VIỆN CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

TIN HỌC ĐẠI CƯƠNG

Phần 1: TIN HỌC CĂN BẢN

Nội dung chính

Chương 1: Thông tin và biểu diễn thông tin

- Các khái niệm cơ bản về thông tin và tin học
- Biểu diễn dữ liệu trong máy tính

Chương 2: Hệ thống máy tính

- Hệ thống máy tính
- Mạng máy tính
- Hệ điều hành

Chương 3: Các hệ thống ứng dụng

- Hệ thống thông tin quản lý
- Hệ thông tin bảng tính
- Hệ quản trị cơ sở dữ liệu
- Các hệ thống thông minh

Nội dung chính

1. Các khái niệm cơ bản về thông tin và tin học
    1. Thông tin và xử lý thông tin
    2. Máy tính điện tử và phân loại
    3. Tin học và các ngành liên quan
2. Biểu diễn dữ liệu trong máy tính
    1. Biểu diễn số trong các hệ đếm
    2. Biểu diễn dữ liệu trong MT & đơn vị thông tin
    3. Biểu diễn số nguyên
    4. Biểu diễn số thực
    5. Biểu diễn ký tự

1.1 Thông tin và xử lý thông tin

- Thông tin
- Dữ liệu
- Tri thức
- Hệ thống thông tin
- Xử lý thông tin

Thông tin

Thông tin (information) là gì ?

- Là khái niệm trừu tượng mô tả tất cả những gì đem lại cho con người sự *hiểu biết, nhận thức* tốt hơn về những đối tượng trong đời sống xã hội, trong thiên nhiên,...
- Giúp cho con người thực hiện hợp lý công việc cần làm để đạt tới mục đích một cách tốt nhất.
- Là ngữ cảnh trong đó dữ liệu được xem xét

Dữ liệu

Dữ liệu (data) là gì ?

- Là biểu diễn của thông tin được thể hiện bằng các tín hiệu vật lý.
- Là vật liệu thô mang tin,
  - Dữ liệu sau khi được tập hợp và xử lý sẽ cho ra thông tin.

Dữ liệu trong thực tế có thể là

- Các số liệu: Dữ liệu số như trong các bảng biểu.
- Các ký hiệu quy ước; ví dụ chữ viết…
- Các tín hiệu vật lý; ví dụ như ánh sáng, âm thanh, nhiệt độ, áp suất….

Dữ liệu→Ghi chú

- Thông tin chứa đựng ý nghĩa
- Dữ liệu chỉ là các sự kiện không có cấu trúc và không có ý nghĩa nếu không được tổ chức và xử lý.

Ví dụ

Nhiệt độ cơ thể - dữ liệu số

- 39°C: Thông tin đang bị sốt
- 37°C: Thông tin bình thường

Tri thức

- **Tri thức (Knowledge) là gì?**
  - Tri thức theo nghĩa thường là thông tin ở mức trừu tượng hơn
- Tri thức khá đa dạng:
  - Có thể là sự kiện , là thông tin
  - Là cách mà một người thu thập được qua kinh nghiệm hoặc qua đào tạo.
  - Có thể là sự hiểu biết chung hay về một lĩnh vực cụ thể nào đó.

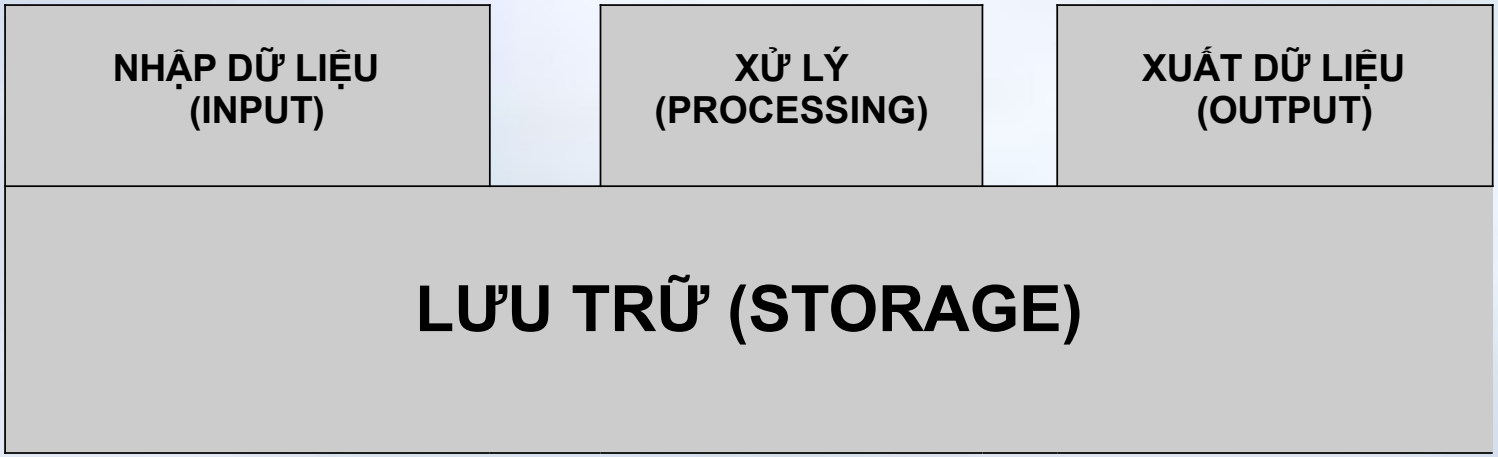
Hệ thống thông tin

Hệ thống thông tin (*information system*) là một hệ thống ghi nhận dữ liệu, xử lý chúng để tạo nên thông tin có ý nghĩa hoặc dữ liệu mới.

Dữ liệu —Xử lý→ **Thông tin** —Xử lý→ **Tri thức**

Xử lý thông tin

Quy trình xử lý thông tin (*máy tính/con người*)



Xử lý thông tin bằng máy tính điện tử

- Tiết kiệm rất nhiều thời gian, công sức
- Tăng độ chính xác cao trong tự động hóa một phần hay toàn phần của quá trình xử lý dữ liệu hay thông tin

1.2 Máy tính điện tử và phân loại

- Lịch sử hình thành và phát triển
  - Máy tính điện tử và chương trình
  - Các thế hệ của máy tính điện tử
- Phân loại máy tính
  - Phân loại theo hiệu năng tính toán
  - Phân loại khác

Lịch sử hình thành và phát triển

- **Máy tính điện tử** (*computer*):
- Là thiết bị điện tử thực hiện các công việc:
  - Nhận thông tin vào
  - Xử lý thông tin theo chương trình được nhớ sẵn bên trong
  - Đưa thông tin ra
- **Chương trình** (*program*)
- Là một dãy các lệnh trong bộ nhớ nhằm yêu cầu máy tính thực hiện công việc cụ thể.

  ⇒ Máy tính hoạt động theo chương trình.

Lịch sử hình thành và phát triển

Các thế hệ của máy tính điện tử

- Thế hệ 1 (1950-1958)
- Thế hệ 2(1958-1964)
- Thế hệ 3 (1965-1974)
- Thế hệ 4 (1974 – nay)
- Thế hệ 5 (1990 – nay)

Lịch sử hình thành và phát triển→Thế hệ 1

Von Neumann Machine (1950-1958)

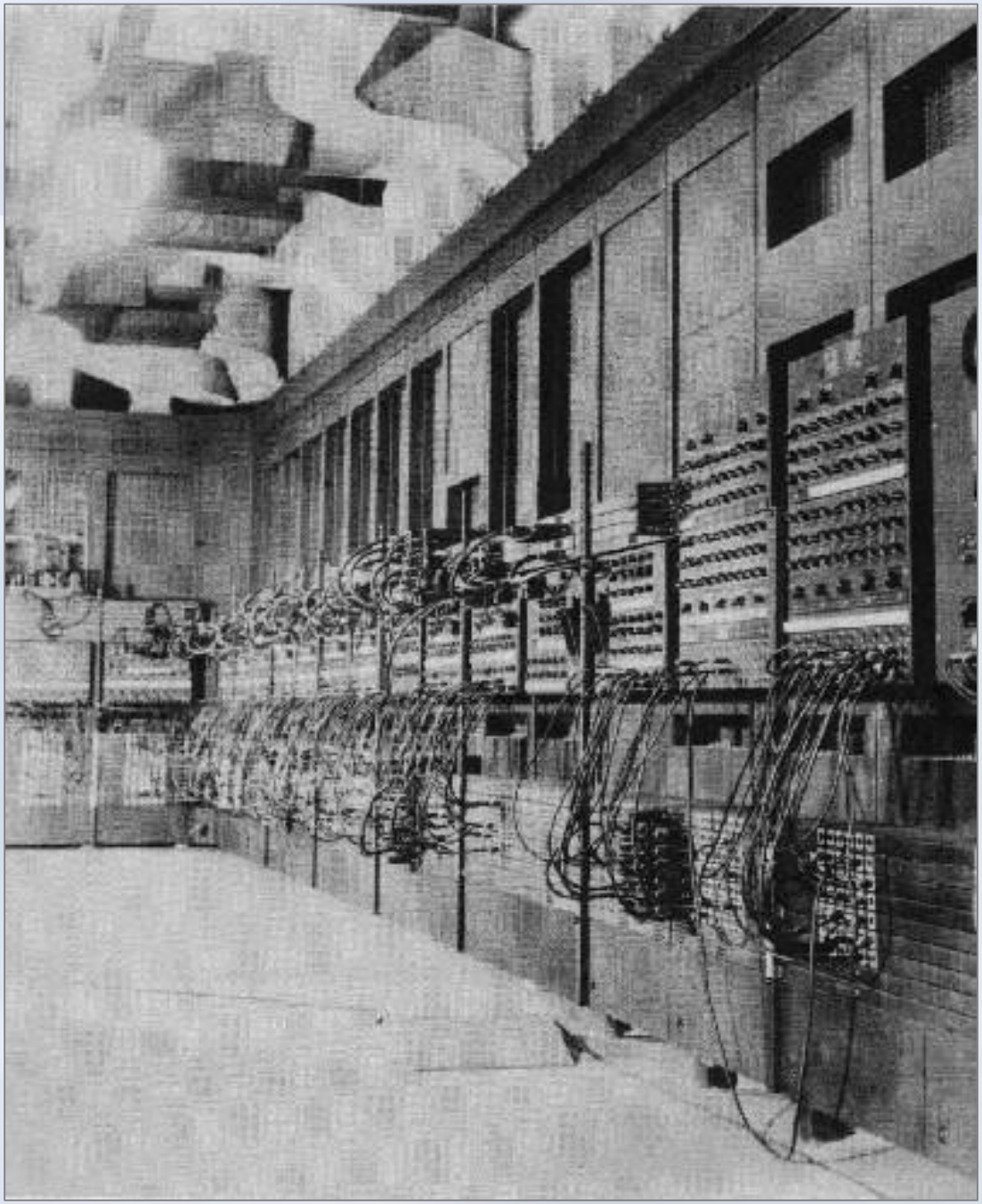
- Sử dụng các bóng đèn điện tử chân không
- Mạch riêng rẽ, vào số liệu bằng phiếu đục lỗ
- Điều khiển bằng tay, kích thước rất lớn
- Tiêu thụ năng lượng nhiều, tốc độ tính chậm khoảng 300 - 3.000 phép tính/s.

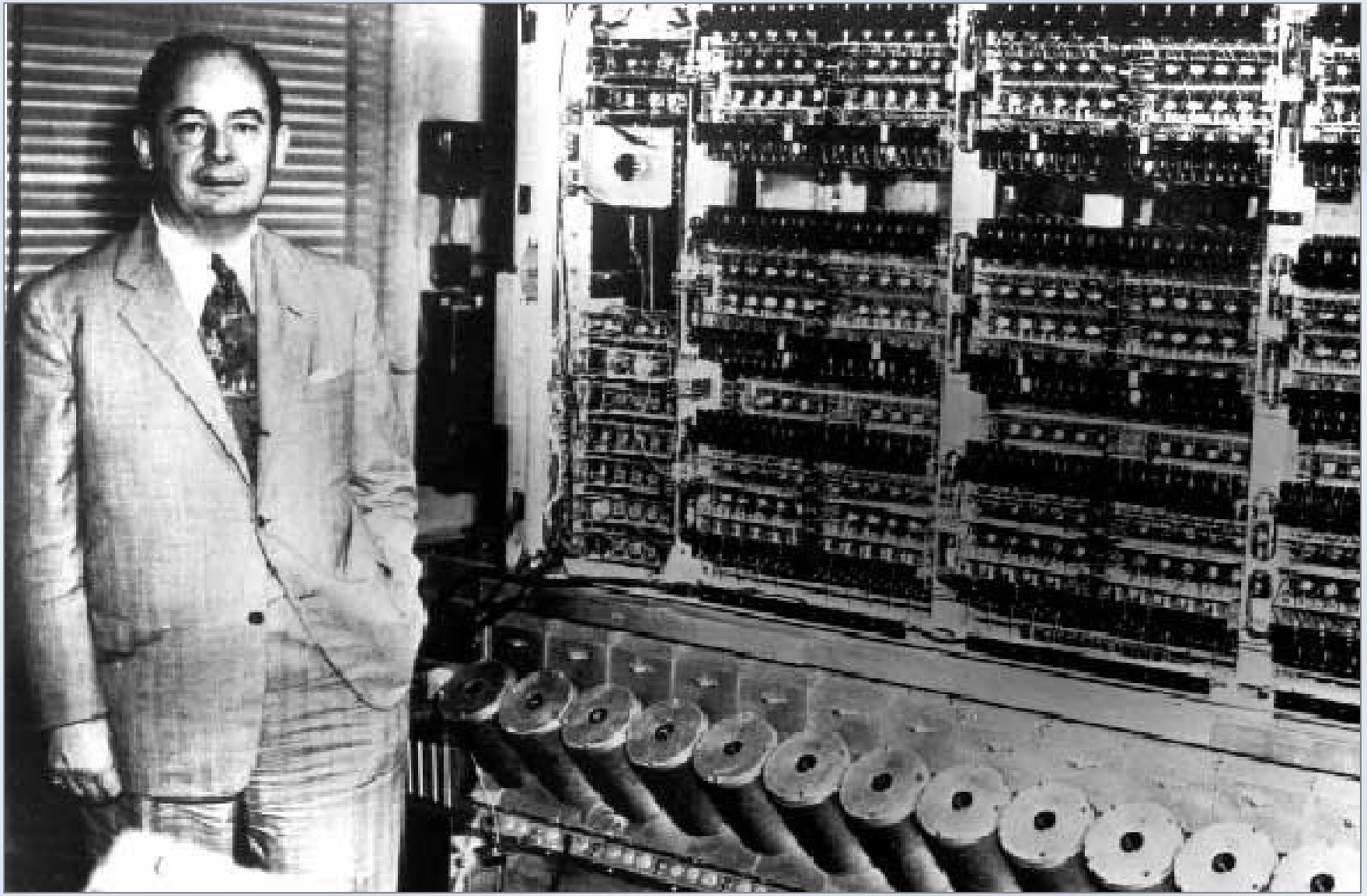
Ví dụ: EDVAC (Mỹ), BESEM (Liên xô cũ)

Bóng đèn chân không

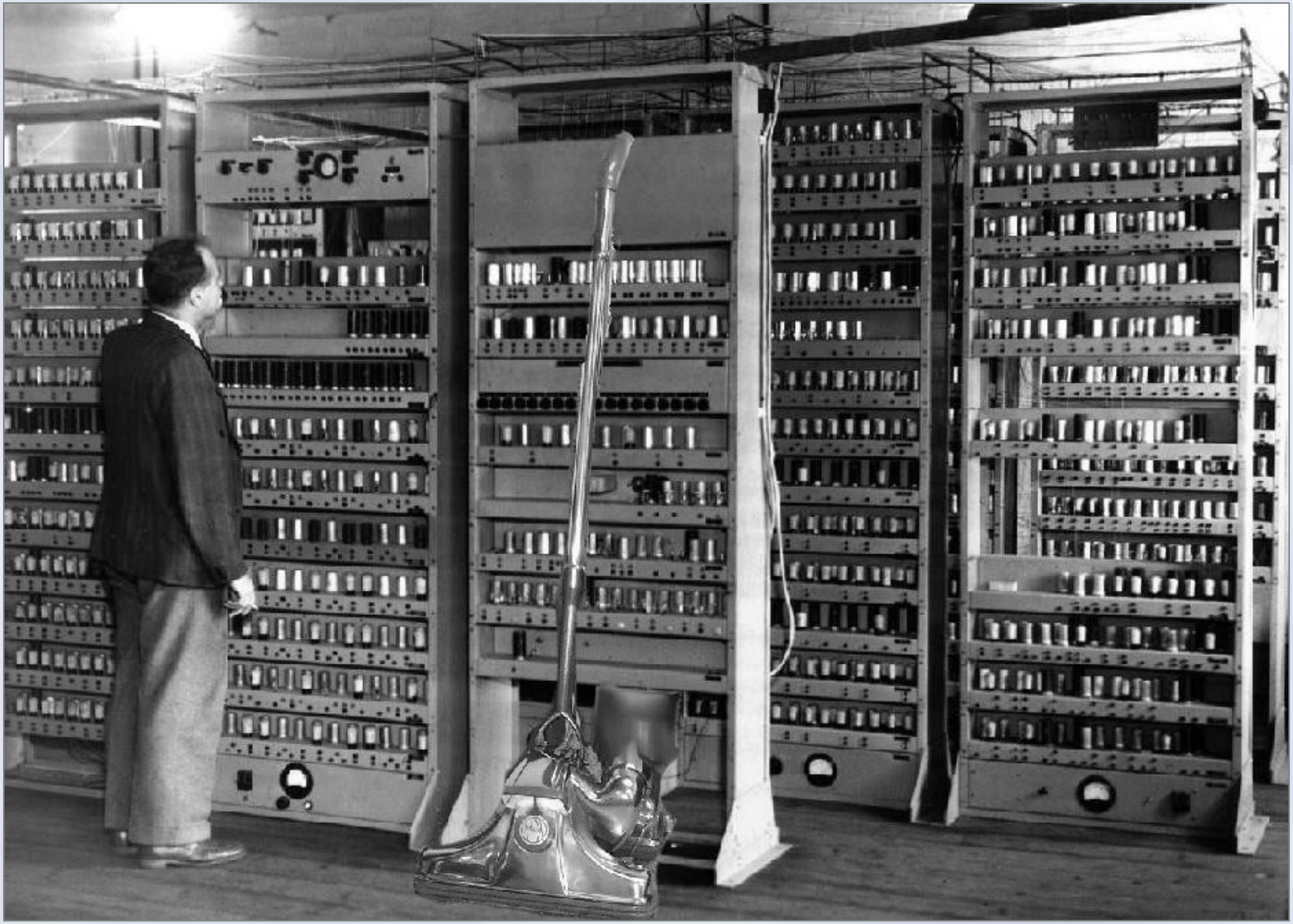
Máy tính đầu tiên: **ENIAC**

Electronic
Numerical
Integrator **A**nd
Computer





Von Neumann & **UNIVAC** (Universal Automatic Computer)



EDVAC: Electronic Discrete Variable Automatic Computer

Lịch sử hình thành và phát triển→Thế hệ 2

Transistors (1958 - 1964):

- Sử dụng **bộ xử lý bằng đèn bán dẫn, mạch in**
- Đã có chương trình dịch như Cobol, Fortran và hệ điều hành đơn giản.
- Kích thước máy còn lớn
- Tốc độ tính khoảng 10.000 - 100.000 phép tính/s

Ví dụ

- IBM 7000 series (Mỹ)
- MINSK (Liên Xô cũ)

IBM 7030

MINSK (Liên Xô cũ)

Lịch sử hình thành và phát triển→Thế hệ 3

Integrated Circuits (1965 - 1974):

- Các bộ vi xử lý được gắn vi mạch điện tử cỡ nhỏ
- Tốc độ tính khoảng 100.000 - 1 triệu phép tính/s.
- Có các hệ điều hành đa chương trình, nhiều người đồng thời theo kiểu phân chia thời gian.
- Kết quả từ máy tính có thể in trực tiếp từ máy in.

Ví dụ

-

IBM 360/91

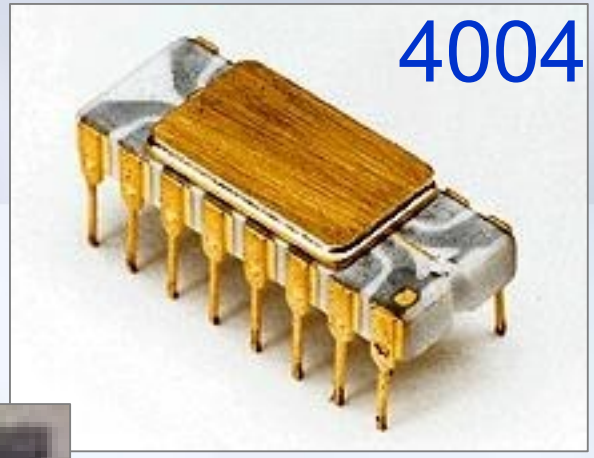
Lịch sử hình thành và phát triển→Thế hệ 4

LSI(Large Scale Integration), **Multiprocessors:**

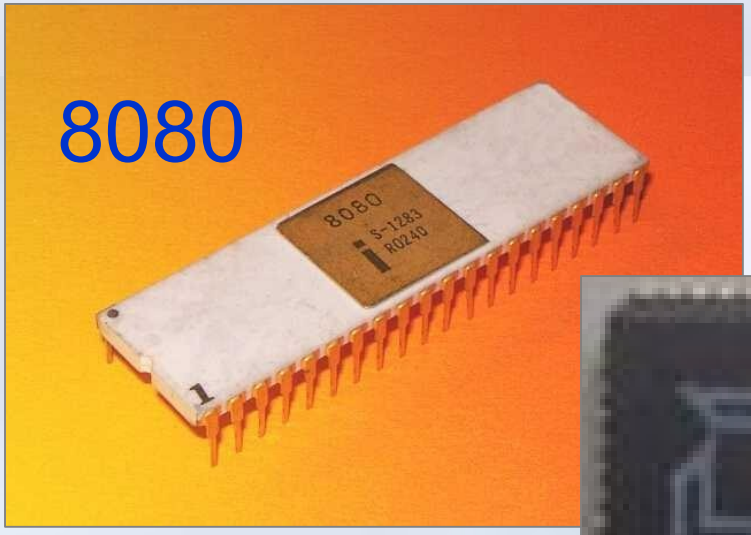
- Máy tính có các vi mạch đa xử lý
- Tốc độ: hàng chục triệu đến hàng tỷ phép tính/s.
- Hai loại máy tính chính:
  - Máy tính cá nhân để bàn (Personal Computer - PC) hoặc xách tay (Laptop hoặc Notebook computer)
  - Các loại máy tính chuyên nghiệp thực hiện đa chương trình, đa xử lý,…
- Hình thành các hệ thống mạng tính máy (Computer Networks).
- Các ứng dụng phong phú, đa phương tiện

Vi mạch Intel

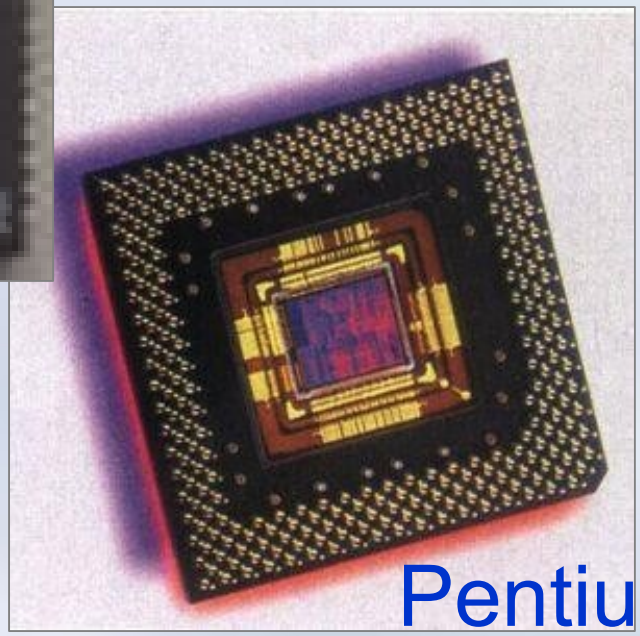
4004



8080



80386



Pentium

Lịch sử hình thành và phát triển→Thế hệ 5

VLSI (Very Large Scale Integration**), ULSI** (Ultra), Artificial Intelligence **(AI)**

- Công nghệ vi điện tử với tốc độ tính toán cao và khả năng xử lý song song.
- Mô phỏng các hoạt động của não bộ và hành vi con người
- Có trí khôn nhân tạo với khả năng tự suy diễn phát triển các tình huống nhận được
- Hệ quản lý kiến thức cơ bản để giải quyết các bài toán đa dạng.

