ĐỀ CƯƠNG ÔN TẬP

TIN HỌC ĐẠI CƯƠNG

Dành cho sinh viên đại học năm đầu chưa từng học tin học, cần ôn thi học phần Tin học đại cương và dùng được máy tính, mạng, lập trình cơ bản

Introduction to Computing: Exam Review Guide – a bilingual guide

Biểu diễn dữ liệu · Phần cứng · Phần mềm · Hệ điều hành

Cơ sở dữ liệu · Mạng và Internet · An toàn thông tin

Hệ thống thông tin · Thuật toán · Lập trình Python

Song ngữ Việt – Anh · Vietnamese – English

Phiên bản 1.0 – Tháng 11 năm 2026

https://tailieuso.vin

MỤC LỤC

CONTENTS

Chương 1. Máy tính lưu và xử lý thông tin như thế nào – *Chapter 1. How Computers Store and Process Information* 6

1.1. Tin học, khoa học máy tính và ứng dụng trong các ngành – *1.1. Computing, Computer Science and Their Uses Across Fields* 6

1.2. Thông tin, dữ liệu và các đơn vị đo bit, byte – *1.2. Information, Data and Units of Measurement* 9

1.3. Hệ đếm nhị phân, thập phân, thập lục phân và cách đổi – *1.3. Binary, Decimal and Hexadecimal Number Systems* 12

1.4. Biểu diễn số nguyên và số thực trong máy – *1.4. Representing Integers and Real Numbers* 14

1.5. Biểu diễn ký tự, hình ảnh và âm thanh – *1.5. Representing Text, Images and Sound* 17

1.6. Phép toán logic, bảng chân trị và cổng logic – *1.6. Logic Operations, Truth Tables and Logic Gates* 20

1.7. Các thế hệ máy tính và mô hình tính toán – *1.7. Generations of Computers and Models of Computation* 23

1.8. Từ ngôn ngữ tự nhiên đến ngôn ngữ hình thức của máy – *1.8. From Natural Language to Formal Languages* 25

1.9. Bài tập vận dụng có lời giải – *1.9. Applied exercises with solutions* 29

1.10. Câu hỏi tự luận và dàn ý trả lời – *1.10. Essay questions and answer outlines* 30

1.11. Câu hỏi trắc nghiệm – *1.11. Multiple-choice questions* 33

Chương 2. Phần cứng: bên trong một chiếc máy tính – *Chapter 2. Hardware: Inside a Computer* 35

2.1. Mô hình Von Neumann và chu trình nạp, giải mã, thực thi lệnh – *2.1. The Von Neumann Model and the Fetch–Decode–Execute Cycle* 35

2.2. Bộ xử lý trung tâm: thành phần, xung nhịp và số nhân – *2.2. The CPU: Components, Clock Speed and Cores* 38

2.3. Bộ nhớ trong: RAM, ROM và bộ nhớ đệm – *2.3. Primary Memory: RAM, ROM and Cache* 40

2.4. Bộ nhớ ngoài: ổ cứng HDD, SSD, USB và lưu trữ đám mây – *2.4. Secondary Storage: HDD, SSD, USB Drives and Cloud Storage* 43

2.5. Thiết bị vào, thiết bị ra và cổng kết nối – *2.5. Input Devices, Output Devices and Ports* 46

2.6. Các loại máy tính từ máy chủ đến thiết bị nhúng – *2.6. Types of Computers: From Servers to Embedded Devices* 48

2.7. Đọc cấu hình và chọn máy theo nhu cầu học tập – *2.7. Reading Specifications and Choosing a Computer* 51

2.8. Xử lý song song và hiệu năng phần cứng – *2.8. Parallel Processing and Hardware Performance* 54

2.9. Bài tập vận dụng có lời giải – *2.9. Applied exercises with solutions* 57

2.10. Câu hỏi tự luận và dàn ý trả lời – *2.10. Essay questions and answer outlines* 59
2.11. Câu hỏi trắc nghiệm – *2.11. Multiple-choice questions* 60

Chương 3. Phần mềm, hệ điều hành và ứng dụng văn phòng – *Chapter 3. Software, Operating Systems and Office Applications* 63
3.1. Phần mềm hệ thống và phần mềm ứng dụng – *3.1. System Software and Application Software* 63
3.2. Chức năng của hệ điều hành – *3.2. What an Operating System Does* 66
3.3. Tệp, thư mục, đường dẫn và phần mở rộng – *3.3. Files, Folders, Paths and File Extensions* 69
3.4. Cài đặt, cập nhật và gỡ bỏ phần mềm an toàn – *3.4. Installing, Updating and Removing Software Safely* 73
3.5. Soạn thảo văn bản đúng quy cách – *3.5. Formatting Documents Properly* 75
3.6. Bảng tính: ô, địa chỉ ô, công thức và hàm cơ bản – *3.6. Spreadsheets: Cells, References, Formulas and Basic Functions* 78
3.7. Bài trình chiếu rõ ràng và dễ theo dõi – *3.7. Clear and Easy-to-Follow Presentations* 81
3.8. Giấy phép phần mềm, mã nguồn mở và phần mềm dịch vụ – *3.8. Software Licenses, Open Source and Software as a Service* 84
3.9. Bài tập vận dụng có lời giải – *3.9. Applied exercises with solutions* 87
3.10. Câu hỏi tự luận và dàn ý trả lời – *3.10. Essay questions and answer outlines* 89
3.11. Câu hỏi trắc nghiệm – *3.11. Multiple-choice questions* 90

Chương 4. Dữ liệu và cơ sở dữ liệu – *Chapter 4. Data and Databases* 93
4.1. Dữ liệu, thông tin và tri thức – *4.1. Data, Information and Knowledge* 93
4.2. Cơ sở dữ liệu và hệ quản trị cơ sở dữ liệu – *4.2. Databases and Database Management Systems* 95
4.3. Mô hình quan hệ: bảng, bản ghi, trường và khoá – *4.3. The Relational Model: Tables, Records, Fields and Keys* 98
4.4. Thiết kế cơ sở dữ liệu nhỏ và chuẩn hoá – *4.4. Designing a Small Database and Normalization* 101
4.5. Truy vấn dữ liệu với SQL cơ bản – *4.5. Querying Data with Basic SQL* 103
4.6. Kho dữ liệu, dữ liệu lớn và khai phá dữ liệu – *4.6. Data Warehouses, Big Data and Data Mining* 107
4.7. Chất lượng dữ liệu, sao lưu và quản trị dữ liệu – *4.7. Data Quality, Backups and Data Governance* 109
4.8. Biểu diễn dữ liệu bằng biểu đồ để ra quyết định – *4.8. Visualizing Data for Decision-Making* 112
4.9. Bài tập vận dụng có lời giải – *4.9. Applied exercises with solutions* 115
4.10. Câu hỏi tự luận và dàn ý trả lời – *4.10. Essay questions and answer outlines* 117
4.11. Câu hỏi trắc nghiệm – *4.11. Multiple-choice questions* 119

Chương 5. Mạng máy tính và Internet – *Chapter 5. Computer Networks and the Internet* 121
5.1. Mạng máy tính là gì và các loại mạng LAN, WAN – *5.1. What a Network Is: LAN, WAN and Other Types* 121

5.2. Thiết bị mạng và đường truyền có dây, không dây – *5.2. Network Devices and Wired or Wireless Media* 124
5.3. Giao thức và mô hình TCP/IP – *5.3. Protocols and the TCP/IP Model* 127
5.4. Địa chỉ IP, tên miền và hệ thống DNS – *5.4. IP Addresses, Domain Names and DNS* 130
5.5. World Wide Web, URL, HTTP và trình duyệt – *5.5. The World Wide Web, URLs, HTTP and Browsers* 133
5.6. Thư điện tử và các dịch vụ trên Internet – *5.6. Email and Other Internet Services* 136
5.7. Điện toán đám mây và làm việc cộng tác trực tuyến – *5.7. Cloud Computing and Online Collaboration* 139
5.8. Lịch sử Internet và Internet vạn vật – *5.8. A Short History of the Internet and the Internet of Things* 142
5.9. Bài tập vận dụng có lời giải – *5.9. Applied exercises with solutions* 145
5.10. Câu hỏi tự luận và dàn ý trả lời – *5.10. Essay questions and answer outlines* 147
5.11. Câu hỏi trắc nghiệm – *5.11. Multiple-choice questions* 149

Chương 6. An toàn thông tin và bảo vệ bản thân trên mạng – *Chapter 6. Information Security and Staying Safe Online* 151
6.1. Bộ ba bí mật, toàn vẹn, sẵn sàng – *6.1. Confidentiality, Integrity and Availability* 151
6.2. Mã độc: virus, sâu, trojan và mã độc tống tiền – *6.2. Malware: Viruses, Worms, Trojans and Ransomware* 154
6.3. Lừa đảo trực tuyến và tấn công phi kỹ thuật – *6.3. Phishing and Social Engineering* 157
6.4. Xác thực: mật khẩu mạnh và xác thực nhiều lớp – *6.4. Authentication: Strong Passwords and Multi-Factor Authentication* 159
6.5. Mã hoá đối xứng, khoá công khai và chữ ký số – *6.5. Symmetric Encryption, Public Keys and Digital Signatures* 162
6.6. Tường lửa, phân quyền truy cập và sao lưu – *6.6. Firewalls, Access Control and Backups* 165
6.7. Quyền riêng tư và dấu chân số – *6.7. Privacy and Your Digital Footprint* 168
6.8. Thói quen an toàn hằng ngày và ứng phó sự cố – *6.8. Everyday Security Habits and Incident Response* 171
6.9. Bài tập vận dụng có lời giải – *6.9. Applied exercises with solutions* 174
6.10. Câu hỏi tự luận và dàn ý trả lời – *6.10. Essay questions and answer outlines* 176
6.11. Câu hỏi trắc nghiệm – *6.11. Multiple-choice questions* 178

Chương 7. Hệ thống thông tin trong tổ chức và xã hội – *Chapter 7. Information Systems in Organizations and Society* 181
7.1. Năm thành phần của một hệ thống thông tin – *7.1. The Five Components of an Information System* 181
7.2. Công nghệ thông tin có tạo lợi thế cạnh tranh không – *7.2. Can Information Technology Create a Competitive Advantage?* 184
7.3. Quy trình nghiệp vụ và hệ thống hoạch định nguồn lực doanh nghiệp – *7.3. Business Processes and ERP Systems* 187

7.4. Con người và nghề nghiệp trong hệ thống thông tin – *7.4. People and Careers in Information Systems* 190
7.5. Vòng đời phát triển hệ thống và phương pháp linh động Agile – *7.5. The Systems Development Life Cycle and Agile Methods* 192
7.6. Từ thiết kế đến sản phẩm dùng được: tái sử dụng và kiến trúc phần mềm – *7.6. From Design to Usable Product: Reuse and Software Architecture* 195
7.7. Toàn cầu hoá, khoảng cách số, đạo đức và sở hữu trí tuệ – *7.7. Globalization, the Digital Divide, Ethics and Intellectual Property* 198
7.8. Xu hướng công nghệ: trí tuệ nhân tạo, tự động hoá và tác động xã hội – *7.8. Technology Trends: Artificial Intelligence, Automation and Social Impact* 201
7.9. Bài tập vận dụng có lời giải – *7.9. Applied exercises with solutions* 204
7.10. Câu hỏi tự luận và dàn ý trả lời – *7.10. Essay questions and answer outlines* 206
7.11. Câu hỏi trắc nghiệm – *7.11. Multiple-choice questions* 207

Chương 8. Tư duy máy tính và thuật toán – *Chapter 8. Computational Thinking and Algorithms* 210
8.1. Tư duy máy tính: phân rã, nhận dạng mẫu, trừu tượng hoá – *8.1. Computational Thinking: Decomposition, Patterns and Abstraction* 210
8.2. Thuật toán là gì và các tính chất bắt buộc – *8.2. What an Algorithm Is and Its Required Properties* 212
8.3. Sơ đồ khối, mã giả và ba cấu trúc điều khiển – *8.3. Flowcharts, Pseudocode and the Three Control Structures* 215
8.4. Cấu trúc dữ liệu cơ bản: mảng, danh sách liên kết, ngăn xếp, hàng đợi – *8.4. Basic Data Structures: Arrays, Linked Lists, Stacks and Queues* 218
8.5. Tìm kiếm tuần tự và tìm kiếm nhị phân – *8.5. Linear Search and Binary Search* ... 221
8.6. Sắp xếp chọn, sắp xếp nổi bọt và sắp xếp chèn – *8.6. Selection Sort, Bubble Sort and Insertion Sort* 224
8.7. Độ phức tạp thời gian và ký hiệu O lớn – *8.7. Time Complexity and Big O Notation* 227
8.8. Chiến lược thiết kế thuật toán và giới hạn của máy tính – *8.8. Algorithm Design Strategies and the Limits of Computation* 230
8.9. Bài tập vận dụng có lời giải – *8.9. Applied exercises with solutions* 233
8.10. Câu hỏi tự luận và dàn ý trả lời – *8.10. Essay questions and answer outlines* 235
8.11. Câu hỏi trắc nghiệm – *8.11. Multiple-choice questions* 237

Chương 9. Lập trình Python: biến, biểu thức, rẽ nhánh, vòng lặp và hàm – *Chapter 9. Python Programming: Variables, Expressions, Branching, Loops and Functions* 239
9.1. Chương trình, ngôn ngữ lập trình và cách chạy Python – *9.1. Programs, Programming Languages and Running Python* 239
9.2. Giá trị, kiểu dữ liệu, biến và câu lệnh gán – *9.2. Values, Data Types, Variables and Assignment* 242
9.3. Toán tử số học, biểu thức và thứ tự ưu tiên phép toán – *9.3. Arithmetic Operators, Expressions and Order of Operations* 245
9.4. Nhập, xuất dữ liệu và chuyển đổi kiểu – *9.4. Input, Output and Type Conversion* .. 248

9.5. Biểu thức logic và câu lệnh if, elif, else – *9.5. Boolean Expressions and if, elif, else Statements*.....251
9.6. Vòng lặp for, while và hàm range – *9.6. for Loops, while Loops and range*.....254
9.7. Gọi hàm có sẵn, module math và lồng ghép lời gọi hàm – *9.7. Calling Built-in Functions, the math Module and Composition*.....258
9.8. Tự định nghĩa hàm: tham số, giá trị trả về, biến cục bộ và sơ đồ ngăn xếp – *9.8. Defining Functions: Parameters, Return Values, Local Variables and Stack Diagrams*.....261
9.9. Tìm và sửa lỗi chương trình – *9.9. Finding and Fixing Bugs*.....264
9.10. Bài tập vận dụng có lời giải – *9.10. Applied exercises with solutions*.....267
9.11. Câu hỏi tự luận và dàn ý trả lời – *9.11. Essay questions and answer outlines*.....269
9.12. Câu hỏi trắc nghiệm – *9.12. Multiple-choice questions*.....271

Chương 10. Lập trình Python: chuỗi, danh sách, từ điển, tệp và đối tượng – *Chapter 10. Python Programming: Strings, Lists, Dictionaries, Files and Objects*..... 274
10.1. Chuỗi: chỉ số, cắt chuỗi, duyệt và phương thức – *10.1. Strings: Indexing, Slicing, Traversal and Methods*.....274
10.2. Danh sách: tạo, sửa, duyệt và các thao tác thường dùng – *10.2. Lists: Creating, Changing, Traversing and Common Operations*.....277
10.3. Từ điển: lưu theo cặp khoá và giá trị – *10.3. Dictionaries: Storing Key–Value Pairs*.....280
10.4. Bộ tuple và tập hợp set – *10.4. Tuples and Sets*.....283
10.5. Đọc và ghi tệp văn bản – *10.5. Reading and Writing Text Files*.....287
10.6. Đệ quy: hàm tự gọi chính nó – *10.6. Recursion: Functions That Call Themselves*289
10.7. Lớp và đối tượng ở mức cơ bản – *10.7. Classes and Objects: The Basics*.....292
10.8. Viết một chương trình hoàn chỉnh: thiết kế hàm, đóng gói và kiểm thử – *10.8. Building a Complete Program: Function Design, Encapsulation and Testing*.....294
10.9. Bài tập vận dụng có lời giải – *10.9. Applied exercises with solutions*.....297
10.10. Câu hỏi tự luận và dàn ý trả lời – *10.10. Essay questions and answer outlines*...299
10.11. Câu hỏi trắc nghiệm – *10.11. Multiple-choice questions*.....301

Phụ lục A. Đáp án trắc nghiệm – *Appendix A. Answers to the multiple-choice questions*..... 303

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. Máy tính lưu và xử lý thông tin như thế nào – *Chapter 1. How Computers Store and Process Information*

Mục tiêu chương – *Chapter objectives*

- Đổi qua lại giữa hệ thập phân, nhị phân và thập lục phân bằng phép chia dần và nhóm bit
  Convert between decimal, binary and hexadecimal using repeated division and bit grouping
- Biểu diễn số nguyên âm bằng bù hai và cộng hai số có dấu trong 8 bit, rồi kiểm tra kết quả
  Represent negative integers in two's complement, add two signed 8-bit numbers and check the result
- Lập bảng chân trị cho biểu thức logic và nói được cổng nào cho ra 1 trong trường hợp nào
  Build a truth table for a logic expression and say which gate outputs 1 under which conditions

1.1. Tin học, khoa học máy tính và ứng dụng trong các ngành – *1.1. Computing, Computer Science and Their Uses Across Fields*

Khi bạn đặt xe qua ứng dụng, điện thoại gửi vị trí của bạn lên máy chủ. Máy chủ tìm tài xế gần nhất, tính giá rồi báo lại sau vài giây. Sáng đến công ty, bạn đặt ngón tay lên máy chấm công. Máy so vân tay với mẫu đã lưu rồi ghi giờ vào bảng. Hai việc này giống nhau ở chỗ: máy nhận dữ liệu, xử lý theo quy tắc, rồi trả kết quả.

When you book a ride in an app, your phone sends your location to a server. The server finds the nearest driver, works out the fare and replies within seconds. At the office in the morning, you press a finger on the time clock. It compares your fingerprint with a stored sample and logs the time in a table. Both cases follow the same pattern: a machine takes in data, processes it by rules and returns a result.

Tin học và khoa học máy tính là gì – *What computing and computer science are*

Tin học là lĩnh vực chung về thu thập, lưu trữ, xử lý và truyền thông tin bằng máy tính. Nó gồm cả việc dùng máy, làm ra phần mềm và nghiên cứu cách máy làm việc. Có thể hình dung nó như "nghề bếp": ai nấu, ai dạy nấu, ai nghiên cứu món ăn đều thuộc về đó.

