngược.
   Type cd, a space and the folder path to move into the right place, for example cd Documents/baitap. On Windows you can use backslashes.
6. Gõ lệnh chạy theo bảng, rồi nhấn Enter. Chữ in ra hiện ngay bên dưới dòng lệnh.
   Type the run command from the table and press Enter. The printed text appears right under the command.
7. Muốn sửa chương trình, sửa trong trình soạn thảo, bấm Save, rồi gõ lại lệnh chạy. Nếu quên lưu, Python vẫn chạy bản cũ.
   To change the program, edit it in the editor, press Save, then type the run command again. If you forget to save, Python still runs the old version.

Hệ điều hành *Operating system*	**Mở cửa sổ dòng lệnh bằng** *Open the command window with*	**Lệnh chạy thường dùng** *Usual run command*
Windows	Tìm Command Prompt hoặc PowerShell trong menu Start. *Search for Command Prompt or PowerShell in the Start menu.*	python chao.py, hoặc py chao.py nếu lệnh đầu không nhận. *python chao.py, or py chao.py if the first one is not recognised.*
macOS	Mở ứng dụng Terminal trong thư mục Utilities. *Open the Terminal app in the Utilities folder.*	python3 chao.py
Linux	Mở ứng dụng Terminal của môi trường desktop. *Open the Terminal app of your desktop environment.*	python3 chao.py

Truyền tham số cho chương trình – *Passing arguments to a program*

Tham số là những chữ bạn gõ thêm sau tên tệp, cách nhau bằng dấu cách. Chúng giúp một chương trình chạy với dữ liệu khác nhau mà không phải sửa tệp.

Arguments are the words you type after the file name, separated by spaces. They let one program run with different data without editing the file.

Python gom các chữ đó vào danh sách sys.argv. Phần tử đầu, sys.argv[0], là tên tệp. Phần tử tiếp theo, sys.argv[1], là tham số thứ nhất, và cứ thế. Mọi phần tử đều là chuỗi chữ.

Python collects those words in the list sys.argv. The first item, sys.argv[0], is the file name. The next one, sys.argv[1], is the first argument, and so on. Every item is a string of text.

Chạy python chao.py Lan thì sys.argv là ['chao.py', 'Lan']. Tên có dấu cách phải đặt trong dấu nháy kép, ví dụ python chao.py "Lan Anh", nếu không Python coi đó là hai tham số.

Running python chao.py Lan gives sys.argv as ['chao.py', 'Lan']. A name with a space must go in double quotes, as in python chao.py "Lan Anh", otherwise Python sees two arguments.

Vì mọi tham số đều là chữ, muốn tính toán bạn phải đổi sang số bằng int(). Ví dụ int("4") cho ra số 4. Bỏ bước này, phép cộng giữa số và chữ sẽ báo lỗi TypeError.

Because every argument is text, you must convert it to a number with int() before doing arithmetic. For example int("4") gives the number 4. Skip this step and adding a number to text raises a TypeError.

Vì sao lưu theo UTF-8 – *Why save as UTF-8*

UTF-8 là cách máy ghi mọi chữ cái thành số, gồm cả chữ có dấu như "ễ" hay "ợ". Python 3 mặc định đọc tệp .py theo UTF-8. Lưu kiểu khác thì chữ tiếng Việt bị đọc sai và Python có thể báo lỗi cú pháp về mã hoá ngay khi chạy.

UTF-8 is the way a computer records every letter as numbers, including accented letters such as "ễ" or "ợ". Python 3 reads .py files as UTF-8 by default. If you save in another encoding, Vietnamese text is misread and Python may stop with a syntax error about encoding.

Chạy nhanh khi không cần tạo tệp – *Quick runs without creating a file*

Với việc ngắn, bạn gõ mã thẳng trên dòng lệnh bằng tuỳ chọn -c. Python chạy đoạn chữ đặt trong dấu nháy rồi thoát, không để lại gì. Ví dụ python -c "print(2 + 3)" in ra 5.

For short jobs, put the code straight on the command line with the -c option. Python runs the quoted text and exits, leaving nothing behind. For example python -c "print(2 + 3)" prints 5.

Python còn chạy được các module có sẵn như một chương trình. Module là tệp mã đã viết trước, giống dụng cụ có sẵn trong bếp. Gõ python -m calendar 2000 để in lịch cả năm 2000. Số 2000 là một tham số, giống tên Lan ở trên.

Python can also run built-in modules as programs. A module is a file of ready-made code, like a tool already in the kitchen. Type python -m calendar 2000 to print the calendar for the whole year 2000. The number 2000 is an argument, just like the name Lan above.

> **VÍ DỤ / EXAMPLE –** Tệp chao.py chào tên người dùng. Nội dung tệp: import sys, rồi if len(sys.argv) > 1: thụt vào ten = sys.argv[1], rồi else: thụt vào ten = "bạn", rồi print("Xin chào,", ten), và dòng cuối print("Chúc bạn học Python vui!"). Chạy python chao.py Lan in ra

"Xin chào, Lan" và dòng chúc. Chạy python chao.py không kèm tên thì in "Xin chào, bạn". Dòng import sys lấy công cụ có sẵn để đọc tham số. Dòng if kiểm tra bạn có gõ thêm tên hay không: len đếm số phần tử, nên lớn hơn 1 nghĩa là có tham số.
The file chao.py greets the user by name. File content: import sys, then if len(sys.argv) > 1: with ten = sys.argv[1] indented, then else: with ten = "bạn" indented, then print("Xin chào,", ten), and a last line print("Chúc bạn học Python vui!"). Running python chao.py Lan prints "Xin chào, Lan" and the wish line. Running python chao.py with no name prints "Xin chào, bạn". The line import sys fetches a built-in tool for reading arguments. The if line checks whether you typed a name: len counts the items, so more than 1 means an argument was given.

```
python chao.py Lan
```

VÍ DỤ / EXAMPLE – Tệp tong.py cộng các số gõ sau tên tệp. Nội dung: import sys, rồi tong = 0, rồi for so in sys.argv[1:]: thụt vào tong = tong + int(so), và dòng cuối print("Tổng là", tong). Chạy python tong.py 4 7 10 thì sys.argv là ['tong.py', '4', '7', '10']. Phần cắt [1:] bỏ tên tệp, chỉ giữ '4', '7', '10'. Vòng for lấy từng chữ, int đổi thành số. Biến tong đi qua các giá trị 0, 4, 11 rồi 21. Chương trình in "Tổng là 21".
The file tong.py adds the numbers typed after the file name. Content: import sys, then tong = 0, then for so in sys.argv[1:]: with tong = tong + int(so) indented, and a last line print("Tổng là", tong). Running python tong.py 4 7 10 makes sys.argv equal to ['tong.py', '4', '7', '10']. The slice [1:] drops the file name and keeps '4', '7', '10'. The for loop takes each text item and int turns it into a number. The variable tong goes through 0, 4, 11 and then 21. The program prints "Tổng là 21".

```
python tong.py 4 7 10
```

LỖI HAY GẶP / COMMON MISTAKE – Báo "can't open file": bạn đang đứng sai thư mục, hoặc gõ sai tên tệp. Dùng cd để vào đúng thư mục, kiểm tra lại tên.
"can't open file" means you are in the wrong folder or mistyped the file name. Use cd to enter the right folder and check the name.

LỖI HAY GẶP / COMMON MISTAKE – Notepad lưu thành chao.py.txt vì để kiểu tệp là văn bản. Chọn kiểu All files (Tất cả tệp) khi lưu rồi đặt lại tên.
Notepad saves chao.py.txt when the file type is left as text. Choose All files when saving, then set the name again.

LỖI HAY GẶP / COMMON MISTAKE – Báo lỗi thụt lề (IndentationError) vì dòng sau if hoặc else không thụt vào, hoặc thụt không đều. Dùng đúng bốn dấu cách.
An IndentationError appears when the line after if or else is not indented, or is indented unevenly. Use exactly four spaces.

LỖI HAY GẶP / COMMON MISTAKE – Báo python không được nhận ra: Python chưa nằm trong đường dẫn tìm lệnh. Thử py chao.py trên Windows, hoặc cài lại và tick ô thêm Python vào PATH.
A "python is not recognised" message means Python is not on the command search path. Try py chao.py on Windows, or reinstall and tick the option that adds Python to PATH.

LỖI HAY GẶP / COMMON MISTAKE – Gõ python chao.py ngay sau dấu nhắc >>> sẽ báo SyntaxError. Dấu >>> nghĩa là bạn còn trong chế độ tương tác. Gõ exit() rồi nhấn Enter để thoát, sau đó chạy lại lệnh.
Typing python chao.py right after the >>> prompt gives a SyntaxError. The >>> sign means you are still in interactive mode. Type exit() and press Enter to leave, then run the command again.

LỖI HAY GẶP / COMMON MISTAKE – Quên đổi tham số sang số: tong + so với so là chữ '4' báo TypeError. Luôn bọc int() quanh tham số khi cần tính.
Forgetting to convert an argument: adding the text '4' to a number raises a TypeError. Always wrap the argument in int() when you need to calculate.

1.5. Tuỳ chọn dòng lệnh và biến môi trường của Python – *1.5. Command-Line Options and Environment Variables*

Biến môi trường giống một tờ ghi chú dán sẵn trên cánh cửa nhà. Ai bước vào, kể cả Python, cũng đọc được. Hệ điều hành giữ các ghi chú này dưới dạng cặp "tên = giá trị", ví dụ PYTHONPATH = C:\mycode. Bạn đặt một lần thì mọi lần chạy Python sau đó đều thấy.
Think of an environment variable as a sticky note on the front door. Anyone who walks in, Python included, can read it. The operating system stores these notes as "name = value" pairs, for example PYTHONPATH = C:\mycode. Set one once and every later Python run can see it.

Tuỳ chọn dòng lệnh thì khác: đó là lời dặn riêng cho một lần chạy, gõ ngay sau chữ python. Khi hai bên mâu thuẫn, tuỳ chọn dòng lệnh thắng biến môi trường.
A command-line option is different: it is an instruction for a single run, typed right after the word python. When the two conflict, the command-line option wins over the environment variable.

GHI CHÚ / NOTE – Trên Windows bạn có thể gõ python hoặc py. Trên macOS và Linux thường phải gõ python3. Các ví dụ sau viết python, bạn đổi theo máy của mình.
On Windows you can type python or py. On macOS and Linux you usually need python3. The examples use python; swap in the name that works on your machine.

Ba tuỳ chọn dùng nhiều nhất – *The three options you will use most*

Trước khi thử, mở Terminal (macOS, Linux) hoặc Command Prompt (Windows). Đó là cửa sổ chữ nơi bạn gõ lệnh.
Before you try them, open Terminal (macOS, Linux) or Command Prompt (Windows). It is the text window where you type commands.