Phân loại máy tính

Nhiều cách phân loại khác nhau

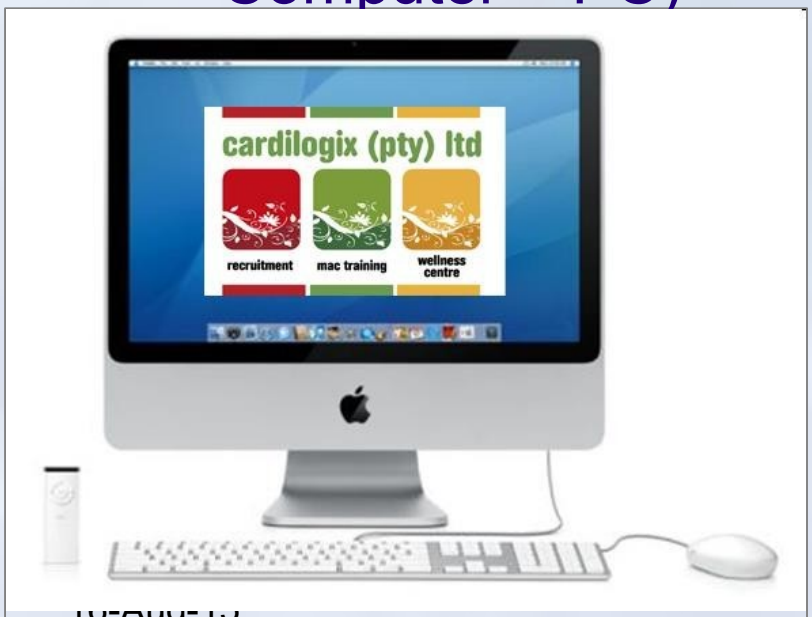
- **Theo hiệu năng tính toán**
  - Máy vi tính (Microcomputer)
  - Máy tính nhỏ (Minicomputer)
  - Máy tính lớn (Mainframe Computer)
  - Siêu máy tính (Supercomputer)
- **Phân loại khác**
  - Máy tính "để bàn" (Desktop Computing)
  - Máy chủ (Server)
  - Máy tính nhúng (Embedded Computer)

Phân loại máy tính→ Máy vi tính

- Thường được thiết kế cho một người dùng
- Giá thành rẻ
- Có nhiều dang máy khác nhau
  - Máy để bàn (*desktop computer*)
  - Máy trạm (*Workstation*)
  - Máy xách tay (*Notebook, Laptop*)
  - Máy tính bỏ túi

Phân loại máy tính→ Máy tính để bàn

- Là loại máy tính phổ biến
- Thiết kế theo hướng tối ưu cả về giá thành và hiệu năng
- Giá thành: từ 500$ đến 10,000$
- Một số loại:
  - Máy tính cá nhân (Personal Computer – PC)



- IBM giới thiệu máy tính IBM-PC sử dụng bộ vi xử lý Intel 8088 năm1981
- Apple đưa ra Macintosh sử dụng bộ xử lý Motorola 68000 năm 1984

Phân loại máy tính→ Máy tính nhỏ

- Thường được thiết kế để sử dụng cho các ứng dụng phức tạp
  - Tốc độ và hiệu năng tinh toán mạnh hơn máy vi tính
- Giá thành: Khoảng chục ngàn USD

Phân loại máy tính→ Máy tính lớn và siêu MT

- Có tổ chức bên trong phức tạp
- Tốc độ rất nhanh, hiệu năng cao
  - Ngàn tỷ phép tính/giây
- Cho phép nhiều người dùng đồng thời
- Được dùng tại các trung tâm tính toán
  - Nhằm giải quyết bài toán lớn, đòi hỏi tốc độ
- Giá thành: Hàng triệu USD

Mainfra
me

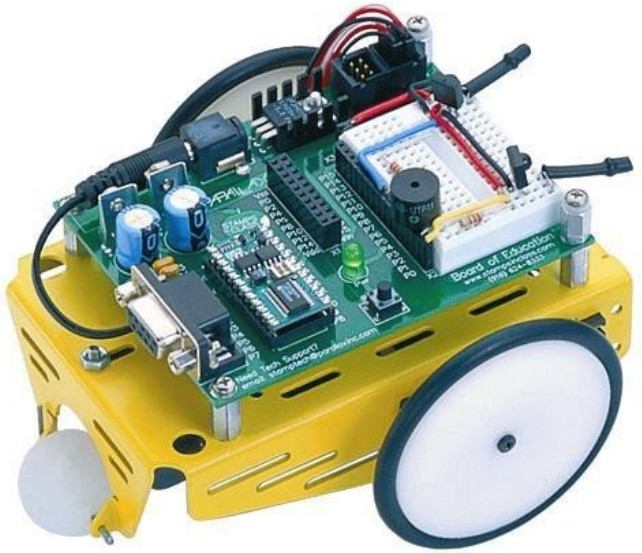
Supercomputer

Phân loại máy tính→ Máy chủ

- Là máy phục vụ; thường dùng trong mạng máy tính theo mô hình Client/Server
- Tốc độ và hiệu năng tính toán cao
- Dung lượng bộ nhớ lớn
- Độ tin cậy cao
- Giá thành: từ hàng chục nghìn đến hàng triệu USD.

Phân loại máy tính→ Máy tính nhúng

- Đặt trong thiết bị khác để điều khiển thiết bị đó làm việc
- Được thiết kế chuyên dụng
  - Điện thoại di động
  - Bộ điều khiển trong máy giặt, điều hòa nhiệt độ
  - Một số thiết bị mạng: Switch, Router, …
- Giá thành: từ vài USD đến hàng trăm ngàn USD



High-Speed Control

- Antilock Braking
- Central Electronics
- Electronic Throttle
- Engine Control
- Steering Wheel
- Transmission Control

Low Speed Control

- Audio
- Climate Control
- Driver's Door
- Driver Information
- Passenger Door
- Phone
- Power Seat
- Rear Electronics
- Sun Roof
- Supplemental Restraint System
- Upper Electronics

1.3 Tin học và các ngành liên quan

- Thuật ngữ tin học
- Công nghệ thông tin
- Công nghệ thông tin và truyền thông

Thuật ngữ tin học

- Nguồn gốc từ tiếng Đức
  - Năm 1957 Karl Steinbuch đề xướng trong bài báo *Informatik: Automatische Informationsverarbeitung* (*Informatics: automatic information processing*).
- Năm 1962, Philippe Dreyfus người Pháp gọi là "*informatique*", tiếp theo là Walter F.Bauer cũng sử dụng tên này.
- Phần lớn các nước Tây Âu, đều chấp nhận.
  - Tại Anh, sử dụng thuật ngữ '**computer science**', hay '**computing science**',
- Nga cũng chấp nhận tên *informatika* (1966)

Tin học (informatic)

- Tin học là ngành khoa học nghiên cứu các phương pháp, công nghệ và kỹ thuật xử lý thông tin một cách tự động.
- Công cụ chủ yếu sử dụng trong tin học là máy tính điện tử và một số thiết bị truyền tin.
- Nội dung nghiên cứu của tin học gồm :
  - **Kỹ thuật phần cứng** (Hardware engineering)
  - **Kỹ thuật phần mềm** (Software engineering)

Công nghệ thông tin: Information Technology

- Thuật ngữ *Công nghệ thông tin* xuất hiện ở Việt nam vào những năm 90.
- Theo ITAA: *Information Technology Association of America*
  - ***CNTT** là ngành nghiên cứu các hệ thống thông tin dựa vào máy tính, đặc biệt là các phần mềm ứng dụng và phần cứng máy tính.*
  - ***CNTT** xử lý với các máy tính điện tử và các phần mềm máy tính nhằm chuyển đổi, lưu trữ, bảo vệ, truyền tin và trích rút thông tin một cách an*

Công nghệ thông tin→Các ứng dụng

- Các bài toán khoa học kỹ thuật
  - Bài toán phức tạp, cần hàng triệu phép tính/giây
- Các bài toán quản lý
  - Quản lý thông tin, CSDL, hỗ trợ quyết định
- Tự động hóa, công tác văn phòng…
- Y tế, Giáo dục
  - Hỗ trợ trình bày bài giảng, chuẩn đoán bệnh,…
- Thương mại điện tử
  - Hỗ trợ mua bán, thanh toán qua mạng
- Các ứng dụng trong đời sống thường ngày
  - Máy móc, đồ điện tử…
- Giải trí: game, xem phim, đọc báo….
- …….

Công nghệ thông tin và truyền thông

- Khuynh hướng hiện thời
  - –Sử dụng "information" thay thế cho "data"
  - –Mở rộng cho lĩnh vực truyền thông và CNTT trở thành **CNTT&TT**
    - ICT: *Information and Communication Technology*

Trường Đại học Bách Khoa Hà nội

Công nghệ thông tin và truyền thông

Truyền thông máy tính

- Là sự kết nối một số lượng máy tính với nhau trong một phạm vi địa lý nhỏ.
- Nhiều máy tính có thể kết nối với nhau theo một phạm vi rộng hơn và việc trao đổi được thực hiện qua một mạng viễn thông nào đó.

• Internet

- *Mạng máy tính toàn cầu*: sản phẩm của ngành Công nghệ thông tin và Truyền thông.

Nội dung chính

1. Các khái niệm cơ bản về thông tin và tin học
    1. Thông tin và xử lý thông tin
    2. Máy tính điện tử và phân loại.
    3. Tin học và các ngành liên quan
2. Biểu diễn dữ liệu trong máy tính
    1. Biểu diễn số trong các hệ đếm
    2. Biểu diễn dữ liệu trong MT & đơn vị thông tin
    3. Biểu diễn số nguyên
    4. Biểu diễn số thực
    5. Biểu diễn ký tự

Nội dung

- Biểu diễn số trong các hệ đếm
- Biểu diễn dữ liệu trong máy tính và đơn vị thông tin
- Biểu diễn số nguyên
- Biểu diễn số thực
- Biểu diễn ký tự

Hệ đếm

- Tập hợp các **ký hiệu** và **qui tắc** sử dụng tập ký hiệu để **biểu diễn** và xác định các giá trị.
  - Hệ La mã: I, V, X, L, C,..
  - Quy tắc: IX, XV, XXX
- Mỗi hệ đếm sử dụng một số ký tự/chữ số (*ký số*) hữu hạn
  - Tổng số ký số của mỗi hệ đếm được gọi là **cơ số** (*base, radix*), ký hiệu là b.
  - **Ví dụ:** Hệ đếm cơ số 10,
    - 10 ký tự là: các chữ số từ 0 đến 9.

Hệ đếm

- Trên lý thuyết, có thể biểu diễn một giá trị theo hệ đếm cơ số bất kì.
- Trong tin học, quan tâm đến các hệ đếm:
  - **Hệ thập phân** (*Decimal System*)
    → Con người sử dụng
  - **Hệ nhị phân** (*Binary System*)
    → Máy tính sử dụng
  - **Hệ đếm bát phân/hệ** cơ số 8 (*Octal System*)
    → Dùng để viết gọn số nhị phân.
  - **Hệ mười sáu** (*Hexadecimal System*)
    → Dùng để viết gọn số nhị phân

Hệ đếm thập phân (*cơ số b =10*)

- Gồm 10 ký số: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9**
- Dùng n chữ số thập phân có thể biểu diễn được 10^n giá trị khác nhau:

  00...000 = 0

  ....

  99...999 = 10^n-1
- Một biểu diễn **A : $a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}$** xác định giá trị:

$$A = a_n 10^n + a_{n-1} 10^{n-1} + \cdots + a_1 10^1 + a_0 10^0 + a \quad 10^{-1} + \cdots + a_{-m} 10^{-m}$$

$$= \sum_{i=-m}^{n-1} a_i 10^i$$

Hệ đếm thập phân (*cơ số b =10*)→Ví dụ

- Biểu diễn: **5246**
  - Có giá trị được tính như sau:

$$5246 = 5 \times 10^3 + 2 \times 10^2 + 4 \times 10^1 + 6 \times 10^0$$

- Biểu diễn **254.68**
  - Có giá trị được tính như sau:

$$254.68 = 2 \times 10^2 + 5 \times 10^1 + 4 \times 10^0 + 6 \times 10^{-1} + 8 \times 10^{-2}$$

Hệ đếm cơ số b

- **Điều kiện**: b nguyên, và b ≥ 2
- Có b ký tự để thể hiện giá trị số.
  - Ký số nhỏ nhất là **0** và lớn nhất là **b-1**
- Biểu diễn A_b : $\mathbf{a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}}$
  - **n+1** ký số biểu diễn cho phần nguyên và **m** ký số biểu diễn cho phần lẻ
  - Xác định giá trị

$$A = a_n b^n + a_{n-1} b^{n-1} + \cdots + a_1 b^1 + a_0 b^0 + a_{-1} b^{-1} + \cdots + a_{-m} b^{-m} = \sum_{i=-m}^{n} a_i b^i$$

Hệ đếm nhị phân (binary system, b= 2)

- Sử dụng 2 chữ số (*nhị phân*): **0,1**
- Chữ số nhị phân gọi là ***bit*** (**bi**nary digi**t)**
  - bit là đơn vị thông tin nhỏ nhất
- Sử dụng n bit biểu diễn được 2^n giá trị

  $00...000_2 \Leftrightarrow 0_{10}$

  ....

  $11...111_2 \Leftrightarrow 2^n\text{-}1_{10}$

  - Ví dụ, sử dụng 3 bit biểu diễn 8 giá trị

    000 001 010 011

    100 101 110 111

Hệ đếm nhị phân (binary system, b = 2)

- Biểu diễn A_b : $\mathbf{a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}}$
  - $\mathbf{a_i}$ là các số nhị phân (**0,1**), xác định giá trị

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + \cdots + a_{-m} 2^{-m}$$
$$= \sum_{i=-m}^{n} a_i 2^i$$

Ví dụ: Số nhị phân A: $\mathbf{1101001.1011_2}$ có giá trị $A = 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$

$= 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$

$= 105.6875_{(10)}$

Hệ đếm nhị phân→Phép toán trên bit nhị phân

Phép cộng	Phép trừ
0 + 0 = 0; 1 + 0 = 0 + 1 = 1; 1 + 1 =10;	1 – 0 = 1 1 – 1 = 0; 0 – 0 = 0; 0 – 1 = 1; (*nợ 1*)
Phép nhân	**Phép chia**
0 x 0 = 1 x 0 = 0 x 1 = 0 1 x 1 = 1	

Hệ đếm nhị phân→Phép toán →Ví dụ

Phép cộng

```
  1 0 1
+ 1 1 1
---------
1 1 0 0
```

Phép trừ

```
1 1 0 0
- 1 1 1
-------
0 1 0 1
```

Hệ đếm bát phân (Octal system, b=8)

- Gồm 8 ký số: **0, 1, 2, 3, 4, 5, 6, 7**
- Dùng **n** chữ số thập phân có thể biểu diễn được $\mathbf{8^n}$ giá trị khác nhau:

00...000 = 0 (*hệ thập phân*)

....

77...777 = 8^n-1 (*hệ thập phân*)

Hệ đếm bát phân (Octal system, b=8)

- Biểu diễn A_8 : $\mathbf{a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}}$
  - $\mathbf{a_i}$ là các số bát phân (**0,1,..,7**), xác định giá trị

$$A = a_n 8^n + a_{n-1} 8^{n-1} + \cdots + a_1 8^1 + a_0 8^0 + a \quad 8^{-1} + \cdots + a_{-m} 8^{-m}$$
$$= \sum_{i=-m}^{n-1} a_i 8^i$$

Ví dụ: Số bát phân A: $\mathbf{235.64_8}$ có giá trị

A = $\mathbf{2}x8^2 + \mathbf{3}x8^1 + \mathbf{5}x8^0 + \mathbf{6}x8^{-1} + \mathbf{4}x8^{-2}$

= 157. 8125(10)

Hệ đếm thập lục phân (*Hexadecimal, b =16*)

- Gồm 16 ký số:

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- Các chữ cái A, **B, C, D, E, F** biểu diễn các giá trị số tương ứng trong hệ 10:

  10, 11, 12, 13, 14, 15

Hệ đếm cơ số b:					
2	**10**	**16**	**2**	**10**	**16**
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

Hệ đếm thập lục phân (*Hexadecimal, b =16*)

- Một biểu diễn $A_h : a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}$
  - a_i là các số hệ hexadecimal xác định giá trị:

$$A = a_n 16^n + a_{n-1} 16^{n-1} + \cdots + a_1 16^1 + a_0 16^0 + a_{-1} 16^{-1} + \cdots + a_{-m} 16^{-m} = \sum_{i=-m}^{n} a_i 16^i$$

Ví dụ: Số hexa A: $2FD.C8_h$ có giá trị

$$A = 2x16^2 + 15x16^1 + 13x16^0 + 12x16^{-1} + 8x16^{-2}$$
$$= 675_d + 0.78125_d$$
$$= 675.78125_d$$

Dùng 2 chữ số hệ 16 biểu diễn được bao nhiêu giá trị?

Chuyển đổi giữa các hệ đếm

- Hệ đếm bất kỳ (b) → Hệ đếm thập phân
- Hệ đếm thập phân → Hệ đếm bất kỳ (b)
- Từ hệ đếm b sang b^k và ngược lại (*b=2*)
- Từ hệ đếm b^k sang b^n (*thường b= 2*)

Chuyển đổi hệ cơ số **b** bất kỳ → hệ thập phân

- Biểu diễn A_b : $\mathbf{a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m}}$
  - n+1 ký số biểu diễn cho phần nguyên
  - m ký số biểu diễn cho phần lẻ
  - Các ký số $\mathbf{a_i}$ thuộc tập ký số của hệ b

  Có giá trị tương đương trong hệ 10

$$A = a_n b^n + a_{n-1} b^{n-1} + \cdots + a_1 b^1 + a_0 b^0 + a \quad b^{-1} + \cdots + a_{-m} b^{-m}$$

$$= \sum_{i=-m}^{-1} a_i \quad {}^{i}b$$

11010.11_2 = $2^4 + 2^3 + 2^1 + 2^{-1} + 2^{-2}$ = **26.75_{10}**

$1A.C_{16}$ = $1x16^1+10x16^0+12x16^{-1}$ = **26.75_{10}**

Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

- Giả thiết **R** là một giá trị số hệ 10
  - R_{10} = Phần nguyên(R) + Phần thập phân(R)
  - Ví dụ: 12.6875 = 12 + 0.6875
- Chuyển đổi R sang hệ b theo 2 bước
  - Chuyển đổi phần nguyên sang hệ cơ số b
    - Chia liên tiếp cho cơ số b cho đến khi thương số là 0
    - Kết quả là các số dư viết theo thứ tự ngược lại
  - Chuyển đổi phần thập phân sang hệ cơ số b
    - Nhân liên tiếp cho cơ số b cho tới khi phần thập phân của tích bằng 0
    - Kết quả là các phần nguyên trong các phép nhân được viết theo thứ tự tính toán

Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

Phần nguyên - Int(R)

- Chia cho cơ số b. Thương số t_0, số dư d_0
- Nếu $t_0 \neq 0$, chia t_0 cho b, được t_1 và dư d_1
- .....
- Tiếp tục cho tới khi thương số $t_n = 0$, dư d_n

Kết quả

$$Int(R_{10}) = d_n d_{n-1} \ldots d_1 d_0 \,(b)$$

Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

Ví dụ: Chuyển đổi phần nguyên 12.6875 sang hệ nhị phân

- 12 chia 2 = 6 dư 0
- 6 chia 2 = 3 dư 0
- 3 chia 2 = 1 dư 1
- 1 chia 2 = 0 dư 1

Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

Phần thập phân - Frac(R)

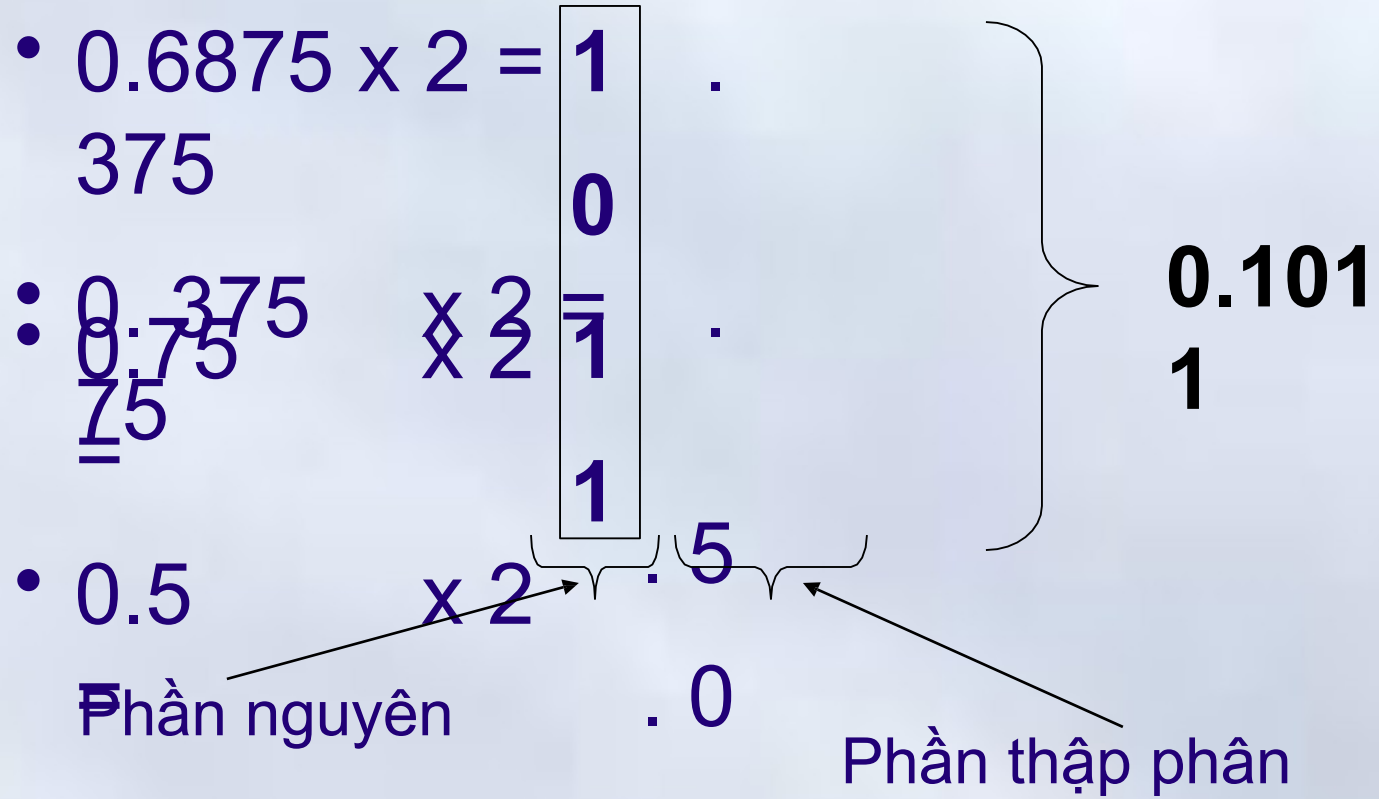
- Nhân Frac(R) với cơ số b. được phần nguyên n_1, phần thập phân t_1 (***Frac***$(R) \times b = n_1 . t_1$)
- Nếu $t_1 \neq 0$, tiếp tục nhân t_1 cho b, được $n_2 . t_2$
- …..
- Tiếp tục cho tới khi được phần thập phân $t_k = 0$, và phần nguyên n_k(được $n_k . 0$)

Kết quả

$$Frac(R_{10}) = 0.n_1n_2\ldots n_{n-1}n_k\ (b)$$

Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

- **Ví dụ:** Chuyển đổi phần thập phân 12.6875 sang hệ nhị phân



Chuyển đổi hệ thập phân → hệ cơ số **b** bất kỳ

Chuyển đổi số 45.375 sang

- Hệ nhị phân
  - 45 → 101101
  - 0.375 → 0.011
  - 45.375 → 101101.011
- Hệ thập lục phân
  - 45 → 2D
  - 0.375 → 0.6
  - 45.375 →2D.6

Chuyển đổi hệ thập phân → hệ cơ số **2**

Nguyên tắc tính nhanh

- Tách số nguyên cần chuyển đổi thành tổng các lũy thừa của 2.
- Các bit của số nhị phân tương ứng sẽ mang giá trị 1/0 tùy theo có thành phần lũy thừa tương ứng trong tổng

Ví dụ

$$67 = 64 + 2 + 1 = 2^6 + 2^1 + 2^0$$

$$= 1\,0\,0\,0\,0\,1\,1_{(2)}$$

$$117 = 64 + 32 + 16 + 4 + 1 =$$

$$2^6 + 2^5 + 2^4 + 2^2 + 2^0$$

Chuyển đổi hệ đếm b ↔ b^k

- Một số hệ **b^k** tương ứng với **k** chữ số hệ **b**
  - Hệ bát phân: 2^3 → tương ứng 3 số hệ nhị phân
  - Hệ Hexa 2^4 → tương ứng 4 số hệ nhị phân
- **Từ b^k → b**
  - Thay mỗi số hạng trong số hệ ***b^k*** bằng ***k*** số trong hệ cơ số b
- **Từ b → b^k**
  - Nhóm từng k số hạng trong số hệ b, kể từ dấu phẩy (.) thập phân về 2 phía.
    - Thêm số 0 ở đầu và cuối số hệ b nếu cần
  - Thay mỗi nhóm bằng một số hệ b^k

Chuyển đổi hệ đếm b ↔ b^k → Ví dụ

25.34_8 = 010 101.011 100_2=10101.0111_2

$5A.2C_{16}$ = 0101 1010 . 0010 1100_2

= 1011010.001011_2

$1011111010 1.01101_2$

= 010 111 110 101.011 010_2

= 2765.32_8

$1011111010 1.01101_2$

= 0101 1111 0101.0110 1000_2

= $5F5.68_{16}$

Chuyển đổi hệ đếm $b^k \rightarrow$ hệ b^n

Nguyên tắc: sử dụng hệ trung gian b

$$b^k \rightarrow b \rightarrow b^n$$

Ví dụ:

1234.67_o $= 001\ 010\ 011\ 100\ .\ 110\ 111_b$

$= 0010\ 1001\ 1100\ .\ 1101\ 11\mathbf{00}_b$

$= 29C.DC_h$

$5D.4C_h$ $= 0101\ 1101\ .\ 0100\ 1100_b$

$= \mathbf{0}01\ 011\ 101\ .\ 010\ 01\ 100_b$

$= 135.23_o$

Bài tập trên lớp

Chuyển sang hệ nhị phân và hệ thập lục phân

- **124.75**
  - Hệ nhị phân :
  - Hệ thập lục phân:
- **65.125**
  - Hệ nhị phân:
  - Hệ thập lục phân:
- **7.3**
  - Hệ nhị phân:

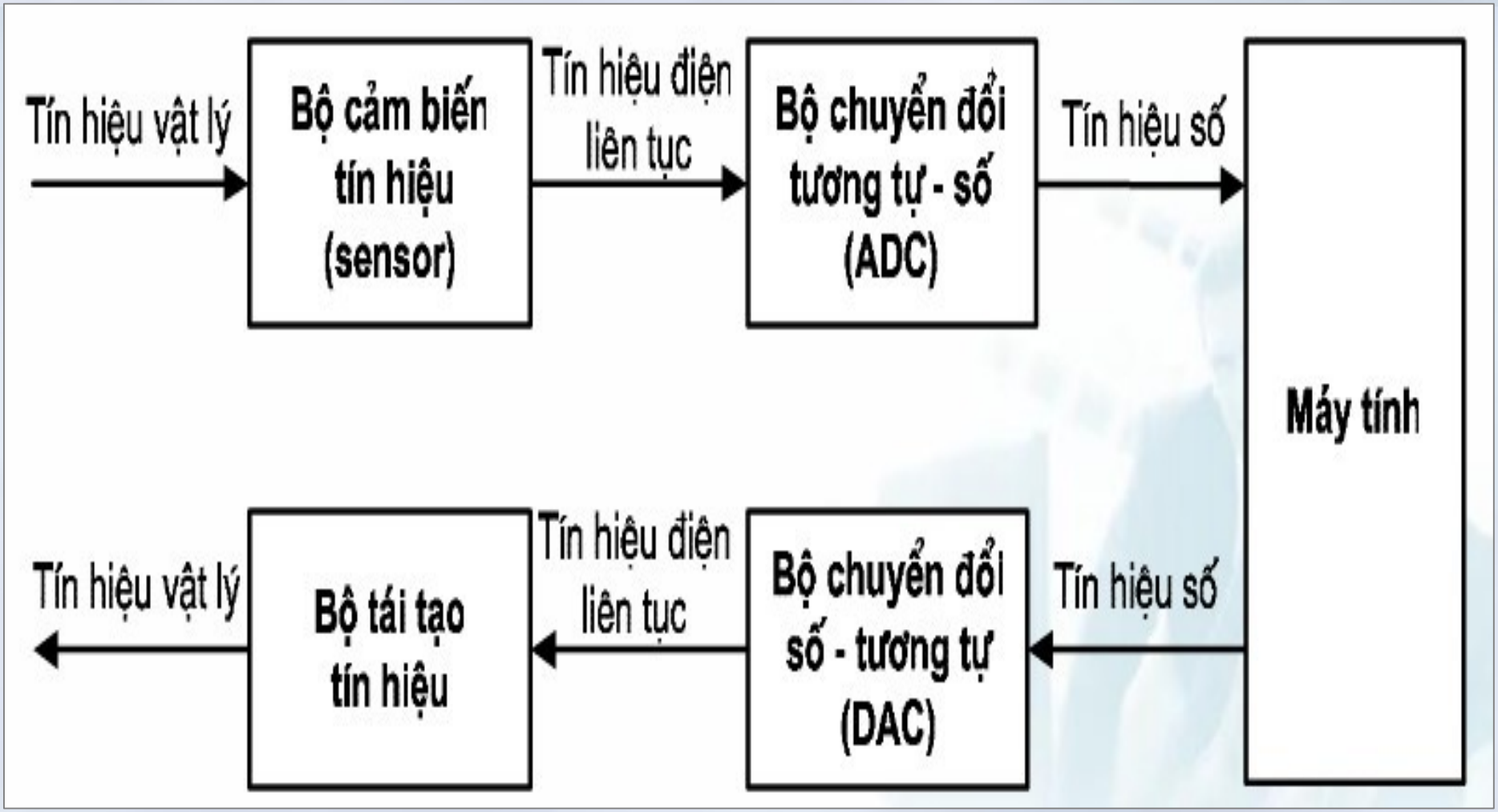
Nội dung

- Biểu diễn số trong các hệ đếm
- Biểu diễn dữ liệu trong máy tính và đơn vị thông tin
- Biểu diễn số nguyên
- Biểu diễn số thực
- Biểu diễn ký tự

Mã hóa dữ liệu cho máy tính

- Dữ liệu đưa vào máy tính phải được mã hóa thành số nhị phân
- Các loại dữ liệu:
  - Dữ liệu nhân tạo: Do con người quy ước
  - Dữ liệu tự nhiên:
    - Tồn tại khách quan với con người.
    - Phổ biến là các tín hiệu vật lý: âm thanh, hình ảnh,…
- Nguyên tắc mã hóa dữ liệu
  - Dữ liệu nhân tạo:
    - Dữ liệu số: Mã hóa theo các chuẩn quy ước
    - Dữ liệu ký tự: Mã hóa theo bộ mã ký tự
  - Dữ liệu tự nhiên:
    - Cần phải số hóa trước khi đưa vào máy tính

Sơ đồ mã hóa và tái tạo tín hiệu vật lý



Dữ liệu trong máy tính

- Dữ liệu cơ bản
  - Số nguyên, số thực, ký tự
- Dữ liệu có cấu trúc
  - Mảng, xâu ký tự, tập hợp, bản ghi

Dữ liệu cơ bản

- Số nguyên:
  - Không dấu: Biểu diễn theo mã nhị phân
  - Có dấu: Biểu diễn dưới dạng mã bù hai.
- Số thực:
  - Biểu diễn bằng số dấu chấm (phẩy) động.
- Ký tự:
  - Biểu diễn bằng mã ký tự trên các bộ mã ký tự.

Độ dài từ dữ liệu:

- Số bit được sử dụng để mã hóa loại dữ liệu tương ứng
- Thực tế, độ dài từ dữ liệu thường là bội số của

Dữ liệu có cấu trúc

- Là tập hợp các loại dữ liệu cơ bản được cấu thành theo một cách nào đó.
  - Ví dụ: kiểu dữ liệu mảng, kiểu xâu ký tự, kiểu tập hợp, bản ghi,…
  - *Các dữ liệu có cấu trúc sẽ được nghiên cứu cụ thể trong phần học về ngôn ngữ lập trình.*

Đơn vị đo thông tin

- Bit (**BI**nary digi**T**)
  - Là đơn vị thông tin nhỏ nhất
    - Mỗi bit ứng với một sự kiện có 2 trạng thái
      - ***Ví dụ:*** Khóa điện có thể tắt khi mạch hở, bật khi mạch đóng
  - Nhận một trong hai giá trị nhị phân 0/1
- Byte (**B**): Chuỗi 8 bit
  - Kilobyte (**KB**): 1 KB = 2^{10}B =
  - Megabyte 1024B (**MB**): 1MB = 2^{10}KB
  - Gigabyte =2^{20}B
  - Terabyte (**GB**): 1GB =2^{10}MB = 2^{20}KB = 2^{30}B
  - Petabyte (**TB**): 1TB = 2^{10}GB = …= 2^{40}B
  - (**PB**): 1TB = 2^{10}TB = …= 2^{50}B

Nội dung

- Biểu diễn số trong các hệ đếm
- Biểu diễn dữ liệu trong máy tính và đơn vị thông tin
- Biểu diễn số nguyên
- Biểu diễn số thực
- Biểu diễn ký tự

Nguyên tắc

- Dùng một chuỗi bít để biểu diễn
- Trường hợp số nguyên có dấu
  - Sử dụng bit đầu tiên (*Most significant bit*) để biểu diễn dấu
    - Bit đầu tiên được gọi là bit dấu

Số nguyên không dấu

- Dùng n bit để biểu diễn cho một số nguyên không dấu **A: $a_{n-1}a_{n-2}...a_2a_1a_0$**
  - Trong đó **a_i** là các số nhị phân (**0,1**)
- Giá trị của A được tính :

$$A = a_{n-1}2^{n-1} + \cdots + a_1 2^1 + a_0 2^0$$
$$= \sum_{i=0}^{n-1} a_i 2^i$$

- Dải biểu diễn :[0..2^n-1]

Số nguyên không dấu→Ví dụ

Dùng 8 bit, biểu diễn các số nguyên không dấu

- **A = 45**

A = 45 = 32 + 8 + 4 + 1

= $2^5 + 2^3 + 2^2 + 2^0$

→ = 0010 1101

- **B = 156**

A = 128 + 16 + 8 + 4

B = 156 = $2^7 + 2^4 + 2^3 + 2^2$

Số nguyên không dấu→Ví dụ

Tính các số nguyên không dấu được biểu diễn bằng 8 bit

- **0010 1011**

$X = 0010\ 1011 = 2^5 + 2^3 + 2^1 + 2^0$

$= 32 + 8 + 2 + 1 = 43$

- **1001 0110**

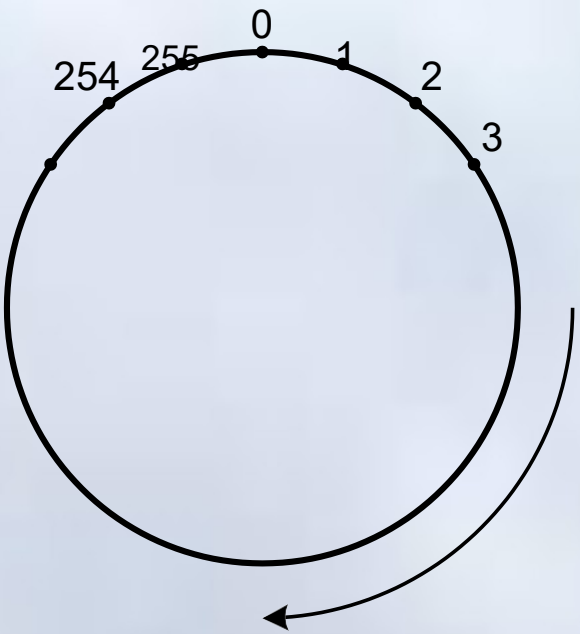
$Y = 1001\ 0110 = 2^7 + 2^4 + 2^2 + 2^1$

$= 128 + 16 + 4 + 2 = 150$

Vấn đề tràn số

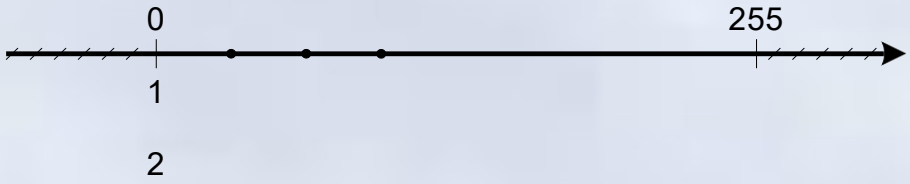
- Xét trường hợp dùng 8 bit biểu diễn số nguyên
  - Phạm vi biểu diễn: [0-255]

00000000 → 0
00000001 → 1
00000010 → 2
.....
11111110 →254
11111111 →255



Trục số học máy tính

Trục số học



Vấn đề tràn số → Chú ý

- Tràn số có thể gây sai sót trong viết chương trình cho máy tính
  - Tránh tràn số, dùng nhiều bit để biểu diễn

```
{ unsigned char N; //Kiểu số nguyên không dấu, 1 byte
      N = 255;
  N = N + 1;
  printf(« %d »,N); //Ra kết quả 0 , do tràn số
} //Nếu dùng unsigned int(số 2 byte)có kết quả256,
```

- Dùng 16 bit, phạm vi biểu diễn: $0 \div 2^{16}-1$ (65535)
- Dùng 32 bit, phạm vi biểu diễn: $0 \div 2^{32}-1$
- Dùng 64 bit, phạm vi biểu diễn: $0 \div 2^{64}-1$

Số nguyên có dấu

- Dùng n bit để biểu diễn cho một số nguyên có dấu **A: $a_{n-1}a_{n-2}...a_2a_1a_0$**
  - Trong đó **a_i** là các số nhị phân (**0,1**)
- Giá trị của A được tính :

$$A = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

- Dải biểu diễn :[$-2^{n-1}..2^{n-1}-1$]
  - -2^{n-1} : 1000…000 //bit dấu bằng 1
  - $2^{n-1}-1$: 0111…111 //bit dấu bằng 0

Số nguyên có dấu → Nhận xét

- Xét số nguyên có dấu n bit $\mathbf{a_{n-1}a_{n-2}...a_2a_1a_0}$
  - Nếu tất cả bit $\mathbf{a_i}$ mang giá trị 1, giá trị là

$$- 2^{n-1} + 2^{n-1} - 1 = - 1$$

- Nhận xét [$\mathbf{a_{n-1}a_{n-2}...a_2a_1a_0}$]
  - Nếu bit a_{n-1} có giá trị 0, biểu diễn giá trị dương
  - Nếu bit a_{n-1} có giá trị 1, giá trị âm sẽ phụ thuộc các bit mang ý nghĩa dương: $a_{n-2},..,a_1, a_0$
    - Các bit dương đều bằng 0→ số âm nhỏ nhất: $\mathbf{-2^{n-1}}$
    - Các bit dương đều bằng 1→ số âm lớn nhất: **- 1**

Số nguyên có dấu → Ví dụ

Tính các số nguyên có dấu được biểu diễn bằng 8 bit :

- **A = 0101 0110**

$$A = 2^6 + 2^4 + 2^2 + 2^1$$
$$= 64 + 16 + 4 + 2$$
$$= +86$$

- **B = 1101 0010**

$$B = -2^7 + 2^6 + 2^4 + 2^1$$
$$= -128 + 64 + 16 + 2$$
$$= -46$$

Chuyển số nguyên có dấu về dạng biểu diễn của máy tính

- **Số bù một** của một số nhị phân A n bit là một số thu được bởi nghịch đảo tất cả các bit trong A
  - A là số nhị phân n bit
  - A^r là số bù 1 (*nghịch đảo bit*) của A
    - $A+A^r$ = 111…..111 (n bit)
- **Số bù hai** của một số nhị phân A n bit là số bù 1 của A + 1
- Xét số 8 bit 10110011
  - Số bù 1: 01001100
  - Số bù 2: 01001100

Chuyển số nguyên có dấu về dạng biểu diễn của máy tính

- Với số nhị phân n bit
  - $A+A^r+1 = 111....111+1=1\ 000…000(n+1\ bit)$ $=0$
  - Vậy A^r+1 chính là -A
- **Nhận xét**: *Số nguyên có dấu n bit –A được biểu diễn bởi số bù 2 của A*

Ví dụ: số nguyên có dấu 8 bit: A = -70

Biểu diễn 70 = 0100 0110

Bù 1 của 70: 1011 1001

\+ 1

Bù 2 của 70: 1011 1010

Vậy: **A = 10111010**

Bài tập trên lớp

Biểu diễn dưới dạng nhị phân 8 bit các số

- 46 :
- -46 :

Biểu diễn dưới dạng nhị phân giá trị sau

-159

Tính toán số học với số nguyên

- Cộng trừ số nguyên không dấu
- Cộng/trừ số nguyên có dấu
- Nhân chia các số nguyên không dấu
- Các phép toán logic với số nhị phân

Cộng trừ số nguyên không dấu

Nguyên tắc

- Tiến hành cộng/trừ lần lượt từng bít từ phải qua trái.
- Khi cộng hai số nguyên không dấu n bits, thu được một số nguyên không dấu cũng n bits.
  - Nếu tổng của hai số đó lớn hơn 2^n-1 $\Rightarrow$tràn số và kết quả sẽ là sai.
- Trường hợp số bị trừ, nhỏ hơn số trừ
  - Kết quả không hợp lệ (*tràn số dưới*)

Ví dụ

- Dùng 8 bit biểu diễn số nguyên không dấu
- Trường hợp không xảy ra tràn số (carry-out):
  - X = 1001 0110 = 150
  - Y = 0001 0011 = 19
  - S = 1010 1001 = 169
  - Cout = 0
- Trường hợp có xảy ra tràn số (carry-out):
  - X = 1100 0101 = 197
  - Y = 0100 0110 = 70
  - S = 0000 1011 $\neq$ 267 (267 = 256+11)
  - Cout = 1 $\rightarrow$ carry-out

Cộng/trừ số nguyên có dấu

- **Phép cộng:**
  - Thực hiện lần lượt các cặp bit từ phải qua trái, bỏ qua bit tràn số (nếu có).

11011011	(-37)
01001011	(75)
- - - - - - - -	- - - - -
00100110	(38)

- **Phép trừ:** X – Y = X + (-Y)
  - Lấy bù hai của Y
  - Cộng X với (–Y) theo nguyên tắc phép cộng.

Cộng/trừ số nguyên có dấu → Nhận xét

- Cộng hai số khác dấu: kết quả luôn đúng
- Cộng hai số cùng dấu:
  - Tổng nhận được cùng dấu với 2 số hạng: Kết quả là đúng
  - Tổng nhận được khác dấu với 2 số hạng: Đã xảy ra hiện tượng *tràn số học* (Overflow) và kết quả nhận được là sai
- Tràn số học xảy ra khi tổng của hai số nằm ngoài dải biểu diễn của số nguyên có dấu n bit: $[-2^{n-1}, 2^{n-1}-1]$

Ví dụ

- Không tràn số số

```
  X = 0100 0110 =  +70
+ Y = 0010 1010 =  +42
  S = 0111 0000 = +112
```

```
  X = 1010 0110 =  -90
+ Y = 0010 0100 =  +36
  S = 1100 1010 =  -54
```

- Bỏ qua bit tràn

```
  X = 0110 0001 =  +97
+ Y = 1100 1100 =  -52
  S = 0010 1101 =  +45
```

$C_{out} = 1 \rightarrow$ bỏ qua

```
  X = 1011 0110 =  -74
+ Y = 1110 0010 =  -30
  S = 1001 1000 = -104
```

$C_{out} = 1 \rightarrow$ bỏ qua

Ví dụ

- Có xảy ra tràn số, kết quả sai:

$$
\begin{array}{ll}
 & X = 0100\ 1011 = +75 \\
+ & Y = 0101\ 0001 = +81 \\
\hline
 & S = 1001\ 1100 \neq +156
\end{array}
$$

$(S = -2^7 + 2^4 + 2^3 + 2^2 = -100)$

$$
\begin{array}{ll}
 & X = 1001\ 1000 = -104 \\
+ & Y = 1011\ 0110 = -74 \\
\hline
 & S = 0100\ 1110 \neq -178
\end{array}
$$

$C_{out} = 1 \rightarrow$ bỏ qua

$(S = 2^6 + 2^3 + 2^2 + 2^1 = 78)$

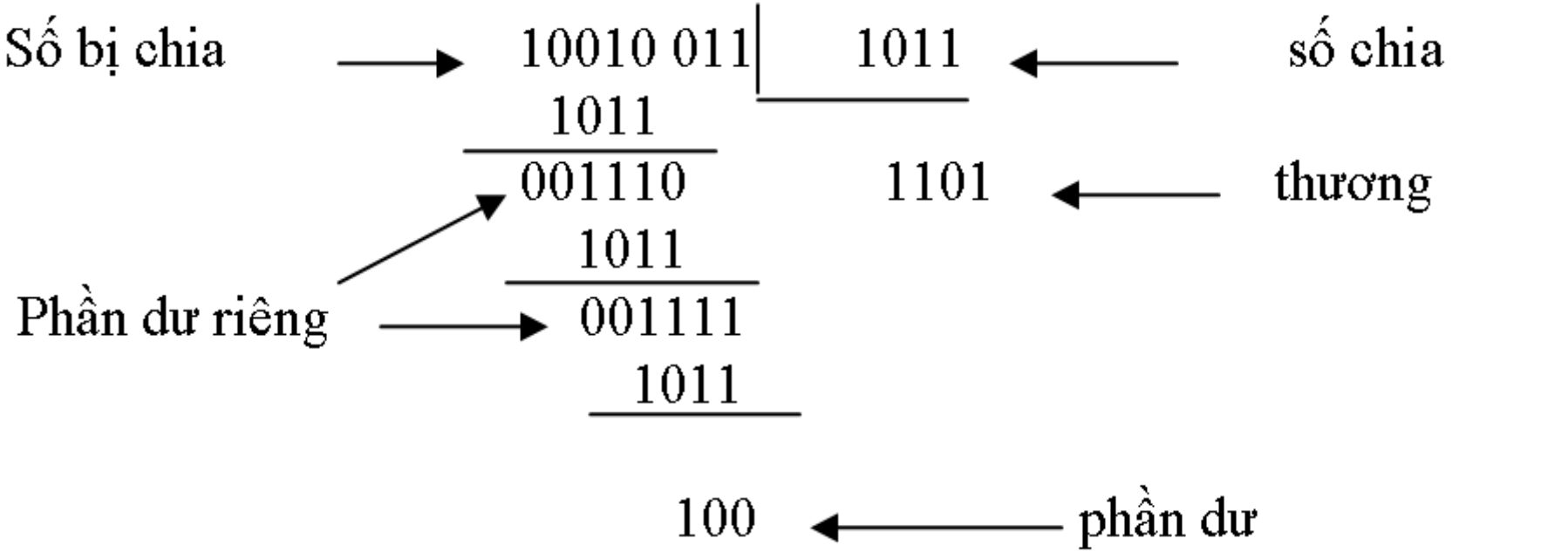
Nhân/chia số nguyên không dấu

- Các bước thực hiện như trọng hệ 10.
- **Phép nhân**

```
          1011
        (11 cơ số 10)
x     ----------
          1101
        (13 cơ số 10)

         1011
      ---------
        0000
       1011
      1011
```

Nhân/chia số nguyên không dấu

- **Phép chia**



- Nhân/chia cho lũy thừa của cơ số !?