Computing is the broad field of collecting, storing, processing and transmitting information with computers. It covers using machines, building software and studying how machines work. Think of it as the "world of cooking": people who cook, people who teach cooking and people who research food all belong to it.

Khoa học máy tính là phần nghiên cứu bên trong tin học. Nó hỏi: máy giải được bài toán nào, giải bằng cách nào cho đúng và nhanh, rồi biến cách đó thành chương trình. Cách giải từng bước, rõ ràng, như một công thức nấu ăn, gọi là thuật toán. Ứng dụng đặt xe cần một thuật toán tìm tài xế gần nhất trong hàng chục nghìn xe mà không bắt bạn chờ cả phút. Nghĩ ra thuật toán đó là việc của khoa học máy tính.

Computer science is the research core inside computing. It asks which problems a machine can solve, how to solve them correctly and quickly, and how to turn the method into a program. A clear step-by-step method, like a recipe, is called an algorithm. A ride app needs an algorithm that finds the nearest driver among tens of thousands of cars without making you wait a minute. Devising that algorithm is computer science work.

Các nhánh chính – *Main branches*

- Thuật toán và lý thuyết tính toán: tìm cách giải hiệu quả và biết bài nào máy không giải nổi.
  Algorithms and theory of computation: finding efficient methods and knowing which problems machines cannot solve.
- Hệ thống máy tính và mạng: phần cứng, hệ điều hành, kết nối giữa các máy.
  Computer systems and networks: hardware, operating systems and links between machines.
- Phần mềm và kỹ nghệ phần mềm: thiết kế, viết, kiểm thử và bảo trì chương trình.
  Software and software engineering: designing, writing, testing and maintaining programs.
- Dữ liệu và cơ sở dữ liệu: tổ chức, lưu trữ và truy xuất lượng thông tin lớn.
  Data and databases: organizing, storing and retrieving large amounts of information.
- Trí tuệ nhân tạo và học máy: cho máy học từ dữ liệu để dịch ngôn ngữ, gợi ý quảng cáo, lái xe tự động.
  Artificial intelligence and machine learning: letting machines learn from data to translate languages, suggest ads or drive cars.
- Tương tác người – máy: thử nghiệm và đánh giá giao diện để máy dễ dùng hơn.
  Human-computer interaction: testing and evaluating interfaces so machines are easier to use.
- An toàn thông tin: giữ dữ liệu khỏi bị lộ, bị sửa trái phép hay bị mất.
  Information security: keeping data from being leaked, altered without permission or lost.

Tin học trong các ngành – *Computing across fields*

Các ngành khác mượn công cụ của khoa học máy tính, và bản thân họ cũng trở nên "tính toán" hơn. Bảng sau tóm tắt bốn ngành quen thuộc.

Other fields borrow tools from computer science, and they themselves are becoming more computational. This table summarizes four familiar fields:

Ngành *Field*	**Tin học giúp việc gì** *What computing does*	**Ví dụ cụ thể** *Concrete example*
Y tế *Healthcare*	Lưu hồ sơ bệnh án, hỗ trợ đọc ảnh chụp, phân tích gen. *Storing patient records, helping read scans, analyzing genes.*	Bác sĩ xem lại toàn bộ lịch sử khám của bệnh nhân trên máy ngay khi bệnh nhân đến. *A doctor sees a patient's full visit history on screen the moment the patient arrives.*

Ngành *Field*	**Tin học giúp việc gì** *What computing does*	**Ví dụ cụ thể** *Concrete example*
Kinh tế *Business*	Xử lý giao dịch, quản lý kho, phân tích hành vi khách hàng. *Processing transactions, managing stock, analyzing customer behavior.*	Cửa hàng xem bảng tính doanh thu để quyết định nhập thêm mặt hàng nào. *A shop reads a sales spreadsheet to decide which product to restock.*
Kỹ thuật *Engineering*	Mô phỏng và tính toán trước khi làm thật. *Simulating and calculating before building for real.*	Kỹ sư chạy mô phỏng chịu tải của cầu rồi mới đổ bê tông; trạm khí tượng dự báo thời tiết bằng mô hình số. *Engineers simulate a bridge under load before pouring concrete; weather services forecast with numerical models.*
Giáo dục *Education*	Dạy từ xa, chấm bài tự động, theo dõi tiến độ học. *Remote teaching, automatic grading, tracking progress.*	Sinh viên làm trắc nghiệm trực tuyến và nhận điểm ngay khi nộp. *Students take an online quiz and get a score the moment they submit.*

Khi dữ liệu quá lớn so với bảng tính, người ta gọi đó là dữ liệu lớn. Ngành khoa học dữ liệu kết hợp kỹ năng tính toán với hiểu biết riêng của lĩnh vực để rút thông tin từ khối dữ liệu ấy.

When data grows too large for a spreadsheet, it is called big data. Data science combines computing skills with knowledge of a specific field to extract information from such data.

Người dùng, người lập trình và nhà khoa học máy tính – ***Users, programmers and computer scientists***

Ba vai này khác nhau ở việc họ làm với phần mềm. Người dùng chỉ sử dụng phần mềm để hoàn thành công việc. Người lập trình viết chương trình theo yêu cầu có sẵn. Nhà khoa học máy tính nghiên cứu cách làm mới, so sánh, chứng minh cách nào tốt hơn.

The three roles differ in what they do with software. A user simply uses software to get a job done. A programmer writes programs to given requirements. A computer scientist researches new methods and proves which is better.

1. Nhìn vào việc cụ thể và hỏi: người này đang dùng sản phẩm có sẵn, hay đang tạo ra sản phẩm?
   Look at the specific job and ask: is this person using a finished product or creating one?
2. Nếu đang tạo, hỏi tiếp: họ viết mã theo yêu cầu đã rõ, hay đang tìm một cách giải chưa ai có?
   If creating, ask next: are they writing code to clear requirements, or searching for a solution nobody has yet?
3. Gán vai: dùng sản phẩm là người dùng, viết mã theo yêu cầu là người lập trình, tìm cách giải mới là nhà khoa học máy tính.
   Assign the role: using a product is a user, writing code to requirements is a programmer, finding a new method is a computer scientist.

VÍ DỤ / EXAMPLE – Với ứng dụng đặt xe: chị Lan mở app gọi xe đi làm, chị là người dùng. Anh Minh viết màn hình đặt xe và nút hủy chuyến theo bản thiết kế, anh là người lập trình. Cô Hà nghiên cứu cách ghép khách với tài xế sao cho tổng thời gian chờ ngắn nhất, rồi chứng minh cách của cô nhanh hơn cách cũ, cô là nhà khoa học máy tính. Cùng một ứng dụng, ba người ở ba vai khác nhau.
Take the ride app: Lan opens it to get to work, so she is a user. Minh builds the booking screen and cancel button from a design, so he is a programmer. Ha researches how to match riders and drivers to minimize total waiting time, then shows her method beats the old one, so she is a computer scientist. One app, three people in three roles.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm tin học với biết dùng Word hay Excel. Dùng thành thạo phần mềm mới là một phần nhỏ của tin học.
Equating computing with knowing Word or Excel. Using software well is only a small part of computing.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm khoa học máy tính với sửa máy hay cài phần mềm. Đó là việc kỹ thuật viên, còn khoa học máy tính xoay quanh thuật toán, dữ liệu và hệ thống.
Confusing computer science with repairing machines or installing software. That is technician work; computer science centers on algorithms, data and systems.

LỖI HAY GẶP / COMMON MISTAKE – Cho rằng mỗi người chỉ có một vai. Một sinh viên có thể là người dùng buổi sáng và người lập trình buổi tối.
Assuming each person has only one role. A student can be a user in the morning and a programmer in the evening.

PHẢI THUỘC / MEMORISE – Tin học là lĩnh vực rộng về xử lý thông tin bằng máy tính; khoa học máy tính là phần nghiên cứu thuật toán, hệ thống và dữ liệu. Ba vai: người dùng sử dụng, người lập trình viết mã, nhà khoa học máy tính tìm cách giải mới.
Computing is the broad field of processing information with computers; computer science is the study of algorithms, systems and data. Three roles: users use, programmers write code, computer scientists find new methods.

1.2. Thông tin, dữ liệu và các đơn vị đo bit, byte – *1.2. Information, Data and Units of Measurement*

Hãy nhìn một bảng điểm: cột tên môn, cột điểm, các con số 8, 6, 7 nằm rời rạc. Đó là dữ liệu: những ký hiệu, con số, chữ cái ghi lại sự việc, chưa cho bạn biết điều gì cần làm. Khi bạn xử lý chúng thành "điểm trung bình của Minh là 7,0, cả ba môn đều qua", đó là thông tin: kết quả có nghĩa, giúp bạn quyết định.
P: Picture a grade sheet: a column of subject names and a column of scores like 8, 6 and 7. That is data: raw symbols, numbers or letters that record facts but don't yet tell you what to do. Once you process them into "Minh's average is 7.0 and he passed all three subjects", you have information: a result that means something and helps you decide.

VÍ DỤ / EXAMPLE – Dữ liệu: Toán 8, Lý 6, Tin 7. Xử lý: (8 + 6 + 7) / 3 = 21 / 3 = 7,0. Thông tin: điểm trung bình 7,0, đều trên mức qua môn là 5. Cùng một dữ liệu, người khác

> tính cách khác sẽ ra thông tin khác, ví dụ môn thấp nhất là Lý.
> *EXAMPLE: Data: Math 8, Physics 6, IT 7. Processing: (8 + 6 + 7) / 3 = 21 / 3 = 7.0. Information: an average of 7.0, above the pass mark of 5. The same data processed another way gives different information, for example that Physics is the lowest subject.*

Bit và byte – *Bit and byte*

Máy tính chạy bằng điện, nên chỉ phân biệt được hai trạng thái: có tín hiệu hoặc không có. Ta ghi hai trạng thái ấy là 1 và 0. Một chữ số 1 hoặc 0 gọi là bit, viết tắt của "binary digit" (chữ số nhị phân). Giống như một công tắc đèn: chỉ bật hoặc tắt.

P: A computer runs on electricity, so it can tell only two states apart: signal or no signal. We write them as 1 and 0. A single 1 or 0 is called a bit, short for "binary digit". Think of a light switch: it is either on or off.

Một bit chứa quá ít, nên người ta gom 8 bit thành một byte. Một byte có $2^8 = 256$ tổ hợp khác nhau, đủ để ghi một chữ cái, một chữ số hay một dấu câu. Văn bản tiếng Anh thường tốn khoảng 1 byte cho mỗi ký tự.

P: One bit holds very little, so bits are grouped in eights to make a byte. A byte has $2^8 = 256$ different patterns, enough to store a letter, a digit or a punctuation mark. English text typically costs about 1 byte per character.

KB, MB, GB, TB và hai cách quy đổi – *KB, MB, GB, TB and two ways to convert*

Khi dữ liệu lớn, ta dùng tiền tố cho gọn, như đổi 1000 mét thành 1 ki-lô-mét. Có hai hệ cùng tồn tại. Hệ 1000 (thập phân) dùng bội của 10^3: 1 KB = 1000 byte. Hệ 1024 (nhị phân) dùng bội của 2^{10}: 1 KB = 1024 byte. Số 1024 xuất hiện vì máy tính tính theo luỹ thừa của 2, và 2^{10} gần 1000 nhất.

P: For large amounts we use prefixes, like turning 1000 metres into 1 kilometre. Two systems exist side by side. The decimal system uses powers of 10^3: 1 KB = 1000 bytes. The binary system uses powers of 2^{10}: 1 KB = 1024 bytes. The number 1024 appears because computers count in powers of 2, and 2^{10} is the closest to 1000.

Đơn vị *Unit*	**Hệ 1000 (nhà sản xuất ổ đĩa, USB hay ghi)** *Decimal (used on disk and USB labels)*	**Hệ 1024 (bộ nhớ, nhiều phần mềm hệ điều hành)** *Binary (memory, many operating system displays)*
1 KB (kilobyte)	1000 byte	1024 byte
1 MB (megabyte)	1000 KB = 1 000 000 byte	1024 KB = 1 048 576 byte
1 GB (gigabyte)	1000 MB = 10^9 byte	1024 MB = 1 073 741 824 byte
1 TB (terabyte)	1000 GB = 10^{12} byte	1024 GB = 2^{40} byte

Để tránh nhầm, tiêu chuẩn quốc tế đặt tên riêng cho hệ 1024: KiB, MiB, GiB, TiB (đọc là kibibyte, mebibyte...). Trong bài thi, nếu đề không nói gì thì dùng 1024, và ghi rõ bạn dùng hệ nào.

P: To avoid confusion, an international standard gives the 1024 system its own names: KiB, MiB, GiB, TiB (kibibyte, mebibyte and so on). In an exam, use 1024 unless the question says otherwise, and state which system you used.

Cách tính dung lượng – *How to calculate size*

1. Đổi mọi thứ về byte, vì byte là đơn vị chung nhỏ nhất dễ so sánh.
   Convert everything to bytes, the smallest common unit that is easy to compare.
2. Nhân hoặc chia để ra số byte cần tìm (số điểm ảnh nhân số byte mỗi điểm, hoặc dung lượng chia cho kích thước một tệp).
   Multiply or divide to get the byte count you need (pixels times bytes per pixel, or capacity divided by one file's size).
3. Đổi kết quả sang KB, MB hay GB bằng cách chia cho 1000 hoặc 1024, mỗi bậc một lần.
   Convert the result to KB, MB or GB by dividing by 1000 or 1024, once per step.

VÍ DỤ / EXAMPLE – Ảnh 4000 × 3000 điểm ảnh (pixel, các chấm nhỏ tạo nên ảnh), mỗi điểm 24 bit, chưa nén. Mỗi điểm = 24 / 8 = 3 byte. Số điểm = 4000 × 3000 = 12 000 000. Dung lượng = 12 000 000 × 3 = 36 000 000 byte. Hệ 1000: 36 MB. Hệ 1024: 36 000 000 / 1 048 576 ≈ 34,33 MiB.
EXAMPLE: A 4000 × 3000 pixel photo (pixels are the tiny dots that make up an image), 24 bits per pixel, uncompressed. Each pixel = 24 / 8 = 3 bytes. Pixels = 4000 × 3000 = 12,000,000. Size = 12,000,000 × 3 = 36,000,000 bytes. Decimal: 36 MB. Binary: 36,000,000 / 1,048,576 ≈ 34.33 MiB.

VÍ DỤ / EXAMPLE – USB ghi 64 GB, mỗi bài hát 4 MB. USB tính theo hệ 1000: 64 GB = 64 × 10^9 byte. Bài hát: 4 × 10^6 byte. Số bài = 64 000 000 000 / 4 000 000 = 16 000 bài. Máy tính hiển thị theo hệ 1024 sẽ báo 64 × 10^9 / 1 073 741 824 ≈ 59,6 GiB, nên bạn đừng ngạc nhiên vì "hụt". Thực tế còn ít hơn chút do hệ thống tệp chiếm chỗ.
EXAMPLE: A USB stick labelled 64 GB, each song 4 MB. The label uses the decimal system: 64 GB = 64 × 10^9 bytes. A song is 4 × 10^6 bytes. Songs = 64,000,000,000 / 4,000,000 = 16,000. A computer that displays in binary units will report 64 × 10^9 / 1,073,741,824 ≈ 59.6 GiB, so the "missing" space is not a defect. In practice you get slightly fewer songs because the file system takes some room.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm b với B: b nhỏ là bit, B hoa là byte. Mạng 100 Mbps là 100 triệu bit mỗi giây, tức chỉ 100 / 8 = 12,5 MB mỗi giây.
MISTAKE: Mixing up b and B: lowercase b is a bit, uppercase B is a byte. A 100 Mbps line moves 100 million bits per second, which is only 100 / 8 = 12.5 MB per second.

LỖI HAY GẶP / COMMON MISTAKE – Trộn hệ 1000 với hệ 1024 trong cùng một phép tính, hoặc quên đổi bit sang byte (chia 8) trước khi nhân.
MISTAKE: Mixing the 1000 and 1024 systems in one calculation, or forgetting to turn bits into bytes (divide by 8) before multiplying.

PHẢI THUỘC / MEMORISE – Dữ liệu là con số, chữ chưa qua xử lý; thông tin là kết quả có nghĩa sau khi xử lý. 1 byte = 8 bit. 1 KB = 1000 hoặc 1024 byte, mỗi bậc kế tiếp (MB, GB, TB) nhân thêm 1000 hoặc 1024.
MEMORIZE: Data is raw numbers and symbols; information is the meaningful result of processing. 1

> *byte = 8 bits. 1 KB = 1000 or 1024 bytes, and each next step (MB, GB, TB) multiplies by another 1000 or 1024.*

1.3. Hệ đếm nhị phân, thập phân, thập lục phân và cách đổi – *1.3. Binary, Decimal and Hexadecimal Number Systems*

Hệ đếm theo vị trí nghĩa là cùng một chữ số, đứng ở chỗ khác nhau thì có giá trị khác nhau. Trong số 505, chữ số 5 đầu là năm trăm, chữ số 5 cuối chỉ là năm đơn vị. Mỗi vị trí có một "quyền số", tính bằng cơ số nâng lên lũy thừa, đếm từ phải sang trái và bắt đầu từ 0.

A positional number system means the same digit has a different value depending on where it sits. In 505, the first 5 is five hundred and the last 5 is just five. Each position has a "weight": the base raised to a power, counted from right to left starting at 0.

Cơ số (base) là số ký hiệu mà hệ đó dùng. Hệ thập phân dùng cơ số 10 với các chữ số 0 đến 9. Số 1010 thập phân bằng $1\times1000 + 0\times100 + 1\times10 + 0\times1$. Máy tính dùng nhị phân, cơ số 2, vì mạch điện chỉ cần phân biệt hai trạng thái: có điện và không có điện.

The base is the number of symbols a system uses. Decimal has base 10 and the digits 0 to 9. The decimal number 1010 equals $1\times1000 + 0\times100 + 1\times10 + 0\times1$. Computers use binary, base 2, because a circuit only needs to tell two states apart: current or no current.

Nhị phân chỉ có hai chữ số là 0 và 1, mỗi chữ số gọi là một bit. Quyền số từ phải sang trái là 1, 2, 4, 8, 16, 32, 64, 128. Số 1010 nhị phân bằng 8 + 2 = 10 thập phân. Cùng dãy chữ số nhưng khác cơ số thì khác giá trị, nên người ta viết cơ số nhỏ bên dưới: 1010_2 và 1010_{10}.