1. Gõ lệnh sau rồi nhấn Enter để xem phiên bản đang cài:
   Type the following and press Enter to see which version is installed:

```
python -V
```

1. Gõ lệnh sau để chạy một đoạn mã ngắn mà không cần tạo tệp. Chữ -c là viết tắt của "command" (lệnh), và đoạn mã đặt trong dấu nháy kép:
   Type the following to run a short piece of code without creating a file. The letter c stands for "command", and the code goes inside double quotes:

```
python -c "print(2 + 3)"
```

1. Gõ lệnh sau để chạy một module (tệp mã có sẵn) theo tên. Tên viết không có đuôi .py:
   Type the following to run a module (a ready-made code file) by name. Write the name without the .py ending:

```
python -m pip --version
```

> **VÍ DỤ / EXAMPLE –** Bước thứ hai in ra số 5, vì Python tính 2 + 3 rồi in kết quả. Bước thứ ba cho bạn biết bản pip đang gắn với đúng bản Python bạn vừa gọi. Cách gọi "python -m pip" sẽ còn dùng nhiều ở chương về cài thư viện.
> *The second step prints 5, because Python works out 2 + 3 and prints the result. The third step shows which pip belongs to the exact Python you just called. You will use "python -m pip" a lot in the chapter on installing libraries.*

Khi không gõ tuỳ chọn nào – *Running Python with no options*

Gõ python một mình sẽ mở chế độ tương tác. Bạn thấy dấu nhắc >>> và gõ từng dòng mã, Python trả lời ngay. Thư mục hiện tại được thêm vào đầu đường tìm module, nên module nằm cạnh bạn import được luôn. Muốn thoát, gõ exit().

Typing python on its own opens interactive mode. You see the >>> prompt, type one line of code at a time, and Python answers at once. The current folder is added to the front of the module search path, so a module sitting next to you can be imported straight away. To leave, type exit().

Xem trợ giúp ngay trong dòng lệnh – *Getting help from the command line*

Gõ python -h (hoặc python --help) để in danh sách tuỳ chọn và biến môi trường. Từ bản 3.11 còn có python --help-env, chỉ liệt kê các biến môi trường riêng của Python, và python --help-all, in đầy đủ mọi thông tin. Nếu danh sách trôi quá nhanh, kéo thanh cuộn của cửa sổ lên đọc lại.

Type python -h (or python --help) to print the list of options and environment variables. Since version 3.11 there is also python --help-env, which lists only Python's own environment variables, and python --help-all, which prints everything. If the text scrolls past too fast, scroll the window back up.

Ba tuỳ chọn cần biết thêm – *Three more options worth knowing*

- -B: Python không ghi các tệp .pyc (bản mã đã biên dịch sẵn) khi nạp module. Biến môi trường tương ứng là PYTHONDONTWRITEBYTECODE.
  -B: Python does not write .pyc files (precompiled copies of your code) when it imports modules. The matching environment variable is PYTHONDONTWRITEBYTECODE.
- -I: chạy ở chế độ cô lập. Python bỏ qua mọi biến bắt đầu bằng PYTHON, kể cả PYTHONPATH, và không thêm thư mục của tệp đang chạy vào đường tìm module. Hữu ích khi bạn chạy tệp tải từ mạng về.
  -I: runs Python in isolated mode. Python ignores every variable starting with PYTHON, PYTHONPATH included, and does not add the script's folder to the module search path. Handy when you run a file you downloaded.
- -O: bỏ các câu lệnh assert. Assert là câu kiểm tra, ví dụ "assert tuổi > 0" sẽ báo lỗi nếu tuổi không dương. Có -O thì các câu kiểm tra này bị bỏ qua. Tệp biên dịch cũng đổi tên, thêm .opt-1 trước đuôi .pyc.
  -O: removes assert statements. An assert is a self-check, for example "assert age > 0" raises an error if age is not positive. With -O these checks are skipped. Compiled files are also renamed, with .opt-1 added before the .pyc ending.

> **VÍ DỤ / EXAMPLE –** Gõ python -c "assert 1 == 2" thì Python báo AssertionError, vì 1 không bằng 2. Gõ python -O -c "assert 1 == 2" thì không có gì hiện ra, vì câu assert đã bị bỏ. Vì vậy đừng dùng assert để chặn dữ liệu xấu từ người dùng.
> *Run python -c "assert 1 == 2" and Python reports an AssertionError, because 1 is not equal to 2. Run python -O -c "assert 1 == 2" and nothing appears, because the assert was removed. That is why you should not rely on assert to block bad input from users.*

Đặt biến môi trường theo từng hệ điều hành – *Setting environment variables on each system*

Ví dụ dùng PYTHONPATH: danh sách thư mục mà Python tìm thêm khi bạn viết import. Các thư mục cách nhau bằng dấu hai chấm trên macOS, Linux và dấu chấm phẩy trên Windows.
The example uses PYTHONPATH: a list of extra folders Python searches when you write import. Folders are separated by a colon on macOS and Linux, and by a semicolon on Windows.

1. Windows, Command Prompt, chỉ cho cửa sổ hiện tại: gõ lệnh sau. Đóng cửa sổ là mất.
   Windows, Command Prompt, current window only: type the following. Closing the window discards it.

```
set PYTHONPATH=C:\mycode
```

1. Windows, PowerShell, chỉ cho cửa sổ hiện tại:
   Windows, PowerShell, current window only:

```
$env:PYTHONPATH = "C:\mycode"
```

1. Windows, lưu lâu dài: gõ lệnh sau, rồi mở cửa sổ mới mới thấy hiệu lực.
   Windows, saved permanently: type the following, then open a new window to see the effect.

```
setx PYTHONPATH "C:\mycode"
```

1. macOS và Linux, cho cửa sổ hiện tại:
   macOS and Linux, current window only:

```
export PYTHONPATH=$HOME/mycode
```

1. macOS và Linux, lưu lâu dài: thêm đúng dòng export trên vào cuối tệp cấu hình của shell (thường là ~/.zshrc trên macOS, ~/.bashrc trên nhiều bản Linux), rồi mở cửa sổ mới.
   macOS and Linux, saved permanently: add the same export line to the end of your shell's startup file (usually ~/.zshrc on macOS, ~/.bashrc on many Linux systems), then open a new window.
2. Kiểm tra kết quả bằng lệnh sau; thư mục của bạn phải nằm trong danh sách in ra:
   Check the result with the following; your folder should appear in the printed list:

```
python -c "import sys; print(sys.path)"
```

Thử xem tuỳ chọn thắng biến môi trường – *Seeing an option override a variable*

Sau khi đã đặt PYTHONPATH ở bước trên, bạn có thể tự kiểm chứng chế độ cô lập. Vẫn trong cửa sổ đó, gõ thêm một lệnh có -I và so hai danh sách in ra.

Once PYTHONPATH is set as above, you can check isolated mode yourself. In the same window, run one more command with -I and compare the two printed lists.

```
python -I -c "import sys; print(sys.path)"
```

VÍ DỤ / EXAMPLE – Giả sử bạn đặt PYTHONPATH trỏ tới thư mục mycode. Lệnh không có -I in ra danh sách có chứa mycode. Lệnh có -I in ra danh sách không có mycode, vì Python đã bỏ qua biến đó. Hai kết quả khác nhau là dấu hiệu chế độ cô lập đang chạy đúng.
Suppose you set PYTHONPATH to point at the mycode folder. The command without -I prints a list that contains mycode. The command with -I prints a list without it, because Python ignored the variable. Two different results show that isolated mode is working.

LỖI HAY GẶP / COMMON MISTAKE – Dùng setx xong mà cửa sổ đang mở vẫn không thấy biến. Setx chỉ ảnh hưởng cửa sổ mở sau đó. Trên macOS và Linux, export gõ tay cũng mất khi đóng Terminal.
After running setx, the window you already have open still does not see the variable. Setx only affects windows opened afterwards. On macOS and Linux, a hand-typed export is also lost when you close Terminal.

LỖI HAY GẶP / COMMON MISTAKE – Gõ python -m hello.py thay vì python -m hello. Sau -m là tên module, không có đuôi tệp. Muốn chạy tệp thì gõ python hello.py.
Typing python -m hello.py instead of python -m hello. After -m comes a module name with no file ending. To run a file, type python hello.py.

LỖI HAY GẶP / COMMON MISTAKE – Trên Windows, ngăn các thư mục trong PYTHONPATH bằng dấu hai chấm. Ổ đĩa như C: đã có sẵn dấu hai chấm, nên Python hiểu sai. Windows dùng dấu chấm phẩy.
On Windows, separating the folders in PYTHONPATH with colons. A drive like C: already contains a colon, so Python misreads the list. Windows uses semicolons.

LỖI HAY GẶP / COMMON MISTAKE – Cài thư viện cho một bản Python rồi chạy bằng bản khác, nên báo không tìm thấy module. Các bản Python cài riêng có thư mục thư viện riêng. Hãy gọi "python -m pip" với đúng bản bạn sẽ chạy, hoặc dùng môi trường ảo (venv) cho mỗi dự án.
Installing a library for one Python and running another, then getting a "module not found" error. Separate Python installations have separate library folders. Call "python -m pip" with the exact Python you will run, or use a virtual environment (venv) for each project.

MẸO / TIP – Đừng đặt PYTHONHOME nếu chưa hiểu rõ. Biến này đổi nơi Python tìm thư viện chuẩn, đặt sai thì Python không khởi động nổi. Lỡ đặt rồi, hãy xoá biến đó đi.
Leave PYTHONHOME alone unless you know why you need it. It changes where Python looks for its standard library, and a wrong value can stop Python from starting. If you set it by mistake, remove it.

MẸO / TIP – Máy có hai bản Python cùng đọc một biến PYTHONPATH, và một bản có thể nạp nhầm module của bản kia. Khi thấy lỗi lạ sau lúc đặt biến, thử chạy lại với -I. Hết lỗi thì thủ phạm nằm ở biến môi trường.
If two Python versions on one machine read the same PYTHONPATH, one may load modules meant for the other. When you see odd errors after setting a variable, rerun with -I. If the error disappears, an environment variable is the culprit.

1.6. Tìm trợ giúp: help(), tài liệu chính thức và hỏi đáp chung – *1.6. Getting Help: help(), Official Docs and General FAQ*

Khi quên một hàm làm gì, bạn không cần mở trình duyệt. Python có sẵn lời giải thích ngay trong cửa sổ đang gõ lệnh, giống như cuốn từ điển bỏ túi nằm cạnh bàn học. Cách dùng cuốn từ điển đó, cách đọc tài liệu chính thức và nơi hỏi khi vẫn bí nằm ngay sau đây.

When you forget what a function does, you don't need a browser. Python carries its own explanations inside the window where you type, like a pocket dictionary on your desk. How to use that dictionary, how to read the official documentation, and where to ask when you are still stuck come next.

Hỏi Python bằng help() – *Asking Python with help()*

help() là hàm có sẵn, in ra phần mô tả của một hàm, kiểu dữ liệu hay module. Bạn cần mở chế độ tương tác (dấu nhắc >>>) như đã làm ở phần trước.

help() is a built-in function that prints the description of a function, data type or module. You need the interactive mode (the >>> prompt) from the earlier part.

1. Mở cửa sổ dòng lệnh, gõ python rồi nhấn Enter để thấy dấu nhắc >>>.
   Open a terminal window, type python and press Enter until the >>> prompt appears.
2. Gõ help(len) rồi nhấn Enter. Chú ý chỉ ghi tên hàm, không thêm cặp ngoặc sau len.
   Type help(len) and press Enter. Write only the function name, with no extra parentheses after len.
3. Đọc kết quả. Dòng đầu cho biết len là hàm dựng sẵn. Dòng kế là cách gọi, rồi đến lời mô tả: len trả về số phần tử của một vật chứa như chuỗi hay danh sách. Dấu / trong cách gọi sẽ học ở chương 4, tạm bỏ qua.
   Read the output. The first line says len is a built-in function. The next shows how to call it, followed by a description: len returns the number of items in a container such as a string or list. The / in the call line is covered in chapter 4, so skip it for now.
4. Nếu màn hình dừng lại và hiện dấu hai chấm hoặc chữ END ở cuối, nhấn q để thoát về dấu nhắc >>>. Dùng phím mũi tên hoặc phím cách để cuộn khi văn bản dài.
   If the screen pauses and shows a colon or the word END at the bottom, press q to return to the >>> prompt. Use the arrow keys or the space bar to scroll when the text is long.