Các phép toán logic với số nhị phân

		AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

	NOT
0	1
1	0

- Thực hiện các phép toán logic với 2 số nhị phân:
  - Thực hiện các phép toán logic với từng cặp bit của 2 số
    - Các phép toán logic chỉ tác động lên từng cặp bit, không ảnh hưởng đến bit khác.
  - Kết quả là 1 số nhị phân

Ví dụ

		NOT
A	1010 1010	01010101
B	0000 1111	11110000
AND	**00001010**	
OR	**10101111**	
XOR	**10100101**	

- Phép **AND** dùng để xoá một số bit và giữ nguyên các bit còn lại (Ví dụ: **AND** 11**0**11**0**11)
- Phép **OR** dùng để thiết lập 1 số bit và giữ nguyên các bit khác. (Ví dụ: **OR** 00**1**00**1**00)

Nội dung

- Biểu diễn số trong các hệ đếm
- Biểu diễn dữ liệu trong máy tính và đơn vị thông tin
- Biểu diễn số nguyên
- **Biểu diễn số thực**
- Biểu diễn ký tự

Nguyên tắc chung

- Số thực trong máy tính được biểu diễn thông qua ký pháp dấu phẩy động (*Floating Point Number)*
- Một số thực V được biểu diễn theo ký pháp dấu phẩy động như sau: $V = M * R^E$

  Trong đó:
  - **R** là cơ số (*Radix*) thường là 2 hoặc 10.
  - **M** là phần định trị (*Mantissa*) thỏa mãn $-R < M < R$
  - **E** là phần mũ (*Exponent*)
- Nhận xét:
  - Nếu R cố định, để lưu trữ V chỉ cần lưu trữ M và E (*dưới dạng số nguyên*)

Ví dụ

Với cơ số R = 10

$$V = -52.5 = -5.2500 \times 10^{1}$$

– Phần định trị: M = -5.2500 / Phần số mũ: E = 1

Ghi chú

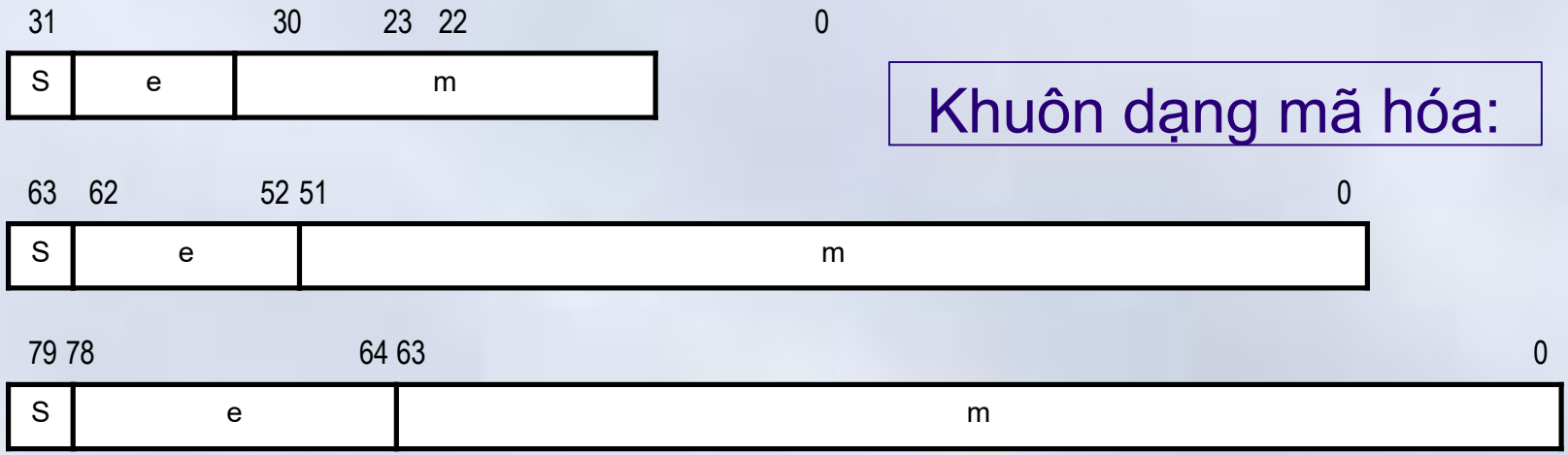
- Để chuẩn hóa, phần định trị luôn nằm trong khoảng (-R,R) và số đầu tiên là khác 0 (*nếu được*)
  – -52.5×10^{0} và -0.5250×10^{2} tương đương với -5.25×10^{1} nhưng không phải số chuẩn hóa
  – Việc chuẩn hóa nhằm duy trì độ chính xác và biểu diễn
  các số rất lớn/bévà biểu diễn các số rất lớn/bé
    - 0.000162 biểu diễn với 4 chữ số thập phân $\Rightarrow$0.0001
    - $+0.001 \times 10^{9} = +0.000.000.001$

Phép toán với dấu phẩy động

- Được tiến hành trên cơ sở các giá trị của phần định trị và phần mũ
- R_1 và R_2 là số dấu phẩy động:
  - $R_1 = M_1 \times R^{E_1}$ và $R_2 = M_2 \times R^{E_2}$
- Việc thực hiện các phép toán số học sẽ được tiến hành:
  - $R_1 \pm R_2 = (M_1 \times R^{E_1-E_2} \pm M_2) \times R^{E_2}$,
  - Giả thiết $E_1 \geq E_2$
  - $R_1 \times R_2 = (M_1 \times M_2) \times R^{E_1+E_2}$
  - $R_1 / R_2 = (M_1 / M_2) \times R^{E_1-E_2}$

Chuẩn IEEE 754/85

- Là chuẩn mã hóa số dấu phẩy động
- Cơ số R = 2
- Các dạng cơ bản:
  - Dạng có độ chính xác đơn, 32-bit
  - Dạng có độ chính xác kép, 64-bit
  - Dạng có độ chính xác kép mở rộng, 80-bit



Khuôn dạng mã hóa:

Chuẩn IEEE 754/85 → Mã hóa

- **S** là bit dấu,
  - S = 0 đó là số dương/ S=1 đó là số âm.
- **e** là mã lệch (excess) của phần mũ E

  E = e – b

  Trong đó b là độ lệch (bias):
  - Dạng 32-bit : b = 127, hay E = e - 127
  - Dạng 64-bit : b = 1023, hay E = e - 1023
  - Dạng 80-bit : b = 16383, hay E = e – 16383

Chuẩn IEEE 754/85 → Mã hóa

$$R = (-1)^S \times 1.m \times 2^{e-b}$$

Chuẩn IEEE 754/85 → Ví dụ 1

- Dạng biểu diễn nhị phân số thực R theo chuẩn IEEE 754 dạng 32 bit:

  1100 0001 0101 0110 0000 0000 0000 0000

S = 1 → R là số âm

- e = 1000 0010 = 130
  - Số 32 bit: b = 127 ⇒ E = 130 - 127
- m = 10101100...00 ⇒ M=1.m
- Vậy R = $(-1)^1 \times 1.10101100...00 \times 2^{130-127}$

Chuẩn IEEE 754/85 → Ví dụ 2

R được biểu diễn:

0011 1111 1000 0000 0000 0000 0000 0000

- s = 0 → R là số dương
- e = 0111 1111= 127
- m = 000000...00
- $R = (-1)^0 \times 1.0000...00 \times 2^{127-127}$

  $= 1.0 \times 2^0 = 1$

Chuẩn IEEE 754/85 → Ví dụ 3

- Biểu diễn số thực R = 9.6875 về dạng số dấu chấm động theo chuẩn IEEE 754 dạng 32 bit

$R = 9.6875_{(10)} = 1001.1011_{(2)} = 1.0011011 \times 2^3$

Ta có:

– R là số dương → **s = 0**

– E = 3, b = 127 (biểu diễn 32 bit) → e = 127 + 3 = $130_{(10)}$

e = 1000 0010$_{(2)}$

– M = 1.0011011 → **m = 001101100...00** (23 bit)

Biểu diễn nhị phân của R:

0100 0001 0001 1011 0000 0000 0000 0000

Các quy ước đặc biệt

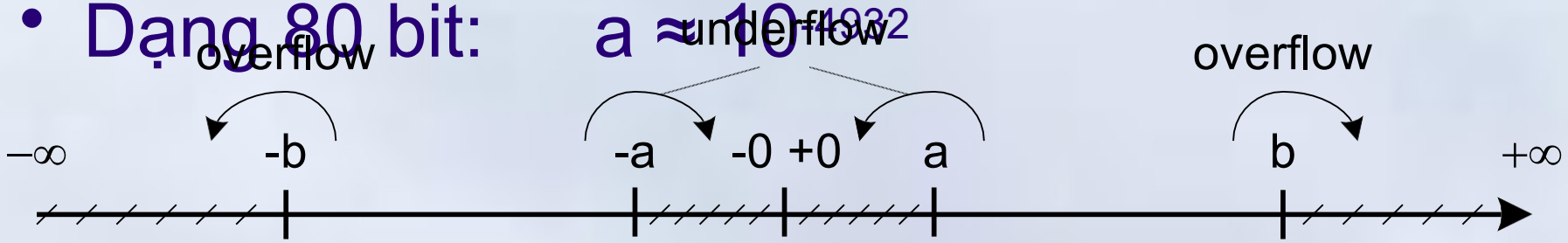
- Nếu tất cả các bit của e đều bằng 0
  - Nếu các bit của m đều bằng 0, thì X = ± 0
  - Phần định trị không cần chuẩn hóa (M ≡ m) và E = 1 - b
- Nếu tất cả các bit của e đều bằng 1,
  - Các bit của m đều bằng 0, thì X = ± ∞
  - m có ít nhất một bit bằng 1 : không phải là số (not a number - NaN)

Số thực dương bé nhất =?

Số thực dương lớn nhất = ?

Phạm vi biểu diễn

- Dạng 32 bit: $a \approx 1.18 \times 10^{-38}$ $b \approx 3.4 \times 10^{+38}$
- Dạng 64 bit: $a \approx 2.2 \times 10^{-308}$ $b \approx 1.79 \times 10^{+308}$
- Dạng 80 bit: $a \approx 10^{-4932}$ $b \approx 10^{+4932}$



Ghi chú

- Khoảng cách giữa 2 số thực liên tiếp ?
  - Sai số máy (Machine epsilon)
- Sai số làm tròn
  - 0.7 = ?
    - 0.1011001100 = 0.69921875
      - 0.1011001100011001100=0.69999980926..
    - 0.1011001101 = 0.700195312

Nội dung

- Biểu diễn số trong các hệ đếm
- Biểu diễn dữ liệu trong máy tính và đơn vị thông tin
- Biểu diễn số nguyên
- Biểu diễn số thực
- Biểu diễn ký tự

Nguyên tắc chung

- Các ký tự cần được chuyển đổi thành chuỗi bit nhị phân gọi là **mã ký tự**.
- Số bit dùng cho mỗi ký tự theo các mã khác nhau là khác nhau.

Ví dụ: Bộ mã ASCII dùng 8 bit cho 1 ký tự.

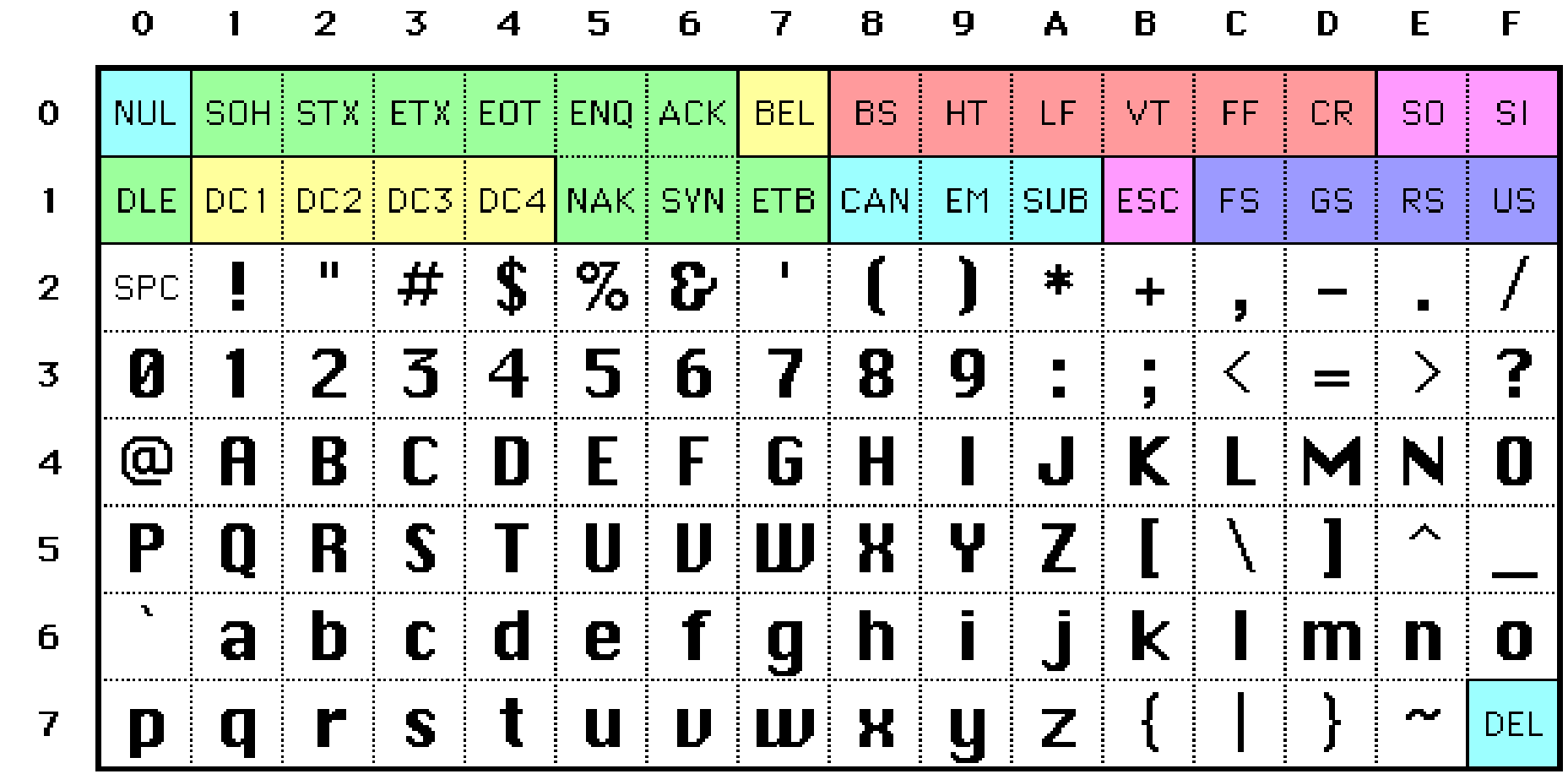
Bộ mã Unicode dùng 16 bit.

ASCII: American Standard **C**odes for **I**nformation **I**nterchangeable

Bộ mã ASCII

- Do ANSI (**A**merican **N**ational **S**tandard **I**nstitute) thiết kế
- Là bộ mã được dùng để *trao đổi thông tin chuẩn của Mỹ*.
  - Lúc đầu chỉ dùng 7 bit (128 ký tự) sau đó mở rộng cho 8 bit và có thể biểu diễn 256 ký tự khác nhau trong máy tính
- Bộ mã 8 bit → mã hóa được cho $2^8 = 256$ kí tự, có mã từ $00_{16} \div FF_{16}$, bao gồm:
  - 128 kí tự chuẩn có mã từ $00_{16} \div 7F_{16}$
  - 128 kí tự mở rộng có mã từ $80_{16} \div FF_{16}$

ASCII Chart



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Physical Device Controls: Format Effectors

Bộ mã ASCII

- **95** kí tự hiển thị được:có mã từ $20_h \div 7E_h$
  - 26 chữ cái Latin hoa 'A' ÷ 'Z', mã từ $41_h \div 5A_h$
  - 26 chữ cái Latin thường 'a' ÷ 'z', mã từ $61_h \div 7A_h$
  - 10 chữ số thập phân '0' ÷ '9', mã từ $30_h \div 39_h$
  - Các dấu câu: . , ? ! : ; …
  - Các dấu phép toán: + - * / …
  - Một số kí tự thông dụng: #, $, &, @, ...
  - Dấu cách (mã là 20_h)
- **33** mã điều khiển: $00_h \div 1F_h$ và $7F_h$
  - Điều khiển định dạng, phân cách thông tin..

Các kí tự mở rộng của bảng ASCII

- Được định nghĩa bởi:
  - Nhà chế tạo máy tính
  - Người phát triển phần mềm

Ví dụ:

- Bộ mã ký tự mở rộng của IBM: được dùng trên máy tính IBM-PC.
- Bộ mã ký tự mở rộng của Apple: được dùng trên máy tính Macintosh.
- Các nhà phát triển phần mềm tiếng Việt cũng đã thay đổi phần này để mã hoá cho các ký tự riêng của chữ Việt, ví dụ như bộ mã TCVN 5712.

Bộ mã Unicode

- Mã thống nhất/ mã đơn nhất
  - Do các hãng máy tính hàng đầu thiết kế
- Bộ mã 16-bit: Số ký tự có thể biểu diễn (mã hoá) là 2^{16}
- Được thiết kế cho đa ngôn ngữ, trong đó có tiếng Việt

Bài tập trên lớp

1. Cho A= 11010_b, B=$0A_h$. Hãy tính C=A*B
2. Hãy biểu diễn các số nguyên sau với 8 bit

   A= +58, B = -80
3. Cho A=2002_d, B=010101111111_b, C=ABC_h. Hãy sắp xếp A, B, C theo thứ tự tăng dần
4. Đổi số nguyên có dấu, 2 byte -1234 sang hệ hệ hexadecimal
5. Giá trị 1234567 có thể là số trong hệ đếm nào: 2/8/10/16?
6. Đổi giá trị 0101_o sang hệ thập phân và thập lục phân
7. Cho biết kết quả cácphép tính sau nếu chỉ dùng 8 bit để biểu diễn các giá trị dưới dạng số nguyên có

Nội dung chính

Chương 1: Thông tin và biểu diễn thông tin

- Các khái niệm cơ bản về thông tin và tin học
- Biểu diễn dữ liệu trong máy tính

Chương 2: Hệ thống máy tính

- Hệ thống máy tính
- Mạng máy tính
- Hệ điều hành

Chương 3: Các hệ thống ứng dụng

- Hệ thống thông tin quản lý
- Hệ thông tin bảng tính
- Hệ quản trị cơ sở dữ liệu
- Các hệ thống thông minh

Nội dung chính

1. Hệ thống máy tính
    1. Tổ chức bên trong máy tính
    2. Phần mềm máy tính
2. Mạng máy tính
    1. Lịch sử phát triển của mạng máy tính
    2. Phân loại mạng máy tính
    3. Các thành phần cơ bản của một mạng máy tính
    4. Mạng Internet
3. Giới thiệu hệ điều hành
    1. Các khái niệm cơ bản
    2. Hệ lệnh của hệ điều hành
    3. Hệ điều hành Window

Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Chức năng của hệ thống máy tính

- **Xử lý dữ liệu**
- **Lưu trữ dữ liệu**
- **Trao đổi dữ liệu**
- **Điều khiển**

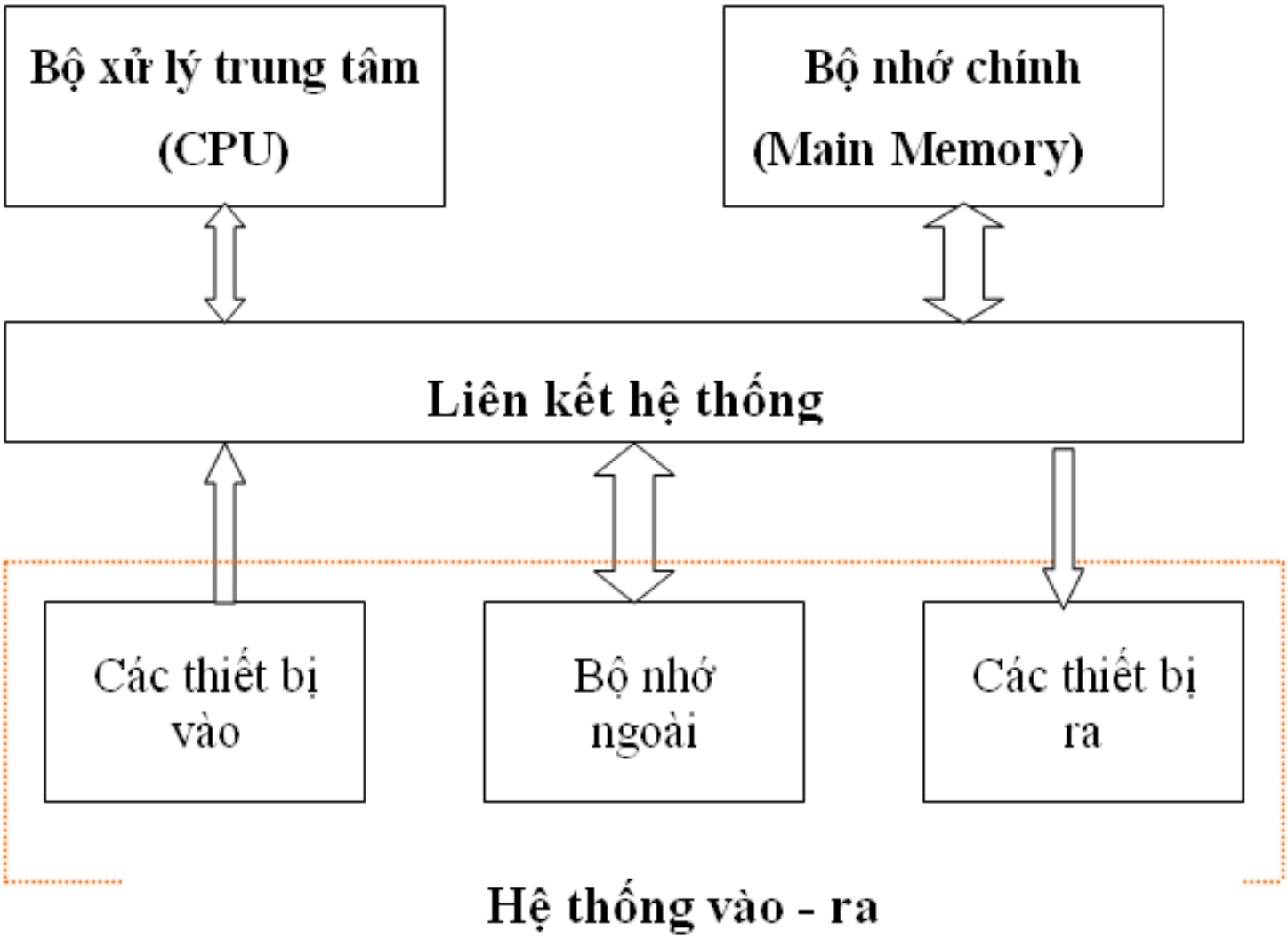
Chức năng của hệ thống máy tính

- **Xử lý dữ liệu**:
  - Chức năng quan trọng nhất của máy tính
  - Dữ liệu có thể có rất nhiều dạng khác nhau và có yêu cầu xử lý khác nhau.
- **Lưu trữ dữ liệu**:
  - Dữ liệu đưa vào máy tính được xử lý ngay hoặc có thể được lưu trong bộ nhớ.
  - Khi cần chúng sẽ được lấy ra xử lý.

Chức năng của hệ thống máy tính

- **Trao đổi dữ liệu**:
  - Giữa các thành phần bên trong và bên ngoài thông qua thiết bị ngoại vi
    - Là quá trình vào ra (input-output)
    - Các thiết bị vào-ra được coi là nguồn cung cấp dữ liệu hoặc nơi tiếp nhận dữ liệu.
  - Khi dữ liệu được vận chuyển trên khoảng cách xa với các thiết bị hoặc máy tính gọi là *truyền dữ liệu* (data communication).
- **Điều khiển**:
  - Máy tính cần phải điều khiển ba chức năng trên

Cấu trúc của hệ thống máy tính



Cấu trúc của hệ thống máy tính

- Bộ xử lý trung tâm – CPU (Central Processor Unit)
  - Điều khiển các hoạt động của máy tính và thực hiện xử lý dữ liệu.
- Bộ nhớ chính (Main Memory)
  - Lưu trữ chương trình và dữ liệu.
- Hệ thống vào ra (Input-Output System):
  - Trao đổi thông tin giữa thế giới bên ngoài với máy tính.
- Liên kết hệ thống (System Interconnection):

– Kết nối và vận chuyển thông tin giữa CPU, bộ nhớ chính và hệ thống vào ra của máy tính với nhau.

Hoạt động của máy tính

- Hoạt động cơ bản của máy tính là thực hiện chương trình.
- Chương trình gồm một tập các lệnh được lưu trữ trong bộ nhớ

Thực hiện chương trình

Lặp lại *chu trình lệnh*, bao gồm các bước

- CPU phát ra địa chỉ ô chớ chứa lệnh
- CPU nhận lệnh từ bộ nhớ, đưa về thanh ghi lệnh
- Tăng nội dung con trỏ lệnh để trỏ tới lệnh tiếp
- CPU giải mã lệnh, để xác định thao tác
- Nếu lệnh sử dụng dữ liệu từ bộ nhớ hay cổng vào ra, cần xác định địa chỉ nới chứa dữ liệu
- CPU nạp các dữ liệu cần thiết vào các thanh ghi
- Thực thi lệnh
- Ghi kết quả vào nơi được yêu cầu
- Quay lại bước đầu tiên để thực hiện lệnh tiếp

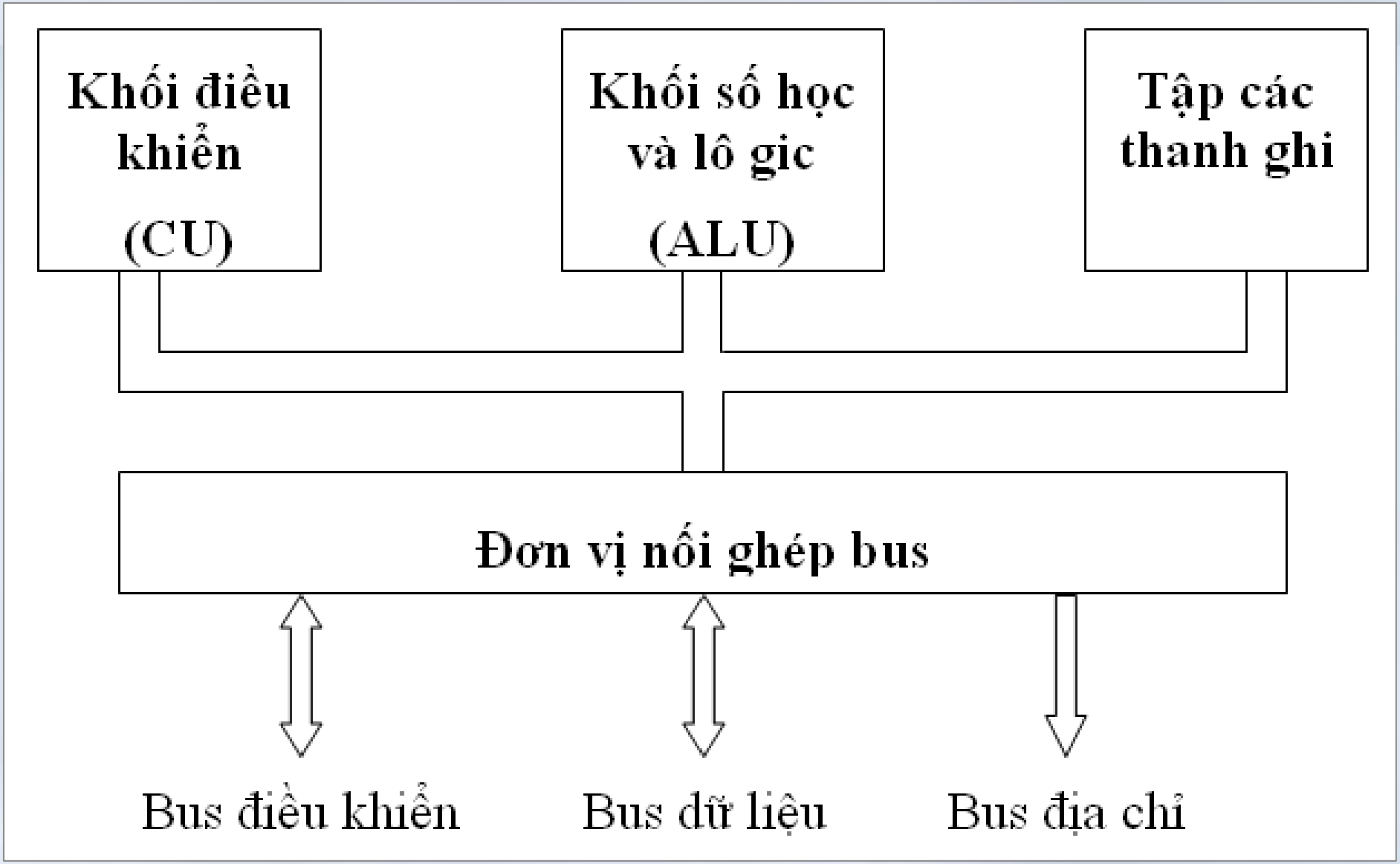
Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Chức năng và hoạt động

- Chức năng
  - Điều khiển hoạt động của toàn bộ hệ thống máy tính
  - Xử lý dữ liệu
- Nguyên tắc hoạt động:
  - Hoạt động theo chương trình nằm trong bộ nhớ chính, bằng cách:
    - Nhận lần lượt lệnh từ bộ nhớ chính
    - Tiến hành giải mã lệnh và phát các tín hiệu điều khiển thực thi lệnh
    - Trong quá trình thực thi lệnh, CPU có thể trao đổi dữ liệu với bộ nhớ chính hay hệ thống vào-ra.