Binary has only two digits, 0 and 1, and each one is a bit. The weights from right to left are 1, 2, 4, 8, 16, 32, 64, 128. Binary 1010 equals 8 + 2 = 10 in decimal. The same digits in a different base mean a different value, so a small subscript marks the base: 1010_2 and 1010_{10}.

Thập lục phân (hexadecimal, gọi tắt là hex) có cơ số 16. Nó cần 16 ký hiệu, nên sau chữ số 9 người ta mượn chữ cái. Hex hay dùng vì mỗi chữ số hex ứng đúng 4 bit, nên viết gọn hơn nhị phân rất nhiều.

Hexadecimal (hex for short) has base 16. It needs 16 symbols, so after the digit 9 it borrows letters. Hex is popular because each hex digit matches exactly 4 bits, which makes long binary strings much shorter to write.

Chữ hex *Hex letter*	**Giá trị thập phân** *Decimal value*	**Nhóm 4 bit** *4-bit group*
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110

Chữ hex *Hex letter*	**Giá trị thập phân** *Decimal value*	**Nhóm 4 bit** *4-bit group*
F	15	1111

Đổi thập phân sang nhị phân – *Decimal to binary*

Cách làm là chia liên tiếp cho 2 và ghi lại số dư. Mỗi số dư chính là một bit, bit đầu tiên tìm được là bit thấp nhất.

The method is to divide by 2 repeatedly and record the remainders. Each remainder is one bit, and the first one you find is the lowest bit.

1. Chia số cần đổi cho 2, ghi thương và số dư (0 hoặc 1).
   Divide the number by 2 and write down the quotient and the remainder (0 or 1).
2. Lấy thương vừa có chia tiếp cho 2, ghi số dư mới.
   Divide the new quotient by 2 and write down the new remainder.
3. Lặp lại đến khi thương bằng 0.
   Repeat until the quotient reaches 0.
4. Đọc các số dư từ dưới lên trên, đó là kết quả.
   Read the remainders from bottom to top. That is the result.

> **VÍ DỤ / EXAMPLE –** Đổi 156. Ta có 156 : 2 = 78 dư 0; 78 : 2 = 39 dư 0; 39 : 2 = 19 dư 1; 19 : 2 = 9 dư 1; 9 : 2 = 4 dư 1; 4 : 2 = 2 dư 0; 2 : 2 = 1 dư 0; 1 : 2 = 0 dư 1. Đọc từ dưới lên: 10011100_2. Thử lại bằng phép cộng 128 + 16 + 8 + 4 = 156.
> *Convert 156. We get 156 ÷ 2 = 78 r 0; 78 ÷ 2 = 39 r 0; 39 ÷ 2 = 19 r 1; 19 ÷ 2 = 9 r 1; 9 ÷ 2 = 4 r 1; 4 ÷ 2 = 2 r 0; 2 ÷ 2 = 1 r 0; 1 ÷ 2 = 0 r 1. Read bottom to top: 10011100_2. Check by adding 128 + 16 + 8 + 4 = 156.*

Đổi nhị phân sang thập phân – *Binary to decimal*

1. Đánh số vị trí cho từng bit, bắt đầu từ 0 ở bit ngoài cùng bên phải.
   Number each bit's position, starting from 0 at the far right.
2. Với mỗi bit, lấy bit đó nhân với 2 mũ vị trí.
   For each bit, multiply the bit by 2 raised to its position.
3. Cộng tất cả các tích lại.
   Add all the products.

> **VÍ DỤ / EXAMPLE –** Đổi 101101_2. Từ trái sang phải, các vị trí là 5, 4, 3, 2, 1, 0. Tổng bằng $1\times32 + 0\times16 + 1\times8 + 1\times4 + 0\times2 + 1\times1 = 32 + 8 + 4 + 1 = 45$.
> *Convert 101101_2. From left to right the positions are 5, 4, 3, 2, 1, 0. The sum is $1\times32 + 0\times16 + 1\times8 + 1\times4 + 0\times2 + 1\times1 = 32 + 8 + 4 + 1 = 45$.*

Nhị phân và thập lục phân – *Binary and hexadecimal*

1. Từ dấu phân cách, chia dãy bit thành các nhóm 4 bit, đi từ phải sang trái.
   Split the bits into groups of 4, working from right to left.

2. Nếu nhóm bên trái cùng thiếu bit, thêm số 0 vào đầu cho đủ 4.
   If the leftmost group is short, add zeros at its front to make 4.
3. Đổi từng nhóm thành một chữ số hex theo bảng ở trên.
   Turn each group into one hex digit using the table above.

VÍ DỤ / EXAMPLE – Đổi 10011100_2. Tách thành 1001 và 1100, tức 9 và C, vậy kết quả là $9C_{16}$. Thử lại: 9×16 + 12 = 156. Một ví dụ cần thêm số 0: 1101011_2 có 7 bit, viết thành 0110 1011, tức 6 và B, ra $6B_{16}$ = 107.
Convert 10011100_2. Split into 1001 and 1100, which are 9 and C, so the result is $9C_{16}$. Check: 9×16 + 12 = 156. An example that needs padding: 1101011_2 has 7 bits, so write it as 0110 1011, which is 6 and B, giving $6B_{16}$ = 107.

Đổi ngược từ hex sang nhị phân thì thay mỗi chữ số bằng đúng 4 bit. Với $3F_{16}$: 3 là 0011, F là 1111, nên được 00111111_2, tức 63. Đổi hex sang thập phân thì nhân từng chữ số với lũy thừa của 16: $2A_{16} = 2×16 + 10 = 42$.
To go from hex back to binary, replace each hex digit with exactly 4 bits. For $3F_{16}$: 3 is 0011 and F is 1111, so you get 00111111_2, which is 63. To convert hex to decimal, multiply each digit by a power of 16: $2A_{16} = 2×16 + 10 = 42$.

LỖI HAY GẶP / COMMON MISTAKE – Đọc số dư từ trên xuống thay vì từ dưới lên, kết quả sẽ bị đảo ngược. Với 156, đọc sai ra 00111001 thay vì 10011100.
Reading the remainders top to bottom instead of bottom to top reverses the result. For 156, the wrong reading gives 00111001 instead of 10011100.

LỖI HAY GẶP / COMMON MISTAKE – Chia nhóm 4 bit từ trái sang phải. Phải chia từ phải sang trái, và chỉ thêm số 0 ở đầu bên trái.
Grouping the bits in 4s from the left. Always group from the right, and add zeros only at the left end.

LỖI HAY GẶP / COMMON MISTAKE – Quên rằng vị trí đếm từ 0, nên bit ngoài cùng bên phải nhân với $2^0 = 1$ chứ không phải 2^1.
Forgetting that positions start at 0, so the rightmost bit is multiplied by $2^0 = 1$, not 2^1.

PHẢI THUỘC / MEMORISE – Quyền số nhị phân: 1, 2, 4, 8, 16, 32, 64, 128. Các chữ hex: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. Một chữ số hex bằng đúng 4 bit.
Binary weights: 1, 2, 4, 8, 16, 32, 64, 128. Hex letters: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. One hex digit equals exactly 4 bits.

- Tự luyện: đổi 100 sang nhị phân rồi sang hex. Đáp án: 1100100_2 và 64_{16}. Kiểm tra: 64 + 32 + 4 = 100, và 6×16 + 4 = 100.
  Practice: convert 100 to binary and then to hex. Answer: 1100100_2 and 64_{16}. Check: 64 + 32 + 4 = 100, and 6×16 + 4 = 100.

1.4. Biểu diễn số nguyên và số thực trong máy – *1.4. Representing Integers and Real Numbers*

Bộ nhớ máy tính chỉ chứa bit, nên mọi số đều phải viết bằng 0 và 1. Số nguyên là số không có phần lẻ, như 7 hay −3. Số có phần lẻ như 3,14 gọi là số thực. Hai loại này được lưu theo hai cách khác nhau.

A computer's memory holds only bits, so every number must be written with 0s and 1s. An integer has no fractional part, like 7 or −3. A number with a fractional part, like 3.14, is a real number. The two kinds are stored in different ways.

Số nguyên không dấu – *Unsigned integers*

Số nguyên không dấu chỉ có 0 hoặc số dương, nên không cần chỗ ghi dấu. Hãy nghĩ tới đồng hồ cây số có 3 chữ số: nó chỉ chạy từ 000 đến 999. Máy cũng vậy: n bit cho 2^n giá trị, từ 0 đến 2^n − 1. Với 8 bit, bạn có 256 giá trị, từ 0 đến 255. Loại này dùng để đếm những thứ không thể âm, như số viên bi trong túi hay số trang của một tệp.

An unsigned integer is either 0 or positive, so it needs no room for a sign. Think of a car odometer with 3 digits: it only runs from 000 to 999. A computer works the same way: n bits give 2^n values, from 0 to 2^n − 1. With 8 bits you get 256 values, from 0 to 255. Use this type for things that can never be negative, such as the number of marbles in a bag or the number of pages in a file.

Số nguyên có dấu theo bù 2 – *Signed integers in two's complement*

Số có dấu có thể dương, âm hoặc 0. Máy dùng cách gọi là bù 2 (two's complement). Bit ngoài cùng bên trái là bit dấu: 0 là dương hoặc 0, 1 là âm. Với 8 bit, vẫn có 256 giá trị nhưng chia đôi: từ −128 đến +127. Công thức chung là từ −2^(n−1) đến 2^(n−1) − 1.

A signed integer can be positive, negative or 0. Computers use a scheme called two's complement. The leftmost bit is the sign bit: 0 means positive or zero, 1 means negative. With 8 bits there are still 256 values, but they are split in half: from −128 to +127. The general range is from −2^(n−1) to 2^(n−1) − 1.

Bù 2 được chọn vì phép cộng và phép trừ dùng chung một mạch phần cứng. Trừ một số chỉ là cộng với số đối của nó.

Two's complement is the standard choice because addition and subtraction can share one piece of hardware. Subtracting a number is just adding its negative.

Đổi số âm sang bù 2 – *Converting a negative number to two's complement*

1. Viết giá trị tuyệt đối của số cần đổi bằng 8 bit. Với −5, viết 5 thành 00000101.
   Write the absolute value of the number in 8 bits. For −5, write 5 as 00000101.
2. Đảo mọi bit: 0 thành 1, 1 thành 0. Ta được 11111010.
   Flip every bit: each 0 becomes 1 and each 1 becomes 0. This gives 11111010.
3. Cộng 1 vào kết quả: 11111010 + 1 = 11111011. Đây là dạng bù 2 của −5.
   Add 1 to the result: 11111010 + 1 = 11111011. This is −5 in two's complement.
4. Kiểm tra: đọc 11111011 như số không dấu thì được 251, và 251 − 256 = −5. Đúng.
   Check: read 11111011 as an unsigned number to get 251, and 251 − 256 = −5. Correct.

> **VÍ DỤ / EXAMPLE –** Tính 13 + (−5) trên 8 bit. Ta có 13 = 00001101 và −5 = 11111011. Cộng từng cột từ phải sang trái, nhớ sang trái như cộng thập phân: 00001101 + 11111011 = 1 00001000. Kết quả có 9 bit, bạn bỏ bit nhớ ngoài cùng, còn 00001000 = 8. Đúng với 13 − 5 = 8.
> *Compute 13 + (−5) in 8 bits. We have 13 = 00001101 and −5 = 11111011. Add column by column*

> *from right to left, carrying leftward as in decimal: 00001101 + 11111011 = 1 00001000. The result has 9 bits, so drop the outer carry and keep 00001000 = 8. This matches 13 − 5 = 8.*

Muốn đọc ngược một dãy có bit đầu bằng 1, bạn cũng đảo bit rồi cộng 1 để tìm độ lớn, sau đó gắn dấu trừ.

To read a pattern that starts with 1, flip the bits and add 1 to find the magnitude, then put a minus sign in front.

Tràn số – *Overflow*

Kết quả vượt phạm vi thì máy không báo lỗi mà quay vòng, như đồng hồ cây số về 000. Với 8 bit có dấu: 01111111 + 00000001 = 10000000, tức 127 + 1 ra −128. Vì vậy phải chọn số bit đủ rộng cho dữ liệu.

When a result goes beyond the range, the machine does not complain; it wraps around, like an odometer returning to 000. With signed 8 bits: 01111111 + 00000001 = 10000000, so 127 + 1 gives −128. Choose a bit width wide enough for your data.

Số thực dấu phẩy động – *Floating-point numbers*

Bạn đã quen ký hiệu khoa học: 12300 = 1,23 × 10^4. Máy làm tương tự, nhưng trong hệ nhị phân. Một số được tách ba phần: dấu, số mũ (cho biết dấu phẩy nằm ở đâu, nên gọi là dấu phẩy "động") và phần định trị (các chữ số có nghĩa). Chuẩn IEEE 754 dùng 32 bit cho số đơn: 1 bit dấu, 8 bit mũ, 23 bit định trị. Nhờ cách này, 32 bit biểu diễn được khoảng giá trị rất rộng, nhưng chỉ gần đúng.

You already know scientific notation: 12300 = 1.23 × 10^4. Computers do the same in binary. A number is split into three parts: a sign, an exponent (which says where the point sits, hence "floating") and a mantissa (the significant digits). The IEEE 754 standard uses 32 bits for a single-precision number: 1 sign bit, 8 exponent bits and 23 mantissa bits. This way 32 bits cover a very wide range of values, but only approximately.

Vì sao 0.1 + 0.2 không ra đúng 0.3 – *Why 0.1 + 0.2 is not exactly 0.3*

Trong thập phân, 1/3 = 0,333... kéo dài mãi, nên viết hữu hạn chữ số thì phải làm tròn. Trong nhị phân, 0,1 cũng lặp vô hạn: 0,0001100110011... Máy chỉ có số bit hữu hạn nên cắt bớt, và mỗi số đã lệch một chút. Cộng hai số lệch thì ra kết quả lệch nhẹ. Trong Python:

In decimal, 1/3 = 0.333... never ends, so a finite number of digits must be rounded. In binary, 0.1 also repeats forever: 0.0001100110011... The machine has a limited number of bits, so it cuts the tail, and each number is slightly off. Adding two slightly-off numbers gives a slightly-off result. In Python:

```
print(0.1 + 0.2) # 0.30000000000000004
```

Sai số này rất nhỏ nhưng có thật, và nó không phải lỗi của Python mà do cách biểu diễn. Số nguyên thì không gặp chuyện này vì chúng được lưu chính xác trong phạm vi.

The error is tiny but real, and it comes from the representation, not from Python. Integers avoid this problem because they are stored exactly within their range.

LỖI HAY GẶP / COMMON MISTAKE – Quên bước cộng 1 khi lấy bù 2, nên chỉ ra bù 1 (11111010 thay vì 11111011).
Forgetting to add 1 when taking two's complement, which gives one's complement (11111010 instead of 11111011).

LỖI HAY GẶP / COMMON MISTAKE – Giữ lại bit nhớ thứ 9 khi cộng trên 8 bit, hoặc quên rằng 8 bit có dấu chỉ chứa từ −128 đến 127.
Keeping the 9th carry bit when adding in 8 bits, or forgetting that signed 8 bits hold only −128 to 127.

LỖI HAY GẶP / COMMON MISTAKE – Dùng == để so sánh hai số thực. Hãy so sánh theo sai số nhỏ, ví dụ hàm math.isclose của Python.
Using == to compare two real numbers. Compare within a small tolerance instead, for example with Python's math.isclose function.

PHẢI THUỘC / MEMORISE – 8 bit không dấu: 0 đến 255. 8 bit có dấu bù 2: −128 đến 127. Đổi sang bù 2: viết dạng dương, đảo bit, cộng 1. Số thực lưu gần đúng gồm dấu, mũ, định trị.
Unsigned 8 bits: 0 to 255. Signed 8 bits in two's complement: −128 to 127. To get two's complement: write the positive form, flip the bits, add 1. Real numbers are stored approximately as sign, exponent and mantissa.

1.5. Biểu diễn ký tự, hình ảnh và âm thanh – *1.5. Representing Text, Images and Sound*

Máy tính chỉ lưu được số 0 và 1. Chữ, ảnh và âm thanh đều phải đổi thành số trước khi lưu. Giống như hai người thống nhất mỗi chữ cái ứng với một dãy chấm gạch khi gửi điện tín: chỉ cần chung một bảng quy ước là hiểu nhau.
A computer can only store 0s and 1s. Text, pictures and sound must all be turned into numbers before they are saved. It works like two people agreeing that each letter matches a pattern of dots and dashes in a telegram: as long as both use the same code table, they understand each other.

Ký tự: ASCII và Unicode – *Characters: ASCII and Unicode*

Bảng mã là bảng ghi mỗi ký tự ứng với một số. ASCII (American Standard Code for Information Interchange) dùng 7 bit, nên có 128 mã. Bảng này chứa chữ cái Latin không dấu, chữ số, dấu câu và một số ký tự điều khiển như xuống dòng.
A character code is a table that gives each character a number. ASCII (American Standard Code for Information Interchange) uses 7 bits, so it has 128 codes. It holds unaccented Latin letters, digits, punctuation and a few control characters such as the line break.

Vài mã nên nhớ: "A" là 65, "a" là 97, "0" là 48. Chữ thường luôn lớn hơn chữ hoa tương ứng đúng 32 đơn vị. Chữ số "0" ở đây là ký tự, khác với con số 0.
A few codes are worth remembering: "A" is 65, "a" is 97, "0" is 48. A lowercase letter is always exactly 32 more than its uppercase twin. The digit "0" here is a character, not the number zero.

VÍ DỤ / EXAMPLE – Chuỗi "Hi" gồm "H" mã 72 và "i" mã 105. Viết nhị phân 8 bit: 72 là 01001000 (64 + 8), 105 là 01101001 (64 + 32 + 8 + 1). Chuỗi này chiếm 2 byte.

> *The string "Hi" is "H" with code 72 and "i" with code 105. In 8-bit binary, 72 is 01001000 (64 + 8) and 105 is 01101001 (64 + 32 + 8 + 1). The string takes 2 bytes.*

ASCII không có "ă, â, đ, ê, ô, ơ, ư" và các dấu thanh, nên không viết được tiếng Việt chuẩn. Vì vậy ra đời Unicode: bảng mã gán cho mỗi ký tự của hầu hết chữ viết trên thế giới một số riêng, gọi là điểm mã. 128 mã đầu của Unicode trùng ASCII, nên văn bản cũ vẫn đọc được.