Muốn tra nhiều thứ liên tiếp, gõ help() không có gì trong ngoặc. Dấu nhắc đổi thành help>. Bạn gõ tên cần tra, ví dụ str, và gõ quit để quay lại >>>. Nếu chỉ gõ help không ngoặc, Python nhắc cách dùng.

To look up several things in a row, type help() with nothing inside. The prompt changes to help>. Type a name such as str, then type quit to go back to >>>. If you type help without parentheses, Python reminds you how to use it.

> **VÍ DỤ / EXAMPLE –** Gõ help(abs). Python mô tả abs trả về giá trị tuyệt đối của một số. Thử tiếp abs(-7) ra 7, abs(3.5) ra 3.5. Cách làm: đọc mô tả, rồi gõ thử một vài giá trị để kiểm tra mình hiểu đúng.
> *Type help(abs). Python describes abs as returning the absolute value of a number. Then try abs(-7),*

which gives 7, and abs(3.5), which gives 3.5. The routine: read the description, then test a few values to check you understood.

LỖI HAY GẶP / COMMON MISTAKE – Gõ help(len()) khiến Python báo TypeError vì len() bị gọi mà chưa có gì để đếm. Hãy viết help(len).
Typing help(len()) makes Python raise a TypeError because len() is called with nothing to count. Write help(len) instead.

Quên tên hàm thì dùng dir() – *Forgetting the name? Use dir()*

Đôi khi bạn nhớ mang máng có cách đổi chữ thường thành chữ hoa mà quên mất tên. dir() là hàm có sẵn, liệt kê tên mọi thứ gắn với một đối tượng, giống mục lục đầu sách.
Sometimes you vaguely remember a way to turn lowercase letters into capitals but forget its name. dir() is a built-in function that lists every name attached to an object, like the table of contents of a book.

1. Gõ dir(str) rồi nhấn Enter. Python in một danh sách tên. Những tên bắt đầu bằng hai dấu gạch dưới là loại nội bộ, tạm bỏ qua.
   Type dir(str) and press Enter. Python prints a list of names. Names that start with two underscores are internal ones, so skip them for now.
2. Tìm tên giống thứ cần. Trong danh sách có upper và lower.
   Look for a name that matches what you need. The list contains upper and lower.
3. Gõ help(str.upper) để đọc mô tả, rồi thử với một chuỗi cụ thể.
   Type help(str.upper) to read the description, then try it on a real string.

VÍ DỤ / EXAMPLE – Gõ "xin chao".upper(), Python trả về 'XIN CHAO'. Gõ tiếp "Ha Noi".lower(), kết quả là 'ha noi'. Chuỗi gốc không đổi, vì cả hai hàm tạo ra một chuỗi mới.
Type "xin chao".upper() and Python returns 'XIN CHAO'. Then "Ha Noi".lower() gives 'ha noi'. The original string stays the same, because both methods build a new string.

Tra hàm nằm trong module – *Looking up a function inside a module*

Module là một tệp chứa sẵn nhiều hàm cùng chủ đề, như math cho các phép toán. Chúng nằm "trên kệ" và bạn phải lấy xuống bằng lệnh import trước khi dùng.
A module is a file holding many ready-made functions on one theme, such as math for calculations. They sit "on a shelf", and you must take them down with the import statement before use.

1. Gõ import math rồi nhấn Enter. Không có thông báo gì nghĩa là đã nạp xong.
   Type import math and press Enter. No message means it loaded fine.
2. Gõ help(math.sqrt) để đọc mô tả: hàm trả về căn bậc hai của một số.
   Type help(math.sqrt) to read the description: the function returns the square root of a number.
3. Gõ thử math.sqrt(81), kết quả là 9.0. Gõ math.sqrt(2), kết quả là 1.4142135623730951.
   Try math.sqrt(81), which gives 9.0. Try math.sqrt(2), which gives 1.4142135623730951.

> **LỖI HAY GẶP / COMMON MISTAKE –** Gõ help(math) khi chưa import sẽ báo NameError: name 'math' is not defined. Hãy chạy import math trước.
> *Typing help(math) before importing gives NameError: name 'math' is not defined. Run import math first.*

Tài liệu chính thức trên python.org – *The official documentation on python.org*

Tài liệu cho bản Python ổn định hiện hành nằm ở docs.python.org/3/. Trang có ô tìm kiếm. Bạn cũng tải được bản PDF, văn bản thường hoặc HTML về máy tại docs.python.org/3/download.html để đọc khi không có mạng.

The documentation for the current stable Python is at docs.python.org/3/. The site has a search box. You can also download PDF, plain text or HTML versions from docs.python.org/3/download.html to read offline.

Người mới nên đọc theo thứ tự này: phần Tutorial trước, vì nó dẫn từng bước qua ngôn ngữ. Phần Library Reference dùng để tra từng module sau khi đã quen. Lưu ý Tutorial của trang chính thức viết cho người từng lập trình ở ngôn ngữ khác. Nếu bạn chưa biết gì, hãy bắt đầu từ trang Beginner's Guide trên python.org, nơi có danh sách giáo trình và sách cho người mới.

Beginners should read in this order: the Tutorial first, since it walks through the language step by step. The Library Reference is for looking up modules once you are comfortable. Note that the official Tutorial is written for people who have programmed in another language. If you know nothing yet, start from the Beginner's Guide on python.org, which lists tutorials and books for newcomers.

Một trang mô tả hàm thường có ba phần: dòng cách gọi, lời giải thích và ví dụ. Trong dòng cách gọi, chỗ ghi dạng tên=giá trị, chẳng hạn sep=None, là tham số có giá trị mặc định. Bạn có thể bỏ qua nó và Python tự dùng giá trị đó.

A function's page usually has three parts: the call line, an explanation and examples. In the call line, anything written as name=value, such as sep=None, is a parameter with a default value. You may leave it out and Python uses that value.

- Cần xem nhanh một hàm đã biết tên: dùng help() ngay trong cửa sổ, vì không cần mạng và kết quả khớp đúng bản Python bạn đang chạy.
  Need a quick look at a function whose name you know: use help() in the window, since it needs no internet and matches the exact Python version you are running.
- Quên tên hàm: dùng dir() để xem danh sách tên, rồi help() để đọc kỹ từng tên.
  Forgot the name: use dir() to see the list of names, then help() to read each one closely.
- Muốn học có hệ thống hoặc xem ví dụ dài: mở tài liệu trên python.org.
  Want structured learning or longer examples: open the documentation on python.org.

Vài câu hỏi hay gặp – *Common questions*

- Python có tốn tiền hay bị hạn chế bản quyền không? Python miễn phí, tải về dùng và phát hành lại được, kể cả khi làm sản phẩm bán tiền. Giấy phép do Python Software Foundation (tổ chức phi lợi nhuận chăm sóc Python) quản lý; nếu cần phát hành lại, hãy đọc nội dung giấy phép đi kèm.

Is Python paid or copyright-restricted? Python is free to download, use and redistribute, even inside products you sell. The license is managed by the Python Software Foundation, the non-profit that looks after Python. If you plan to redistribute it, read the license that ships with it.

- Có ổn định không? Mỗi phiên bản có số hiệu riêng và tài liệu ghi rõ bản nào đang là ổn định. Hãy chọn bản ổn định hiện hành để học.
  Is it stable? Each version has its own number, and the documentation states which one is the current stable release. Pick the current stable version for learning.
- Có hợp để học lập trình đầu tiên không? Có. Cú pháp gọn và nhất quán, thư viện chuẩn rộng, nên bạn tập trung vào cách chia nhỏ bài toán thay vì khai báo rườm rà. Bạn còn có chế độ tương tác để thử ngay mọi thứ, và IDLE, trình soạn thảo đi kèm Python, để viết mã.
  Is it a good first language? Yes. The syntax is short and consistent and the standard library is large, so you focus on breaking problems into steps instead of heavy declarations. You also get interactive mode for trying things at once, and IDLE, the editor bundled with Python, for writing code.

Báo lỗi và hỏi cộng đồng – *Reporting bugs and asking the community*

Trước khi nghĩ Python bị lỗi, hãy chắc là mã của bạn sai chưa. Đa số trường hợp, lỗi nằm ở mã. Hỏi cộng đồng thì làm như sau:

Before you decide Python itself is broken, check your own code first. In most cases the fault is there. To ask the community, do this:

1. Tìm trong tài liệu và gõ nguyên thông báo lỗi lên công cụ tìm kiếm. Rất có thể người khác đã gặp.
   Search the documentation, then paste the exact error message into a search engine. Someone has probably met it before.
2. Nếu chưa ra, đăng câu hỏi lên diễn đàn discuss.python.org. Ghi rõ bản Python, hệ điều hành, đoạn mã ngắn nhất gây lỗi và toàn bộ thông báo lỗi.
   If nothing turns up, post on the forum at discuss.python.org. State your Python version, operating system, the shortest code that triggers the problem, and the full error message.
3. Chỉ khi chắc lỗi nằm ở Python, hãy mở một issue trên kho mã python/cpython ở GitHub, kèm các thông tin trên.
   Only when you are sure the fault is in Python itself, open an issue in the python/cpython repository on GitHub with the same details.

> **MẸO / TIP –** Chụp hoặc sao chép đầy đủ thông báo lỗi, đừng tóm tắt bằng lời. Người trả lời cần đúng từng chữ để giúp bạn.
> *Copy the full error message rather than summarising it. Helpers need the exact wording to assist you.*
>
> **MẸO / TIP –** Muốn biết mình đang chạy bản nào để ghi vào câu hỏi, gõ python --version trong cửa sổ dòng lệnh. Thông báo có dạng Python 3 kèm số phiên bản.
> *To find which version you run for your question, type python --version in the terminal. The output reads Python 3 followed by the version number.*

1.7. Thực hành: chương trình chào và tính tuổi – *1.7. Practice: A Greeting and Age Calculator*

Bài này gom những gì bạn đã học thành một chương trình nhỏ. Chương trình hỏi tên và năm sinh, rồi chào và tính tuổi. Bạn sẽ dùng input() để nhận chữ gõ từ bàn phím, int() để đổi chữ thành số, và print() để in kết quả.

This exercise puts what you have learned into one small program. It asks for your name and birth year, greets you, and works out your age. You will use input() to take typed text, int() to turn text into a number, and print() to show the result.

Chuẩn bị: Python đã cài và gõ được lệnh kiểm tra phiên bản trong cửa sổ dòng lệnh, cùng một trình soạn thảo văn bản bất kỳ (Notepad, TextEdit, gedit hoặc trình soạn thảo mã).

What you need: Python installed and a command window where the version check works, plus any text editor (Notepad, TextEdit, gedit, or a code editor).