Cấu trúc cơ bản của CPU



Các thành phần cơ bản của CPU

- **Khối điều khiển** (*Control Unit* – CU)
  - Nhận lệnh từ bộ nhớ trong đưa vào CPU
  - Giải mã các lệnh
  - Tạo ra các tín hiệu điều khiển các thành phần khác theo chương trình đã định sẵn
- **Khối tính toán số học và logic** (*Arithmetic – Logic Unit* - ALU):
  - Bao gồm các thiết bị thực hiện các phép tính số học, logic (AND, OR..) và các phép tính so sánh
  - Dữ liệu được lấy từ các thanh ghi, sau khi tính toán, được ghi trở lại các thanh ghi
  - ALU thường được tăng cường thêm khối tính toán dấu phẩy động (FPU)-còn gọi là bộ đồng xử lý

Các thành phần cơ bản của CPU

- **Tập các thanh ghi** (*Register File - RF*)
  - Lưu trữ thông tin tạm thời phục vụ hoạt động của CPU
    - Dữ liệu từ bộ nhớ, thiết bị vào ra chuyển vào các thanh ghi
    - ALU tính toán trên các dữ liệu của thanh ghi
    - Dữ liệu sau khi thính toán sẽ chuyển từ thanh ghi về bộ nhớ hay thiết bị vào ra
  - Số lượng và kích thước các thanh ghi phụ thuộc CPU
- **Bus bên trong (**Internal Bus**)**
  - Kết nối các thành phần bên trong CPU với nhau
- **Đơn vị ghép nối bus** (Bus Interface Unit – BIU**)**
  - Kết nối và trao đổi thông tin với nhau giữa các bus bên trong với các bus bên ngoài.
- **Đồng hồ (**Clock**)**
  - Tạo xung nhịp đồng bộ hóa các bộ phận

Bộ vi xử lý (Microprocessoer)

- CPU được chế tạo trên một vi mạch và được gọi là bộ vi xử lý.
  - Các bộ vi xử lý hiện nay có cấu trúc phức tạp hơn nhiều so với một CPU cơ bản.
- Tốc độ của bộ vi xử lý
  - Số lệnh được thực hiện trong 1s (**MIPS:** Milliions of Intructions per Second)
  - Khó đánh giá chính xác
- Tần số xung nhịp của bộ xử lý
  - Bộ xử lý hoạt động theo một xung nhịp (clock) có tần số xác định
  - Tốc độ của bộ xử lý được đánh giá gián tiếp thông qua tần số xung nhịp (Ví dụ: Tần số *2GHz* $\Rightarrow$*Chu kỳ 0.5*η*s*)

Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Chức năng và thành phần

- Chức năng
  - Lưu trữ thông tin cần thiết trong quá trình xử lý
- Các thành phần chính
  - Bộ nhớ trong (Internal Memory)
    - Bộ nhớ chính (main memory)
      - ROM: Read Only Memory
      - RAM: Random Access Memory
  - Bộ nhớ ngoài (External Memory)

Bộ nhớ trong (internal memory)

- Cho phép CPU có thể trao đổi trực tiếp thông tin ghi trong đó
  - Các mã lệnh thực thi, các dữ liệu đang xử lý..
- Tốc độ rất nhanh
- Dung lượng không lớn

Các loại bộ nhớ trong

- Bộ nhớ chính
  - Thành phần quan trọng nhất với bộ nhớ trong, có thể đồng nhất với bộ nhớ trong
- Bộ nhớ cache (bộ nhớ đệm nhanh)

Bộ nhớ chính (main memory)

- Là thành phần nhớ tồn tại trên mọi hệ thống máy tính
- Chứa các chương trình và dữ liệu đang được CPU sử dụng
- Tổ chức thành các ngăn nhớ được đánh địa chỉ; Số bit dùng để đánh địa chỉ quyết định dung lượng tối đa bộ nhớ (n bit, 2^n)
  - Pentium III có 36 bít địa chỉ ⇒ quản lý 64GB
- Ngăn nhớ thường được tổ chức theo Byte
- Nội dung ngăn nhớ có thể thay đổi, song địa chỉ vật lý của ngăn nhớ luôn cố định
- Thông thường, bộ nhớ chính gồm 2 phần:
  - ROM (Read Only Memory)
  - RAM (Random Access Memory)

Nội dung	Địa chỉ
00101011	0000
11010101	0001
00001010	0010
01011000	0011
11111011	0100
00001000	0101
11101010	0110
00000000	0111
10011101	1000
00101010	1001
11101011	1010
00000010	1011
00101011	1100
00101011	1101
11111111	1110
10101010	1111

ROM & RAM

- ROM (*Read Only Memory*)
  - Là bộ nhớ chỉ đọc, dữ liệu được ghi sẵn từ nơi sản xuất
  - Dữ liệu không bị mất khi tắt máy
  - Dùng lưu trữ chương trình hệ thống, chương trình điều khiển việc nhập xuất cơ bản (**ROM-BIOS**: ROM Basic Input-Output System)

- RAM (*Random Access Memory*)
  - Bộ nhớ truy xuất ngẫu nhiên
  - Dùng lưu trữ dữ liệu và chương trình trong quá trình tính toán của CPU
  - Thông tin trong RAM bị mất đi khi tắt máy

Bộ đệm (cache memory)

- Là thành phần nhớ tốc độ truy nhập nhanh
  - Tốc độ cache nhanh hơn bộ nhớ
  - Dung lượng cache thường nhỏ hơn bộ nhớ chính
- Được đặt đệm giữa CPU và bộ nhớ chính
  - Mục đích: tăng tốc độ truy cập bộ nhớ của CPU
    - Thông tin trong bộ nhớ được sao chép tạm thời vào cache
    - CPU tìm trong cache trước khi tìm trong bộ nhớ chính
- Cache thường được chia ra thành một số mức:
  - Cache L1, Cache L2,…
- Hiện nay cache được tích hợp trên các chip VXL
- Cache có thể có hoặc không

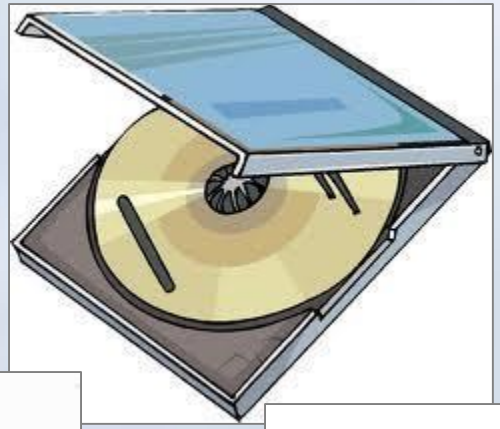
Bộ nhớ ngoài (external memory)

- Cho phép lưu trữ thông tin với dung lượng lớn
  - Tốc độ truy nhập chậm
- Thông tin không bị mất khi mất điện
  - Lưu trữ và di chuyển bộ nhớ ngoài độc lập với máy tính
- Thông tin thường là lưu dữ liệu và chương trình
  - Để sử dụng, cần phải đọc các thông tin trên bộ nhớ ngoài vào bộ nhớ trong
- Bộ nhớ ngoài được kết nối với máy tính thông qua mô đun nối ghép vào ra
  - Về chức năng: Bộ nhớ ngoài là bộ nhớ
  - Về cấu trúc: Bộ nhớ ngoài thuộc hệ thống vào ra

Bộ nhớ ngoài (external memory)

Floppy disk

Compact disk



USB flash drive

Hard disk

External hard disk

Compact flash card

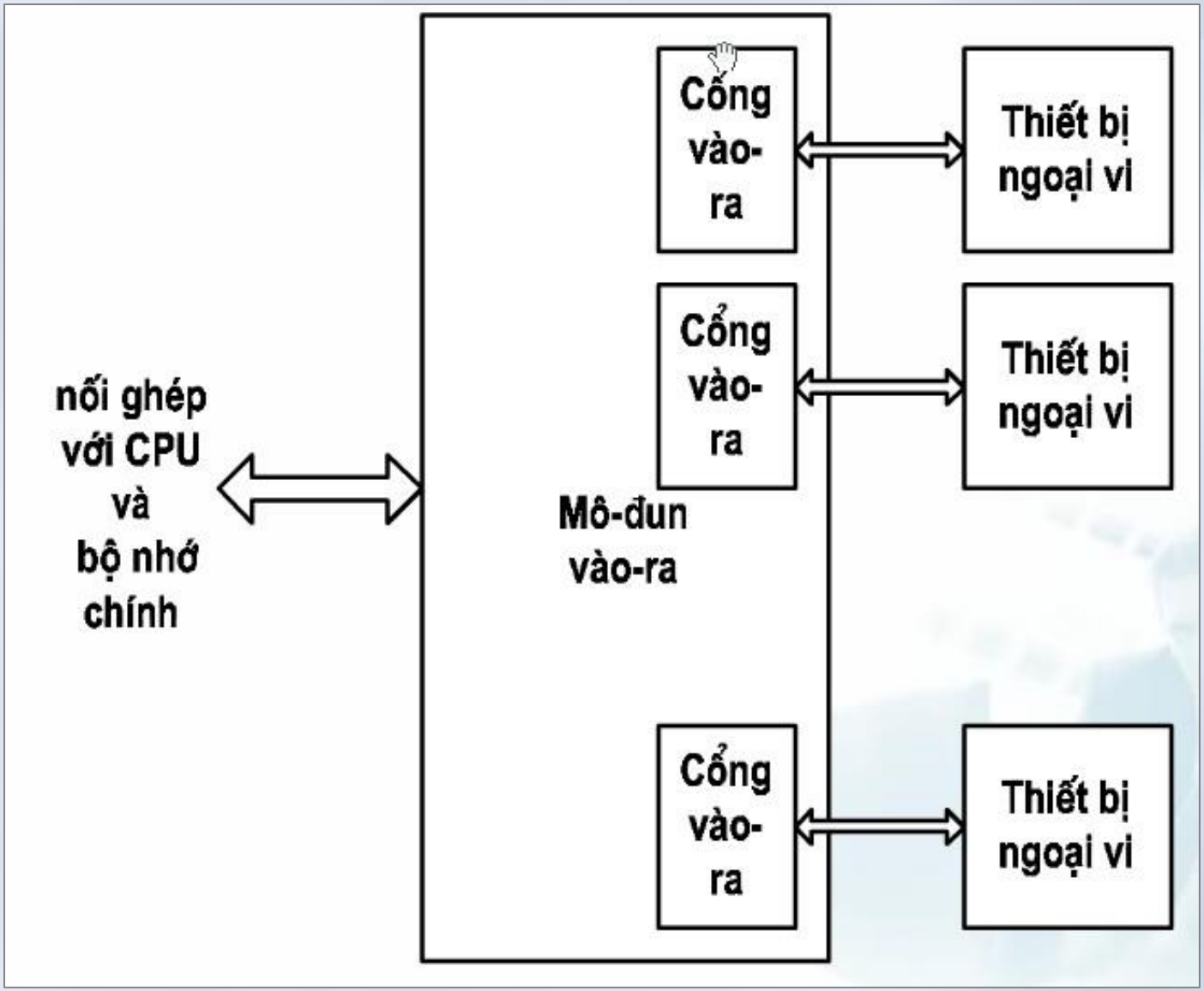
Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Chức năng và hoạt động

- Trao đổi thông tin giữa máy tính với thế giới bên ngoài.
- Các thành phần chính:
  - Các ***thiết bị vào-ra*** (IO devices) hay còn gọi là thiết bị ngoại vi (Peripheral devices)
  - Các ***mô-đun ghép nối vào-ra*** (IO Interface modules)

Cấu trúc cơ bản



Mô đun ghép nối vào ra

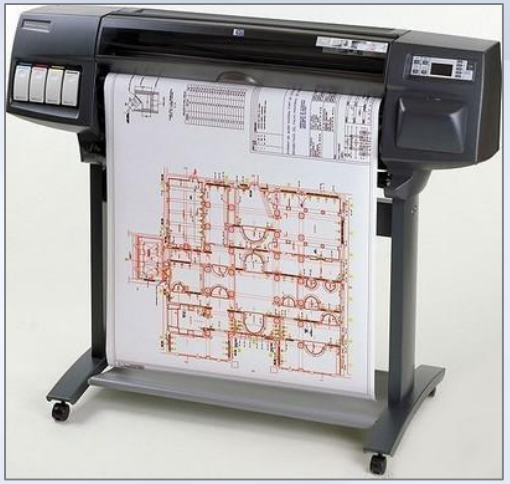
- Thiết bị vào ra không kết nối trực tiếp với CPU
  - Được kết nối thông qua các mô-đun ghép nối vào/ra.
- Các mô đun ghép nối vào-ra có các cổng vào-ra
  - Cổng vào ra: IO Port
- Các cổng vào ra được đánh địa chỉ bởi CPU
  - Mỗi cổng cũng có một địa chỉ xác định.
- Mỗi thiết bị vào-ra kết nối với CPU thông qua cổng tương ứng với địa chỉ xác định.

Các thiết bị vào ra

- Chức năng
  - Chuyển đổi thông tin từ dạng vật lý về dạng dữ liệu phù hợp với máy tính hoặc ngược lại
- Các thiết bị thông dụng
  - Thiết bị vào: Bàn phím, chuột, máy quét,…
    - Thiết bị vào chuẩn: Bàn phím
  - Thiết bị ra: Màn hình, máy in,…
    - Thiết bị ra chuẩn: Màn hình
  - Thiết bị nhớ: Các ổ đĩa,…
  - Thiết bị truyền thông: Modem

Các thiết bị vào

Các thiết bị ra



Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Mục đích

- CPU, bộ nhớ chính và hệ thống vào-ra cần được kết nối để trao đổi thông tin khi hoạt động
  - Nhiệm vụ kết nối được thược đảm bảo bởi một tập các đường kết nối, gọi là bus
- Các bus trong máy tính khá phức tạp,
  - Bus được thể hiện bằng các đường dẫn trên các bản mạch, các khe cắm trên bản mạch chính, các cáp nối,..
- Độ rộng của bus
  - Số đường dây của bus có thể truyền thông tin đồng thời.
- Về chức năng, bus được chia làm 3 loại chính:
  - Bus địa chỉ
  - bus dữ liệu
  - bus điều khiển

Các loại bus

- **Bus điều khiển** (Control bus)
  - Chuyển các tín hiệu điều khiển từ thành phần này đến thành phần khác
    - Ví dụ: CPU phát tín hiệu đọc/ghi tới bộ điều khiển bộ nhớ
- **Bus dữ liệu** (Data bus)
  - Chuyển tải dữ liệu từ CPU (*các thanh ghi*) tới bộ nhớ (*các ngăn nhớ*) và ngược lại hoặc từ bộ nhớ/CPU tới các thiết bị ngoại vi
  - Là loại bus 2 chiều
  - Các máy tính hiện tại thường có 32/64 đường bít dữ liệu
- **Bus địa chỉ** (Address bus)
  - Xác lập địa chỉ của ngăn nhớ hoặc cổng vào ra mà CPU muốn đọc/ghi dữ liệu
  - Độ rộng bus địa chỉ cho biết kích thước tối đa bộ nhớ.

Máy tính cá nhân

Các thành phần cơ bản

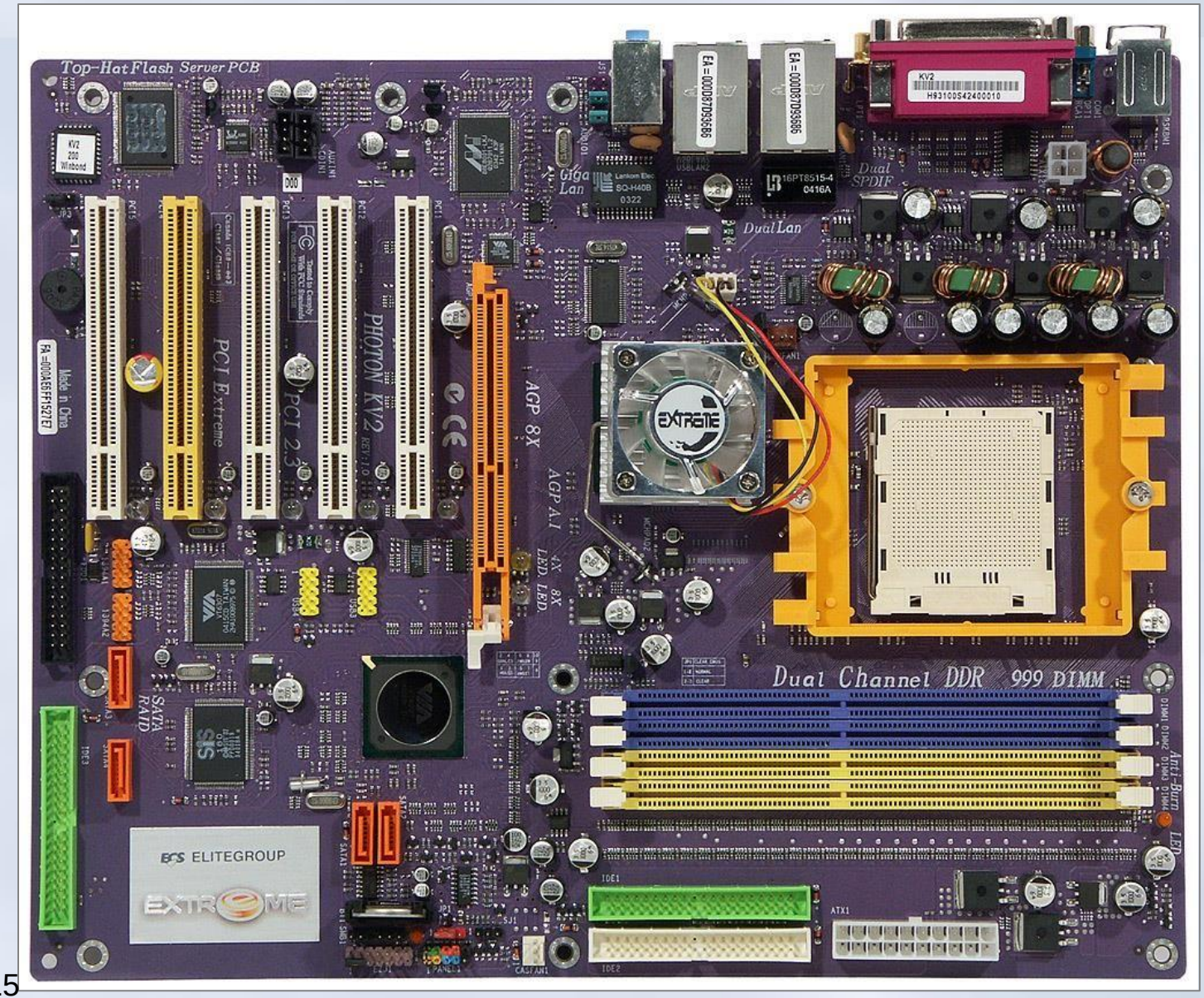
Máy tính cá nhân

Các thành phần

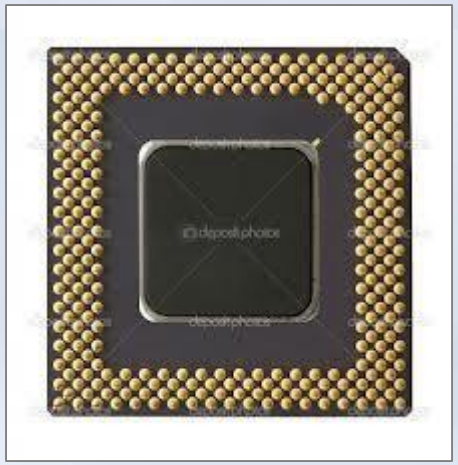
- Hộp máy tính (Case):
  - Bản mạch chính (Mainboard):
    - Bộ vi xử lý
    - Bộ nhớ hệ thống: chip nhớ ROM và các module nhớ RAM
    - Các vi mạch điều khiển tổng hợp (chipset)
    - Các khe cắm mở rộng
    - Các kênh truyền tín hiệu (bus)
  - Các loại ổ đĩa: ổ đĩa cứng, ổ đĩa mềm, ổ đĩa quang, ...
  - Các cổng vào-ra
  - Bộ nguồn và quạt
- Các thiết bị ngoại vi (Peripheral Devices):
  - Màn hình (monitor), bàn phím (keyboard), chuột (mouse)
  - Loa (speaker), máy in (printer), máy quét ảnh (scanner), modem, projector,...