ASCII has no letters with extra marks, such as a with a breve, o with a horn or d with a stroke, and no tone marks, so it cannot write proper Vietnamese. Unicode was created to fix this: it gives every character of nearly all the world's writing systems its own number, called a code point. Its first 128 codes match ASCII, so old text still reads correctly.

Cách lưu phổ biến nhất là UTF-8, dùng từ 1 đến 4 byte cho mỗi ký tự. Chữ ASCII chiếm 1 byte, nên tiết kiệm chỗ. Chữ tiếng Việt có dấu chiếm 2 hoặc 3 byte.

The most common storage format is UTF-8, which uses 1 to 4 bytes per character. ASCII letters take 1 byte, which saves space. Accented Vietnamese letters take 2 or 3 bytes.

Ký tự *Character*	**Điểm mã Unicode** *Unicode code point*	**Số byte trong UTF-8** *Bytes in UTF-8*
a	U+0061	1 byte
đ (chữ đ) *đ (d with a stroke)*	U+0111	2 byte *2 bytes*
ơ (chữ ơ) *ơ (o with a horn)*	U+01A1	2 byte *2 bytes*
ế (chữ ê dấu sắc) *ế (e with circumflex and acute)*	U+1EBF	3 byte *3 bytes*

Ảnh số: điểm ảnh và màu RGB – *Digital Images: Pixels and RGB Colour*

Ảnh số là một lưới ô vuông rất nhỏ, mỗi ô là một điểm ảnh (pixel) mang đúng một màu. Nhìn xa, các ô hòa vào nhau thành hình liền mạch, như tranh ghép gạch men. Ảnh rộng 1920 điểm và cao 1080 điểm gọi là ảnh 1920 × 1080.

A digital image is a grid of tiny squares. Each square, called a pixel, holds exactly one colour. From a distance the squares blend into a smooth picture, like a mosaic of floor tiles. An image 1920 pixels wide and 1080 pixels high is called a 1920 × 1080 image.

Màu của mỗi điểm ảnh thường lưu theo RGB: ba số cho đỏ (Red), lục (Green), lam (Blue). Mỗi số từ 0 đến 255, vừa một byte. Một điểm ảnh vậy tốn 3 byte, tức 24 bit, và có $2^{24} = 16\,777\,216$ màu.

Each pixel's colour is usually stored as RGB: three numbers for Red, Green and Blue. Each runs from 0 to 255, which fits in one byte. One pixel therefore takes 3 bytes, or 24 bits, and can show 2^24 = 16,777,216 colours.

- (255, 0, 0) là đỏ tươi; (0, 0, 0) là đen; (255, 255, 255) là trắng; (255, 255, 0) là vàng vì đỏ trộn lục.

(255, 0, 0) is pure red; (0, 0, 0) is black; (255, 255, 255) is white; (255, 255, 0) is yellow, because red mixed with green gives yellow.

1. Lấy chiều rộng nhân chiều cao để ra số điểm ảnh.
   Multiply width by height to get the number of pixels.
2. Nhân số điểm ảnh với số byte mỗi điểm (RGB là 3 byte).
   Multiply the pixel count by the bytes per pixel (3 for RGB).
3. Đổi sang đơn vị lớn bằng cách chia cho 1024 mỗi bậc.
   Convert to a larger unit by dividing by 1024 per step.

> **VÍ DỤ / EXAMPLE –** Ảnh 1920 × 1080, RGB, chưa nén. Số điểm ảnh: 1920 × 1080 = 2 073 600. Dung lượng: 2 073 600 × 3 = 6 220 800 byte. Chia 1024 được 6075 KB, chia tiếp 1024 được khoảng 5,93 MB.
> *A 1920 × 1080 uncompressed RGB image. Pixels: 1920 × 1080 = 2,073,600. Size: 2,073,600 × 3 = 6,220,800 bytes. Divide by 1024 for 6075 KB, then by 1024 again for about 5.93 MB.*

Âm thanh số: lấy mẫu – *Digital Sound: Sampling*

Âm thanh ngoài đời là sóng liên tục. Để lưu, máy đo độ lớn của sóng rất nhiều lần mỗi giây, mỗi lần ghi lại một con số. Việc đo này gọi là lấy mẫu. Giống như quay phim: nhiều khung hình rời nhau, chiếu nhanh thì thấy chuyển động liền.

Real-world sound is a continuous wave. To store it, the computer measures the wave's height many times per second and writes down one number each time. This is called sampling. It is like filming: many separate frames, played fast, look like smooth motion.

Ba thông số quyết định dung lượng. Tần số lấy mẫu là số mẫu mỗi giây, đo bằng hertz (Hz); đĩa CD dùng 44 100 Hz. Độ sâu bit là số bit cho mỗi mẫu, thường 16. Số kênh là 1 với mono, 2 với stereo.

Three settings decide the size. The sampling rate is the number of samples per second, measured in hertz (Hz); a CD uses 44,100 Hz. The bit depth is the number of bits per sample, commonly 16. The channel count is 1 for mono and 2 for stereo.

1. Tính số bit mỗi giây: tần số × độ sâu bit × số kênh.
   Work out bits per second: sampling rate × bit depth × channels.
2. Nhân với số giây để ra tổng bit.
   Multiply by the number of seconds to get total bits.
3. Chia 8 để ra byte, rồi đổi đơn vị.
   Divide by 8 to get bytes, then convert units.

> **VÍ DỤ / EXAMPLE –** Ghi âm stereo 1 phút (60 giây), 44 100 Hz, 16 bit, chưa nén. Mỗi giây: 44 100 × 16 × 2 = 1 411 200 bit, tức 176 400 byte. Một phút: 176 400 × 60 = 10 584 000 byte, khoảng 10,09 MB.
> *A one-minute (60 s) stereo recording at 44,100 Hz and 16 bits, uncompressed. Per second: 44,100 × 16 × 2 = 1,411,200 bits, or 176,400 bytes. For one minute: 176,400 × 60 = 10,584,000 bytes, about 10.09 MB.*

LỖI HAY GẶP / COMMON MISTAKE – Quên nhân số kênh, nên ra nửa dung lượng của bản stereo. Hoặc quên chia 8 và lẫn bit với byte.
Forgetting the channel count, which gives half the true size of a stereo file. Or forgetting to divide by 8 and mixing up bits with bytes.

LỖI HAY GẶP / COMMON MISTAKE – Lấy kết quả này so với file JPEG hay MP3. Hai định dạng đó đã nén, nên nhỏ hơn nhiều; công thức trên chỉ đúng cho dữ liệu chưa nén.
Comparing this result with a JPEG or MP3 file. Those formats are compressed and much smaller; the formulas here only fit uncompressed data.

PHẢI THUỘC / MEMORISE – ASCII: 7 bit, 128 mã, "A" = 65, "a" = 97. Ảnh: dung lượng = rộng × cao × 3 byte. Âm thanh: dung lượng = tần số × (độ sâu bit ÷ 8) × số kênh × số giây.
ASCII: 7 bits, 128 codes, "A" = 65, "a" = 97. Image: size = width × height × 3 bytes. Sound: size = sampling rate × (bit depth ÷ 8) × channels × seconds.

1.6. Phép toán logic, bảng chân trị và cổng logic – *1.6. Logic Operations, Truth Tables and Logic Gates*

Máy tính chỉ phân biệt hai trạng thái: có điện và không có điện. Ta ghi là 1 (đúng, true) và 0 (sai, false). Phép toán logic nhận các giá trị đúng/sai và trả ra một giá trị đúng/sai. Chính nhờ các phép này mà máy so sánh, chọn đường đi và tính toán được.
A computer tells apart only two states: current or no current. We write them as 1 (true) and 0 (false). A logic operation takes true/false values and returns a true/false value. This is how a machine compares, chooses a path and calculates.

Bốn phép toán qua công tắc đèn – *Four operations through light switches*

AND (và): hai công tắc nối tiếp trên một dây. Đèn chỉ sáng khi cả hai cùng bật. Trong đề thi: "đậu môn nếu điểm giữa kỳ ≥ 5 VÀ điểm cuối kỳ ≥ 5", thiếu một vế là rớt.
AND: two switches in series on one wire. The lamp lights only when both are on. In an exam rule such as "pass if midterm ≥ 5 AND final ≥ 5", missing either condition means failing.

OR (hoặc): hai công tắc song song. Chỉ cần một cái bật là đèn sáng, cả hai bật cũng sáng. Ví dụ: "được miễn phí nếu là sinh viên HOẶC trên 60 tuổi".
OR: two switches in parallel. One switch on is enough, and both on also lights the lamp. Example: "free entry if you are a student OR over 60".

NOT (phủ định): đảo ngược giá trị. Đúng thành sai, sai thành đúng. Giống một công tắc lắp ngược: gạt lên thì tắt, gạt xuống thì sáng.
NOT: flips the value. True becomes false and false becomes true. Think of a switch mounted upside down: up means off, down means on.

XOR (hoặc loại trừ): đúng khi hai giá trị khác nhau. Ví dụ đèn cầu thang có hai công tắc, một ở tầng trệt, một ở tầng trên. Gạt bất kỳ công tắc nào cũng đổi trạng thái đèn, nên đèn sáng khi hai công tắc đang ở vị trí lệch nhau.
XOR (exclusive or): true when the two values differ. Take a staircase light with one switch downstairs and one upstairs. Flipping either one changes the lamp, so it is on when the two switches are in different positions.

Bảng chân trị (truth table) là bảng liệt kê mọi tổ hợp giá trị đầu vào và kết quả tương ứng. Hãy xem bảng cho hai biến A, B:

A truth table lists every combination of input values and the matching result. Look at the table for two variables A and B:

A	B	A AND B	A OR B	A XOR B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

NOT chỉ có một đầu vào: NOT 0 = 1 và NOT 1 = 0.

NOT has a single input: NOT 0 = 1 and NOT 1 = 0.

Lập bảng chân trị cho một biểu thức – *Building a truth table for an expression*

1. Đếm số biến n. Bảng có 2^n hàng: hai biến có 4 hàng, ba biến có 8 hàng.
   Count the variables n. The table has 2^n rows: two variables give 4 rows, three give 8.
2. Ghi các tổ hợp đầu vào theo thứ tự đếm nhị phân: 000, 001, 010, 011, 100, 101, 110, 111. Cách này giúp không sót, không trùng hàng.
   List the input combinations in binary counting order: 000, 001, 010 and so on up to 111. This prevents missing or repeated rows.
3. Thêm một cột cho mỗi phần trong ngoặc hoặc mỗi phép NOT.
   Add one column for each bracketed part and each NOT.
4. Tính từng cột từ trong ra ngoài. Thứ tự ưu tiên: NOT trước, rồi AND, rồi OR.
   Fill the columns from the inside out. Precedence: NOT first, then AND, then OR.
5. Cột cuối cùng là kết quả của cả biểu thức.
   The last column is the result of the whole expression.

> **VÍ DỤ / EXAMPLE –** Hai biến: tính A AND (NOT B). Hàng (A,B) = (0,0): NOT B = 1, kết quả 0. Hàng (0,1): NOT B = 0, kết quả 0. Hàng (1,0): NOT B = 1, kết quả 1. Hàng (1,1): NOT B = 0, kết quả 0. Biểu thức chỉ đúng khi A đúng và B sai.
> *Two variables: evaluate A AND (NOT B). Row (A,B) = (0,0): NOT B = 1, result 0. Row (0,1): NOT B = 0, result 0. Row (1,0): NOT B = 1, result 1. Row (1,1): NOT B = 0, result 0. The expression is true only when A is true and B is false.*

Ba biến: tính (A AND B) OR (NOT C). Bảng có 8 hàng:

Three variables: evaluate (A AND B) OR (NOT C). The table has 8 rows:

A	B	C	A AND B	NOT C	Kết quả *Result*
0	0	0	0	1	1
0	0	1	0	0	0

A	B	C	A AND B	NOT C	Kết quả *Result*
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	0	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	0	1

Đọc cột cuối: biểu thức đúng ở 5 hàng trên 8, sai khi C = 1 mà A, B không cùng bằng 1.

Reading the last column: the expression is true in 5 of 8 rows, and false when C = 1 unless A and B are both 1.

Từ cổng logic đến bộ xử lý – *From logic gates to the processor*

Cổng logic (logic gate) là một mạch điện nhỏ thực hiện đúng một phép logic. Bên trong là các transistor, tức công tắc điện tử cực nhỏ, đóng mở bằng tín hiệu điện. Ngoài AND, OR, NOT, XOR còn có NAND (AND rồi đảo), NOR (OR rồi đảo) và XNOR (XOR rồi đảo).

A logic gate is a small circuit that performs exactly one logic operation. Inside are transistors, tiny electronic switches opened and closed by electrical signals. Besides AND, OR, NOT and XOR there are NAND (AND then invert), NOR (OR then invert) and XNOR (XOR then invert).

Bộ xử lý ghép hàng tỉ cổng để làm mọi việc. Ví dụ cộng hai bit A và B: bit tổng = A XOR B, bit nhớ = A AND B. Cộng 1 + 1 cho tổng 0 và nhớ 1, tức kết quả 10 ở hệ nhị phân. Các lệnh and, or, xor của máy cũng chạy thẳng trên những cổng này. Ổ SSD còn dùng cổng NAND để lưu bit trong bộ nhớ flash.

A processor combines billions of gates to do all its work. To add two bits A and B: sum bit = A XOR B, carry bit = A AND B. Adding 1 + 1 gives sum 0 and carry 1, which is 10 in binary. The machine instructions and, or, xor run directly on such gates. SSDs also use NAND gates to store bits in flash memory.

Trong lập trình, các phép này viết thành and, or, not để đặt điều kiện, chẳng hạn điểm ≥ 5 and vắng ≤ 3. Với A AND B, nếu A đã sai thì máy bỏ qua B vì kết quả chắc chắn sai (đoản mạch, short circuit).

In programming these operations appear as and, or, not inside conditions, for example score ≥ 5 and absences ≤ 3. For A AND B, if A is already false the machine skips B because the result must be false (short circuit).

> **PHẢI THUỘC / MEMORISE –** AND chỉ đúng khi mọi đầu vào đúng. OR chỉ sai khi mọi đầu vào sai. XOR đúng khi hai đầu vào khác nhau. n biến cho 2^n hàng.
> *AND is true only when all inputs are true. OR is false only when all inputs are false. XOR is true when the two inputs differ. n variables give 2^n rows.*

LỖI HAY GẶP / COMMON MISTAKE – Hiểu OR theo kiểu "hoặc là" trong đời thường. Với OR logic, cả hai cùng đúng vẫn cho kết quả đúng; muốn loại trường hợp đó phải dùng XOR.
Reading OR as the everyday "either one or the other". Logical OR stays true when both inputs are true; to exclude that case you need XOR.

LỖI HAY GẶP / COMMON MISTAKE – Cho rằng NOT (A AND B) bằng (NOT A) AND (NOT B). Hai biểu thức này khác nhau: ở A=1, B=0, vế đầu bằng 1 còn vế sau bằng 0. Vế đầu thực ra bằng (NOT A) OR (NOT B).
Assuming NOT (A AND B) equals (NOT A) AND (NOT B). They differ: at A=1, B=0 the first gives 1 and the second gives 0. The first one actually equals (NOT A) OR (NOT B).

LỖI HAY GẶP / COMMON MISTAKE – Bỏ sót hàng hoặc quên ngoặc khi lập bảng. Hãy đếm đủ 2^n hàng trước khi tính và tách từng phần trong ngoặc thành cột riêng.
Skipping rows or ignoring brackets when building a table. Count the full 2^n rows before calculating and give each bracketed part its own column.

1.7. Các thế hệ máy tính và mô hình tính toán – *1.7. Generations of Computers and Models of Computation*

Cách chia "thế hệ" máy tính giống cách chia các đời điện thoại: mỗi đời đổi linh kiện chính, máy nhỏ đi, nhanh hơn và rẻ hơn. Bạn chỉ cần nhớ bốn mốc linh kiện, theo thứ tự thời gian. Mốc nào cũng là câu trắc nghiệm quen thuộc.
Generations of computers work like phone models: each one swaps the main component, and machines get smaller, faster and cheaper. You only need four hardware milestones, in time order. Each one is a familiar multiple-choice topic.

Bốn mốc linh kiện – *Four hardware milestones*

- Thế hệ 1 (khoảng 1940–1956): đèn điện tử. Đèn là ống thuỷ tinh hút chân không, đóng mở dòng điện như công tắc. Đèn to, nóng, hay cháy. ENIAC (hoàn thành 1945) dùng khoảng 18.000 đèn và chiếm cả một căn phòng.
  Generation 1 (about 1940–1956): vacuum tubes. A tube is a sealed glass bulb that switches current on and off. Tubes were big, hot and burned out often. ENIAC (finished 1945) used about 18,000 of them and filled a whole room.
- Thế hệ 2 (khoảng 1956–1963): bóng bán dẫn (transistor). Bóng bán dẫn là công tắc điện làm từ vật liệu bán dẫn, được phát minh năm 1947. Nó nhỏ, ít toả nhiệt, bền hơn đèn nhiều lần.
  Generation 2 (about 1956–1963): transistors. A transistor is an electric switch made of semiconductor material, invented in 1947. It is small, runs cooler and lasts far longer than a tube.
- Thế hệ 3 (khoảng 1964–1970): mạch tích hợp (integrated circuit, IC). Mạch tích hợp gom nhiều bóng bán dẫn lên một mảnh silicon nhỏ, ra đời cuối thập niên 1950. Giống việc nén cả bảng mạch rời rạc vào một con chip.
  Generation 3 (about 1964–1970): integrated circuits (ICs). An IC packs many transistors onto one small slice of silicon; it first appeared in the late 1950s. Think of squeezing a whole loose circuit board into a single chip.

- Thế hệ 4 (từ 1971): vi xử lý (microprocessor). Đó là toàn bộ bộ xử lý trung tâm đặt trên một chip. Chip Intel 4004 năm 1971 là vi xử lý thương mại đầu tiên, mở đường cho máy tính cá nhân.
  Generation 4 (from 1971): the microprocessor. This is an entire central processing unit on one chip. Intel's 4004 in 1971 was the first commercial microprocessor and opened the way to personal computers.

Thế hệ thứ năm thường gắn với trí tuệ nhân tạo và xử lý song song, nhưng các tài liệu chia khác nhau. Đề thi hay dừng ở bốn thế hệ trên.