Viết chương trình – *Write the program*

1. Tạo một thư mục riêng, ví dụ baitap, ở nơi dễ nhớ như thư mục Documents. Việc gom tệp vào một chỗ giúp bạn khỏi lạc khi chạy.
   Create a folder of its own, for example baitap, somewhere easy to remember such as Documents. Keeping files in one place stops you from getting lost when you run them.
2. Mở trình soạn thảo, tạo tệp mới và lưu vào thư mục đó với tên chao.py. Chọn mã hoá UTF-8 nếu trình soạn thảo cho chọn, để tiếng Việt có dấu không bị vỡ chữ.
   Open the editor, create a new file and save it in that folder as chao.py. Pick UTF-8 encoding if the editor offers a choice, so accented Vietnamese text stays intact.
3. Gõ đúng các dòng sau, mỗi dòng bắt đầu sát lề trái, không thụt vào:
   Type these lines exactly, each starting at the left edge with no indentation:

```
ten = input("Ban ten la gi? ")
nam_sinh = int(input("Ban sinh nam nao? "))
nam_nay = 2026
tuoi = nam_nay - nam_sinh
print("Xin chao,", ten)
print("Nam", nam_nay, "ban khoang", tuoi, "tuoi")
```

Giải thích từng dòng. input() in câu hỏi trong ngoặc, chờ bạn gõ rồi nhấn Enter, và trả về những gì bạn gõ dưới dạng chuỗi (chữ). Kể cả khi bạn gõ 1990, Python vẫn coi đó là chữ "1990", giống tờ giấy ghi số chứ chưa phải con số để cộng trừ được.

Line by line: input() prints the question in the brackets, waits for you to type and press Enter, then hands back what you typed as a string (text). Even if you type 1990, Python treats it as the text "1990", like a number written on paper that you cannot yet add or subtract.

int() đổi chuỗi chữ số thành số nguyên. Dòng thứ hai bọc input() trong int(), nên `nam_sinh` là số thật và trừ được. Dòng `nam_nay` gán năm hiện tại vào một biến; tuoi lấy năm nay trừ năm sinh.

int() converts a string of digits into a whole number. The second line wraps input() in int(), so `nam_sinh` is a real number you can subtract. The `nam_nay` line stores the current year in a variable, and tuoi is that year minus the birth year.

Biến là một chiếc hộp dán nhãn để cất giá trị, như hộp ghi "muối" trong bếp. Dấu = nghĩa là "cất vào hộp", không phải so sánh bằng. Tên hộp phải viết đúng từng chữ mỗi lần dùng, vì ten và Ten là hai hộp khác nhau.

A variable is a labelled box for keeping a value, like a jar marked "salt" in the kitchen. The = sign means "put into the box", not "is equal to". The label must be spelled the same way every time, because ten and Ten are two different boxes.

Khi print() nhận nhiều phần ngăn bằng dấu phẩy, nó tự chèn một dấu cách giữa các phần. Vì vậy print("Xin chao,", ten) ra Xin chao, Lan mà bạn không phải gõ khoảng trắng. Chữ "khoảng" trong kết quả có lý do: nếu sinh nhật năm nay chưa tới, tuổi thật nhỏ hơn con số tính ra một tuổi.

When print() receives several items separated by commas, it puts one space between them by itself. So print("Xin chao,", ten) gives Xin chao, Lan without you typing a space. The word "khoang" (about) in the output is there for a reason: if your birthday has not come yet this year, your real age is one less than the number calculated.

Chạy chương trình – *Run the program*

1. Mở cửa sổ dòng lệnh (Command Prompt hoặc PowerShell trên Windows, Terminal trên macOS và Linux).
   Open a command window (Command Prompt or PowerShell on Windows, Terminal on macOS and Linux).
2. Chuyển vào thư mục chứa tệp bằng lệnh cd kèm đường dẫn, ví dụ trên macOS:
   Move into the folder that holds the file with cd and its path, for example on macOS:

```
cd ~/Documents/baitap
```

1. Nếu chưa chắc Python có sẵn, gõ python3 --version (Windows: py --version). Thấy dòng bắt đầu bằng Python và một số phiên bản là dùng được.
   If you are not sure Python is available, type python3 --version (Windows: py --version). A line starting with Python and a version number means it works.
2. Chạy tệp. Trên Windows gõ py chao.py (hoặc python chao.py); trên macOS và Linux thường gõ python3 chao.py.
   Run the file. On Windows type py chao.py (or python chao.py); on macOS and Linux usually type python3 chao.py.
3. Khi chương trình hỏi, gõ tên rồi Enter, gõ năm sinh rồi Enter.
   When the program asks, type your name and press Enter, then type your birth year and press Enter.

> **VÍ DỤ / EXAMPLE –** Gõ tên Lan và năm sinh 1990. Python tính 2026 - 1990 = 36. Màn hình hiện: Ban ten la gi? Lan, Ban sinh nam nao? 1990, Xin chao, Lan, Nam 2026 ban khoang 36 tuoi. Thử lại với 2004: 2026 - 2004 = 22, kết quả là 22 tuổi.
> *Type the name Lan and birth year 1990. Python works out 2026 - 1990 = 36. The screen shows: Ban ten la gi? Lan, Ban sinh nam nao? 1990, Xin chao, Lan, Nam 2026 ban khoang 36 tuoi. Try 2004: 2026 - 2004 = 22, so the result is 22.*
>
> **VÍ DỤ / EXAMPLE –** Thử một năm sinh vô lý để thấy chương trình không tự kiểm tra. Gõ tên Minh và năm sinh 2030: 2026 - 2030 = -4, nên màn hình in Nam 2026 ban khoang -4 tuoi.

Python chỉ làm phép tính bạn viết, nó không biết tuổi âm là sai.
Try an impossible birth year to see that the program does no checking. Enter the name Minh and birth year 2030: 2026 - 2030 = -4, so the screen prints Nam 2026 ban khoang -4 tuoi. Python only does the arithmetic you wrote; it does not know a negative age makes no sense.

Lỗi hay gặp – *Common mistakes*

LỖI HAY GẶP / COMMON MISTAKE – Quên đổi kiểu. Nếu viết `nam_sinh` = input(...) mà không có int(), dòng trừ báo TypeError: unsupported operand type(s) for -: 'int' and 'str'. Nghĩa là bạn đang lấy số trừ chữ. Bọc input() bằng int() là hết lỗi.
Forgetting the conversion. If you write `nam_sinh` = input(...) without int(), the subtraction line raises TypeError: unsupported operand type(s) for -: 'int' and 'str'. You are subtracting text from a number. Wrapping input() in int() fixes it.

LỖI HAY GẶP / COMMON MISTAKE – Sai thư mục. Nếu dòng lệnh báo can't open file ... [Errno 2] No such file or directory, Python không thấy chao.py ở thư mục hiện tại. Kiểm tra bạn đã cd đúng chưa, tên tệp có đúng chao.py không, và tệp có bị lưu thành chao.py.txt không.
Wrong folder. If the command window says can't open file ... [Errno 2] No such file or directory, Python cannot find chao.py in the current folder. Check that you ran cd correctly, that the name is exactly chao.py, and that the file was not saved as chao.py.txt.

LỖI HAY GẶP / COMMON MISTAKE – Gõ chữ thay vì số ở câu hỏi năm sinh, ví dụ abc, sẽ gây ValueError: invalid literal for int() with base 10: 'abc'. Chạy lại và gõ số nguyên.
Typing letters instead of digits for the birth year, such as abc, causes ValueError: invalid literal for int() with base 10: 'abc'. Run it again and type a whole number.

LỖI HAY GẶP / COMMON MISTAKE – Viết sai tên biến, ví dụ print("Xin chao,", Ten) khi biến đặt là ten. Python báo NameError: name 'Ten' is not defined. Sửa lại cho khớp từng chữ, kể cả chữ hoa chữ thường.
Misspelling a variable, such as print("Xin chao,", Ten) when the variable is ten. Python reports NameError: name 'Ten' is not defined. Fix it so it matches letter for letter, including capitals.

LỖI HAY GẶP / COMMON MISTAKE – Thiếu một dấu ngoặc kép đóng, như print("Xin chao,, ten), gây SyntaxError: unterminated string literal. Đọc dòng được báo và đếm xem mỗi dấu mở đã có dấu đóng chưa.
Leaving out a closing quote, as in print("Xin chao,, ten), causes SyntaxError: unterminated string literal. Read the line it points to and check that every opening quote has a closing one.

MẸO / TIP – Thụt lề thừa ở đầu dòng gây IndentationError. Dòng nào cũng phải bắt đầu sát lề trái ở bài này.
A stray indent at the start of a line causes IndentationError. In this exercise every line must start at the left edge.

Tự luyện thêm – *Try it yourself*

- Thêm một dòng in tuổi của bạn vào năm 2030. Gợi ý: `nam_sinh` 1990 thì 2030 - 1990 = 40. Đáp án: print("Nam 2030 ban se", 2030 - `nam_sinh`, "tuoi").
  Add a line that prints your age in 2030. Hint: for birth year 1990, 2030 - 1990 = 40. Answer: print("Nam 2030 ban se", 2030 - `nam_sinh`, "tuoi").

- Đổi `nam_nay` thành năm bạn muốn rồi chạy lại, và kiểm tra kết quả bằng cách trừ tay.
  Change `nam_nay` to any year you like, run it again, and check the output by subtracting by hand.
- Thêm một dòng in năm bạn tròn 50 tuổi. Gợi ý: sinh năm 1990 thì 1990 + 50 = 2040. Đáp án: print("Ban tron 50 tuoi nam", `nam_sinh` + 50).
  Add a line that prints the year you turn 50. Hint: born in 1990 gives 1990 + 50 = 2040. Answer: print("Ban tron 50 tuoi nam", `nam_sinh` + 50).

1.8. Bài thực hành có hướng dẫn – *1.8. Guided practice exercises*

Bài 1. Tính tiền trà sữa trong chế độ tương tác – *Exercise 1. Calculating a milk-tea bill in interactive mode*

Một cốc trà sữa giá 25000 đồng, bạn mua 4 cốc và có phiếu giảm 10000 đồng. Hãy mở chế độ tương tác của Python, tính số tiền phải trả rồi thoát ra đúng cách.

A cup of milk tea costs 25000 dong, you buy 4 cups and have a 10000 dong voucher. Open Python's interactive mode, work out the amount to pay, then exit properly.

1. Mở Terminal (macOS, Linux) hoặc Command Prompt (Windows), gõ python rồi nhấn Enter. Trên Windows, nếu máy không nhận lệnh này thì gõ py. Khi thấy dấu nhắc >>> là Python đã sẵn sàng.
   Open Terminal (macOS, Linux) or Command Prompt (Windows), type python and press Enter. On Windows, if the command is not recognised, type py. When you see the >>> prompt, Python is ready.
2. Tại dấu >>> gõ 25000 * 4 rồi Enter. Python in ra 100000, là tiền 4 cốc chưa giảm.
   *At the >>> prompt type 25000 * 4 and press Enter. Python prints 100000, the price of 4 cups before the discount.*
3. Gõ tiếp print(25000 * 4-10000) rồi Enter. Python trừ phiếu giảm 10000 khỏi 100000 và in kết quả.
   *Now type print(25000 * 4-10000) and press Enter. Python subtracts the 10000 voucher from 100000 and prints the result.*
4. Gõ exit() rồi Enter để quay lại dấu nhắc của Terminal. Trên macOS và Linux bạn cũng có thể nhấn Ctrl-D.
   Type exit() and press Enter to return to the Terminal prompt. On macOS and Linux you can also press Ctrl-D.
5. Kết quả: bạn trả 90000 đồng.
   Result: you pay 90000 dong.