Hộp máy tính (case)

Bản mạch chính

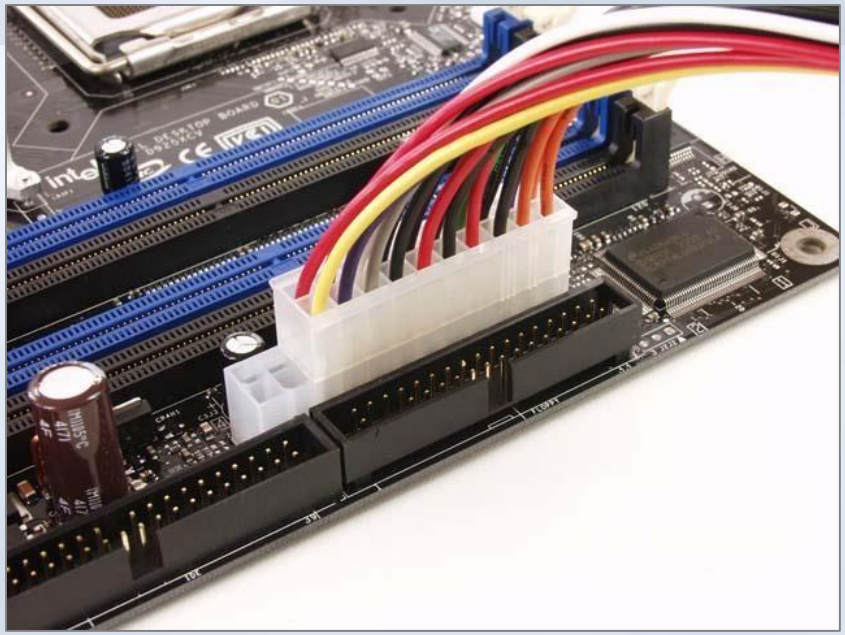


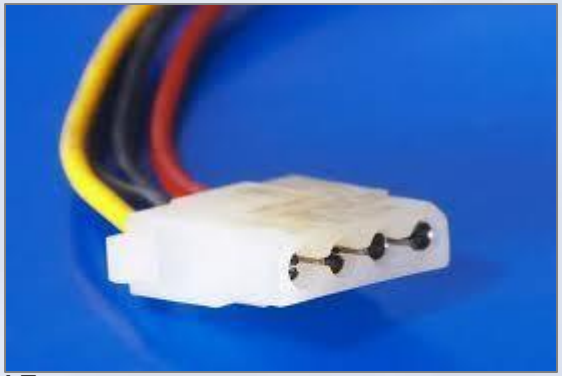
Bộ Vi xử lý

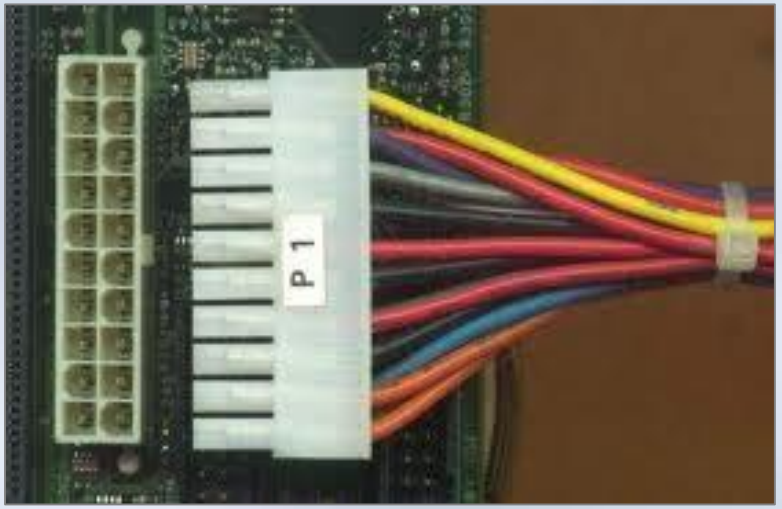


Bộ nhớ hệ thống

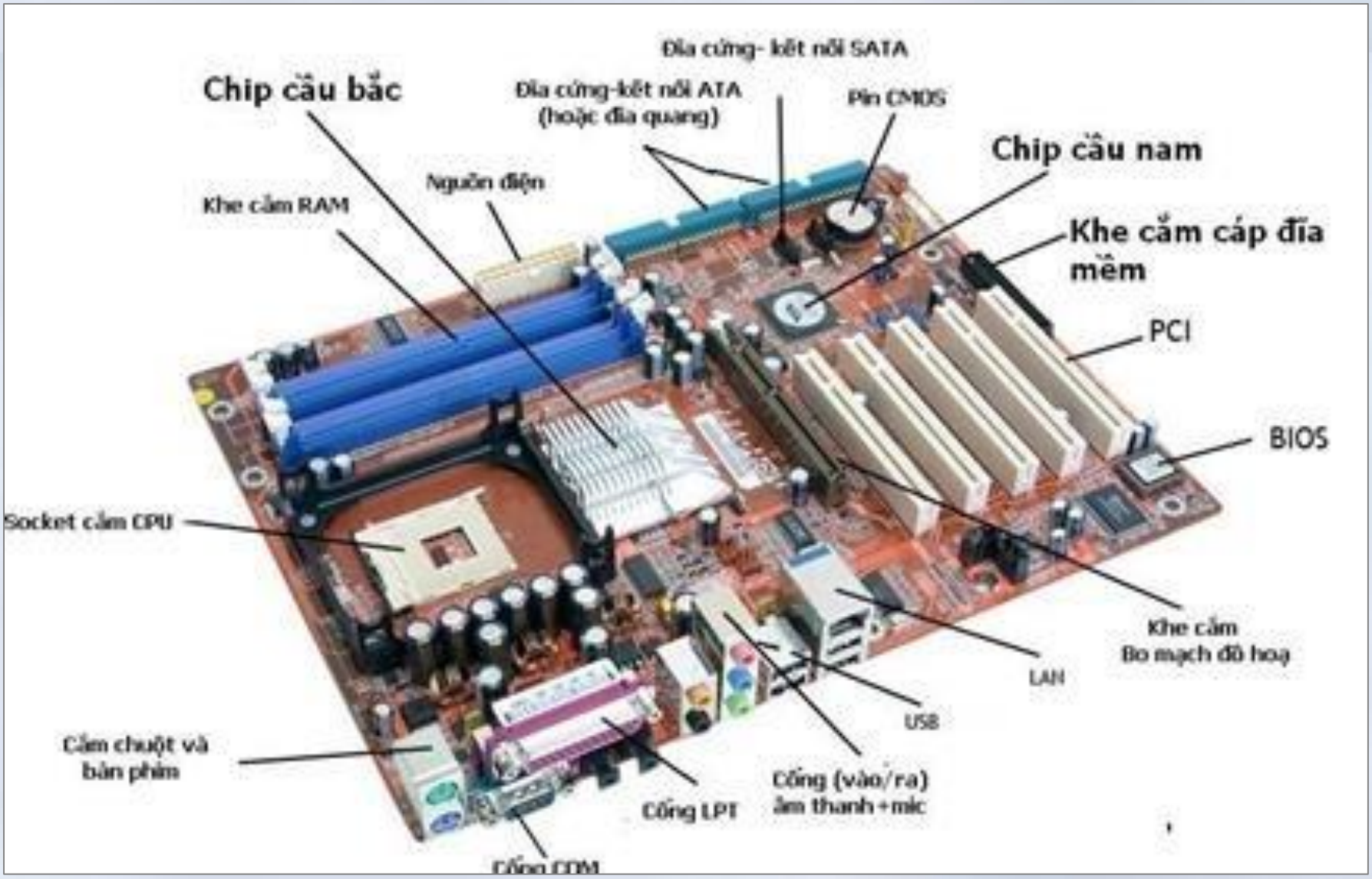
Bộ nguồn và dây nguồn





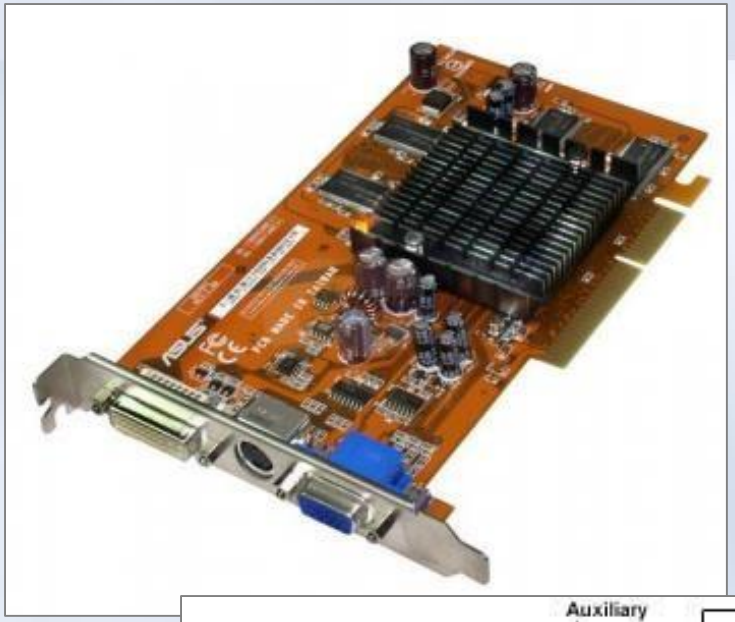


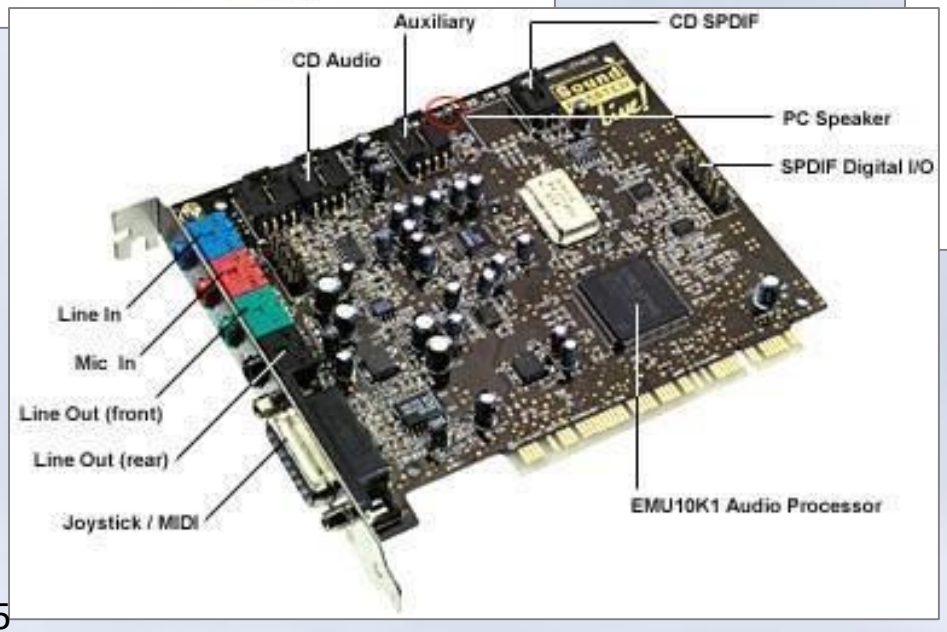
Các khe cắm mở rộng



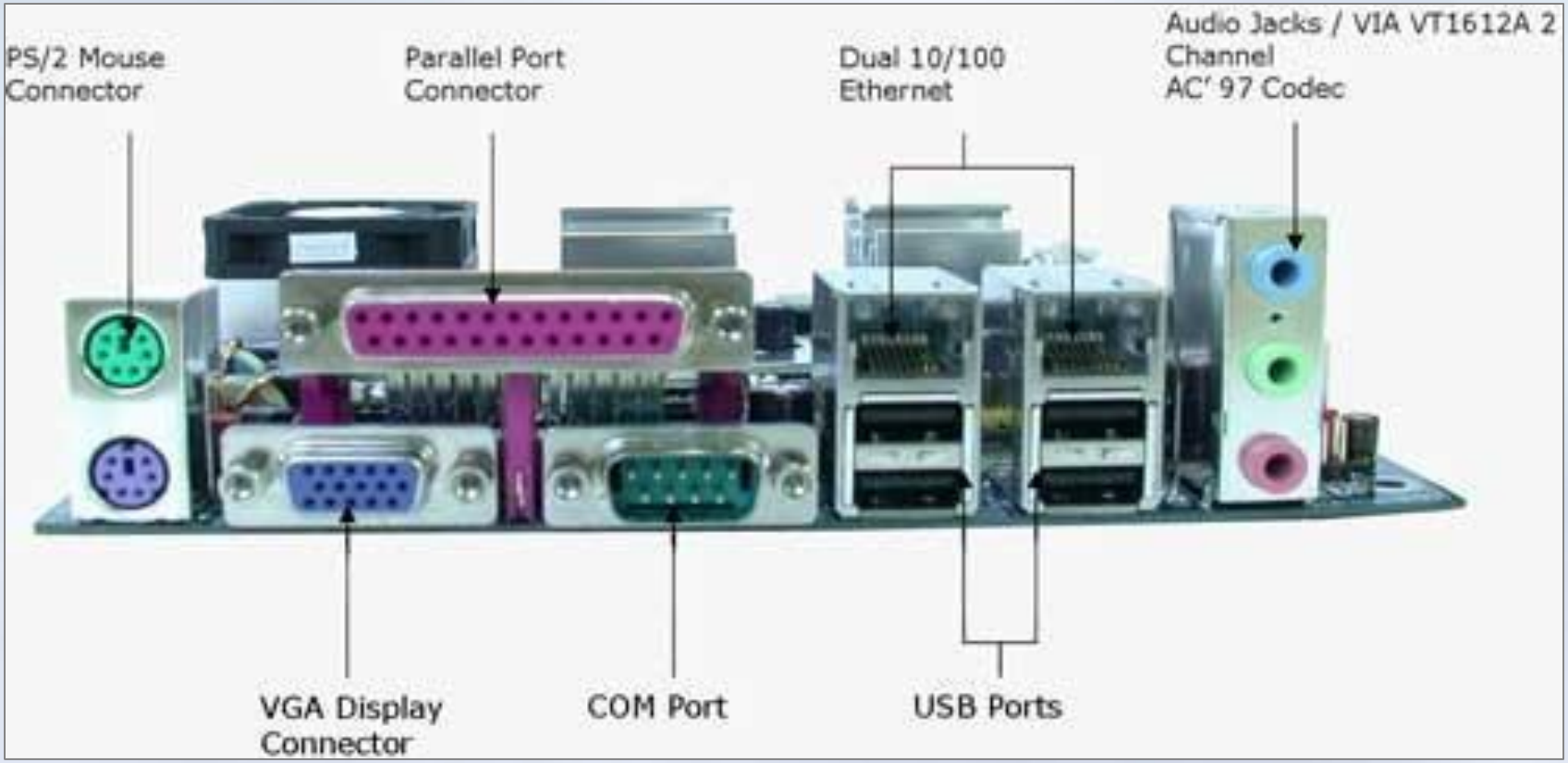
http://tip4pc.com/mainboard/

I/O card




Auxiliary
CD SPDIF
CD Audio
PC Speaker
SPDIF Digital I/O
Line In
Mic In
Line Out (front)
Line Out (rear)
Joystick / MIDI
EMU10K1 Audio Processor

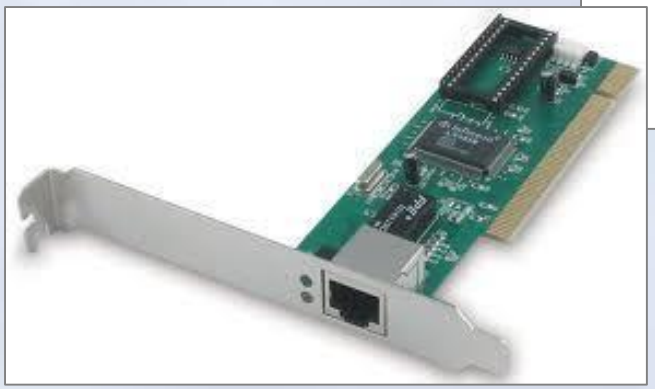
Các cổng vào ra



Các loại ổ đĩa

Các thiết bị vào ra thông dụng

Các thiết bị mạng



Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Giới thiệu

- Máy tính hoạt động theo một qui trình tự động, định sẵn gọi là chương trình (*program*)
  - Mỗi nhiệm vụ/bài toán (task/problem) của người dùng cần một quy trình/ chương trình
- Phần mềm máy tính (*Computer Software*)
  - Là khái niệm tương đương song thường mang ý nghĩa rộng hơn

Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phần mềm máy tính

Dữ liệu & Giải thuật

Mỗi bài toán phải giải quyết thường bao gồm 2 phần:

- Phần *dữ liệu*
  - Liên quan đến các thông tin cụ thể của bài toán: Dữ liệu vào, dữ liệu ra
- Phần *xử lý*:
  - Thao tác phải được máy tính tiến hành nhằm đáp ứng yêu cầu người dùng
  - **Xác đinh các thao tác cần thực hiên ⇒ Xây dưng giải thuật/Thuật toán (Algorithm)**

Dữ liệu

Dữ liệu:

- Đầu vào (Input):
  - Các dữ liệu được cung cấp để xử lý
  - Ví dụ: kết quả học tập của sinh viên
- Đầu ra (Output):
  - Kết quả xử lý
  - Ví du: Danh sách sinh viên phải học lại

Biểu diễn dữ liệu là vấn đề phức tạp

- Kết quả học tập được lưu trữ trong máy tính ntn?

Giải thuật

- Xây dựng thuật toán /giải thuật (*algorithm*)
  - Xác định các thao tác xử lý cần thiết để đạt được yêu cầu của nhiệm vụ
- *Khái niệm thuật toán*
  - *Thuật toán để giải một bài toán là một dãy hữu hạn các thao tác và trình tự thực hiện các thao tác đó sao cho sau khi thực hiện dãy thao tác này theo trình tự đã chỉ ra, với đầu vào (input) ta thu được kết quả đầu ra (output) mong muốn.*

Mô tả giải thuật/thuật toán

1. Diễn giải tuần tự các bước
2. Lưu đồ/sơ đồ khối
3. Các ngôn ngữ lập trình như Pascal, C/C++ hay Java. Tuy nhiên, không nhất thiết phải sử dụng đúng ký pháp của các ngôn ngữ đó mà có thể được bỏ một số ràng buộc.
4. Giả ngữ (p*seudocode*) gọi là ngôn ngữ mô phỏng chương trình PDL (*Programming Description Language*).

Ví dụ

Tìm giá trị lớn nhất của một dãy số nguyên có N số

- **Đầu vào:** Số số nguyên dương N và N số nguyên a_1, a_2,…, a_N
- **Đầu ra:** số nguyên lớn nhất của dãy ak, k trong khoảng [1…N]

Phương pháp:

- Khởi tạo giá trị Max = a_1
- Lần lượt so sánh Max với a_i với i = 2,3,…, N; nếu a_i > Max ta gán giá trị mới cho Max

Ví dụ → Diễn giải tuần tự các bước

- B1: Nhập N và dãy số $a_1, a_2,\ldots,a_N$.
- B2: Max ← a_1; i=2.
- B3: Nếu i > N, Nhảy tới B7
- B4: Nếu a_i > Max, Max ← a_i
- B5: Tăng i lên 1 đơn vị.
- B6: Quay lên B3.
- B7: Hiển thị Max là giá trị lớn nhất của dãy
- B8: Kết thúc.

Các tiêu chí giải thuật cần thỏa mãn

- **Tính xác định**
  - Các bước trong thuật toán phải chính xác rõ ràng, không gây sự nhập nhằng nhầm lẫn
- **Tính hữu hạn:**
  - Phải dừng sau một thời gian hữu hạn.
  - Khi kết thúc, phải cung cấp kết quả đúng đắn.
- **Tính hiệu quả:**
  - Thời gian tính toán nhanh
  - Sử dụng ít tài nguyên không gian như bộ nhớ, thiết bị,…
- **Tính phổ dụng:**
  - dễ hiểu, dễ cài đặt và mở rộng cho các lớp bài toán khác.

Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Chương trình và ngôn ngữ lập trình

- **Lập trình:**
  - Để máy có thể hiểu và tiến hành xử lý được ta phải biến các bước thao tác thành các chỉ thị (statement) và biểu diễn trong dạng mà máy tính hiểu được.
- **Chương trình:**
  - Giải thuật được biểu diễn dưới dạng một tập các chỉ thị của một ngôn ngữ nào đó.
- **Ngôn ngữ lập trình:**
  - Ngôn ngữ dùng để lập trình: Dùng để trao đổi với máy tính, máy tính hiểu và thực thi nhiệm vụ đã chỉ ra

Chương trình và ngôn ngữ lập trình

- Cấu trúc dữ liệu
  - Cách thức tổ chức để lưu trữ dữ liệu.:

Cấu trúc dữ liệu + thuật toán= Chương trình

Niklaus E. Wirth

Các loại ngôn ngữ lập trình (1/3)

Ngôn ngữ máy:

- Mỗi loại máy tính đều có ngôn ngữ máy riêng.
- Loại ngôn ngữ duy nhất để viết chương trình mà máy tính hiểu trực tiếp và thực hiện được.
- Các chỉ thị (lệnh) của ngôn ngữ này viết bằng mã nhị phân hay mã hec-xa.
- Gắn với kiến trúc phần cứng của máy, do vậy khai thác được các đặc điểm phần cứng.
- Không thuận lợi cho người lập trình do tính khó nhớ của mã, tính thiếu cấu trúc,…

Các loại ngôn ngữ lập trình (2/3)

Hợp ngữ:

- Cho phép người lập trình sử dụng một số từ gợi nhớ viết tắt để thể hiện các câu lệnh thực hiện.
- Ví dụ: cộng nội dung của 2 thanh ghi AX và BX rồi ghi kết quả vào AX,
  - Mã máy (8086): 01D8
  - Câu lệnh hợp ngữ: **ADD AX, BX**
- Chương trình hợp ngữ phải được dịch ra ngôn ngữ máy trước khi máy tính có thể thực hiện
  - Sử dụng chương trình hợp dịch.

Các loại ngôn ngữ lập trình (3/3)

Ngôn ngữ bậc cao:

- Ít phụ thuộc vào kiến trúc phần cứng máy tính
- Gần với tiếng Anh tự nhiên
- Có tính độc lập cao nhằm khắc phục những hạn chế của hợp ngữ
- Cần phải chuyển đổi sang ngôn ngữ máy để thực hiện
  - Quá trình chuyển đổi gọi là quá trình dịch.
- Ví dụ: FORTRAN, COBOL, ALGOL60, BASIC, Pascal, Foxpro, Visual Foxpro, Visual Basic, C, Visual C, C++, Java, C#, Python,…

Các phương thức dịch (1/2)

Thông dịch (Interpreter):

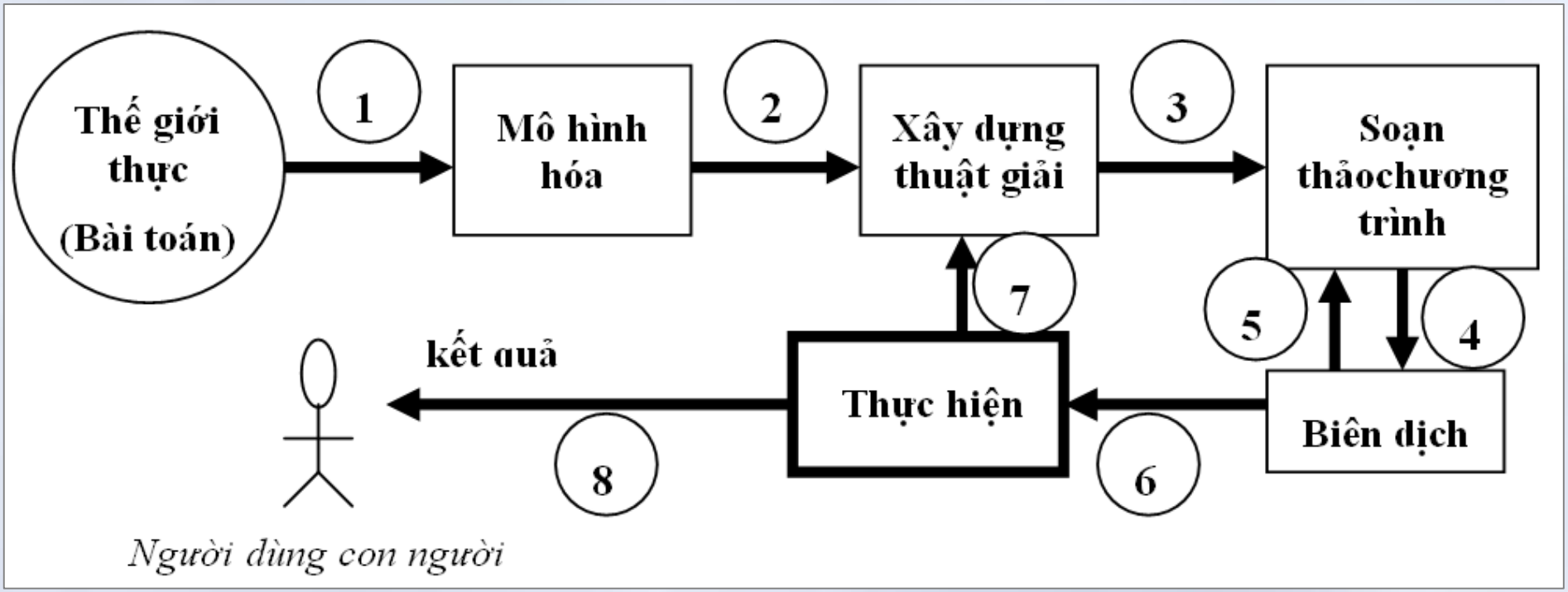
- Bộ thông dịch đọc từng lệnh của chương trình nguồn, phân tích cú pháp của câu lệnh đó và nếu đúng thì thực hiện.
- Quá trình bắt đầu từ lệnh đầu tiên của chương trình đến lệnh cuối cùng nếu không có lỗi.
- Bộ thông dịch này giống như vai trò của 1 thông dịch viên (translator).

Các phương thức dịch (2/2)

***Biên dịch* (Compiler):**

- Trình biên dịch dịch toàn bộ chương trình nguồn sang ngôn ngữ đích.
- Với chương trình đích này, máy đã có thể hiểu được và biết cách thực thi.
- Quá trình biên dịch sẽ tạo ra chương trình đích chỉ khi các lệnh trong chương trình nguồn không có lỗi.

Qui trình giải quyết một bài toán trên máy tính



Quy trình phát triển phần mềm máy tính

- B1: Xác định bài toán:
  - Xác định yêu cầu người dùng
- B2: Phân tích bài toán:
  - Tìm hiểu nhiệm vụ (chức năng) mà phần mềm cần xây dựng phải có và các dữ liệu cần thiết.
  - Xây dựng các giải pháp khả thi.

  → Tìm hiểu **hệ thống là gì? Và làm gì? (What)**
- B3: Thiết kế hệ thống:
  - Thực hiện thiết kế kiến trúc hệ thống, thiết kế các mô đun chương trình, thiết kế giao tiếp, thiết kế an toàn,…
  - Thiết kế mô đun chính là xây dựng giải thuật cho mô đun đó và cách diễn tả giải thuật.

  → Hệ thống cần được làm **như thế nào? (How)**

Quy trình phát triển phần mềm máy tính

- B4: Xây dựng chương trình:
  - Viết mã nguồn (source code) cho các mô đun theo ngôn ngữ lập trình đã xác định.
- B5: Kiểm thử chương trình:
  - Nhằm kiểm tra tính đúng đắn của từng mô đun và cả hệ thống trước khi bàn giao cho khách hàng.
- B7: Triển khai:
  - Cài đặt, triển khai cho khách hàng (người dùng) sử dụng chương trình
  - Viết tài liệu hướng dẫn sử dụng cho phần mềm
- B8: Bảo trì:
  - Sửa các lỗi trong quá trình người sử dụng dùng thử chương trình trong thời gian đầu.

Hệ thống máy tính

- **Tổ chức bên trong của máy tính**
  1. Mô hình cơ bản của máy tính
  2. Bộ xử lý trung tâm – CPU
  3. Bộ nhớ
  4. Hệ thống vào-ra
  5. Liên kết hệ thống (buses)
- **Phần mềm máy tính**
  1. Dữ liệu và giải thuật
  2. Chương trình và ngôn ngữ lập trình
  3. Phân loại phần mềm máy tính

Theo quan điểm sử dụng chung

- *Phần mềm hệ thống*:
  - Điều khiển hoạt động bên trong của máy tính và cung cấp môi trường giao tiếp giữa người dùng và máy tính nhằm khai thác hiệu quả phần cứng phục vụ cho nhu cầu sử dụng.
  - Đòi hỏi tính ổn định, tính an toàn cao.
  - Ví dụ: Hệ điều hành máy đơn hay hệ điều hành mạng, các tiện ích hệ thống,…
- *Phần mềm ứng dụng*:
  - Dùng giải quyết các vấn đề phục vụ các hoạt động của con người như quản lý, kế toán, soạn thảo văn bản,..
  - Nhu cầu về phần mềm ứng dụng ngày càng tăng và đa dạng.