A fifth generation is often linked to artificial intelligence and parallel processing, but sources divide it differently. Exams usually stop at the four generations above.

Năm 1965, Gordon Moore, người đồng sáng lập Intel, nhận xét số bóng bán dẫn trên một chip tăng gấp đôi đều đặn. Về sau nó được gọi là định luật Moore: cứ khoảng hai năm, số bóng bán dẫn trên chip lại gấp đôi. Hiểu đơn giản, cùng một số tiền, sức tính tăng gấp đôi sau mỗi hai năm.

In 1965, Gordon Moore, a co-founder of Intel, noticed that the number of transistors on a chip kept doubling at a steady pace. This became Moore's Law: about every two years, the transistor count on a chip doubles. Put simply, for the same money you get twice the computing power every two years.

Máy Turing: một máy tưởng tượng để định nghĩa "tính toán được" – *The Turing machine: an imaginary device that defines "computable"*

Năm 1936, Alan Turing mô tả một cỗ máy chỉ có trên giấy. Mục đích là trả lời câu hỏi: việc nào thì làm được bằng cách làm theo các bước máy móc? Máy có ba phần: một cuộn băng dài vô hạn chia ô để ghi ký hiệu (bộ nhớ), một đầu đọc ghi đứng trên một ô, và một bảng quy tắc.

In 1936, Alan Turing described a machine that exists only on paper. It answers a question: which tasks can be done by following mechanical steps? The machine has three parts: an endless tape divided into cells for symbols (the memory), a head that sits on one cell, and a rule table.

Mỗi bước, đầu đọc xem ô hiện tại và trạng thái của mình, tra bảng quy tắc, rồi ghi một ký hiệu, dịch sang trái hoặc phải, và đổi trạng thái. Chỉ vậy mà cỗ máy tính được mọi thứ một thuật toán tính được. Vì thế, máy Turing được dùng làm thước đo: một máy tính thật mà mô phỏng được máy Turing gọi là đầy đủ Turing (Turing-complete).

At each step, the head reads the current cell and its own state, looks up the rule table, then writes a symbol, moves left or right, and changes state. With just that, it can compute anything an algorithm can compute. So it serves as a yardstick: a real computer that can imitate a Turing machine is called Turing-complete.

Hệ quả dễ hiểu: máy tính hôm nay chỉ nhanh hơn ENIAC hàng nghìn tỉ lần, còn việc nó tính được thì cũng là việc ENIAC tính được. Luận đề Church–Turing nói các mô hình tính toán hợp lý đều mạnh ngang nhau.

An easy consequence: today's computers are trillions of times faster than ENIAC, yet what they can compute is the same as what ENIAC could compute. The Church–Turing thesis says all reasonable models of computation are equally powerful.

Máy tính dùng chương trình lưu sẵn – *The stored-program computer*

Máy thế hệ đầu như ENIAC muốn đổi việc phải cắm lại dây và gạt lại công tắc, mất cả ngày. Năm 1945, John von Neumann mô tả cách khác: cất chương trình vào bộ nhớ, ngay cạnh dữ liệu. Muốn đổi việc, chỉ cần nạp chương trình mới. Máy chạy được chương trình đầu tiên theo kiểu này vào năm 1948 tại Manchester (Anh).

Early machines like ENIAC needed rewiring and switch changes to switch jobs, which took a day. In 1945, John von Neumann described another way: keep the program in memory, right next to the data. To change jobs, just load a new program. The first machine to run a program this way did so in 1948 in Manchester, England.

Bạn thấy chỗ giống máy Turing: bộ nhớ chứa cả dữ liệu lẫn lệnh, và một bộ đếm chỉ ra lệnh nào chạy tiếp. Máy Turing là ý tưởng toán học, còn máy lưu chương trình là cách chế tạo thực tế.

You can spot the link to the Turing machine: memory holds both data and instructions, and a counter points to the next instruction. The Turing machine is a mathematical idea; the stored-program computer is how it was built in practice.

VÍ DỤ / EXAMPLE – Đề hỏi: "Máy dùng hàng chục nghìn đèn chân không, phải đổi dây để đổi việc, là thế hệ nào?" Đèn chân không ⇒ thế hệ 1. Đổi dây ⇒ chưa lưu chương trình. Đáp án: thế hệ 1, ví dụ ENIAC.

Question: "A machine uses tens of thousands of vacuum tubes and must be rewired to change jobs. Which generation?" Vacuum tubes ⇒ generation 1. Rewiring ⇒ no stored program. Answer: generation 1, for example ENIAC.

PHẢI THUỘC / MEMORISE – Thứ tự linh kiện: đèn điện tử, bóng bán dẫn, mạch tích hợp, vi xử lý. Mốc năm: 1947 bóng bán dẫn, 1965 định luật Moore, 1971 Intel 4004. Máy Turing có băng, đầu đọc và bảng quy tắc.

Component order: vacuum tube, transistor, integrated circuit, microprocessor. Key years: 1947 transistor, 1965 Moore's Law, 1971 Intel 4004. A Turing machine has a tape, a head and a rule table.

LỖI HAY GẶP / COMMON MISTAKE – Hay nhầm bóng bán dẫn (1947) với mạch tích hợp: chip là nhiều bóng bán dẫn gộp lại, nên đến sau. Cũng đừng tưởng máy Turing là một chiếc máy từng được chế tạo để bán.

A common slip is mixing up the transistor (1947) with the integrated circuit: a chip is many transistors combined, so it came later. Also, a Turing machine was never a product built for sale.

MẸO / TIP – Nhớ "Đèn, Bán dẫn, Tích hợp, Vi xử lý" theo kích thước: từ cả căn phòng thu dần về một con chip.

Remember "Tube, Transistor, IC, Microprocessor" by size: from a whole room shrinking down to a single chip.

1.8. Từ ngôn ngữ tự nhiên đến ngôn ngữ hình thức của máy – *1.8. From Natural Language to Formal Languages*

Khi bạn nhờ bạn cùng phòng "mua giúp tớ cái gì ăn sáng đi", bạn ấy tự đoán được ý bạn. Máy tính không đoán được. Tiếng Việt, tiếng Anh là ngôn ngữ tự nhiên: ngôn ngữ lớn

lên cùng con người, không ai ngồi thiết kế. Ngôn ngữ hình thức là ngôn ngữ do con người thiết kế cho một mục đích hẹp, như ký hiệu toán học, công thức hoá học hay ngôn ngữ lập trình. Mọi ngôn ngữ lập trình, kể cả Python, đều là ngôn ngữ hình thức.

When you ask a roommate to "grab something for breakfast", they work out what you mean. A computer cannot. English and Vietnamese are natural languages: they grew up with people, and nobody sat down and designed them. A formal language is one that people design for a narrow purpose, such as mathematical notation, chemical formulas, or programming languages. Every programming language, Python included, is a formal language.

Ba điểm khác nhau cần nắm – ***Three differences you need to know***

Điểm thứ nhất là tính nhập nhằng. Một câu tự nhiên có thể hiểu theo nhiều cách, và người nghe chọn nghĩa đúng nhờ hoàn cảnh, giọng nói, chuyện đã biết từ trước. Ngôn ngữ hình thức được thiết kế để mỗi câu chỉ có đúng một nghĩa. Không có hoàn cảnh nào để máy dựa vào, nên nghĩa phải nằm hết trong câu lệnh.

The first difference is ambiguity. A natural sentence can be read in several ways, and listeners pick the right one from context, tone of voice, and what they already know. A formal language is designed so that each statement has exactly one meaning. A machine has no context to lean on, so the meaning must sit entirely inside the statement.

Điểm thứ hai là tính chặt chẽ. Ngôn ngữ tự nhiên dư thừa và dễ tha thứ: bạn viết sai chính tả, thiếu dấu câu, người đọc vẫn hiểu. Thành ngữ cũng quen thuộc, như "nước đến chân mới nhảy", chẳng ai nhảy cả. Ngôn ngữ hình thức ít thừa hơn, ngắn gọn hơn, và nói đúng nghĩa đen những gì nó viết. Một dấu sai có thể đổi nghĩa cả câu lệnh.

The second difference is strictness. Natural language is redundant and forgiving: with a spelling slip or a missing comma, readers still follow you. It is also full of idioms, such as "break a leg", where nobody breaks anything. A formal language is less redundant, more concise, and means literally what it says. One wrong character can change the whole statement.

Điểm thứ ba là cú pháp, tức bộ luật quy định câu được viết thế nào. Cú pháp có hai tầng. Tầng một là token, những mẩu nhỏ nhất có nghĩa của ngôn ngữ, giống "từ" trong câu thường: tên lệnh, con số, dấu cộng, dấu ngoặc. Tầng hai là cấu trúc, tức cách xếp các token cạnh nhau. Câu "5 $ 2" sai ở tầng một vì ký hiệu $ không phải token hợp lệ trong phép tính của Python. Câu "5 + ∗ 2" sai ở tầng hai: từng token đều hợp lệ nhưng hai dấu phép tính không được đứng liền nhau.

The third difference is syntax, the rulebook for how statements are written. Syntax has two layers. The first is tokens, the smallest meaningful pieces of the language, like words in an ordinary sentence: command names, numbers, plus signs, brackets. The second is structure, the way tokens are placed next to each other. "5 $ 2" fails at the first layer, because $ is not a valid token in a Python calculation. "5 + ∗ 2" fails at the second: every token is valid, but two operators cannot sit side by side.

Tiêu chí *Criterion*	**Ngôn ngữ tự nhiên** *Natural language*	**Ngôn ngữ hình thức** *Formal language*
Nghĩa của câu *Meaning of a sentence*	Một câu có thể có nhiều nghĩa, người nghe tự chọn theo hoàn cảnh. *One sentence can carry several meanings, and the listener chooses from context.*	Một câu lệnh chỉ có một nghĩa duy nhất. *A statement has one meaning only.*
Lỗi nhỏ *Small slips*	Thường vẫn hiểu được. *Usually still understood.*	Có thể làm câu lệnh bị từ chối hoặc chạy sai. *Can make the statement rejected or run wrongly.*
Cách dùng từ *Use of words*	Có thành ngữ, ẩn dụ, nói bóng gió. *Idioms, metaphors, and hints are common.*	Chỉ có nghĩa đen, theo luật đã định. *Only literal meaning, following fixed rules.*

Một câu nói hai nghĩa, một câu lệnh một nghĩa – *One sentence, two meanings; one command, one meaning*

> **VÍ DỤ / EXAMPLE –** Câu "Cô ấy thích nấu ăn hơn chồng" có thể hiểu là cô ấy thích nấu ăn hơn là chồng cô ấy thích nấu ăn. Cũng có thể hiểu là cô ấy thích nấu ăn hơn là thích chồng. Chỉ người biết chuyện mới chọn được nghĩa đúng. Giờ xem lệnh Python print(7 - 2 * 3). Luật của Python quy định phép nhân làm trước phép trừ. Máy tính 2 * 3 = 6, rồi 7 - 6 = 1, và in ra 1. Không có cách đọc thứ hai. Nếu muốn trừ trước, bạn phải viết rõ print((7 - 2) * 3), kết quả 15.
> *The sentence "She likes cooking more than her husband" can mean she likes cooking more than her husband likes cooking. It can also mean she likes cooking more than she likes her husband. Only someone who knows the story can pick the right one. Now take the Python command print(7 - 2 * 3). Python's rules say multiplication goes before subtraction. The machine works out 2 * 3 = 6, then 7 - 6 = 1, and prints 1. There is no second reading. To subtract first, you must say so with brackets: print((7 - 2) * 3) gives 15.*

Vì sao máy cần lệnh đúng tuyệt đối – *Why a machine needs perfectly exact commands*

Bộ xử lý chỉ thực hiện những thao tác đã cài sẵn. Nó không có kinh nghiệm sống, không hỏi lại "ý bạn là gì". Trước khi chạy, chương trình viết bằng Python phải được một phần mềm dịch kiểm tra và chuyển sang dạng máy làm được. Phần mềm này cắt dòng lệnh thành token, rồi kiểm tra các token có xếp đúng cấu trúc không. Việc kiểm tra cấu trúc này gọi là phân tích cú pháp (parsing). Nếu cấu trúc sai, nó dừng và báo lỗi cú pháp. Nếu cho phép "gần đúng", máy sẽ phải đoán, và đoán sai thì tiền trong bảng lương hay điểm trong bảng điểm có thể sai theo.

A processor only carries out the operations built into it. It has no life experience and cannot ask "what do you mean?". Before a Python program runs, a translator program checks it and converts it into a form the machine can carry out. It cuts each line into tokens, then checks whether the tokens are arranged in a valid structure. That structure check is called parsing. If

the structure is wrong, it stops and reports a syntax error. If "close enough" were allowed, the machine would have to guess, and a wrong guess could corrupt a payroll or a grade sheet.

Cách đọc một dòng lệnh như máy đọc – ***How to read a command the way a machine does***

1. Tách dòng lệnh thành các token. Với print("Xin chao"), ta có bốn token: print, dấu (, chuỗi "Xin chao", dấu).
   Split the line into tokens. In print("Xin chao"), there are four: print, the symbol (, the text "Xin chao", and the symbol).
2. Kiểm tra từng token có hợp lệ không: tên lệnh viết đúng, dấu nháy đóng mở đủ cặp, không có ký hiệu lạ.
   Check that each token is valid: the command name is spelled right, quotation marks come in pairs, and no strange symbols appear.
3. Kiểm tra cấu trúc: tên lệnh đứng trước, ngoặc mở, nội dung, ngoặc đóng. Thiếu một dấu ngoặc là sai cấu trúc.
   Check the structure: command name first, then opening bracket, content, closing bracket. A missing bracket breaks the structure.
4. Chỉ khi hai kiểm tra trên đạt, bạn mới hỏi đến nghĩa: lệnh này làm gì. Ở ví dụ trên, nó in dòng chữ Xin chao ra màn hình.
   Only when both checks pass do you ask about meaning: what does the command do? Here, it prints the words Xin chao on the screen.

LỖI HAY GẶP / COMMON MISTAKE – Viết Print thay vì print, vì Python phân biệt chữ hoa và chữ thường. Dùng dấu nháy cong “ ” copy từ Word thay cho nháy thẳng " ". Quên đóng ngoặc hoặc đóng nháy. Cả ba lỗi đều là lỗi cú pháp, và người đọc thường nhìn mà không thấy.
Writing Print instead of print, because Python tells capital and small letters apart. Pasting curly quotes “ ” from a word processor instead of straight quotes " ". Forgetting to close a bracket or a quotation mark. All three are syntax errors, and the eye often misses them.

GHI CHÚ / NOTE – Lỗi trong chương trình gọi là bug, việc tìm và sửa lỗi gọi là debugging (gỡ lỗi). Bạn sẽ gặp lỗi liên tục, kể cả khi đã nhiều kinh nghiệm. Đọc thông báo lỗi chậm rãi là cách nhanh nhất để sửa.
A mistake in a program is called a bug, and finding and fixing it is called debugging. You will meet bugs constantly, even with plenty of experience. Reading the error message slowly is the fastest way to fix it.

PHẢI THUỘC / MEMORISE – Ngôn ngữ hình thức khác ngôn ngữ tự nhiên ở ba điểm: ít nhập nhằng (mỗi câu một nghĩa), chặt chẽ (nghĩa đen, ít thừa), và có cú pháp rõ gồm token và cấu trúc. Máy cần lệnh đúng tuyệt đối vì nó không có hoàn cảnh để đoán ý.
A formal language differs from natural language in three ways: little ambiguity (one meaning per statement), strictness (literal meaning, little redundancy), and clear syntax made of tokens and structure. A machine needs exact commands because it has no context to guess your intent.

1.9. Bài tập vận dụng có lời giải – *1.9. Applied exercises with solutions*

Bài 1. Đổi số 173 sang nhị phân và thập lục phân – *Exercise 1. Converting 173 to binary and hexadecimal*

Đổi số thập phân 173 sang nhị phân và thập lục phân, rồi cho biết cần tối thiểu bao nhiêu bit để lưu số này.

Convert the decimal number 173 to binary and hexadecimal, then state the minimum number of bits needed to store it.

1. Chia 173 liên tiếp cho 2 và ghi số dư: 173 chia 2 được 86 dư 1; 86 chia 2 được 43 dư 0; 43 chia 2 được 21 dư 1; 21 chia 2 được 10 dư 1.
   Divide 173 by 2 repeatedly and write down the remainders: 173 gives 86 remainder 1; 86 gives 43 remainder 0; 43 gives 21 remainder 1; 21 gives 10 remainder 1.
2. Tiếp tục: 10 chia 2 được 5 dư 0; 5 chia 2 được 2 dư 1; 2 chia 2 được 1 dư 0; 1 chia 2 được 0 dư 1.
   Continue: 10 gives 5 remainder 0; 5 gives 2 remainder 1; 2 gives 1 remainder 0; 1 gives 0 remainder 1.
3. Đọc các số dư từ dưới lên trên được 10101101. Đây là dạng nhị phân của 173.
   Read the remainders from bottom to top to get 10101101. This is 173 in binary.
4. Tách từ phải sang trái thành các nhóm 4 bit: 1010 và 1101. Nhóm 1010 là 10, viết là A. Nhóm 1101 là 13, viết là D. Vậy dạng thập lục phân là AD.
   Split from the right into 4-bit groups: 1010 and 1101. The group 1010 is 10, written A. The group 1101 is 13, written D. So the hexadecimal form is AD.
5. Kiểm tra: 10 nhân 16 cộng 13 bằng 160 cộng 13 bằng 173, khớp với số ban đầu.
   Check: 10 times 16 plus 13 is 160 plus 13, which is 173, matching the original number.
6. Vì $128 \le 173 < 256$ nên số này có đúng 8 chữ số nhị phân. Đáp số: 173 = 10101101 (nhị phân) = AD (thập lục phân), cần tối thiểu 8 bit, tức 1 byte.
   Since $128 \le 173 < 256$, the number has exactly 8 binary digits. Answer: 173 = 10101101 in binary = AD in hexadecimal, needing at least 8 bits, which is 1 byte.

Bài 2. Cộng –45 và 20 bằng bù hai 8 bit – *Exercise 2. Adding –45 and 20 with 8-bit two's complement*

Dùng số nguyên có dấu 8 bit theo cách bù hai. Viết –45 và 20, cộng chúng lại, rồi đọc kết quả thành số thập phân.