Bài 2. Viết tệp tuoi.py hỏi tên và tính tuổi – *Exercise 2. Writing tuoi.py to ask a name and work out an age*

Tạo tệp tuoi.py hỏi tên và năm sinh, lấy năm nay là 2025 để tính tuổi, rồi chạy từ dòng lệnh với tên Lan, năm sinh 2001.

Create tuoi.py that asks for a name and a birth year, uses 2025 as the current year to work out the age, then run it from the command line with the name Lan and birth year 2001.

1. Mở trình soạn thảo văn bản bất kỳ, tạo tệp mới và lưu với tên tuoi.py trong một thư mục dễ nhớ, ví dụ thư mục baitap.
   Open any text editor, create a new file and save it as tuoi.py in a folder that is easy to remember, for example a folder called baitap.
2. Gõ dòng thứ nhất: ten = input("Tên của bạn: "). Hàm input chờ người dùng gõ chữ rồi lưu vào biến ten.
   Type the first line: ten = input("Tên của bạn: "). The input function waits for the user to type text and stores it in the variable ten.
3. Gõ dòng thứ hai: `nam_sinh` = int(input("Năm sinh: ")). Hàm int đổi chữ thành số nguyên để trừ được.
   Type the second line: `nam_sinh` = int(input("Năm sinh: ")). The int function turns the text into a whole number so it can be subtracted.
4. Gõ dòng thứ ba: `nam_nay` = 2025. Gõ dòng thứ tư: print("Chào", ten + ",", "năm", `nam_nay`, "bạn", `nam_nay` - `nam_sinh`, "tuổi."). Lưu tệp.
   Type the third line: `nam_nay` = 2025. Type the fourth line: print("Chào", ten + ",", "năm", `nam_nay`, "bạn", `nam_nay` - `nam_sinh`, "tuổi."). Save the file.
5. Trong Terminal dùng lệnh cd để vào thư mục baitap, gõ python tuoi.py rồi Enter. Nhập Lan, Enter, rồi nhập 2001, Enter.
   In the Terminal use cd to enter the baitap folder, type python tuoi.py and press Enter. Enter Lan, press Enter, then enter 2001 and press Enter.
6. Kết quả: màn hình in Chào Lan, năm 2025 bạn 24 tuổi. vì 2025 - 2001 = 24.
   Result: the screen shows Chào Lan, năm 2025 bạn 24 tuổi. because 2025 - 2001 = 24.

Bài 3. Dùng tuỳ chọn dòng lệnh và help() – ***Exercise 3. Using command-line options and help()***

Không mở tệp nào, hãy xem số phiên bản Python đang cài, tính 12 * 8 chỉ bằng một lệnh, rồi tra cách dùng hàm abs bằng help.

*Without opening any file, check which Python version is installed, work out 12 * 8 with a single command, then look up how the abs function works with help.*

1. Trong Terminal gõ python -V rồi Enter. Python in một dòng bắt đầu bằng Python 3. theo sau là số phiên bản đầy đủ và thoát ngay.
   In the Terminal type python -V and press Enter. Python prints one line starting with Python 3. followed by the full version number, then exits at once.
2. Gõ python -c "print(12 * 8)" rồi Enter. Tuỳ chọn -c bảo Python chạy đoạn lệnh nằm trong dấu ngoặc kép.
   *Type python -c "print(12 * 8)" and press Enter. The -c option tells Python to run the code inside the quotation marks.*
3. Gõ python rồi Enter để vào chế độ tương tác, sau đó gõ help(abs) và Enter. Python hiện phần mô tả hàm abs, trong đó có câu Return the absolute value of the argument.
   Type python and press Enter to enter interactive mode, then type help(abs) and press Enter. Python shows the description of abs, including the sentence Return the absolute value of the argument.

4. Gõ exit() rồi Enter để thoát.
   Type exit() and press Enter to leave.

5. Kết quả: lệnh thứ hai in 96 vì 12 × 8 = 96, và help(abs) cho biết abs trả về giá trị tuyệt đối.
   Result: the second command prints 96 because 12 × 8 = 96, and help(abs) shows that abs returns the absolute value.

1.9. Câu hỏi trắc nghiệm – *1.9. Multiple-choice questions*

1. Trên macOS hoặc Linux, tổ hợp phím nào thoát khỏi chế độ tương tác của Python khi dấu nhắc >>> đang chờ lệnh?
   On macOS or Linux, which key combination leaves Python's interactive mode when the >>> prompt is waiting for a command?

   A. Ctrl-C
   B. Ctrl-D
   C. Ctrl-P
   D. Ctrl-S

2. Bạn chỉ muốn xem số phiên bản Python rồi thoát ngay, không chạy gì thêm. Lệnh nào đúng?
   You only want to see the Python version number and exit at once, running nothing else. Which command is right?

   A. python -i
   B. python -c
   C. python -V
   D. python -B

3. Tuỳ chọn -B (hoặc biến môi trường PYTHONDONTWRITEBYTECODE) làm gì?
   What does the -B option (or the PYTHONDONTWRITEBYTECODE environment variable) do?

   A. Ngăn Python ghi tệp .pyc khi nạp module từ mã nguồn / *Stops Python from writing .pyc files when importing source modules*
   B. Mở chế độ tương tác sau khi chạy tệp / *Opens interactive mode after running a file*
   C. Báo cảnh báo khi so sánh bytes với str / *Warns when comparing bytes with str*
   D. Xoá mọi tệp .py trong thư mục hiện tại / *Deletes every .py file in the current folder*

4. Bạn muốn biến tệp script thành tệp chạy trực tiếp trên hệ Unix bằng dòng bắt đầu bằng #!. Dòng này phải nằm ở đâu?
   You want to make a script directly executable on Unix using a line that starts with #!. Where must this line be?

   A. Ở hai ký tự đầu tiên của tệp / *As the first two characters of the file*
   B. Ngay sau dòng import đầu tiên / *Right after the first import line*
   C. Ở bất kỳ dòng nào vì # là chú thích / *On any line because # starts a comment*
   D. Ở dòng cuối cùng của tệp / *On the last line of the file*

5. Điểm nào sau đây là tính năng mới của Python 3.14 (PEP 750)?
Which of these is a new feature of Python 3.14 (PEP 750)?

A. Chỉ chạy được trên Windows / *Running only on Windows*
B. Bỏ hẳn lệnh print / *Removal of the print function*
C. Bỏ chế độ tương tác / *Removal of interactive mode*
D. Chuỗi mẫu (template strings) / *Template strings*

Chương 2. Python như chiếc máy tính: số, chữ và danh sách – *Chapter 2. Python as a Calculator: Numbers, Text and Lists*

Mục tiêu chương – *Chapter objectives*

- Tính được các phép cộng, trừ, nhân, chia, chia lấy phần nguyên, chia lấy dư và lũy thừa bằng Python, rồi lưu kết quả vào biến
  Calculate with addition, subtraction, multiplication, division, floor division, remainder and powers in Python, and store results in variables
- Viết được chuỗi có dấu nháy, ký tự thoát hoặc chuỗi thô, và dùng chỉ số, cắt lát, nối, lặp để lấy đúng phần chuỗi cần
  Write strings with quotes, escape sequences or raw strings, and use indexing, slicing, concatenation and repetition to get the exact part you need
- Chạy được chương trình nhiều bước dùng danh sách và vòng lặp while, như dãy Fibonacci và bảng tính tiền điện bậc thang
  Run multi-step programs that use lists and a while loop, such as the Fibonacci sequence and a tiered electricity bill

2.1. Số nguyên, số thập phân và các phép tính – *2.1. Integers, Decimals and Arithmetic*

Python làm được mọi việc của chiếc máy tính bỏ túi. Bạn gõ phép tính sau dấu nhắc >>>, bấm Enter, kết quả hiện ngay ở dòng dưới. Mọi ví dụ trong mục này đều gõ thử được trong chế độ tương tác.

Python can do everything a pocket calculator does. You type a calculation after the >>> prompt, press Enter, and the result appears on the next line. You can try every example in this section in interactive mode.

Số nguyên và số thập phân – *Integers and decimals*

Số nguyên (kiểu int) là số không có phần lẻ, như 7, 0 hay -12. Nó hợp với thứ đếm được: số ghế, số lần lặp. Số thập phân (kiểu float) có dấu chấm, như 3.5 hay 0.25. Nó hợp với thứ đo được: cân nặng, giá có lẻ. Python dùng dấu chấm, không dùng dấu phẩy.

An integer (type int) has no fractional part, like 7, 0 or -12. It suits things you count: chairs, loop turns. A decimal (type float) has a dot, like 3.5 or 0.25. It suits things you measure: weight, prices with cents. Python uses a dot, never a comma.

Chỉ cần có dấu chấm là Python coi đó là số thập phân. 2 là số nguyên, còn 2.0 là số thập phân. Muốn kiểm tra, gõ lệnh sau và Python trả về <class 'float'>:

A dot is enough to make Python treat a number as a decimal. 2 is an integer, while 2.0 is a decimal. To check, type this command and Python answers <class 'float'>:

```
type(2.0)
```

Trộn hai loại trong một phép tính thì kết quả là số thập phân. Ví dụ 3 + 0.5 ra 3.5, và 2 ∗ 1.5 ra 3.0.

Mixing the two kinds in one calculation gives a decimal. For example, 3 + 0.5 gives 3.5, and 2 ∗ 1.5 gives 3.0.

Bảy phép tính – *The seven operations*

Bảng sau tóm tắt các phép tính, lấy 7 và 2 làm ví dụ. Hai phép lạ nhất là // và %, giải thích ngay sau bảng.

This table sums up the operations, using 7 and 2 as examples. The two unusual ones are // and %, explained right after the table.

Phép tính *Operation*	**Ký hiệu** *Symbol*	**Ví dụ** *Example*	**Kết quả** *Result*
Cộng *Addition*	+	7 + 2	9
Trừ *Subtraction*	-	7 - 2	5
Nhân *Multiplication*	*	7 * 2	14
Chia *Division*	/	7 / 2	3.5
Chia lấy phần nguyên *Floor division*	//	7 // 2	3
Lấy dư *Remainder*	%	7 % 2	1
Luỹ thừa *Power*	**	7 ** 2	49

Hãy nghĩ tới chuyện chia kẹo. Có 17 cái kẹo chia cho 5 bạn: mỗi bạn được 3 cái, còn dư 2 cái. Python viết 17 // 5 ra 3 và 17 % 5 ra 2. Kiểm tra lại: 5 ∗ 3 + 2 = 17.

Think of sharing sweets. You split 17 sweets among 5 friends: each gets 3 and 2 are left over. In Python, 17 // 5 gives 3 and 17 % 5 gives 2. Check: 5 ∗ 3 + 2 = 17.

Phép chia / luôn ra số thập phân, kể cả khi chia hết: 8 / 2 ra 4.0, không phải 4. Lý do: phần lớn phép chia không tròn, như 7 / 2. Python chọn một kiểu kết quả cố định để bạn không phải đoán nó là nguyên hay thập phân. Muốn kết quả nguyên, dùng //.

The / operator always gives a decimal, even when the division is exact: 8 / 2 gives 4.0, not 4. Most divisions do not come out even, like 7 / 2. So Python picks one fixed result type, and you never have to guess whether you get an integer or a decimal. For an integer result, use //.