Phân loại theo đặc thù ứng dụng và môi trường

- Phần mềm thời gian thực (Real-time SW)
- Phần mềm nghiệp vụ (Business SW)
- Phần mềm tính toán KH&KT (Eng.&Scie. SW)
- Phần mềm nhúng (Embedded SW)
- Phần mềm trên Web (Web-based SW)
- Phần mềm trí tuệ nhân tạo (AI SW)
- …

Nội dung chính

1. Hệ thống máy tính
    1. Tổ chức bên trong máy tính
    2. Phần mềm máy tính
2. Mạng máy tính
    1. Lịch sử phát triển của mạng máy tính
    2. Phân loại mạng máy tính
    3. Các thành phần cơ bản của một mạng máy tính
    4. Mạng Internet
3. Giới thiệu hệ điều hành
    1. Các khái niệm cơ bản
    2. Hệ lệnh của hệ điều hành
    3. Hệ điều hành Window

Mạng máy tính

1. Lịch sử phát triển của mạng máy tính
2. Phân loại mạng máy tính
3. Thành phần cơ bản của một mạng máy tính
4. Mạng Internet

Khái niệm mạng máy tính

Mạng máy tính hay ***mạng*** (*computer network, network*) là một tập hợp gồm nhiều máy tính hoặc thiết bị xử lý thông tin **được kết nối với nhau** qua các đường truyền và có sự trao đổi dữ liệu với nhau

Ví dụ:

- Mạng tại Trung tâm Máy tính, Khoa CNTT, Trường ĐHBK Hà Nội
- Mạng LAN của quán Game
- Mạng Internet

Lịch sử phát triển

- 1950: Máy tính ra đời
- 1960 mạng máy tính bắt đầu xuất hiện.
  - Thời gian đầu: mạng có dạng là một máy tính lớn nối với nhiều trạm cuối (*terminal*).
  - 1970s: mạng máy tính là các máy tính độc lập được nối với nhau.
- Hiện nay:
  - Mạng máy tính phát triển trên mọi lĩnh vực
  - Qui mô và mức độ phức tạp ngày càng tăng.

Mạng máy tính

1. Lịch sử phát triển của mạng máy tính
2. Phân loại mạng máy tính
3. Thành phần cơ bản của một mạng máy tính
4. Mạng Internet

Theo mối quan hệ giữa các máy trong mạng

- *Mạng bình đẳng (peer-to-peer)*
  - Các máy có quan hệ ngang hàng
- *Mạng khách/chủ* (*client/server*).
  - Một số máy là server (máy phục vụ/máy chủ) chuyên phục vụ các máy khác gọi là máy khách (client) hay máy trạm (workstation)
  - Ví dụ các server chuyên để tính toán, chuyên quản lý dữ liệu, cung cấp thông tin,..

Theo qui mô địa lý

- ***LAN*** (*Local Area Network*) mạng cục bộ
  - Phạm vi nhỏ, ví dụ bán bán kính 500m,
  - Số lượng máy tính không quá nhiều,
  - Mạng không quá phức tạp.
- ***WAN*** (*Wide Area Network*) mạng diện rộng,
  - Các máy tính có thể ở các thành phố khác nhau. Bán kính có thể 100-200 km.
  - Ví dụ mạng của Tổng cục thuế.
- ***GAN*** (*Global Area Network*) mạng toàn cầu
  - Máy tính ở nhiều nước khác nhau.
  - Thường là kết hợp của nhiều mạng con.
  - Ví dụ mạng Internet.

Mạng máy tính

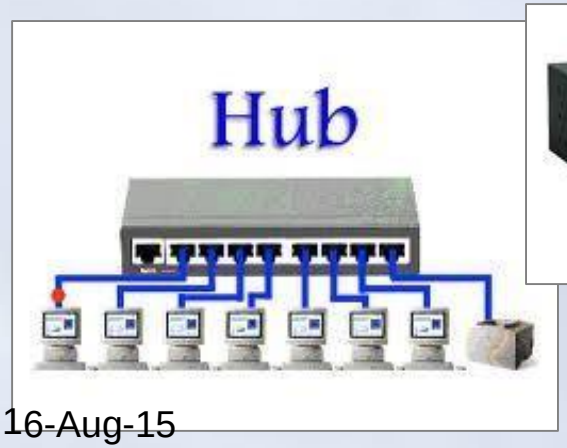
1. Lịch sử phát triển của mạng máy tính
2. Phân loại mạng máy tính
3. Thành phần cơ bản của một mạng máy tính
4. Mạng Internet

Các thành phần cơ bản của một mạng máy tính (1/3)

- Các máy tính
  - Mỗi máy tính là một nút của mạng
  - Để xử lý, lưu trữ và trao đổi thông tin
- Các vỉ mạng (NIC: Network Interface Card)
  - Cho phép giao tiếp giữa máy tính với đường truyền

Các thành phần cơ bản của một mạng máy tính (2/3)

- Đường truyền/đường truyền vật lý
  - Là phương tiện truyền tải thông tin dữ liệu, trên đó thông tin được truyền đi
  - Phân ra thành 2 loại :Hữu tuyến và vô tuyến
- Các thiết bị kết nối mạng
  - HUB, SWITCH, ROUTER



Các thành phần cơ bản của một mạng máy tính (3/3)

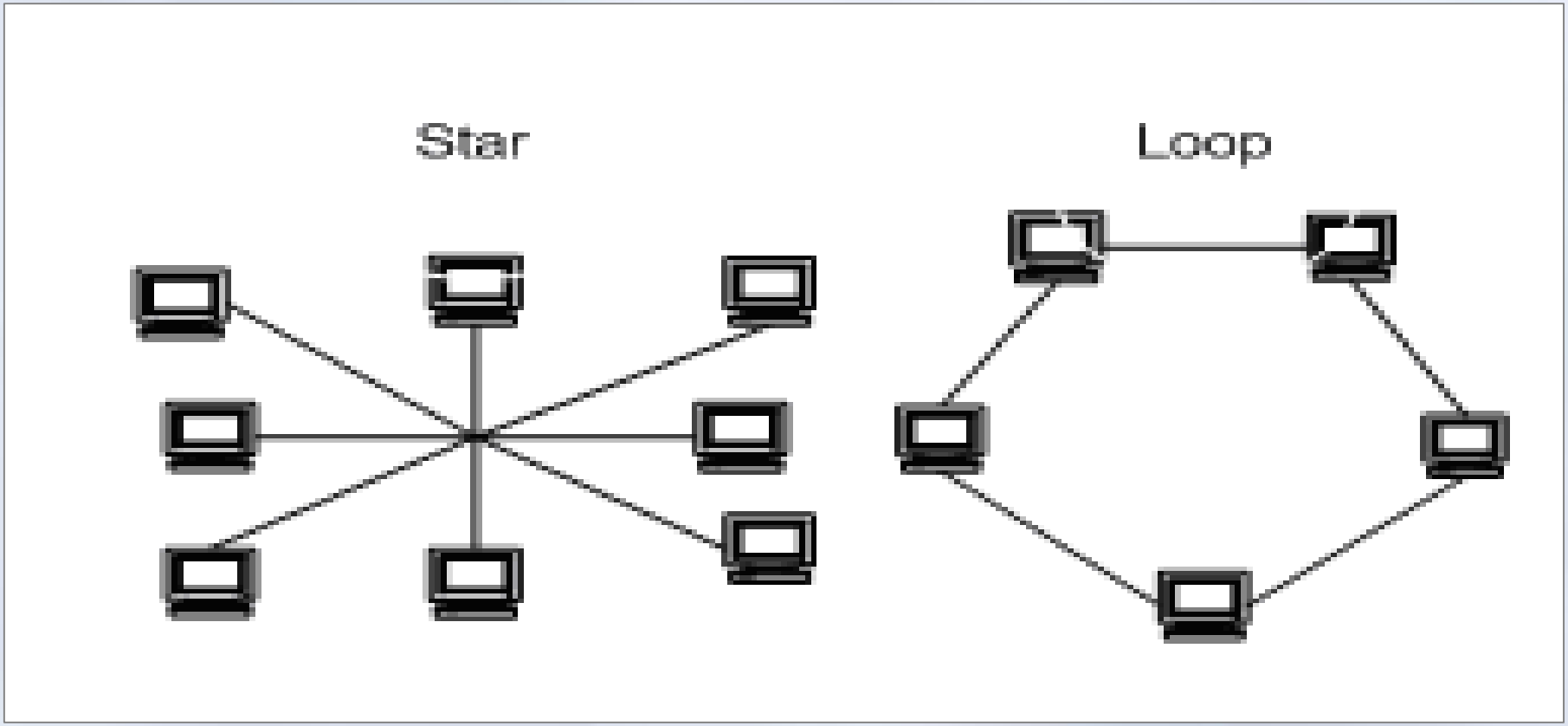
- Các thiết bị đầu cuối (terminal)
  - Máy tính, máy in, máy photo…
- Các phụ kiện mạng
  - Giắc cắm, ổ cắm
- Hệ điều hành mạng
  - Điều khiển hoạt động của mạng
- Phần mềm mạng máy tính
  - Hộ trợ kết nối mạng (nếu HĐH không hỗ trợ)
- Các ứng dụng trên mạng
  - Ví dụ: Web, Mail, Chat, game online,..

Kiến trúc mạng

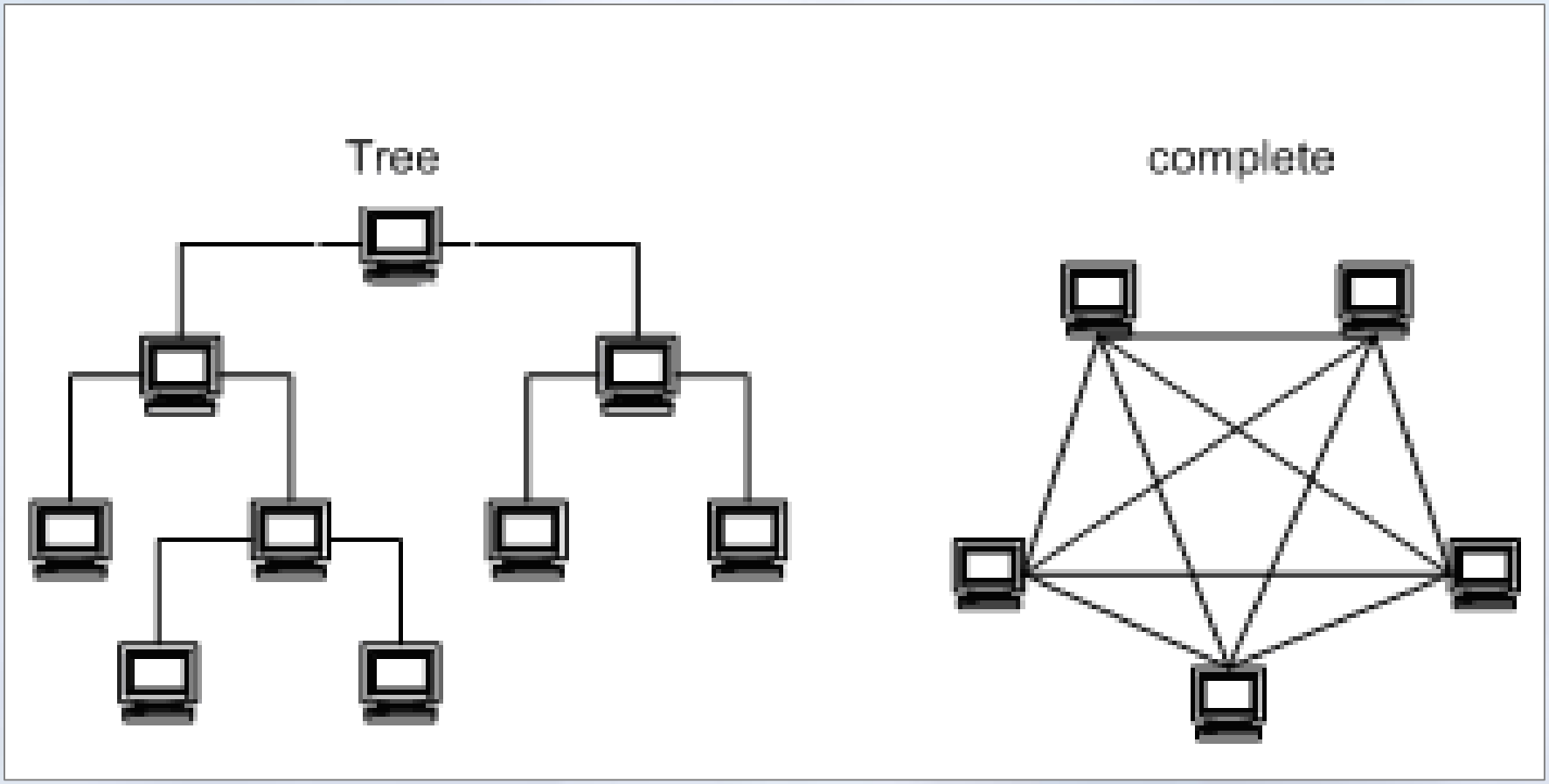
- Network architecture
- Thể hiện cách kết nối máy tính với nhau và qui ước truyền dữ liệu/giao thức giữa các máy tính như thế nào.
  - Cách kết nối: Topology:
    - Điểm điểm (Point to point)
    - Quảng bá (broadcast)
  - Tập các quy ước truyền thông: Protocol

Kiến trúc mạng→Cách kết nối điểm - điểm

- Các nút được nối thành từng cặp
- Các nút sẽ gửi dữ liệu đến nút lân cận nó

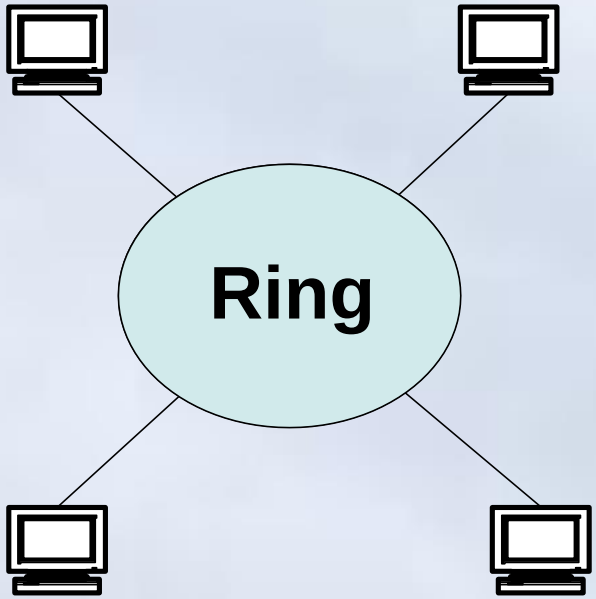


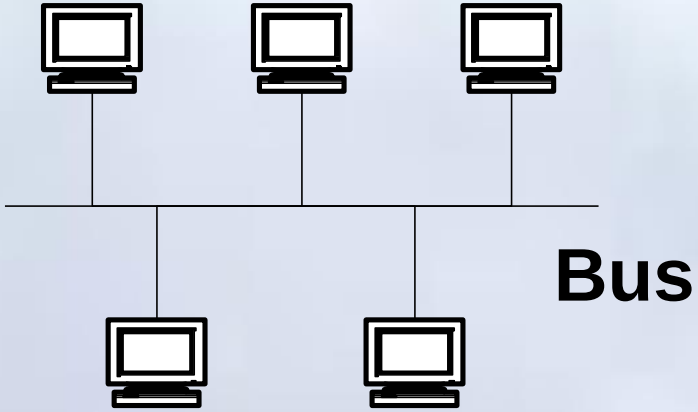
Kiến trúc mạng→Cách kết nối điểm - điểm



Kiến trúc mạng→Cách kết nối quảng bá

Một nút gửi các nút khác đều nhận được





Mạng máy tính

1. Lịch sử phát triển của mạng máy tính
2. Phân loại mạng máy tính
3. Thành phần cơ bản của một mạng máy tính
4. Mạng Internet

Giới thiệu

- *Internet* là một mạng máy tính có qui mô toàn cầu
  - Gồm rất nhiều mạng con và máy tính nối với nhau bằng nhiều loại phương tiện truyền.
- Internet không thuộc sở hữu của ai cả. Chỉ có các uỷ ban điều phối và kỹ thuật giúp điều hành Internet.
  - Ban đầu là mạng *ARPANET* của Bộ Quốc phòng Mỹ (DoD)

Một số dịch vụ

- Truyền tệp tin (**FTP**: File Transfer Protocol)
- Truy nhập máy tính từ xa (telnet)
- Web (WWW) :
  - Tìm kiếm và khai thác thông tin trên mạng
- Thư điện tử (E-mail)
- Tán gẫu (Chat) trên mạng…
- ….

Kết nối internet

- Máy tính có Modem (Dial-up, ADSL) hoặc card mạng.
- Có thuê bao kết nối với Internet: qua mạng, qua đường điện thoại, đường thuê riêng.
  - Hiện nay thường kết nối qua đường điện thoại hoặc qua ADSL
- Có tài khoản Internet ở trên mạng hay ở một nhà cung cấp dịch vụ Internet
  - Ví dụ như VNN, FPT.
- Có phần mềm Internet thông dụng
  - Web browser để xem trang web: IE, FireFox ,
  - Phần mềm để xem thư (Outlook), Chat (YM, Skype)..

Lợi ích của internet

- Truyền tin
- Phổ biến tin
- Thu thập tin
- Trao đổi thông tin
- Kho dữ liệu
  - Tìm hiểu về biểu diễn số thực !!
    - Google.com → IEEE754/85
- …..

Nội dung chính

1. Hệ thống máy tính
    1. Tổ chức bên trong máy tính
    2. Phần mềm máy tính
2. Mạng máy tính
    1. Lịch sử phát triển của mạng máy tính
    2. Phân loại mạng máy tính
    3. Các thành phần cơ bản của một mạng máy tính
    4. Mạng Internet
3. Giới thiệu hệ điều hành
    1. Các khái niệm cơ bản
    2. Hệ lệnh của hệ điều hành
    3. Hệ điều hành Window

Giới thiệu hệ điều hành

1. Các khái niệm cơ bản
2. Hệ lệnh của hệ điều hành
3. Hệ điều hành Windows

Khái niệm hệ điều hành *(Operating System)*

- ***Hệ điều hành*** là **hệ thống chương trình** đảm bảo **quản lý tài nguyên** của hệ thống tính toán và **cung cấp các dịch vụ** cho người sử dụng.
- Trong các máy tính hiện nay, hệ điều hành thường được cài đặt trên đĩa
- Hệ điều hành là ***phần mềm hệ thống***, nên phụ thuộc vào cấu trúc của máy tính.
- Mỗi loại máy tính có hệ điều hành khác nhau.
  - Máy tính lớn IBM360 có hệ điều hành là DOS, TOS.
  - Máy tính lớn EC-1022 có hệ điều hành là OC-EC.
  - Máy tính cá nhân PC-IBM có hệ điều hành MS-DOS.
  - Mạng máy tính có các hệ điều hành mạng NETWARE, UNIX, WINDOWS-NT…

Nhiệm vụ của hệ điều hành

- Khởi động máy tính, tạo môi trường giao tiếp cho người sử dụng.
- Tự động điều khiển và kiểm soát hoạt động của các thiết bị (đĩa, bàn phím, màn hình, máy in,…).
- Quản lý việc cấp phát tài nguyên của máy tính như bộ xử lý trung ương, bộ nhớ, các thiết bị vào ra…
- Quản lý các chương trình đang thực hiện trên máy tính.
- Thực hiện giao tiếp với người sử dụng để nhận lệnh và thực hiện lệnh.

Khái niệm Tệp/Tập tin/File

- ***Tệp*** là đơn vị thông tin mà hệ điều hành quản lý (lưu trữ, tìm kiếm,..) trên bộ nhớ ngoài
  - Thường là tập hợp các dữ liệu có liên quan tới nhau, được tổ chức theo một cấu trúc nào đó.
  - Nội dung có thể là chương trình, dữ liệu, văn bản,..
- Mỗi tệp được lưu trên đĩa với một tên phân biệt
  - Tên tệp được đặt theo quy ước của hệ điều hành
  - Tên tệp tin thường có 2 phần:
    - Phần tên (name): Buộc phải có
    - phần mở rộng (extension)
    - Giữa phần tên và phần mở rộng có một dấu chấm (.) ngăn cách.

Tên tệp → Phần tên

- Bao gồm các ký tự chữ từ A đến Z (thường và hoa) chữ số từ 0 đến 9,
- Ký tự khác như #, $, %, ~, ^, @, (,), !, _, khoảng trắng
  - *MS-DOS không có khoảng trắng*
- Độ dài tối đa phần tên trong MS-DOS là 8, trong Windows có thể tới 128 ký tự
- ***Lưu ý***: *Nên đặt tên mang tính gợi nhớ.*
  - Hợp lệ: dulieu211212.txt, dulieu$211212.dat
  - Không hợp lệ: ‘dulieu211212.txt, ?abc.dat

Tên tệp → Phần mở rộng

- Thường là 3 ký tự hợp lệ, do chương trình ứng dụng tạo ra tập tin tự đặt theo quy ước.
- Một số phần mở rộng trong Windows
  - COM, EXE : Các file khả thi chạy trực tiếp
  - TXT, DOC, ... : Các file văn bản.
  - PAS, BAS, ... : Các file chương trình PASCAL, DELPHI, BASIC, ...
  - WK1, XLS, ... : Các file chương trình bảng tính LOTUS, EXCEL ...
  - BMP, GIF, JPG, ... : Các file hình ảnh.
  - MP3, DAT, WMA, … : Các file âm thanh, video.

Ký hiệu đại diện (Wildcard)

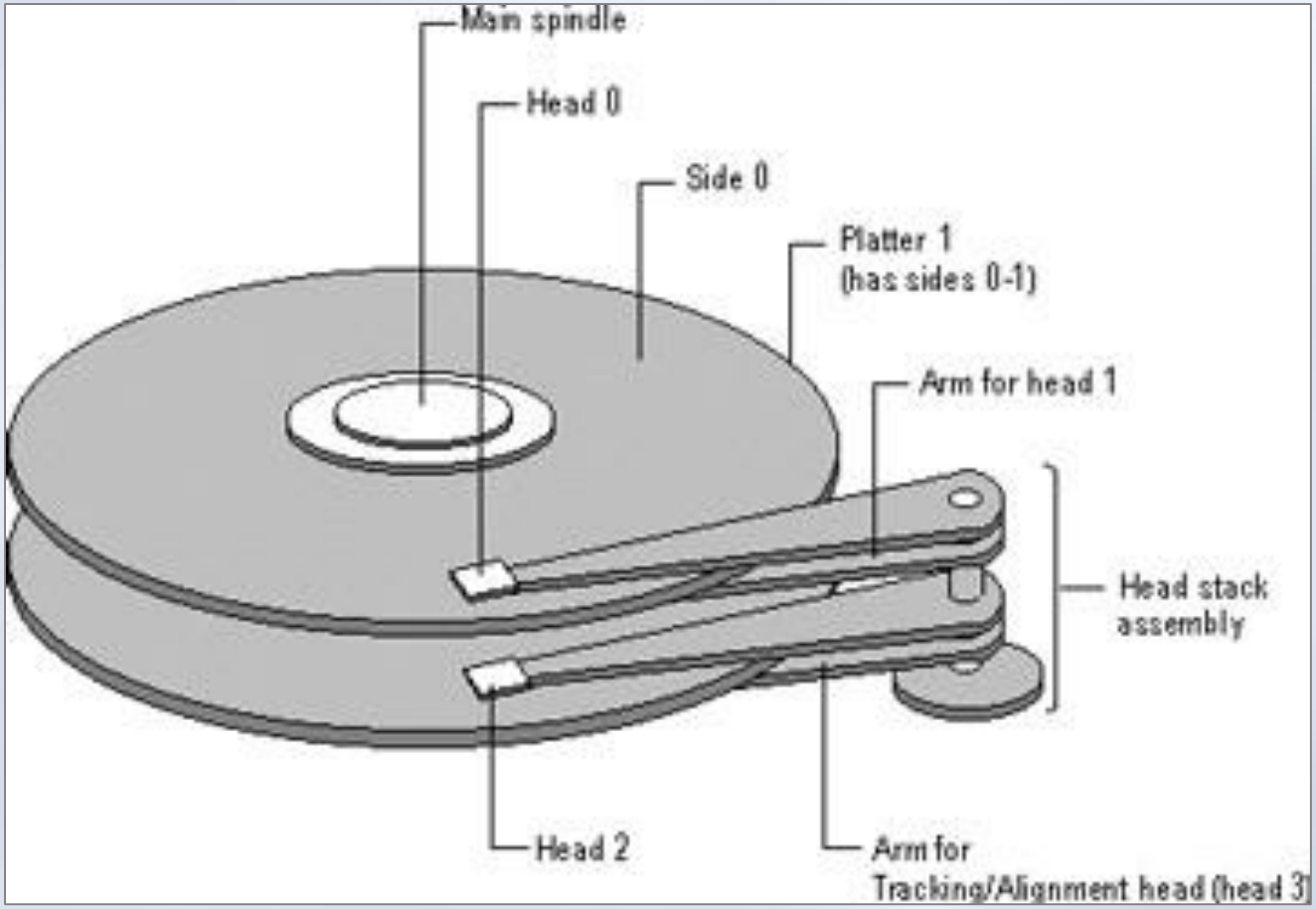
- Dùng để chỉ một nhóm tập tin
- Ký tự '**?**'
  - Đại diện cho một ký tự bất kỳ trong tập tin tại vị trí xuất hiện
  - **Bai?.doc:** Bai1.doc, Bai5.doc, BaiA.doc,..
- Ký tự '*****'
  - Đại diện cho một chuỗi ký tự bất kỳ trong tập tin tại vị trí xuất hiện
  - **Bai*.doc:** Bai.doc, Bai1.doc, Baitap.doc,…
  - **Bai.*:** Bai.doc, Bai.c, Bai.xls, Bai.ppt,…

Bai?.* :….