Use 8-bit signed integers in two's complement. Write –45 and 20, add them, and read the result as a decimal number.

1. Viết 45 dạng nhị phân 8 bit: 45 = 32 + 8 + 4 + 1 nên là 00101101.
   Write 45 in 8-bit binary: 45 = 32 + 8 + 4 + 1, so it is 00101101.
2. Tìm –45: đảo từng bit của 00101101 được 11010010, rồi cộng 1 được 11010011.
   Find –45: flip every bit of 00101101 to get 11010010, then add 1 to get 11010011.
3. Viết 20 dạng 8 bit: 16 + 4 nên là 00010100.
   Write 20 in 8 bits: 16 + 4, so it is 00010100.

4. Cộng hai dãy bit: 11010011 + 00010100 = 11100111, không có nhớ ra ngoài bit thứ 8.
   Add the two bit strings: 11010011 + 00010100 = 11100111, with no carry out of the 8th bit.
5. Bit đầu là 1 nên kết quả âm. Đảo bit của 11100111 được 00011000, cộng 1 được 00011001, bằng 25. Đáp số: 11100111 biểu diễn –25, đúng với –45 + 20 = –25.
   The first bit is 1, so the result is negative. Flip the bits of 11100111 to get 00011000, add 1 to get 00011001, which is 25. Answer: 11100111 represents –25, matching –45 + 20 = –25.

Bài 3. Bảng chân trị của F = (A AND B) OR (NOT C) – ***Exercise 3. Truth table of F = (A AND B) OR (NOT C)***

Lập bảng chân trị cho F = (A AND B) OR (NOT C) với ba biến A, B, C, rồi đếm có bao nhiêu dòng cho F bằng 1.
Build the truth table of F = (A AND B) OR (NOT C) for the three inputs A, B, C, then count how many rows give F = 1.

1. Ba biến có 2^3 = 8 tổ hợp đầu vào. Liệt kê theo thứ tự ABC từ 000 đến 111.
   Three inputs give 2^3 = 8 input combinations. List them in ABC order from 000 to 111.
2. Với C = 0 thì NOT C = 1 nên F = 1 bất kể A, B. Các dòng 000, 010, 100, 110 đều cho F = 1.
   When C = 0, NOT C = 1, so F = 1 whatever A and B are. Rows 000, 010, 100 and 110 all give F = 1.
3. Với C = 1 thì NOT C = 0 nên F chỉ còn bằng A AND B. Dòng 001: 0 AND 0 = 0. Dòng 011: 0 AND 1 = 0. Dòng 101: 1 AND 0 = 0. Dòng 111: 1 AND 1 = 1.
   When C = 1, NOT C = 0, so F equals A AND B. Row 001: 0 AND 0 = 0. Row 011: 0 AND 1 = 0. Row 101: 1 AND 0 = 0. Row 111: 1 AND 1 = 1.
4. Đáp số: F bằng 1 ở 5 dòng (000, 010, 100, 110, 111) và bằng 0 ở 3 dòng (001, 011, 101).
   Answer: F equals 1 in 5 rows (000, 010, 100, 110, 111) and 0 in 3 rows (001, 011, 101).

1.10. Câu hỏi tự luận và dàn ý trả lời – *1.10. Essay questions and answer outlines*

Câu 1. Phân biệt dữ liệu với thông tin, và giải thích bit, byte cùng cách đổi giữa các đơn vị đo.

Question 1. *Distinguish data from information, and explain the bit, the byte and how to convert between units.*

- Dữ liệu là các giá trị thô chưa có ý nghĩa riêng, như con số 36,5; thông tin là dữ liệu đã được đặt vào ngữ cảnh, như "nhiệt độ cơ thể 36,5 độ C là bình thường".
  Data are raw values without meaning of their own, such as the number 36.5; information is data placed in context, such as "a body temperature of 36.5 °C is normal".
- Bit là một chữ số nhị phân, chỉ nhận 0 hoặc 1; 1 byte gồm 8 bit và đủ để phân biệt 2^8 = 256 giá trị.
  A bit is one binary digit, either 0 or 1; 1 byte is 8 bits and can tell apart 2^8 = 256 values.
- Các đơn vị lớn hơn: 1 KB = 1024 byte, 1 MB = 1024 KB, 1 GB = 1024 MB; đổi lên thì chia, đổi xuống thì nhân.

Larger units: 1 KB = 1024 bytes, 1 MB = 1024 KB, 1 GB = 1024 MB; divide to go up a unit, multiply to go down.

- Ví dụ ngắn: tệp 3 MB có 3 × 1024 = 3072 KB.
  Short example: a 3 MB file is 3 × 1024 = 3072 KB.

Câu 2. Vì sao máy tính dùng hệ nhị phân, và hệ thập lục phân giúp ích gì cho con người?

Question 2. *Why do computers use the binary system, and how does hexadecimal help people?*

- Linh kiện điện tử dễ phân biệt hai trạng thái như có điện và không có điện, nên mỗi trạng thái ứng với một bit 1 hoặc 0.
  Electronic parts easily tell two states apart, such as current and no current, so each state maps to a bit, 1 or 0.
- Dãy nhị phân dài rất khó đọc và dễ chép sai; mỗi chữ số thập lục phân thay đúng 4 bit nên ngắn gọn gấp bốn lần.
  Long binary strings are hard to read and easy to copy wrongly; each hexadecimal digit stands for exactly 4 bits, so the text is four times shorter.
- Hệ thập lục phân dùng 16 ký hiệu: các chữ số 0 đến 9 và các chữ A đến F, với A là 10 và F là 15.
  Hexadecimal uses 16 symbols: digits 0 to 9 and letters A to F, where A is 10 and F is 15.
- Nêu cách đổi: nhị phân sang thập lục phân bằng cách nhóm 4 bit từ phải; thập phân sang nhị phân bằng cách chia 2 lấy số dư.
  State the methods: binary to hexadecimal by grouping 4 bits from the right; decimal to binary by dividing by 2 and keeping the remainders.

Câu 3. So sánh cách máy lưu số nguyên có dấu với số thực.

Question 3. *Compare how a machine stores signed integers and real numbers.*

- Số nguyên không dấu chỉ biểu diễn 0 và số dương; n bit cho phạm vi từ 0 đến 2^n – 1.
  Unsigned integers represent 0 and positive values only; n bits cover 0 to 2^n – 1.
- Số nguyên có dấu dùng bù hai, trong đó bit đầu cho biết dấu; n bit cho phạm vi từ –2^(n–1) đến 2^(n–1) – 1, ví dụ 8 bit là –128 đến 127.
  Signed integers use two's complement, where the first bit shows the sign; n bits cover –2^(n–1) to 2^(n–1) – 1, for example –128 to 127 for 8 bits.
- Số thực như 3,14 hay –1,25 phải lưu dạng dấu phẩy động, gồm phần dấu, phần mũ và phần định trị.
  Real numbers such as 3.14 or –1.25 must be stored in floating point, made of a sign, an exponent and a mantissa.
- Phần cứng xử lý dấu phẩy động phức tạp hơn nên chậm hơn; vì vậy chỉ dùng số thực khi cần thiết.
  Floating-point hardware is more complicated and slower, so real numbers are used only when needed.

Câu 4. Máy tính biểu diễn ký tự, hình ảnh và âm thanh bằng bit như thế nào?

Question 4. *How does a computer represent characters, images and sound with bits?*

- Ký tự: mỗi ký tự có một mã nhị phân riêng, ví dụ trong bảng ASCII; chữ hoa và chữ thường có mã khác nhau.
  Characters: each character has its own binary code, for example in the ASCII table; capital and lowercase letters have different codes.
- Hình ảnh: chia thành lưới điểm ảnh, mỗi điểm lưu một giá trị màu; càng nhiều điểm và càng nhiều bit màu thì tệp càng nặng.
  Images: split into a grid of pixels, each storing a colour value; more pixels and more colour bits make a bigger file.
- Âm thanh: đo biên độ sóng âm đều đặn theo thời gian (lấy mẫu), mỗi lần đo lưu thành một số nhị phân.
  Sound: measure the sound wave's height at regular time steps (sampling) and store each reading as a binary number.
- Cách tính dung lượng: lấy số điểm ảnh nhân số bit mỗi điểm, hoặc số mẫu mỗi giây nhân số bit mỗi mẫu nhân số giây.
  Size calculation: pixels times bits per pixel, or samples per second times bits per sample times seconds.

Câu 5. Mô tả đường đi từ ngôn ngữ tự nhiên đến chương trình mà máy chạy được.

Question 5. *Describe the path from natural language to a program the machine can run.*

- Ngôn ngữ tự nhiên như tiếng Việt dễ hiểu với người nhưng có thể mơ hồ, nên máy không xử lý trực tiếp được.
  Natural language such as Vietnamese is easy for people but can be ambiguous, so a machine cannot process it directly.
- Ngôn ngữ hình thức có quy tắc chặt chẽ, mỗi câu lệnh chỉ có một nghĩa; hợp ngữ là ngôn ngữ cấp thấp gần với máy.
  Formal languages have strict rules and each statement has one meaning; assembly language is a low-level language close to the machine.
- Trình hợp dịch (assembler) chuyển chương trình hợp ngữ thành mã máy gồm các bit 0 và 1.
  The assembler turns an assembly program into machine code made of 0s and 1s.
- Bộ xử lý đọc từng lệnh mã máy; bộ đếm chương trình (program counter) cho biết lệnh nào được thực hiện tiếp theo.
  The processor reads machine instructions one by one; the program counter shows which instruction runs next.

1.11. Câu hỏi trắc nghiệm – *1.11. Multiple-choice questions*

1. 1 KB theo cách tính nhị phân bằng bao nhiêu byte?
Using binary counting, how many bytes are in 1 KB?

A. 8 byte / *8 bytes*
B. 1024 byte / *1024 bytes*
C. 1 048 576 byte / *1,048,576 bytes*
D. 1000 byte / *1000 bytes*

2. Số thập lục phân 3F bằng số thập phân nào?
Which decimal number equals the hexadecimal number 3F?

A. 54
B. 63
C. 45
D. 48

3. Một số nguyên không dấu 8 bit biểu diễn được các giá trị trong phạm vi nào?
Which range of values can an 8-bit unsigned integer represent?

A. Từ 0 đến 255 / *From 0 to 255*
B. Từ 1 đến 255 / *From 1 to 255*
C. Từ –128 đến 127 / *From –128 to 127*
D. Từ 0 đến 256 / *From 0 to 256*

4. Trong bù hai 8 bit, số –5 được viết thế nào?
In 8-bit two's complement, how is –5 written?

A. 10000101
B. 01111011
C. 11111011
D. 11111010

5. Phát biểu nào về mã ASCII của chữ A hoa và chữ a thường là đúng?
Which statement about the ASCII codes of capital A and lowercase a is correct?

A. Hai chữ dùng chung một mã vì cùng là chữ cái A / *Both letters share one code because they are the same letter*
B. Chữ hoa không có mã riêng và được suy ra từ chữ thường / *Capital letters have no code of their own and are derived from lowercase*
C. Mã của chữ thường luôn nhỏ hơn mã của chữ hoa / *The code of a lowercase letter is always smaller than that of a capital*
D. Mỗi chữ có một mã nhị phân riêng nên hai mã khác nhau / *Each letter has its own binary code, so the two codes differ*

6. Cổng logic nào cho ra 1 chỉ khi cả hai đầu vào đều bằng 1?
Which logic gate outputs 1 only when both inputs are 1?

A. AND
B. OR
C. XOR
D. NOT

7. Máy tính thế hệ thứ hai dùng linh kiện chính nào?
Which main component did second-generation computers use?

A. Bộ vi xử lý / *Microprocessors*
B. Mạch tích hợp / *Integrated circuits*
C. Đèn điện tử chân không / *Vacuum tubes*
D. Bóng bán dẫn (transistor) / *Transistors*

8. Trình hợp dịch (assembler) làm việc gì?
What does an assembler do?

A. Thực hiện trực tiếp các phép tính số học trong bộ xử lý / *Performs arithmetic directly inside the processor*
B. Quản lý việc cấp phát bộ nhớ cho các chương trình / *Manages memory allocation for programs*
C. Chuyển chương trình hợp ngữ thành mã máy / *Translates an assembly program into machine code*
D. Hiểu câu tiếng Việt thường ngày và chạy thành lệnh / *Understands everyday Vietnamese sentences and runs them as commands*

Chương 2. Phần cứng: bên trong một chiếc máy tính – *Chapter 2. Hardware: Inside a Computer*

Mục tiêu chương – *Chapter objectives*

- Vẽ và mô tả được chu trình nạp, giải mã, thực thi lệnh của CPU, tính được xung nhịp và thời gian chạy của một chương trình đơn giản
  Describe the fetch, decode, execute cycle of a CPU and calculate clock speed and run time for a simple program
- Phân biệt được RAM, ROM, bộ nhớ đệm, HDD, SSD, USB và lưu trữ đám mây, rồi chọn đúng loại cho từng việc cụ thể
  Tell RAM, ROM, cache, HDD, SSD, USB and cloud storage apart, and pick the right one for a given job
- Đọc bảng cấu hình máy và chọn cấu hình đủ dùng cho nhu cầu học tập, có tính dung lượng RAM và thời gian sao chép dữ liệu
  Read a computer spec sheet and choose a configuration that fits study needs, including RAM capacity and file copy time

2.1. Mô hình Von Neumann và chu trình nạp, giải mã, thực thi lệnh – *2.1. The Von Neumann Model and the Fetch–Decode–Execute Cycle*

Gần như mọi máy tính hiện nay đều theo một cách tổ chức chung, gọi là mô hình Von Neumann (đặt theo tên nhà toán học đề xuất nó). Hãy hình dung một nhà bếp nhỏ: đầu bếp nấu, tủ và bàn chứa công thức lẫn nguyên liệu, cửa nhận đơn, cửa trả món và chiếc xe đẩy chở đồ qua lại. Máy tính cũng có đủ năm phần như vậy.

Almost every computer today follows one shared layout, called the Von Neumann model (named after the mathematician who proposed it). Picture a small kitchen: a cook, a cupboard and table holding recipes and ingredients, an order window, a serving window, and a trolley that carries things around. A computer has the same five parts.

Năm khối của máy tính – *The five blocks*

- Bộ xử lý (CPU, central processing unit) là đầu bếp: đọc lệnh và làm theo. Bên trong có đơn vị số học và logic (ALU) để cộng, trừ, so sánh; các thanh ghi là những ô nhớ cực nhanh nằm ngay trong CPU; và khối điều khiển để hiểu lệnh. Hai thanh ghi đáng nhớ: bộ đếm chương trình (PC) giữ địa chỉ lệnh kế tiếp, thanh ghi lệnh giữ lệnh đang xử lý.
  The processor (CPU, central processing unit) is the cook: it reads instructions and carries them out. Inside are the arithmetic logic unit (ALU) for adding, subtracting and comparing; registers, tiny very fast storage cells inside the CPU; and a control unit that understands instructions. Two registers matter here: the program counter (PC) holds the address of the next instruction, and the instruction register holds the one being processed.
- Bộ nhớ trong là chiếc tủ có đánh số ngăn. Mỗi ngăn có một địa chỉ (số thứ tự) và chứa một giá trị. CPU muốn lấy hay cất gì đều phải nói rõ địa chỉ.
  Main memory is a cupboard with numbered shelves. Each shelf has an address (its number) and holds one value. To fetch or store anything, the CPU must name the address.

- Thiết bị vào là cửa nhận đơn: bàn phím, chuột, micro đưa dữ liệu từ người dùng vào máy.
  Input devices are the order window: keyboard, mouse and microphone bring data from the user into the machine.
- Thiết bị ra là cửa trả món: màn hình, loa, máy in đưa kết quả ra cho người dùng.
  Output devices are the serving window: screen, speakers and printer deliver results to the user.
- Đường truyền (bus) là chiếc xe đẩy: các dây dẫn chở tín hiệu giữa các khối. Có ba loại: bus địa chỉ chở địa chỉ ngăn nhớ và chỉ đi một chiều từ CPU, bus dữ liệu chở giá trị và đi hai chiều, bus điều khiển chở tín hiệu như "đọc" hay "ghi".
  The bus is the trolley: wires that carry signals between blocks. There are three kinds: the address bus carries memory addresses and runs one way, from the CPU; the data bus carries values and runs both ways; the control bus carries signals such as "read" or "write".

Chu trình nạp, giải mã, thực thi – *The fetch–decode–execute cycle*

CPU làm việc bằng cách lặp đi lặp lại một chu trình cho từng lệnh, từ lúc bật máy đến khi tắt. Mỗi vòng gồm các bước sau.
The CPU works by repeating one cycle for each instruction, from power-on until shutdown. Each round has these steps.

1. Nạp (fetch): CPU đọc PC để biết địa chỉ lệnh, gửi địa chỉ đó qua bus địa chỉ, bộ nhớ trả lệnh về qua bus dữ liệu và lệnh được đặt vào thanh ghi lệnh.
   Fetch: the CPU reads the PC for the instruction's address, sends it over the address bus, memory returns the instruction over the data bus, and it lands in the instruction register.
2. Tăng PC lên để trỏ tới lệnh kế tiếp.
   Advance the PC so it points to the next instruction.
3. Giải mã (decode): khối điều khiển xem lệnh thuộc loại gì (cộng, nạp, cất…) và cần dữ liệu ở đâu.
   Decode: the control unit works out what kind of instruction it is (add, load, store…) and where its data lives.
4. Lấy toán hạng (operand, giá trị mà lệnh cần dùng), nếu có, từ bộ nhớ hoặc thanh ghi.
   Get the operand (the value the instruction works on), if there is one, from memory or a register.
5. Thực thi (execute): ALU làm phép tính hoặc CPU làm việc lệnh yêu cầu.
   Execute: the ALU does the calculation or the CPU does whatever the instruction asks.
6. Ghi kết quả vào thanh ghi hoặc bộ nhớ, rồi quay lại bước 1.
   Write the result to a register or memory, then return to step 1.

> **VÍ DỤ / EXAMPLE –** Máy giả định đơn giản có ba lệnh, nằm ở địa chỉ 0, 1, 2: LOAD 10 (nạp giá trị ở địa chỉ 10 vào thanh ghi A), ADD 11 (cộng giá trị ở địa chỉ 11 vào A), STORE 12 (cất A vào địa chỉ 12). Ngăn 10 chứa 7, ngăn 11 chứa 5. Vòng 1: PC = 0, nạp LOAD 10, PC thành 1, giải mã, đọc ngăn 10 được 7, A = 7. Vòng 2: PC = 1, nạp ADD 11, PC thành 2, đọc ngăn 11 được 5, ALU tính 7 + 5 = 12, A = 12. Vòng 3: PC = 2, nạp STORE 12, PC thành 3, ghi 12 vào ngăn 12. Kết quả cuối: ngăn 12 chứa 12.