VÍ DỤ / EXAMPLE – Đổi 3725 giây ra giờ, phút, giây. Bước 1: 3725 // 60 ra 62 và 3725 % 60 ra 5, tức 62 phút lẻ 5 giây. Bước 2: lấy 62 chia tiếp cho 60: 62 // 60 ra 1 và 62 % 60 ra 2. Kết quả: 1 giờ 2 phút 5 giây. Kiểm tra: 1 ∗ 3600 + 2 ∗ 60 + 5 = 3725. Cặp // và % hay dùng cho việc đổi đơn vị kiểu này.

Convert 3725 seconds into hours, minutes and seconds. Step 1: 3725 // 60 gives 62 and 3725 % 60

gives 5, so 62 minutes and 5 seconds. Step 2: divide 62 by 60 again: 62 // 60 gives 1 and 62 % 60 gives 2. Result: 1 hour 2 minutes 5 seconds. Check: 1 ∗ 3600 + 2 ∗ 60 + 5 = 3725. The pair // and % is the usual tool for unit conversions like this.

Thứ tự ưu tiên – *Order of operations*

Python tính giống môn toán ở trường: ngoặc trước, rồi luỹ thừa **, rồi nhân chia ∗ / // %, cuối cùng cộng trừ + -. Cùng mức thì tính từ trái sang phải.

*Python follows the order you learned in school maths: brackets first, then powers **, then ∗ / // %, and finally + -. Operations on the same level go from left to right.*

Thử với 20 - 6 / 2 ∗ 3. Làm theo các bước sau:

Try 20 - 6 / 2 ∗ 3. Work through it step by step:

1. Tìm phép ở mức cao nhất. Ở đây / và ∗ cùng mức, cao hơn phép trừ.
   Find the highest-priority operations. Here / and ∗ share a level, above subtraction.
2. Tính từ trái sang: 6 / 2 ra 3.0.
   Go from left to right: 6 / 2 gives 3.0.
3. Tiếp theo 3.0 ∗ 3 ra 9.0.
   Next, 3.0 ∗ 3 gives 9.0.
4. Cuối cùng 20 - 9.0 ra 11.0. Muốn trừ trước, thêm ngoặc: (20 - 6) / 2 ∗ 3 ra 21.0.
   Finally, 20 - 9.0 gives 11.0. To subtract first, add brackets: (20 - 6) / 2 ∗ 3 gives 21.0.

VÍ DỤ / EXAMPLE – Tính tiền hoá đơn có thuế 8%. Bạn mua 2 cái áo giá 150000 đồng mỗi cái và một đôi tất giá 85000 đồng. Tiền hàng: 2 ∗ 150000 + 85000 ra 385000. Tiền thuế: 385000 ∗ 8 / 100 ra 30800.0. Tổng phải trả: 385000 + 30800.0 ra 415800.0, tức 415800 đồng. Cách khác chỉ dùng số nguyên: 385000 ∗ 108 // 100 cũng ra 415800.

Working out a bill with 8% tax. You buy 2 shirts at 150000 dong each and a pair of socks at 85000 dong. Goods total: 2 ∗ 150000 + 85000 gives 385000. Tax: 385000 ∗ 8 / 100 gives 30800.0. Amount to pay: 385000 + 30800.0 gives 415800.0, that is 415800 dong. An integer-only way: 385000 ∗ 108 // 100 also gives 415800.

Đổi kiểu số – *Converting between types*

Hàm int() đổi một số thành số nguyên bằng cách bỏ phần lẻ, không làm tròn. Hàm float() đổi số nguyên thành số thập phân. Hàm abs() lấy giá trị tuyệt đối, tức bỏ dấu âm.

The int() function turns a number into an integer by dropping the fractional part, with no rounding. The float() function turns an integer into a decimal. The abs() function gives the absolute value, meaning it drops the minus sign.

Thử bốn lệnh: int(3.9) ra 3, int(-3.9) ra -3, float(5) ra 5.0, abs(-12) ra 12. Chuỗi chữ viết đúng dạng số cũng đổi được: int("42") ra 42.

Try four commands: int(3.9) gives 3, int(-3.9) gives -3, float(5) gives 5.0, abs(-12) gives 12. A piece of text written like a number can be converted too: int("42") gives 42.

VÍ DỤ / EXAMPLE – Đổi 30 độ C sang độ F theo công thức C ∗ 9 / 5 + 32. Gõ 30 ∗ 9 / 5 + 32. Bước 1: 30 ∗ 9 ra 270. Bước 2: 270 / 5 ra 54.0. Bước 3: 54.0 + 32 ra 86.0. Vậy 30 độ C

bằng 86 độ F. Nếu cần số nguyên, bọc cả phép tính: int(30 * 9 / 5 + 32) ra 86.
*Convert 30 degrees Celsius to Fahrenheit with the formula C * 9 / 5 + 32. Type 30 * 9 / 5 + 32. Step 1: 30 * 9 gives 270. Step 2: 270 / 5 gives 54.0. Step 3: 54.0 + 32 gives 86.0. So 30 degrees Celsius is 86 degrees Fahrenheit. If you need an integer, wrap the whole calculation: int(30 * 9 / 5 + 32) gives 86.*

Chỗ hay sai – *Common mistakes*

LỖI HAY GẶP / COMMON MISTAKE – -3 ** 2 ra -9, không phải 9, vì ** được tính trước dấu trừ. Muốn bình phương số âm, viết (-3) ** 2 ra 9.
*-3 ** 2 gives -9, not 9, because ** is applied before the minus sign. To square a negative number, write (-3) ** 2, which gives 9.*

LỖI HAY GẶP / COMMON MISTAKE – Với số âm, // làm tròn xuống chứ không cắt phần lẻ: -7 // 2 ra -4, và -7 % 2 ra 1. Kiểm tra: -4 * 2 + 1 = -7. Hàm int() thì cắt phần lẻ, nên int(-3.5) ra -3 còn -7 // 2 ra -4.
*With negative numbers, // rounds down rather than cutting off the fraction: -7 // 2 gives -4, and -7 % 2 gives 1. Check: -4 * 2 + 1 = -7. The int() function does cut off the fraction, so int(-3.5) gives -3 while -7 // 2 gives -4.*

LỖI HAY GẶP / COMMON MISTAKE – Số thập phân được máy lưu gần đúng, nên 0.1 + 0.2 ra 0.30000000000000004. Đây không phải lỗi của bạn. Với tiền, hãy tính bằng số nguyên (đơn vị đồng) khi có thể.
The computer stores decimals approximately, so 0.1 + 0.2 gives 0.30000000000000004. It is not your mistake. With money, calculate in whole units (dong) when you can.

LỖI HAY GẶP / COMMON MISTAKE – Chia cho 0, như 5 / 0, làm Python báo ZeroDivisionError. Gõ 3,5 thay cho 3.5 cũng không cho ra số 3.5. Luôn dùng dấu chấm.
Dividing by 0, such as 5 / 0, makes Python report ZeroDivisionError. Typing 3,5 instead of 3.5 does not give you the number 3.5 either. Always use a dot.

LỖI HAY GẶP / COMMON MISTAKE – Nhiều người quen viết 2 ^ 3 để tính luỹ thừa. Python không hiểu như vậy: 2 ^ 3 ra 1, không phải 8, vì ^ là một phép khác. Luỹ thừa phải viết 2 ** 3 ra 8. Đừng gõ cả dấu >>> vào lệnh, vì đó là dấu nhắc Python tự hiện.
*Many people are used to writing 2 ^ 3 for powers. Python reads it differently: 2 ^ 3 gives 1, not 8, because ^ is a different operation. Write powers as 2 ** 3, which gives 8. Do not type the >>> prompt as part of a command either, since Python shows it by itself.*

2.2. Biến và phép gán – *2.2. Variables and Assignment*

Biến giống tờ nhãn bạn dán lên một chiếc hộp. Trong hộp là một giá trị, ví dụ số 12. Tờ nhãn ghi tên, ví dụ dai. Khi cần giá trị, bạn gọi tên trên nhãn thay vì gõ lại số.
A variable works like a label stuck on a box. The box holds a value, say the number 12. The label carries a name, say dai. When you need the value, you call the name instead of retyping the number.

Việc dán nhãn gọi là phép gán (assignment): bạn gán một giá trị cho một tên. Dấu bằng trong Python không có nghĩa "bằng nhau" như ở môn toán. Nó có nghĩa "lấy giá trị bên phải, dán tên bên trái lên". Python tính vế phải trước, rồi mới dán nhãn.

Sticking on the label is called assignment: you assign a value to a name. In Python the equals sign does not mean "is equal to" as in maths. It means "take the value on the right and put the name on the left onto it". Python works out the right side first, then attaches the label.

Làm thử: tính diện tích hình chữ nhật – *Try it: the area of a rectangle*

Bạn cần Python đang ở chế độ tương tác, nơi màn hình hiện dấu nhắc >>>. Mảnh đất của ta dài 12 mét, rộng 5 mét.

You need Python in interactive mode, where the screen shows the >>> prompt. Our plot of land is 12 metres long and 5 metres wide.

1. Gõ dai = 12 rồi nhấn Enter. Python không in gì cả, nghĩa là nhãn dai đã được dán vào số 12.
   Type dai = 12 and press Enter. Python prints nothing, which means the label dai is now attached to 12.
2. Gõ rong = 5 rồi nhấn Enter. Lúc này có hai nhãn: dai và rong.
   Type rong = 5 and press Enter. Now there are two labels: dai and rong.
3. Gõ `dien_tich` = dai ∗ rong rồi nhấn Enter. Dấu ∗ là phép nhân. Python lấy 12 nhân 5 được 60, rồi dán nhãn `dien_tich` lên số 60.
   Type `dien_tich` = dai ∗ rong and press Enter. The ∗ sign means multiplication. Python multiplies 12 by 5 to get 60, then attaches the label `dien_tich` to 60.
4. Gõ `dien_tich` rồi nhấn Enter, hoặc gõ print(`dien_tich`). Màn hình hiện 60.
   Type `dien_tich` and press Enter, or type print(`dien_tich`). The screen shows 60.

VÍ DỤ / EXAMPLE – Cả phiên làm việc trông như sau:
The whole session looks like this:

```
>>> dai = 12
>>> rong = 5
>>> dien_tich = dai * rong
>>> print(dien_tich)
60
```

Bạn có thể dán lại nhãn bất cứ lúc nào. Gõ rong = 8 thì nhãn rong chuyển sang số 8. Nhưng `dien_tich` vẫn là 60, vì nó giữ kết quả của phép tính cũ, không phải công thức. Muốn có số mới, bạn gõ lại `dien_tich` = dai ∗ rong, được 96.

You can move a label at any time. Typing rong = 8 moves the label rong to 8. But `dien_tich` is still 60, because it keeps the result of the old calculation, not the formula. For a new figure, type `dien_tich` = dai ∗ rong again and you get 96.

Ví dụ thứ hai: tính tiền và cập nhật số lượng – *A second example: a bill and an updated quantity*

Bạn mua bút giá 25000 đồng một cây, lấy 4 cây. Sau đó đổi ý, lấy thêm 1 cây. Ví dụ này cho thấy một nhãn có thể tự dùng giá trị cũ của chính nó để tạo giá trị mới.

You buy pens at 25000 dong each and take 4. Then you change your mind and add one more. This example shows a label using its own old value to build a new one.

```
>>> gia = 25000
>>> so_luong = 4
>>> tong = gia * so_luong
>>> print(tong)
100000
>>> so_luong = so_luong + 1
>>> tong = gia * so_luong
>>> print(tong)
125000
```

Ở dòng `so_luong = so_luong + 1`, Python tính vế phải trước: 4 cộng 1 được 5. Sau đó nó chuyển nhãn `so_luong` sang số 5. Phải gõ lại dòng tính tong thì mới ra 25000 nhân 5 bằng 125000.