A simple made-up machine has three instructions at addresses 0, 1, 2: LOAD 10 (load the value at address 10 into register A), ADD 11 (add the value at address 11 to A), STORE 12 (save A to address 12). Shelf 10 holds 7, shelf 11 holds 5. Round 1: PC = 0, fetch LOAD 10, PC becomes 1, decode, read shelf 10 to get 7, A = 7. Round 2: PC = 1, fetch ADD 11, PC becomes 2, read shelf 11 to get 5, the ALU computes 7 + 5 = 12, A = 12. Round 3: PC = 2, fetch STORE 12, PC becomes 3, write 12 into shelf 12. Final result: shelf 12 holds 12.

Vì sao chương trình và dữ liệu cùng nằm trong bộ nhớ – ***Why programs and data share memory***

Lệnh cũng chỉ là dãy bit như số liệu, nên có thể cất chung một nơi. Trong ví dụ trên, lệnh ở ngăn 0 đến 2 còn số liệu ở ngăn 10 đến 12, cùng một chiếc tủ. Nhờ cách này, muốn máy làm việc khác bạn chỉ cần nạp chương trình khác vào bộ nhớ, không phải đấu lại dây. Trình duyệt, trò chơi và bảng tính chạy trên cùng một máy chính vì vậy. Chương trình còn có thể tạo ra hay sửa chương trình khác, như trình biên dịch.

Instructions are just bit patterns like any other data, so they can be stored in one place. In the example, instructions sit in shelves 0 to 2 and data in shelves 10 to 12, all in the same cupboard. This means that to make the machine do a different job you load a different program into memory instead of rewiring it. That is why a browser, a game and a spreadsheet can run on one machine. A program can also create or modify other programs, as a compiler does.

Cái giá phải trả: lệnh và dữ liệu cùng đi qua một đường truyền nên CPU dễ phải chờ bộ nhớ. Ngoài ra, nếu PC trỏ nhầm vào vùng dữ liệu, CPU vẫn cố hiểu số liệu như một lệnh.

The cost: instructions and data travel over the same bus, so the CPU often has to wait for memory. Also, if the PC points into a data area by mistake, the CPU will still try to read that data as an instruction.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm PC chứa lệnh. PC chỉ chứa địa chỉ của lệnh kế tiếp.
Thinking the PC holds the instruction. It holds only the address of the next instruction.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm bộ nhớ trong với ổ cứng. Chương trình phải được nạp vào bộ nhớ trong thì CPU mới chạy được.
Confusing main memory with the hard drive. A program must be loaded into main memory before the CPU can run it.

LỖI HAY GẶP / COMMON MISTAKE – Cho rằng ALU điều khiển cả máy. ALU chỉ tính toán, còn việc hiểu lệnh thuộc khối điều khiển.
Believing the ALU runs the whole machine. The ALU only calculates; understanding instructions is the control unit's job.

PHẢI THUỘC / MEMORISE – Năm khối: bộ xử lý, bộ nhớ, thiết bị vào, thiết bị ra, đường truyền. Chu trình: nạp, giải mã, (lấy toán hạng), thực thi, ghi kết quả. Chương trình và dữ liệu cùng nằm trong bộ nhớ dưới dạng bit.
Five blocks: processor, memory, input, output, bus. Cycle: fetch, decode, (get operand), execute, write back. Programs and data share memory as bits.

2.2. Bộ xử lý trung tâm: thành phần, xung nhịp và số nhân – *2.2. The CPU: Components, Clock Speed and Cores*

CPU (Central Processing Unit, bộ xử lý trung tâm) là con chip làm nhiệm vụ "bộ não" của máy. Nó nhận lệnh từ phần mềm, thực hiện rồi trả kết quả. Hãy hình dung một bếp trưởng nhận phiếu gọi món, nấu và đưa món ra. Hai hãng làm CPU cho máy tính cá nhân phổ biến là Intel và AMD.

The CPU (Central Processing Unit) is the chip that acts as the computer's brain. It takes instructions from software, carries them out and returns results. Picture a head chef who reads an order slip, cooks the dish and sends it out. Intel and AMD are the two best-known makers of CPUs for personal computers.

Ba thành phần bên trong CPU – *Three parts inside a CPU*

Khối điều khiển (control unit) giống bếp trưởng điều phối. Nó nạp từng lệnh từ bộ nhớ, đọc xem lệnh đòi làm gì, rồi ra hiệu cho các bộ phận khác hành động. Nó không tự tính toán.

The control unit works like the chef who directs the kitchen. It fetches each instruction from memory, reads what the instruction asks for, then signals other parts to act. It does no calculating itself.

Khối số học logic (ALU, Arithmetic Logic Unit) là nơi tính toán thật sự. Nó cộng, trừ, nhân, chia và so sánh, ví dụ kiểm tra 5 có lớn hơn 3 không. Bạn có thể coi nó như chiếc máy tính bỏ túi gắn trong CPU.

The ALU (Arithmetic Logic Unit) is where the actual computing happens. It adds, subtracts, multiplies, divides and compares, for example checking whether 5 is greater than 3. Think of it as a pocket calculator built into the CPU.

Thanh ghi (register) là những ô nhớ cực nhỏ nhưng cực nhanh nằm ngay trong CPU. Chúng giống tờ giấy nháp đặt cạnh tay: ghi tạm số đang tính, địa chỉ lệnh kế tiếp, kết quả vừa ra. Kết quả của ALU thường được cất vào thanh ghi hoặc gửi lại bộ nhớ trong.

Registers are tiny, extremely fast memory cells inside the CPU. They are like scratch paper next to your hand: they hold the number being worked on, the address of the next instruction, the result just produced. The ALU's output usually goes into a register or back to main memory.

Xung nhịp, số nhân và số luồng – *Clock speed, cores and threads*

Xung nhịp là nhịp "tích tắc" điều khiển CPU làm việc, đo bằng hertz (Hz), một Hz là một chu kỳ mỗi giây. Một gigahertz (GHz) là một tỷ chu kỳ mỗi giây. CPU 3,2 GHz tích tắc 3,2 tỷ lần trong một giây.

Clock speed is the "tick" that paces the CPU, measured in hertz (Hz), where one hertz is one cycle per second. One gigahertz (GHz) is one billion cycles per second. A 3.2 GHz CPU ticks 3.2 billion times each second.

Nhân (core) là một bộ xử lý hoàn chỉnh nằm trong chip. Chip hai nhân (dual-core) hay bốn nhân (quad-core) làm được nhiều việc cùng lúc, như bếp có nhiều đầu bếp. Luồng (thread) là một dòng công việc mà nhân đang xử lý. Một nhân có thể đảm nhận hai luồng, nên số luồng thường bằng hoặc gấp đôi số nhân.

A core is a complete processor inside the chip. A dual-core or quad-core chip can do several jobs at once, like a kitchen with several cooks. A thread is one stream of work a core is handling. One core can handle two threads, so the thread count is usually equal to or double the core count.

Cách đọc thông số một bộ xử lý – *How to read a processor's specs*

Giả sử có chip tên Vinta X5-6400 với thông số: 3,2 GHz, 6 nhân, 12 luồng, bộ nhớ đệm 16 MB.

Suppose a chip named Vinta X5-6400 lists: 3.2 GHz, 6 cores, 12 threads, 16 MB cache.

1. Tìm số GHz: 3,2 GHz nghĩa là mỗi nhân tích tắc 3,2 tỷ lần mỗi giây.
   Find the GHz figure: 3.2 GHz means each core ticks 3.2 billion times per second.
2. Tìm số nhân: 6 nhân nghĩa là sáu bộ xử lý chạy song song.
   Find the core count: 6 cores means six processors working in parallel.
3. Tìm số luồng: 12 luồng chia cho 6 nhân bằng 2, tức mỗi nhân xử lý hai luồng.
   Find the thread count: 12 threads divided by 6 cores is 2, so each core handles two threads.
4. Xem bộ nhớ đệm (cache): đây là vùng nhớ nhanh gần CPU, chứa dữ liệu hay dùng để CPU đỡ phải chờ RAM. 16 MB là dung lượng của nó.
   Check the cache: a fast memory area close to the CPU that holds frequently used data so the CPU waits less for RAM. 16 MB is its size.

Vì sao xung nhịp cao chưa chắc nhanh hơn – *Why a higher clock speed does not always mean a faster machine*

> **VÍ DỤ / EXAMPLE –** Máy A có chip 4,0 GHz, 2 nhân. Máy B có chip 3,0 GHz, 6 nhân. Khi chạy phần mềm biết chia việc cho nhiều nhân, tổng nhịp của A là 2 × 4,0 = 8 tỷ chu kỳ mỗi giây, của B là 6 × 3,0 = 18 tỷ. B có sức làm việc lớn hơn gấp 2,25 lần (18 : 8). Nhưng với phần mềm chỉ dùng một nhân, A (4,0 GHz) lại nhanh hơn B (3,0 GHz).
> *Machine A has a 4.0 GHz, 2-core chip. Machine B has a 3.0 GHz, 6-core chip. With software that splits work across cores, A's total is 2 × 4.0 = 8 billion cycles per second and B's is 6 × 3.0 = 18 billion. B has 2.25 times the capacity (18 : 8). But with software that uses a single core, A (4.0 GHz) beats B (3.0 GHz).*

Ngoài số nhân, còn ba lý do nữa. Thứ nhất, mỗi chu kỳ chip đời mới có thể xử lý nhiều việc hơn chip đời cũ. Thứ hai, bộ nhớ đệm và RAM chậm sẽ làm CPU phải ngồi chờ. Thứ ba, chip nóng quá sẽ tự hạ xung để bảo vệ mình.

Three more reasons apply beyond core count. First, a newer chip can get more done per cycle than an older one. Second, slow cache or RAM leaves the CPU waiting. Third, a chip that overheats lowers its own speed to protect itself.

> **LỖI HAY GẶP / COMMON MISTAKE –** Chỉ nhìn số GHz để chọn máy, hoặc cộng thẳng số nhân với số GHz rồi coi đó là tốc độ thật. Phép nhân ở ví dụ trên chỉ là ước lượng thô, và chỉ đúng khi phần mềm dùng được nhiều nhân.
> *Choosing a machine by GHz alone, or multiplying cores by GHz and treating the result as real speed. The product in the example is only a rough estimate, and holds only when software can use many cores.*

LỖI HAY GẶP / COMMON MISTAKE – Nhầm nhân với luồng. Chip 6 nhân 12 luồng không có 12 bộ xử lý đầy đủ, chỉ có 6.
Confusing cores with threads. A 6-core, 12-thread chip does not have 12 full processors, only 6.

PHẢI THUỘC / MEMORISE – CPU gồm khối điều khiển (điều phối), ALU (tính toán, so sánh) và thanh ghi (nhớ tạm, rất nhanh). 1 GHz = một tỷ chu kỳ mỗi giây. Hiệu năng phụ thuộc cả xung nhịp, số nhân, bộ nhớ đệm và phần mềm.
A CPU has a control unit (directs), an ALU (calculates and compares) and registers (very fast temporary storage). 1 GHz = one billion cycles per second. Performance depends on clock speed, core count, cache and software together.

2.3. Bộ nhớ trong: RAM, ROM và bộ nhớ đệm – *2.3. Primary Memory: RAM, ROM and Cache*

Hãy hình dung bạn ngồi học ở bàn làm việc, sau lưng có một tủ hồ sơ. Tài liệu đang dùng thì để trên bàn, với tay là có. Tài liệu cất lâu dài thì nằm trong tủ, muốn dùng phải đứng dậy đi lấy. Máy tính cũng vậy: bộ nhớ trong là cái bàn, ổ cứng và SSD là tủ hồ sơ. Bộ nhớ trong là nơi bộ xử lý (CPU) lấy dữ liệu trực tiếp, nên rất nhanh nhưng dung lượng nhỏ. Nó gồm ba loại: RAM, ROM và cache.
Picture yourself studying at a desk with a filing cabinet behind you. Papers you are using sit on the desk, within arm's reach. Papers you keep for the long term stay in the cabinet, and you must get up to fetch them. A computer works the same way: primary memory is the desk, while the hard drive and SSD are the filing cabinet. Primary memory is where the processor (CPU) reads data directly, so it is very fast but small. It has three kinds: RAM, ROM and cache.

RAM: chiếc bàn làm việc – *RAM: the working desk*

RAM (random-access memory, bộ nhớ truy cập ngẫu nhiên) là nơi chứa chương trình và dữ liệu đang chạy. "Ngẫu nhiên" nghĩa là CPU đọc ô nhớ nào cũng được, theo thứ tự nào cũng được, giống như bạn nhặt bất kỳ tờ giấy nào trên bàn mà không phải lật từng tờ từ đầu. Khi bạn mở một chương trình, máy chép nó từ ổ cứng lên RAM rồi mới chạy. RAM nhanh hơn ổ cứng rất nhiều, nên RAM nhiều thì bàn rộng, bạn trải được nhiều việc cùng lúc mà không bị chậm.
RAM (random-access memory) holds the programs and data that are running right now. "Random access" means the CPU can read any cell in any order, like picking any sheet off your desk without flipping through the pile from the top. When you open a program, the computer copies it from the drive into RAM and then runs it. RAM is far faster than a drive, so more RAM means a bigger desk: you can spread out more work at once without slowing down.

RAM có tính "bay hơi" (volatile): có điện thì giữ dữ liệu, mất điện là mất sạch. Giống như bàn làm việc bị dọn trắng khi bạn rời phòng. Vì thế bài chưa lưu mà máy sập nguồn là mất bài. Lưu tệp nghĩa là chép từ RAM xuống ổ lưu trữ lâu dài.
RAM is volatile: it keeps data while powered and loses everything when the power goes off. It is like a desk that gets wiped clean when you leave the room. That is why an unsaved document disappears if the machine shuts down. Saving a file means copying it from RAM to long-term storage.

ROM: tờ hướng dẫn dán sẵn trên tường – *ROM: the instructions taped to the wall*

ROM (read-only memory, bộ nhớ chỉ đọc) giữ dữ liệu cả khi tắt máy. Nó chứa chương trình khởi động, thường gọi là firmware (phần sụn), ví dụ BIOS hoặc UEFI. Lúc vừa bấm nút nguồn, RAM còn trống và ổ cứng chưa ai gọi. CPU cần một đoạn lệnh có sẵn để kiểm tra phần cứng và tìm hệ điều hành cần nạp. Đoạn lệnh ấy nằm trong ROM, như tờ hướng dẫn dán sẵn trên tường: lúc nào vào phòng cũng có để đọc.

ROM (read-only memory) keeps its contents even when the machine is off. It stores the startup program, usually called firmware, such as BIOS or UEFI. The moment you press the power button, RAM is empty and nobody has called the drive yet. The CPU needs ready-made instructions to check the hardware and find the operating system to load. Those instructions live in ROM, like a note taped to the wall that is there every time you walk in.

Cache: tập giấy để sát tay – *Cache: the stack right by your hand*

Cache (bộ nhớ đệm) nằm sát CPU, thậm chí nằm ngay trong chip. Nó dùng công nghệ SRAM (static RAM), rất nhanh nhưng đắt và chỉ có từ vài KB đến vài MB. Nó giống tập giấy bạn đặt sát tay vì đang dùng liên tục. RAM thông thường dùng công nghệ DRAM, chậm hơn một chút nhưng rẻ hơn, nên chứa được hàng GB. Cache cũng bay hơi như RAM. Điểm khác là bạn không tự quản lý được nó: phần cứng tự quyết định chép gì vào cache.

Cache sits right next to the CPU, often inside the chip itself. It uses SRAM (static RAM), which is very fast but expensive and holds only a few KB to a few MB. Think of the stack of papers you keep at hand because you use it constantly. Ordinary RAM uses DRAM, which is a bit slower but cheaper, so it can hold many GB. Cache is volatile like RAM. The difference is that you cannot manage it yourself: the hardware decides what gets copied into it.

Tháp phân cấp bộ nhớ – *The memory hierarchy*

Không có loại bộ nhớ nào vừa nhanh, vừa rẻ, vừa lớn, vừa giữ được dữ liệu khi tắt máy. Vì vậy máy tính xếp nhiều loại thành tầng: càng gần CPU càng nhanh, đắt và nhỏ; càng xa càng chậm, rẻ và lớn. Hình dạng giống một cái tháp, đỉnh hẹp, đáy rộng.

No single memory is fast, cheap, large and able to survive power loss all at once. So computers stack several kinds in layers: the closer to the CPU, the faster, pricier and smaller; the farther away, the slower, cheaper and larger. The shape is a pyramid, narrow at the top and wide at the base.

Tầng *Layer*	**Tốc độ** *Speed*	**Giá trên mỗi GB** *Cost per GB*	**Dung lượng thường gặp** *Typical size*	**Mất điện thì sao** *Without power*
Cache (SRAM)	Nhanh nhất *Fastest*	Đắt nhất *Most expensive*	Vài KB đến vài MB *A few KB to a few MB*	Mất dữ liệu *Data lost*
RAM (DRAM)	Nhanh *Fast*	Vừa phải *Moderate*	Từ 1 GB đến hàng chục GB *From 1 GB to tens of GB*	Mất dữ liệu *Data lost*

Tầng *Layer*	**Tốc độ** *Speed*	**Giá trên mỗi GB** *Cost per GB*	**Dung lượng thường gặp** *Typical size*	**Mất điện thì sao** *Without power*
SSD	Chậm hơn *Slower*	Rẻ *Cheap*	Vài trăm GB đến vài TB *Hundreds of GB to a few TB*	Giữ dữ liệu *Data kept*
HDD	Chậm nhất *Slowest*	Rẻ nhất *Cheapest*	Vài TB *Several TB*	Giữ dữ liệu *Data kept*

Cách tháp vận hành: CPU hỏi tầng gần nhất trước, không có thì xuống tầng kế tiếp.
How the pyramid works: the CPU asks the nearest layer first, and moves down a layer if it finds nothing.