On the line `so_luong = so_luong + 1`, Python first works out the right side: 4 plus 1 is 5. Then it moves the label `so_luong` to 5. You must retype the line for tong to get 25000 times 5, which is 125000.

Cách đặt tên – *Naming rules*

Tên biến chỉ gồm chữ cái, chữ số và dấu gạch dưới _. Ký tự đầu không được là chữ số. Tên không chứa dấu cách hay dấu gạch ngang. Khi tên có nhiều từ, người ta thường nối bằng gạch dưới, như `dien_tich`.

A variable name may contain only letters, digits and the underscore _. The first character cannot be a digit. Spaces and hyphens are not allowed. When a name has several words, people usually join them with underscores, as in `dien_tich`.

Tên *Name*	**Dùng được không** *Valid?*	**Lý do** *Reason*
`so_luong`	Được, chữ và gạch dưới đều hợp lệ. *Valid: letters and an underscore are fine.*	Cách đặt tên phổ biến. *The usual style.*
2lan	Không được, vì bắt đầu bằng chữ số. *Invalid: it starts with a digit.*	Đổi thành lan2. *Rename it lan2.*
tong-tien	Không được, vì có dấu gạch ngang. *Invalid: it contains a hyphen.*	Python hiểu đó là phép trừ. *Python reads it as a subtraction.*
`_tam`	Được, gạch dưới đầu tên vẫn hợp lệ. *Valid: a leading underscore is allowed.*	Thường dùng cho giá trị tạm. *Often used for temporary values.*

Python phân biệt chữ hoa và chữ thường. Dai, dai và DAI là ba nhãn khác nhau, mỗi nhãn dán lên một hộp riêng. Bạn nên giữ một kiểu viết thống nhất, ưu tiên chữ thường.

Python tells uppercase from lowercase. Dai, dai and DAI are three different labels, each on its own box. Pick one style and keep it, preferably lowercase.

Một số từ đã được Python giữ riêng cho ngôn ngữ, gọi là từ khoá (keyword). Ví dụ: if, for, while, class, def, return, import, True, None. Chúng có nhiệm vụ cố định trong cú pháp, nên Python không cho dùng làm tên biến. Gõ class = 3 sẽ báo SyntaxError, nghĩa là sai cú pháp.

Python reserves certain words for the language itself, called keywords. Examples: if, for, while, class, def, return, import, True, None. They have fixed jobs in the syntax, so Python will not let you use them as variable names. Typing class = 3 gives a SyntaxError, meaning invalid syntax.

Tên các hàm có sẵn như print không phải từ khoá, nên Python vẫn cho bạn gán đè. Nếu gõ print = 5, nhãn print bị chuyển sang số 5 và lệnh in không còn dùng được trong phiên đó. Hãy tránh đặt biến trùng tên hàm quen thuộc.

Names of built-in functions such as print are not keywords, so Python lets you overwrite them. If you type print = 5, the label print moves to 5 and the print command stops working in that session. Avoid giving variables the names of familiar functions.

> **MẸO / TIP –** Tên nên cho biết giá trị là gì. `dien_tich` dễ đọc hơn dt, và sau một tuần bạn vẫn nhớ nó nói về cái gì.
> *Choose names that say what the value is. `dien_tich` is easier to read than dt, and a week later you will still remember what it refers to.*

Lỗi NameError – *The NameError error*

Nếu bạn gọi một tên chưa từng được dán nhãn, Python không biết tìm hộp nào và báo NameError. Thử gõ:

If you call a name that was never attached to anything, Python cannot find a box and raises a NameError. Try typing:

```
>>> print(chieu_cao)
NameError: name 'chieu_cao' is not defined
```

Dòng cuối nói rõ tên nào không tìm thấy. Hãy đọc tên đó và hỏi hai câu: mình đã gán nó chưa, và mình có gõ sai không?

The last line names the missing name. Read it and ask two questions: did I assign it yet, and did I mistype it?

Bạn cũng có thể chủ động gỡ nhãn bằng lệnh del. Gõ del gia thì nhãn gia biến mất. Lúc đó print(gia) báo NameError giống như tên chưa từng gán.

You can also remove a label on purpose with the del command. Typing del gia makes the label gia disappear. After that, print(gia) raises a NameError, just like a name that was never assigned.

> **LỖI HAY GẶP / COMMON MISTAKE –** Gán dai = 12 nhưng sau đó gõ Dai hoặc dei sẽ gây NameError, vì đó là tên khác. Lỗi cũng xuất hiện khi bạn đóng Python rồi mở lại: các nhãn cũ đã mất, phải gán lại từ đầu.
> *Assigning dai = 12 and later typing Dai or dei causes a NameError, because those are different names. The error also appears when you close Python and reopen it: the old labels are gone and you must assign them again.*
>
> **LỖI HAY GẶP / COMMON MISTAKE –** Viết dai = rong rồi tưởng hai biến luôn bằng nhau. Thực ra Python chỉ dán thêm nhãn dai lên giá trị mà rong đang giữ lúc đó. Đổi rong sau này không làm dai đổi theo.
> *Writing dai = rong and assuming the two stay equal. Python only attaches the label dai to whatever value rong holds at that moment. Changing rong later does not change dai.*

LỖI HAY GẶP / COMMON MISTAKE – Đặt tên có dấu cách, như tong tien = 5. Python đọc thành hai tên đứng cạnh nhau và báo SyntaxError. Hãy viết `tong_tien`.
Putting a space in a name, as in tong tien = 5. Python reads two names side by side and raises a SyntaxError. Write `tong_tien` instead.

GHI CHÚ / NOTE – Nhớ ba điều: dấu = là gán chứ không phải so sánh; tên phân biệt hoa thường và không trùng từ khoá; dùng tên chưa gán thì gặp NameError.
Remember three things: = assigns and does not compare; names are case-sensitive and cannot be keywords; using an unassigned name gives a NameError.

2.3. Chuỗi ký tự: dấu nháy, ký tự thoát và chuỗi thô – *2.3. Strings: Quotes, Escape Sequences and Raw Strings*

Chuỗi ký tự (tiếng Anh: string) là đoạn chữ đặt giữa hai dấu nháy, như tên bạn, một địa chỉ hay lời chào. Python giữ nguyên những gì nằm trong nháy và không tính toán gì với nó, giống một tờ giấy ghi chú: ghi sao thì đọc vậy.
A string is a piece of text placed between quote marks, such as your name, an address or a greeting. Python keeps whatever sits inside the quotes as it is and does not calculate anything with it, much like a sticky note: what you write is what you read.

Nháy đơn và nháy kép – *Single and double quotes*

Bạn dùng nháy đơn hoặc nháy kép đều được, miễn mở bằng loại nào thì đóng bằng loại đó. Hai cách cho kết quả giống hệt nhau.
You can use single or double quotes, as long as you close with the same kind you opened with. Both give the same result.

```
print('Xin chào')
print("Xin chào")
```

Cả hai dòng đều in ra: Xin chào. Việc có hai loại nháy giúp bạn chèn nháy vào trong chữ. Muốn câu có dấu nháy kép thì bọc ngoài bằng nháy đơn, và ngược lại.
Both lines print: Xin chào. Having two kinds of quotes lets you put a quote mark inside the text. If the sentence contains double quotes, wrap it in single quotes, and the other way round.

VÍ DỤ / EXAMPLE – Gõ print('Cô nói: "Chào bạn"') thì màn hình in ra: Cô nói: "Chào bạn". Gõ print("Quán 'Hoa Sữa' mở cửa") thì in ra: Quán 'Hoa Sữa' mở cửa.
Typing print('Cô nói: "Chào bạn"') prints: Cô nói: "Chào bạn". Typing print("Quán 'Hoa Sữa' mở cửa") prints: Quán 'Hoa Sữa' mở cửa.

LỖI HAY GẶP / COMMON MISTAKE – Mở bằng nháy đơn rồi đóng bằng nháy kép, như print('Xin chào"), thì Python coi chuỗi chưa đóng và báo lỗi cú pháp. Nhìn lại đầu và cuối chuỗi xem hai nháy có cùng loại không.
Opening with a single quote and closing with a double quote, as in print('Xin chào"), leaves the string unclosed and Python reports a syntax error. Check that the first and last quotes are the same kind.

Nối chuỗi viết liền nhau – *Strings written side by side*

Hai chuỗi đứng cạnh nhau, chỉ cách bằng khoảng trắng, được Python dán thành một. Cách này tiện khi một câu quá dài: bạn bọc trong ngoặc tròn và ngắt thành nhiều dòng.

Two strings standing next to each other, separated only by spaces, are joined into one by Python. This helps when a sentence is too long: wrap it in parentheses and break it over several lines.

```
loi = ("Xin " "chào " "bạn")
print(loi)
```

In ra: Xin chào bạn. Python không tự thêm khoảng trắng, nên bạn phải để dấu cách bên trong nháy, như "Xin " ở trên.

It prints: Xin chào bạn. Python adds no spaces of its own, so keep the space inside the quotes, as in "Xin " above.

Ba nháy cho văn bản nhiều dòng – *Triple quotes for multi-line text*

Chuỗi trong một cặp nháy thường nằm gọn trên một dòng. Muốn giữ cả đoạn nhiều dòng, như lá thư hay bảng hướng dẫn, bạn đặt ba dấu nháy ở đầu và ba dấu nháy ở cuối. Dùng được ba nháy đơn hoặc ba nháy kép.

A string in a single pair of quotes normally sits on one line. To keep a whole multi-line passage, such as a letter or a help message, put three quotes at the start and three at the end. Either three single or three double quotes work.

1. Gõ thu = """Kính gửi anh Nam, rồi nhấn Enter để xuống dòng.
   Type thu = """Kính gửi anh Nam, then press Enter to start a new line.
2. Gõ tiếp dòng thứ hai: Hẹn gặp anh lúc 9 giờ., rồi nhấn Enter.
   Type the second line: Hẹn gặp anh lúc 9 giờ., then press Enter.
3. Gõ dòng cuối kèm ba nháy đóng: Trân trọng""" rồi nhấn Enter.
   Type the last line with the closing triple quotes: Trân trọng""" then press Enter.
4. Gõ print(thu).
   Type print(thu).

Kết quả in ra đúng ba dòng như bạn gõ:

The output has exactly three lines, as you typed them:

```
Kính gửi anh Nam,
Hẹn gặp anh lúc 9 giờ.
Trân trọng
```

Mỗi lần nhấn Enter bên trong ba nháy, Python ghi nhớ một dấu xuống dòng. Khoảng trắng đầu dòng cũng được giữ. Trong ba nháy bạn còn gõ tự do nháy đơn và nháy kép, chỉ cần không gõ liền ba nháy cùng loại với nháy mở.

Each time you press Enter inside triple quotes, Python remembers a line break. Spaces at the start of a line are kept too. Inside triple quotes you can type single and double quotes freely, as long as you do not type three in a row of the same kind that opened the string.

Ngay sau ba nháy mở, nếu bạn xuống dòng luôn thì chuỗi bắt đầu bằng một dòng trống. Muốn bỏ dòng trống ấy, đặt dấu \ cuối dòng mở. Dấu \ ở cuối dòng báo Python bỏ qua chỗ xuống dòng đó.

If you press Enter right after the opening triple quotes, the string starts with an empty line. To drop it, put a \ at the end of the opening line. A \ at the end of a line tells Python to skip that line break.

```
print("""\
Dòng một
Dòng hai""")
```

In ra hai dòng, bắt đầu ngay bằng chữ Dòng một, không có dòng trống phía trên.
It prints two lines, starting straight away with Dòng một, with no empty line above.

Ký tự thoát – *Escape sequences*

Ký tự thoát là cách viết một ký tự đặc biệt bằng dấu gạch chéo ngược \ đứng trước một chữ cái hay dấu. Python đọc cặp này như một ký tự duy nhất. Ví dụ \n là xuống dòng, giống việc bạn nhấn Enter khi soạn văn bản.
An escape sequence writes a special character using a backslash \ followed by a letter or symbol. Python reads the pair as one single character. For example, \n means a line break, like pressing Enter when typing a document.

Viết *Written*	**Ý nghĩa** *Meaning*	**Ví dụ và kết quả** *Example and result*
\n	Xuống dòng *Line break*	print('A\nB') in A rồi B ở dòng kế *print('A\nB') prints A, then B on the next line*
\t	Một dấu tab, tức khoảng trống rộng *One tab, a wide gap*	print('A\tB') in A, khoảng rộng, rồi B *print('A\tB') prints A, a wide gap, then B*
\\	Một dấu gạch chéo ngược *One backslash*	print('A\\B') in A\B *print('A\\B') prints A\B*
\' và \" *\' and \"*	Dấu nháy đơn, nháy kép *A single or double quote mark*	print('It\'s') in It's *print('It\'s') prints It's*

Dấu \' và \" cho phép đặt nháy vào chuỗi mà không cần đổi loại nháy bọc ngoài. Dòng print("Cô nói \"Chào\"") in ra: Cô nói "Chào".
The \' and \" sequences let you put a quote inside a string without changing the outer quotes. The line print("Cô nói \"Chào\"") prints: Cô nói "Chào".

Vì sao đường dẫn Windows hay lỗi – *Why Windows paths often break*

Windows ngăn các thư mục bằng dấu \, đúng ký hiệu Python dùng để thoát. Gõ đường dẫn thẳng vào, Python sẽ hiểu \n và \t là ký tự đặc biệt.
Windows separates folders with \, the very symbol Python uses for escapes. If you type a path directly, Python reads \n and \t as special characters.

```
print("C:\new\table.txt")
```

Kết quả không phải đường dẫn, mà là hai dòng, dòng hai có một tab ở giữa:
The result is not the path but two lines, and the second contains a tab:

```
C:
ew   able.txt
```

Có đường dẫn còn gây lỗi cú pháp (SyntaxError), như "C:\Users\An", vì Python đọc \U như phần mở đầu của một ký tự đặc biệt khác và không hiểu tiếp.

Some paths even cause a syntax error (SyntaxError), such as "C:\Users\An", because Python reads \U as the start of another special character and cannot make sense of it.

Chuỗi thô với tiền tố r – *Raw strings with the r prefix*

Chuỗi thô (raw string) là chuỗi có chữ r đặt sát ngay trước nháy mở. Python giữ nguyên mọi dấu \ bên trong, không đổi chúng thành ký tự đặc biệt.

A raw string has the letter r right before the opening quote. Python keeps every backslash inside as it is and does not turn it into a special character.

1. Gõ duongdan = r"C:\new\table.txt". Chữ r nằm sát nháy, không có khoảng trắng.
   Type duongdan = r"C:\new\table.txt". The r touches the quote, with no space.
2. Gõ print(duongdan).
   Type print(duongdan).
3. Đọc kết quả: C:\new\table.txt, đúng như bạn viết.
   Read the result: C:\new\table.txt, exactly as you wrote it.

Cách khác là gấp đôi mỗi dấu: "C:\\new\\table.txt" cũng in ra C:\new\table.txt.

The alternative is to double each backslash: "C:\\new\\table.txt" also prints C:\new\table.txt.

LỖI HAY GẶP / COMMON MISTAKE – Chuỗi thô không được kết thúc bằng một số lẻ dấu \. Dòng r"C:\data\" báo lỗi vì \" bị hiểu là nháy kép nằm trong chuỗi, nên chuỗi chưa đóng. Hãy bỏ dấu \ cuối, hoặc nối thêm: r"C:\data" + "\\".

A raw string cannot end with an odd number of backslashes. The line r"C:\data\" fails because \" is taken as a quote inside the string, so the string is never closed. Drop the final backslash, or join one on: r"C:\data" + "\\".

LỖI HAY GẶP / COMMON MISTAKE – Quên chữ r hoặc đặt cách nháy một khoảng trắng, như r "C:\new", thì chuỗi không còn là chuỗi thô và Python báo lỗi cú pháp.

Forgetting the r, or leaving a space between r and the quote, as in r "C:\new", means the string is no longer raw and Python reports a syntax error.

MẸO / TIP – Dán đường dẫn từ File Explorer vào trong r"..." là cách nhanh và ít sai nhất. Trong chuỗi thô, \n vẫn là hai ký tự \ và n, không xuống dòng.

Pasting a path from File Explorer into r"..." is the quickest and safest way. In a raw string, \n stays as the two characters \ and n, with no line break.

Chèn giá trị vào chuỗi với tiền tố f – *Putting values into a string with the f prefix*

Chuỗi f (f-string) là chuỗi có chữ f sát trước nháy mở. Trong chuỗi, phần nằm giữa cặp ngoặc nhọn { } được Python tính ra rồi dán vào chỗ đó. Giống một mẫu thư có chỗ trống để điền tên.

An f-string has the letter f right before the opening quote. Whatever sits between the curly braces { } is worked out by Python and pasted in at that spot. It is like a letter template with blanks for a name.

1. Gõ ten = "Lan", nhấn Enter, rồi gõ tuoi = 20.
   Type ten = "Lan", press Enter, then type tuoi = 20.
2. Gõ print(f"`{ten}` sang năm {tuoi + 1} tuổi").
   Type print(f"`{ten}` sang năm {tuoi + 1} tuổi").
3. Đọc kết quả: Lan sang năm 21 tuổi. Biểu thức tuoi + 1 được tính trước: 20 + 1 = 21.
   Read the result: Lan sang năm 21 tuổi. The expression tuoi + 1 is worked out first: 20 + 1 = 21.

Sau dấu hai chấm trong ngoặc nhọn, bạn có thể chọn cách hiển thị số. Với gia = 3.14159, dòng print(f"{gia:.2f}") in ra 3.14, tức làm tròn còn hai chữ số thập phân. Thêm dấu = sau tên, như f"{gia=}", thì in cả tên lẫn giá trị: gia=3.14159.

After a colon inside the braces you can choose how a number is shown. With gia = 3.14159, the line print(f"{gia:.2f}") prints 3.14, rounded to two decimal places. Adding = after the name, as in f"{gia=}", prints both name and value: gia=3.14159.

Muốn in chính dấu ngoặc nhọn thì gõ đôi. Dòng print(f"{{ten}} là `{ten}`") in ra: `{ten}` là Lan.

To print the curly braces themselves, type them twice. The line print(f"{{ten}} là `{ten}`") prints: `{ten}` là Lan.

> **LỖI HAY GẶP / COMMON MISTAKE –** Quên chữ f thì Python không điền gì cả. Dòng print("Chào `{ten}`") in nguyên văn Chào `{ten}`. Trong f-string, ngoặc nhọn cũng không được để trống: f"{}" báo lỗi.
> *Forgetting the f means Python fills in nothing. The line print("Chào `{ten}`") prints Chào `{ten}` word for word. Inside an f-string the braces also cannot be empty: f"{}" gives an error.*

Chuỗi không sửa tại chỗ được – *Strings cannot be edited in place*

Chuỗi là bất biến (immutable): đã tạo thì không đổi từng ký tự bên trong được, giống chữ đã in lên giấy. Muốn khác đi, bạn tạo một chuỗi mới.

Strings are immutable: once created, you cannot change single characters inside, like ink already printed on paper. To get something different, you make a new string.

> **VÍ DỤ / EXAMPLE –** Với s = "mèo", dòng s = s.upper() tạo chuỗi mới "MÈO" và gán lại cho s. Còn s[0] = "M" báo lỗi TypeError, vì không được gán vào từng vị trí của chuỗi.
> *With s = "mèo", the line s = s.upper() creates a new string "MÈO" and assigns it back to s. But s[0] = "M" gives a TypeError, because you cannot assign to a single position in a string.*

Có một họ chuỗi khác là bytes, viết với tiền tố b như b"abc". Nó chứa các byte thô và chỉ nhận ký tự ASCII (bộ ký tự cơ bản, không có chữ có dấu). Vì vậy b"chào" báo lỗi, còn "chào" thì bình thường.

A related type is bytes, written with a b prefix such as b"abc". It holds raw bytes and accepts only ASCII characters (the basic character set, with no accented letters). So b"chào" gives an error, while "chào" is fine.

2.4. Nối, lặp, đánh chỉ số và cắt lát chuỗi – *2.4. Joining, Repeating, Indexing and Slicing Strings*

Chuỗi giống một dãy ghế trong rạp: mỗi ký tự ngồi một ghế và mỗi ghế có số riêng. Python cho bạn dán chuỗi lại với nhau, lặp chuỗi nhiều lần, lấy đúng một ký tự hoặc cắt một đoạn theo số ghế.

A string is like a row of seats in a cinema: each character sits in one seat, and each seat has its own number. Python lets you glue strings together, repeat them, pick a single character, or cut out a stretch by seat number.

Nối và lặp chuỗi – *Joining and repeating strings*

Dấu + dán hai chuỗi thành một chuỗi mới. Dấu ∗ lặp một chuỗi theo số lần bạn cho. Gõ vào chế độ tương tác (dấu nhắc >>>) từng dòng sau đây:

The + sign glues two strings into a new one. The ∗ sign repeats a string as many times as you ask. Type these lines at the interactive prompt (>>>):

```
>>> "Xin " + "chào"
'Xin chào'
>>> "ha" * 3
'hahaha'
```

Dấu + chỉ nối chuỗi với chuỗi. Muốn nối với số, bạn đổi số thành chuỗi bằng str(). Ví dụ "Tuổi: " + str(20) cho ra 'Tuổi: 20'.

The + sign only joins a string to a string. To join a number, turn it into a string with str(). For example, "Tuổi: " + str(20) gives 'Tuổi: 20'.

Đánh chỉ số: đếm từ 0 – *Indexing: counting from 0*

Chỉ số là số thứ tự của một ký tự trong chuỗi. Python đếm từ 0, nên ký tự đầu tiên có chỉ số 0, không phải 1. Viết chỉ số trong ngoặc vuông sau tên chuỗi.

An index is the position number of one character in a string. Python counts from 0, so the first character has index 0, not 1. Write the index in square brackets after the string's name.

```
>>> word = "Python"
>>> word[0]
'P'
>>> word[5]
'n'
```

Chỉ số âm đếm từ cuối về: -1 là ký tự cuối, -2 là ký tự áp chót. Chỉ số âm bắt đầu từ -1 vì -0 cũng chỉ là 0.

Negative indices count from the end: -1 is the last character, -2 the one before it. Negative indices start at -1 because -0 is just 0.

```
>>> word[-1]
'n'
>>> word[-2]
'o'
```