1. CPU cần một dữ liệu, tìm trong cache trước.
   The CPU needs a piece of data and checks the cache first.
2. Nếu có trong cache, CPU dùng ngay. Nếu không, CPU xuống RAM tìm.
   If it is in the cache, the CPU uses it at once. If not, it goes down to RAM.
3. Nếu RAM cũng không có, hệ điều hành nạp dữ liệu từ SSD hoặc HDD lên RAM.
   If RAM does not have it either, the operating system loads it from the SSD or HDD into RAM.
4. Bản sao vừa lấy được giữ lại ở tầng gần hơn, để lần sau tìm nhanh.
   The copy just fetched is kept in a closer layer so the next lookup is fast.

VÍ DỤ / EXAMPLE – Bạn mở một bài báo cáo 20 MB trong ổ SSD. Máy chép bài vào RAM, bạn gõ thêm một trang rồi bị cúp điện trước khi bấm lưu. Khi có điện lại, bài trong ổ SSD vẫn là bản cũ, trang mới nằm ở RAM nên đã mất. Còn đoạn lệnh khởi động nằm trong ROM thì máy vẫn bật lên bình thường.
You open a 20 MB report stored on the SSD. The computer copies it into RAM, you type one more page, and then the power cuts out before you save. When power returns, the file on the SSD is still the old version, and the new page, which lived in RAM, is gone. The startup instructions in ROM are untouched, so the machine boots normally.

LỖI HAY GẶP / COMMON MISTAKE – Nhầm RAM với ổ cứng. Máy ghi "16 GB RAM" và "512 GB SSD" là hai thứ khác nhau: một cái là bàn, một cái là tủ.
Confusing RAM with the drive. A spec of "16 GB RAM" and "512 GB SSD" describes two different things: one is the desk, the other the cabinet.

LỖI HAY GẶP / COMMON MISTAKE – Nghĩ cache là RAM loại tốt hơn mà mua thêm được. Cache gắn sẵn trong bộ xử lý, bạn không tự thêm hay chọn dung lượng.
Thinking cache is just better RAM you can buy more of. Cache is built into the processor, and you cannot add to it or pick its size.

PHẢI THUỘC / MEMORISE – RAM: bay hơi, chứa chương trình đang chạy. ROM: giữ dữ liệu khi tắt máy, chứa chương trình khởi động. Cache: SRAM, nhanh nhất, nằm sát CPU. Tháp bộ nhớ: gần CPU thì nhanh, đắt, nhỏ; xa CPU thì chậm, rẻ, lớn.
RAM: volatile, holds running programs. ROM: keeps data when off, holds startup code. Cache:

> *SRAM, fastest, next to the CPU. Memory pyramid: closer to the CPU means faster, pricier, smaller; farther means slower, cheaper, larger.*

2.4. Bộ nhớ ngoài: ổ cứng HDD, SSD, USB và lưu trữ đám mây – *2.4. Secondary Storage: HDD, SSD, USB Drives and Cloud Storage*

Bộ nhớ ngoài là nơi cất dữ liệu lâu dài, tắt máy vẫn còn nguyên. Hãy hình dung RAM như mặt bàn làm việc: rộng vừa đủ, lấy giấy tờ nhanh, nhưng dọn bàn là mất. Bộ nhớ ngoài giống tủ hồ sơ: chứa được nhiều hơn, chậm hơn một chút, và đồ cất vào vẫn ở đó hôm sau.

Secondary storage keeps data for the long term, and it survives power-off. Think of RAM as your desk: just big enough, quick to grab papers from, but cleared when you leave. Secondary storage is the filing cabinet: it holds far more, it is slower to reach into, and whatever you file is still there tomorrow.

Ổ cứng HDD và ổ thể rắn SSD – *Hard disk drives (HDD) and solid-state drives (SSD)*

Ổ cứng HDD (hard disk drive) ghi dữ liệu lên các đĩa kim loại phủ từ, quay liên tục. Một đầu đọc nhỏ lướt sát mặt đĩa để đọc hoặc ghi, giống cần kim trên đĩa than. Vì có bộ phận quay và chuyển động nên HDD phải chờ đầu đọc tới đúng chỗ.

A hard disk drive stores data on magnetically coated metal platters that spin all the time. A tiny read/write head hovers just above the surface, much like the needle on a record player. Because parts spin and move, the drive has to wait for the head to reach the right spot.

Ổ thể rắn SSD (solid-state drive) lưu dữ liệu trong chip nhớ flash. "Thể rắn" nghĩa là không có bộ phận chuyển động, nên SSD tìm dữ liệu gần như tức thì, giống bật công tắc đèn thay vì chờ quạt quay.

A solid-state drive stores data in flash memory chips. "Solid-state" means no moving parts, so the drive finds data almost instantly, like flipping a light switch instead of waiting for a fan to spin up.

Hãy dùng bảng này để đối chiếu hai loại ổ. Các con số là mức thường gặp, mỗi model một khác.

Use this table to compare the two drives. The figures are typical ranges, and every model differs.

Tiêu chí *Criterion*	**HDD (đĩa từ)** *HDD (magnetic disk)*	**SSD (thể rắn)** *SSD (solid-state)*
Cách hoạt động *How it works*	Đĩa quay, đầu đọc di chuyển cơ học. *Spinning platters and a mechanical read/write head.*	Chip flash, chỉ có mạch điện, không chuyển động. *Flash chips with only electronics and nothing moving.*
Tốc độ đọc ghi *Read/write speed*	Khoảng 100 đến 200 MB/s, thời gian chờ tìm dữ liệu dài. *Roughly 100 to 200 MB/s, with a long wait to locate data.*	Khoảng 500 MB/s với loại SATA, vài nghìn MB/s với loại NVMe. *Around 500 MB/s for SATA models and several thousand MB/s for NVMe models.*

Tiêu chí *Criterion*	**HDD (đĩa từ)** *HDD (magnetic disk)*	**SSD (thể rắn)** *SSD (solid-state)*
Độ bền *Durability*	Sợ rơi và rung khi đang chạy vì đầu đọc sát mặt đĩa. *Fears drops and vibration while running, since the head sits so close to the platter.*	Chịu va đập tốt hơn, nhưng mỗi ô nhớ chỉ ghi xoá được một số lần nhất định. *Handles shocks better, but each cell can only be rewritten a limited number of times.*
Giá mỗi GB *Price per GB*	Rẻ hơn, hợp để chứa dữ liệu lớn. *Cheaper, a good fit for bulk data.*	Đắt hơn, nhưng đã giảm dần theo thời gian. *More expensive, though prices have fallen over time.*
Tiếng ồn và nhiệt *Noise and heat*	Có tiếng quay, toả nhiệt nhiều hơn. *Audible spinning, more heat.*	Êm, nhẹ, ít tốn pin laptop. *Silent, light, and gentler on laptop batteries.*

Nhiều máy kết hợp cả hai: SSD nhỏ chứa hệ điều hành và phần mềm hay dùng, HDD lớn chứa phim, ảnh và tài liệu cũ.

Many computers use both: a small SSD for the operating system and frequently used programs, and a large HDD for movies, photos and old files.

Đĩa quang, thẻ nhớ và USB – *Optical discs, memory cards and USB drives*

- Đĩa quang (CD, DVD, Blu-ray) được đọc bằng tia laser. Dung lượng khoảng 700 MB cho CD, 4,7 GB cho DVD một lớp, 25 GB cho Blu-ray một lớp. Giờ ít dùng, chủ yếu để phát phim hoặc lưu trữ lâu.
  Optical discs (CD, DVD, Blu-ray) are read by a laser. A CD holds about 700 MB, a single-layer DVD 4.7 GB, a single-layer Blu-ray 25 GB. They are rarely used now, mostly for movies or long-term archives.
- Thẻ nhớ (SD, microSD) dùng chip flash như SSD, nhỏ bằng móng tay, hay gặp trong máy ảnh và điện thoại.
  Memory cards (SD, microSD) use flash chips like an SSD. They are about the size of a fingernail and common in cameras and phones.
- USB flash là chip flash gắn đầu cắm USB (Universal Serial Bus, cổng nối tiêu chuẩn). Nó nhỏ, rẻ, mang theo túi, hợp để chép bài nộp hoặc chuyển tệp giữa hai máy.
  A USB flash drive is a flash chip with a USB plug (Universal Serial Bus, a standard connector). It is small, cheap and pocket-sized, ideal for carrying assignments or moving files between two computers.

Lưu trữ đám mây – *Cloud storage*

Lưu trữ đám mây nghĩa là tệp nằm trên máy chủ của một công ty, như Google Drive hay OneDrive, và bạn truy cập qua Internet. Ưu điểm: mở được từ nhiều thiết bị, máy hỏng vẫn còn dữ liệu. Nhược điểm: cần mạng, tốc độ phụ thuộc đường truyền, và bạn tin vào nhà cung cấp.

Cloud storage means your files sit on a company's servers, such as Google Drive or OneDrive, and you reach them over the Internet. Upsides: you can open them from many devices, and they

survive a broken laptop. Downsides: you need a connection, speed depends on your line, and you must trust the provider.

Bài mẫu: tính thời gian chép một tệp – ***Worked example: how long does a file copy take?***

Thời gian chép bằng dung lượng chia cho tốc độ. Tốc độ bị giới hạn bởi khâu chậm nhất, gọi là nút thắt cổ chai.

Copy time equals file size divided by speed. The speed is capped by the slowest link, called the bottleneck.

1. Đổi dung lượng và tốc độ về cùng đơn vị. Ở đây lấy 1 GB = 1000 MB cho dễ tính.
   Convert size and speed to the same unit. Here we take 1 GB = 1000 MB to keep the maths simple.
2. So tốc độ đọc của ổ nguồn với tốc độ ghi của ổ đích, lấy số nhỏ hơn.
   Compare the source drive's read speed with the destination's write speed, and take the smaller one.
3. Chia dung lượng (MB) cho tốc độ đó (MB/s) ra số giây.
   Divide the size (MB) by that speed (MB/s) to get seconds.
4. Đổi giây ra phút nếu cần, chia cho 60.
   Convert seconds to minutes if needed by dividing by 60.

VÍ DỤ / EXAMPLE – Chép video 6 GB từ SSD (đọc 500 MB/s) sang USB (ghi 50 MB/s). Đổi: 6 GB = 6000 MB. Nút thắt là USB, 50 MB/s. Thời gian = 6000 ÷ 50 = 120 giây = 2 phút. Nếu chép sang SSD khác ghi 450 MB/s thì nút thắt là 450 MB/s, mất 6000 ÷ 450 ≈ 13,3 giây.

Copy a 6 GB video from an SSD (read 500 MB/s) to a USB drive (write 50 MB/s). Convert: 6 GB = 6000 MB. The bottleneck is the USB drive at 50 MB/s. Time = 6000 ÷ 50 = 120 seconds = 2 minutes. Copying to another SSD that writes at 450 MB/s makes 450 MB/s the bottleneck, so it takes 6000 ÷ 450 ≈ 13.3 seconds.

VÍ DỤ / EXAMPLE – Tải cùng tệp 6 GB lên đám mây bằng đường truyền 80 Mbps. Cứ 8 bit bằng 1 byte, nên 80 Mbps = 80 ÷ 8 = 10 MB/s. Thời gian = 6000 ÷ 10 = 600 giây = 10 phút.

Upload the same 6 GB file to the cloud over an 80 Mbps line. Since 8 bits make 1 byte, 80 Mbps = 80 ÷ 8 = 10 MB/s. Time = 6000 ÷ 10 = 600 seconds = 10 minutes.

LỖI HAY GẶP / COMMON MISTAKE – Lẫn Mb (megabit, chữ b thường) với MB (megabyte, chữ B hoa), nên tính sai gấp 8 lần. Quên nút thắt: lấy tốc độ ổ nhanh làm kết quả. Cho rằng SSD không bao giờ hỏng, hoặc đã lên đám mây là khỏi cần sao lưu.

Mixing up Mb (megabit, lowercase b) with MB (megabyte, capital B), which throws the result off by a factor of 8. Forgetting the bottleneck and using the faster drive's speed. Assuming an SSD never fails, or that files in the cloud need no backup.

PHẢI THUỘC / MEMORISE – HDD dùng đĩa từ quay, rẻ mỗi GB, sợ va đập. SSD dùng chip flash, nhanh, êm, đắt hơn. USB và thẻ nhớ cũng là flash. Đĩa quang dùng laser. Đám mây cần Internet. Thời gian chép = dung lượng ÷ tốc độ của khâu chậm nhất.

HDD uses spinning magnetic disks, is cheap per GB and fears shocks. SSD uses flash chips and is

> *fast, quiet and pricier. USB drives and memory cards are flash too. Optical discs use a laser. The cloud needs the Internet. Copy time = size ÷ speed of the slowest link.*

2.5. Thiết bị vào, thiết bị ra và cổng kết nối – *2.5. Input Devices, Output Devices and Ports*

Máy tính chỉ có ích khi nhận được dữ liệu từ bạn và trả kết quả về cho bạn. Thiết bị vào đưa thông tin vào máy, thiết bị ra đưa thông tin ra. Bên trong, mọi thứ đều đổi thành các số 0 và 1 để lưu trong bộ nhớ, rồi đổi ngược lại thành chữ, hình hoặc âm thanh. Giống như người phiên dịch đứng ở cửa nhà: khách nói tiếng người, nhà trong chỉ hiểu tiếng máy.

A computer is only useful if it can take data from you and hand results back. Input devices send information into the machine, and output devices send it out. Inside, everything becomes 0s and 1s stored in memory, then is turned back into text, pictures or sound. Think of an interpreter at the front door: visitors speak human, the house only understands machine.

Thiết bị vào – *Input devices*

- Bàn phím: bạn bấm phím, máy nhận mã của từng ký tự. Đây là cách nhập chữ và số chính.
  Keyboard: each key press sends the code of one character. It is the main way to enter text and numbers.
- Chuột và bàn di chuột: đưa con trỏ tới chỗ cần chọn rồi bấm. Chúng ra đời cùng giao diện đồ họa, nơi bạn chọn biểu tượng thay vì gõ lệnh.
  Mouse and touchpad: move the pointer to what you want and click. They arrived with graphical interfaces, where you pick icons instead of typing commands.
- Máy quét: chuyển trang giấy hoặc mã vạch thành ảnh số. Ví dụ, quét tờ giấy tờ tùy thân để nộp hồ sơ trực tuyến.
  Scanner: turns a paper page or a barcode into a digital image. For example, scanning an ID card to submit an online application.
- Micro: thu sóng âm rồi đổi thành dữ liệu số, dùng khi học trực tuyến hay ghi âm.
  Microphone: captures sound waves and converts them to digital data, used in online classes and recordings.

Thiết bị ra – *Output devices*

- Màn hình: hiển thị hình ảnh. Ảnh trên màn hình là lưới các điểm nhỏ gọi là điểm ảnh (pixel).
  Monitor: shows images. The picture is a grid of tiny dots called pixels.
- Máy in: đưa nội dung ra giấy. Máy in phun dùng mực lỏng, máy in laser dùng bột mực nung chảy bám vào giấy.
  Printer: puts content on paper. Inkjet printers spray liquid ink, laser printers fuse toner powder onto the page.
- Loa và tai nghe: biến dữ liệu âm thanh thành sóng âm để bạn nghe.
  Speakers and headphones: turn audio data into sound waves you can hear.

Một số thiết bị làm cả hai việc. Màn hình cảm ứng vừa hiển thị (ra) vừa nhận ngón tay chạm (vào). Tai nghe có micro vừa phát tiếng vừa thu tiếng. Ổ USB và ổ cứng ngoài cũng vừa đọc vừa ghi dữ liệu.

Some devices do both jobs. A touchscreen displays (output) and senses your finger (input). A headset plays sound and records your voice. USB drives and external disks also read and write data.

Độ phân giải màn hình và dpi máy in – *Screen resolution and printer dpi*

Độ phân giải màn hình là số điểm ảnh theo chiều ngang nhân chiều dọc. Màn hình 1920 × 1080 có 1920 × 1080 = 2.073.600 điểm ảnh. Cùng kích thước vật lý, nhiều điểm ảnh hơn thì hình mịn hơn, chữ nhỏ vẫn rõ.

Screen resolution is the number of pixels across times the number down. A 1920 × 1080 screen has 1920 × 1080 = 2,073,600 pixels. At the same physical size, more pixels give a smoother picture and sharper small text.

Chỉ số dpi (dots per inch, số chấm trên mỗi inch) cho biết máy in đặt được bao nhiêu chấm mực trên một inch giấy. Số này càng cao, bản in càng mịn nhưng in chậm hơn và tốn mực hơn.

The dpi figure (dots per inch) tells how many ink dots a printer places along one inch of paper. A higher number prints finer detail but runs slower and uses more ink.

> **VÍ DỤ / EXAMPLE –** Bạn muốn in một ảnh 6 × 4 inch ở 300 dpi. Chiều ngang cần 6 × 300 = 1800 chấm, chiều dọc cần 4 × 300 = 1200 chấm. Ảnh cần ít nhất 1800 × 1200 điểm ảnh, tức 2.160.000 điểm ảnh. Ảnh nhỏ hơn mức này sẽ bị vỡ hạt khi in.
> *You want to print a 6 × 4 inch photo at 300 dpi. The width needs 6 × 300 = 1800 dots and the height 4 × 300 = 1200 dots. The photo needs at least 1800 × 1200 pixels, which is 2,160,000 pixels. A smaller image will look grainy in print.*

Cổng kết nối và bus – *Ports and buses*

Cổng là ổ cắm nằm ở thành máy, thường gắn với bo mạch chủ (mạch chính mà CPU, RAM và ổ lưu trữ cùng cắm vào). Thiết bị cắm vào cổng để trao đổi dữ liệu với máy.

A port is a socket on the case, usually wired to the motherboard (the main circuit board that the CPU, RAM and storage plug into). Devices plug into ports to exchange data with the computer.

- USB: cổng chuẩn chung cho bàn phím, chuột, máy in, ổ lưu trữ, tai nghe. Gần như mọi thiết bị ngày nay cắm qua USB.
  USB: the common standard port for keyboards, mice, printers, storage drives and headsets. Almost every device today plugs in through USB.
- HDMI: cổng đưa cả hình và tiếng sang màn hình ngoài, tivi hoặc máy chiếu bằng một sợi cáp.
  HDMI: sends both picture and sound to an external monitor, TV or projector over one cable.
- Bluetooth: không dùng dây, truyền dữ liệu bằng sóng vô tuyến trong khoảng cách ngắn. Cả máy và thiết bị đều phải có chip Bluetooth.
  Bluetooth: wireless, sends data by radio over short distances. Both the computer and the device need a Bluetooth chip.