?E*.* :….

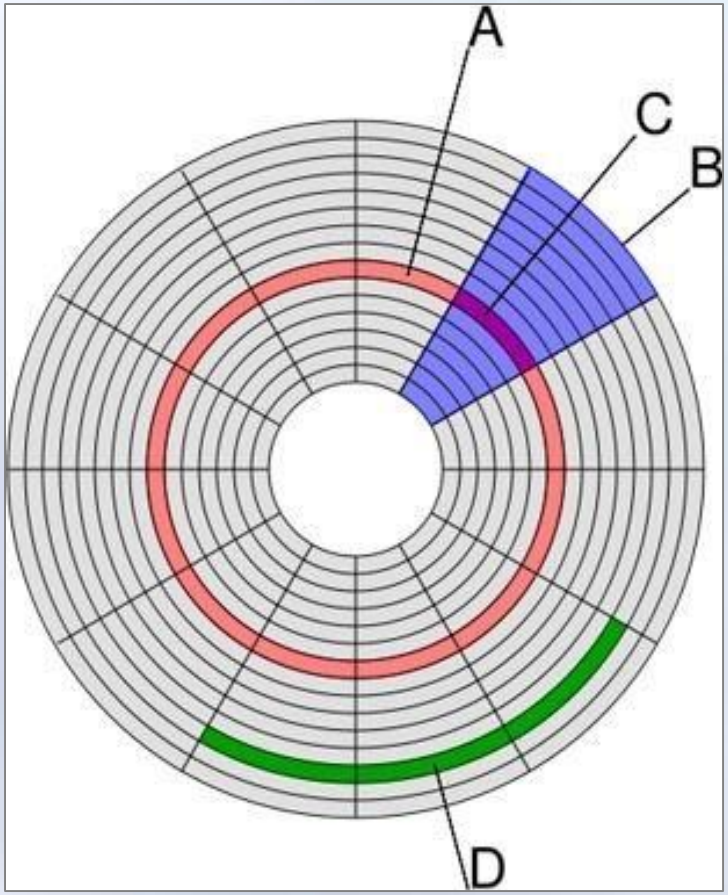
Cấu trúc vật lý đĩa từ

- Gồm nhiều đĩa (Platter) gắn đồng trục
- Các mặt đĩa (side) được đọc/ghi bởi một đầu đọc
  - Các mặt chia thành rãnh (track), cung (sector)



Cấu trúc vật lý đĩa từ

- A - Rãnh từ (track)
  - Rãnh từ được đánh số từ ngoài vào trong bắt đầu từ 0
- B - Dải Cung từ (Sector track)
- C - Cung từ (Sector)
  - Kích thước: 512byte
- D - Liên cung (Cluster)
  - Sector liên tiếp nhau
  - HĐH ghi tệp theo cluster
- Cylinder: các rãnh có cùng bán kính nằm trên các mặt đĩa khác nhau



Tổ chức ghi thông tin trên đĩa

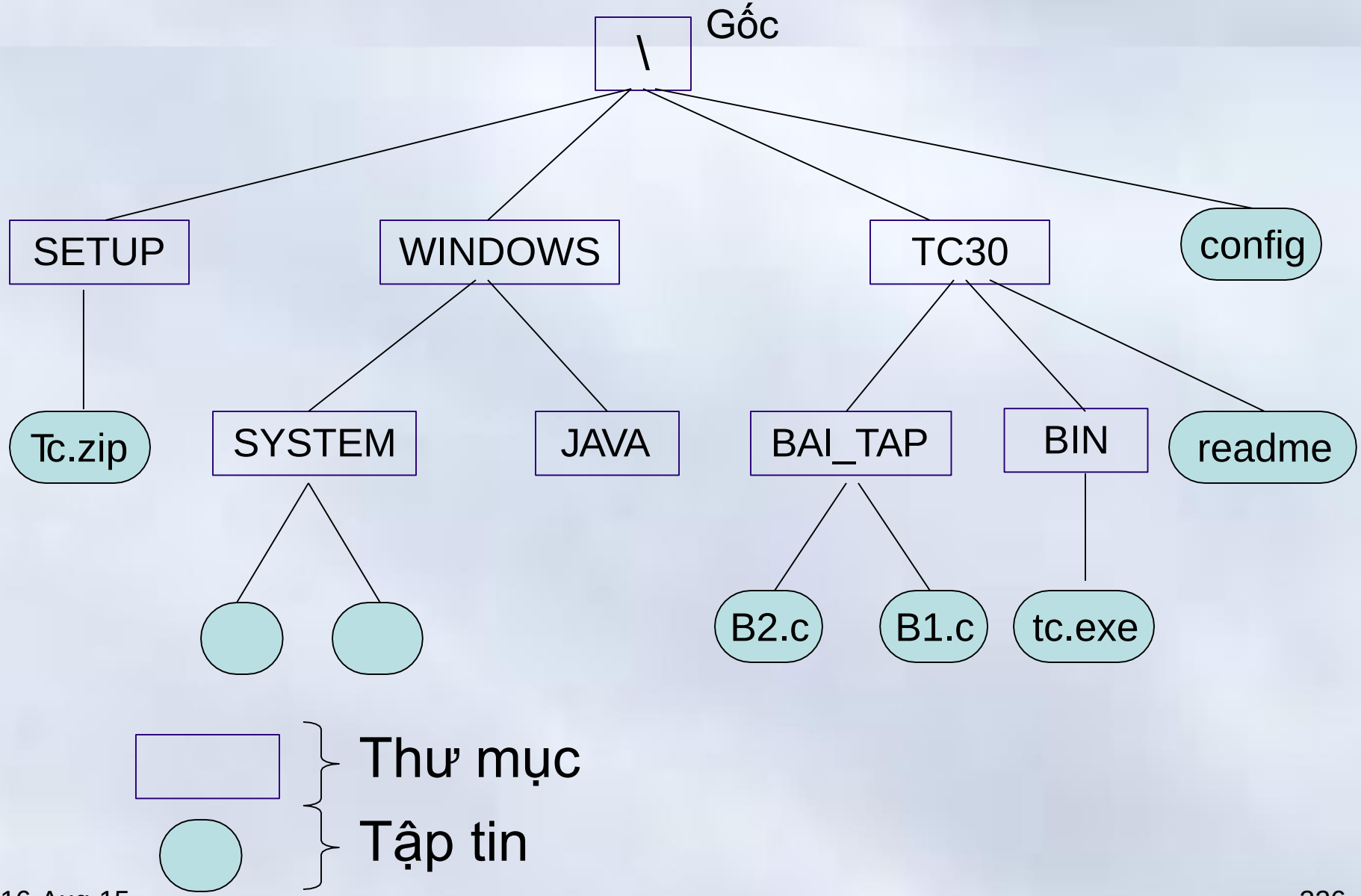
- Thông tin lưu trữ trên đĩa dưới dạng các tệp.
  - Mỗi tệp chiếm 1 hoặc nhiều cluster.
    - Mỗi Cluster gồm 1 hay nhiều sector liên tiếp nhau về mặt logic
- Số lượng tệp trên một đĩa có thể rất lớn
  - Khó quản lý (Ví dụ: Các tệp trùng tên !?)
- HĐH tổ chức lưu thông tin theo thư mục
  - Trong Windows: Thư mục được gọi là *Folder*
- HĐH có thể chia đĩa thành các phân vùng logic (partition)-ổ đĩa logic (*C:, D:, E: ..*)

Thư mục

- Mỗi phân vùng có một thư mục chung gọi là thư mục gốc
  - Thư mục gốc không có tên riêng và được ký hiệu là ****
- Mỗi thư mục có thể tạo ra các thư mục khác: Thư mục con
  - Thư mục con đặt tên theo nguyên tắc tên tệp
  - Thư mục chứa thư mục con: Thư mục cha
- Mỗi file lưu trữ phải thuộc về một thư mục
  - Hoặc thư mục gốc, hoặc thư mục con

Tổ chức dạng cây: Thư mục⇔cành, tệp⇔lá

Cây thư mục


Gốc
\
SETUP
WINDOWS
TC30
config
Tc.zip
SYSTEM
JAVA
BAI_TAP
BIN
readme
B2.c
B1.c
tc.exe
Thư mục
Tập tin

Tên đầy đủ của tệp

- Đường dẫn
  - Chuỗi các thư mục được ngăn cách bởi ký tự '****' chỉ đường vào thư mục con chứa tệp
  - Có thể bắt đầu bằng tên ổ đĩa (A,B,C,D,E,..) theo sau bởi dấu '**:**'
  - Đường dẫn tuyệt đối: Tính từ thư mục gốc
  - Đường dẫn tương đối: Tính từ thư mục hiện thời
- Tên đầy đủ của tệp gồm
  - Tên tệp
  - Đường dẫn tới thư mục con chứa tệp
  - Ví dụ: \TC30\BAI_TAP\B1.c
  - Nếu là thư mục gốc của ổ C: **C:\TC30\BAI_TAP\B1.c**

Giới thiệu hệ điều hành

1. Các khái niệm cơ bản

2. Hệ lệnh của hệ điều hành

3. Hệ điều hành Windows

Hệ lệnh của hệ điều hành

- Thao tác với tệp:
  - Sao chép, di chuyển, xoá, đổi tên , xem nội dung tệp
- Thao tác với thư mục:
  - Tạo, xoá, sao chép, xem nội dung thư mục,..
- Thao tác hệ thống
  - Lấy /đặt thông tin ngày giờ,..
- Thao tác với đĩa:
  - Tạo khuôn (Format), sao chép đĩa

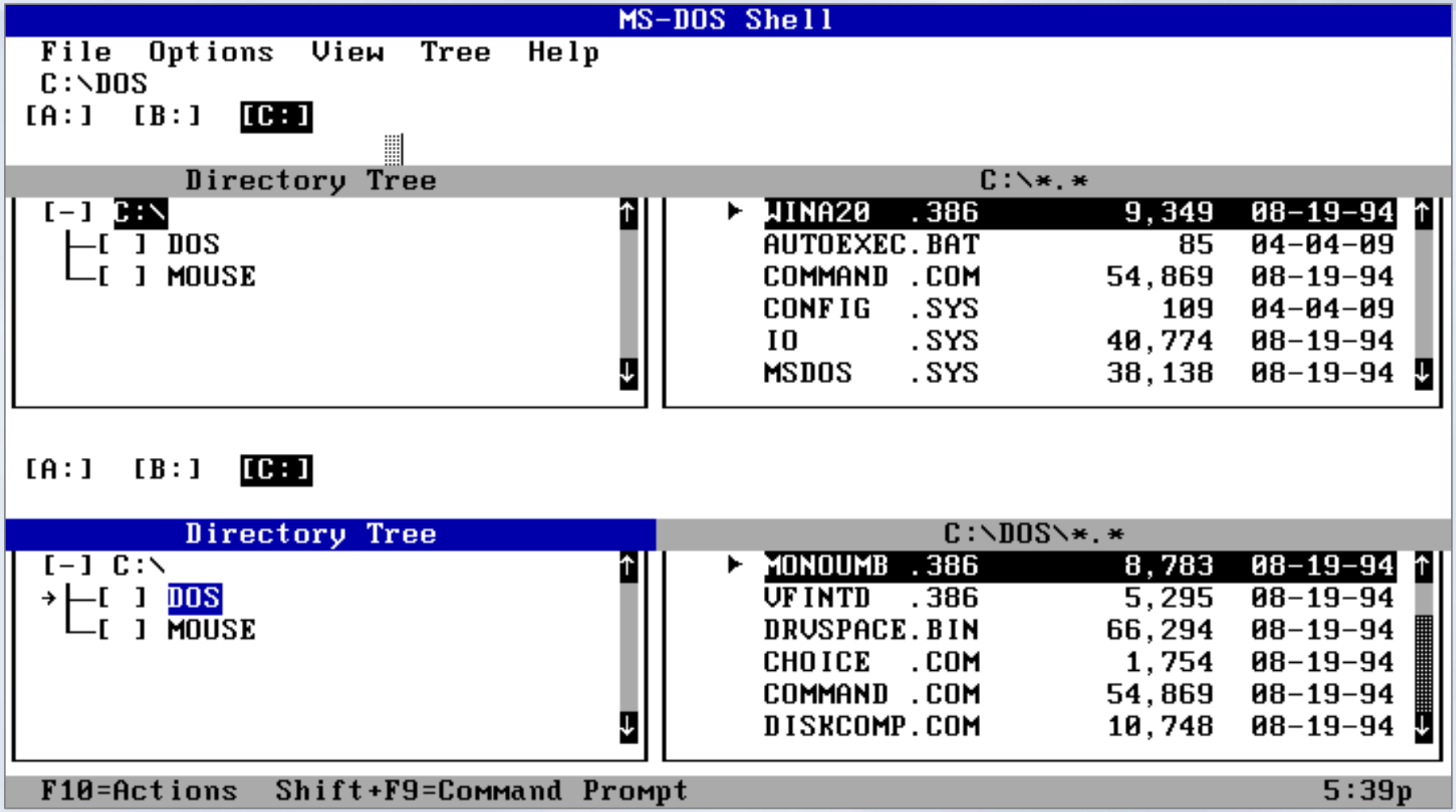
Hình thái giao tiếp

- Giao diện dòng lệnh
  - Thực hiện các chức năng bởi gõ câu lệnh
    - Phức tạp do phải nhớ lệnh, nhớ cú pháp
  - Ví dụ: MS-DOS
    - Thư mục: MD, CD, RD, DIR..
    - File: COPY, REN, DEL, TYPE…
- Giao diện thực đơn/biểu tượng
  - Gọi chức năng thông qua phím tắt, giao diện đồ họa
    - Cần thiết bị vào như con chuột (*mouse*)
  - Ví dụ: Dos-shell, Windows

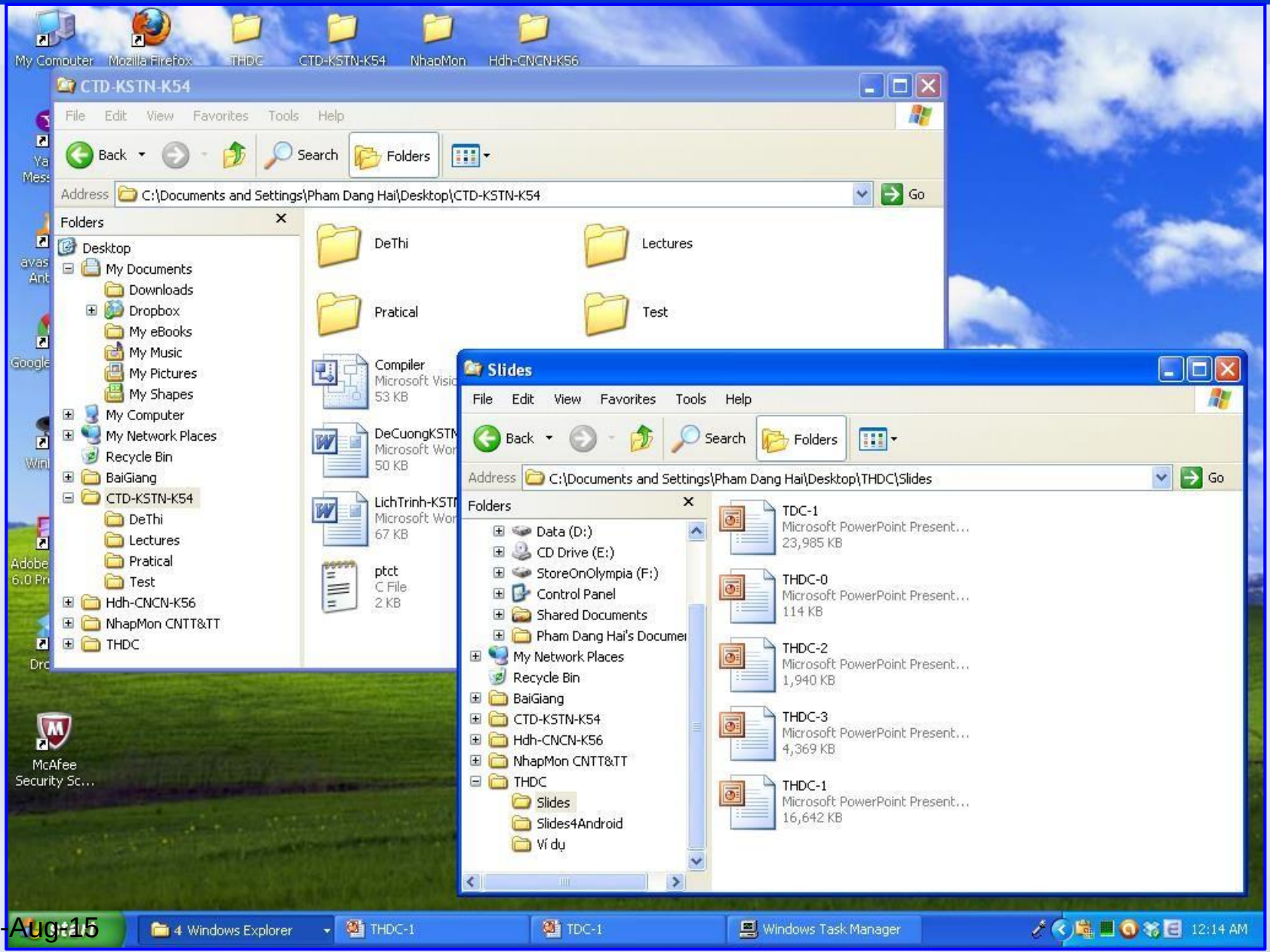
MS-DOS

```
C:\>dir a:/w

 Volume in drive A is BOOT622
 Volume Serial Number is 3505-18E3
 Directory of A:\

SYS.COM         COMMAND.COM     ATTRIB.EXE      CHKDSK.EXE      DELTREE.EXE
EMM386.EXE      FDISK.EXE       LABEL.EXE       MEM.EXE         MSCDEX.EXE
QBASIC.EXE      UNDELETE.EXE    CD2.SYS         EDIT.HLP        UNDELETE.INI
C.BAT           HIMEM.SYS       CONFIG.SYS      AUTOEXEC.BAT    MOUSE.@@@
CD3.SYS         EDIT.EXE        EDIT.INI        DOSKEY.COM      UNFORMAT.COM
TREE.COM        FIND.EXE        RESTORE.EXE     SETVER.EXE      SCANDISK.EXE
SHARE.EXE       XCOPY.EXE       QBASIC.HLP      MOUSE.INI       SCANDISK.INI
MOUSE.SYS       CD4.SYS         CD1.SYS         FORMAT.COM      MOUSE.COM
       40 file(s)      1,250,202 bytes
                          51,200 bytes free

C:\>ver

MS-DOS Version 6.22


C:\>date
Current date is Wed 09-05-2012
Enter new date (mm-dd-yy):
```

Dos - shell


MS-DOS Shell
File Options View Tree Help
C:\DOS
[A:] [B:] [C:]
Directory Tree
C:*.*
[-] C:\
[ ] DOS
[ ] MOUSE
WINA20 .386 9,349 08-19-94
AUTOEXEC.BAT 85 04-04-09
COMMAND .COM 54,869 08-19-94
CONFIG .SYS 109 04-04-09
IO .SYS 40,774 08-19-94
MSDOS .SYS 38,138 08-19-94
[A:] [B:] [C:]
Directory Tree
C:\DOS*.*
[-] C:\
[ ] DOS
[ ] MOUSE
MONOUMB .386 8,783 08-19-94
VFINTD .386 5,295 08-19-94
DRVSPACE.BIN 66,294 08-19-94
CHOICE .COM 1,754 08-19-94
COMMAND .COM 54,869 08-19-94
DISKCOMP.COM 10,748 08-19-94
F10=Actions Shift+F9=Command Prompt
5:39p

Windows


My Computer
Mozilla Firefox
THDC
CTD-KSTN-K54
NhapMon
Hdh-CNCN-K56
CTD-KSTN-K54
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\Pham Dang Hai\Desktop\CTD-KSTN-K54
Go
Folders
Desktop
My Documents
Downloads
Dropbox
My eBooks
My Music
My Pictures
My Shapes
My Computer
My Network Places
Recycle Bin
BaiGiang
CTD-KSTN-K54
DeThi
Lectures
Pratical
Test
Hdh-CNCN-K56
NhapMon CNTT&TT
THDC
DeThi
Lectures
Pratical
Test
Compiler
Microsoft Visio
53 KB
DeCuongKSTN
Microsoft Wor
50 KB
LichTrinh-KSTN
Microsoft Wor
67 KB
ptct
C File
2 KB
Slides
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\Pham Dang Hai\Desktop\THDC\Slides
Go
Folders
Data (D:)
CD Drive (E:)
StoreOnOlympia (F:)
Control Panel
Shared Documents
Pham Dang Hai's Documer
My Network Places
Recycle Bin
BaiGiang
CTD-KSTN-K54
Hdh-CNCN-K56
NhapMon CNTT&TT
THDC
Slides
Slides4Android
Ví dụ
TDC-1
Microsoft PowerPoint Present...
23,985 KB
THDC-0
Microsoft PowerPoint Present...
114 KB
THDC-2
Microsoft PowerPoint Present...
1,940 KB
THDC-3
Microsoft PowerPoint Present...
4,369 KB
THDC-1
Microsoft PowerPoint Present...
16,642 KB
McAfee Security Sc...
4 Windows Explorer
THDC-1
TDC-1
Windows Task Manager
12:14 AM

Giới thiệu hệ điều hành

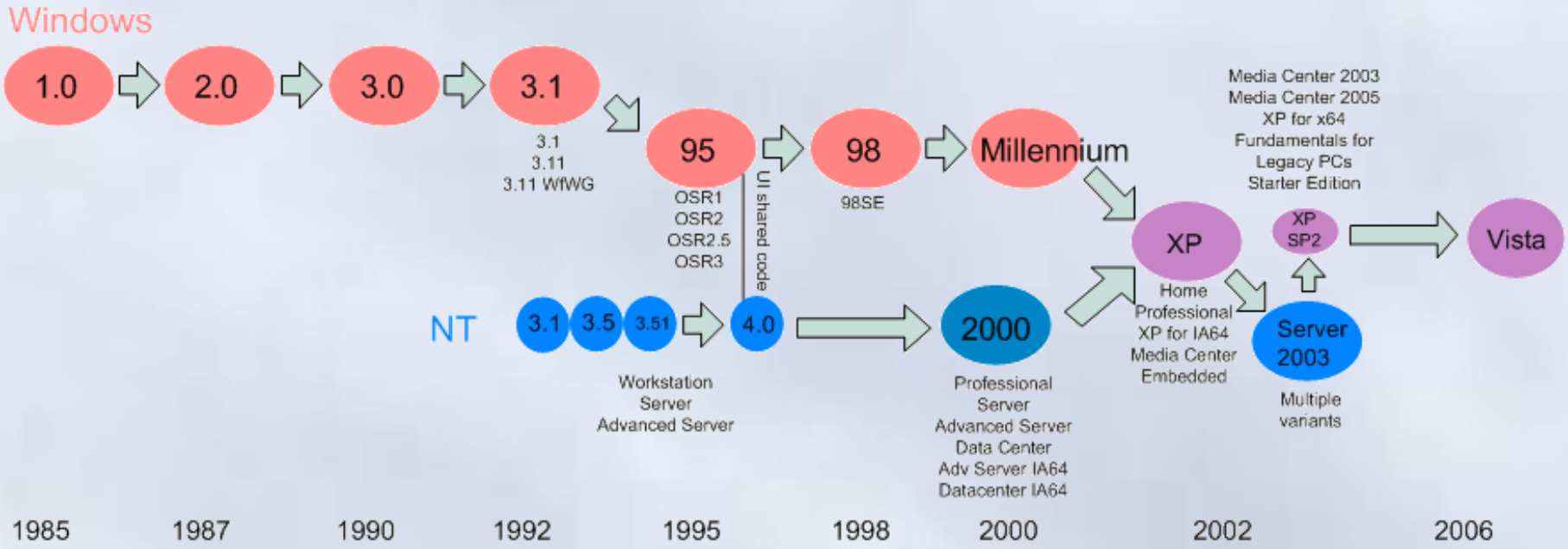
1. Các khái niệm cơ bản

2. Hệ lệnh của hệ điều hành

3. Hệ điều hành Windows

Ra đời và phát triển

- Do hãng Microsoft sản xuất.
  - Version 1.0 ra đời vào năm 1985
    - Windows 1.0, 3.0, 3.1.. Là chương trình ứng dụng
    - Windows 95, 98,.. : Là hệ điều hành



Làm việc với Windows XP

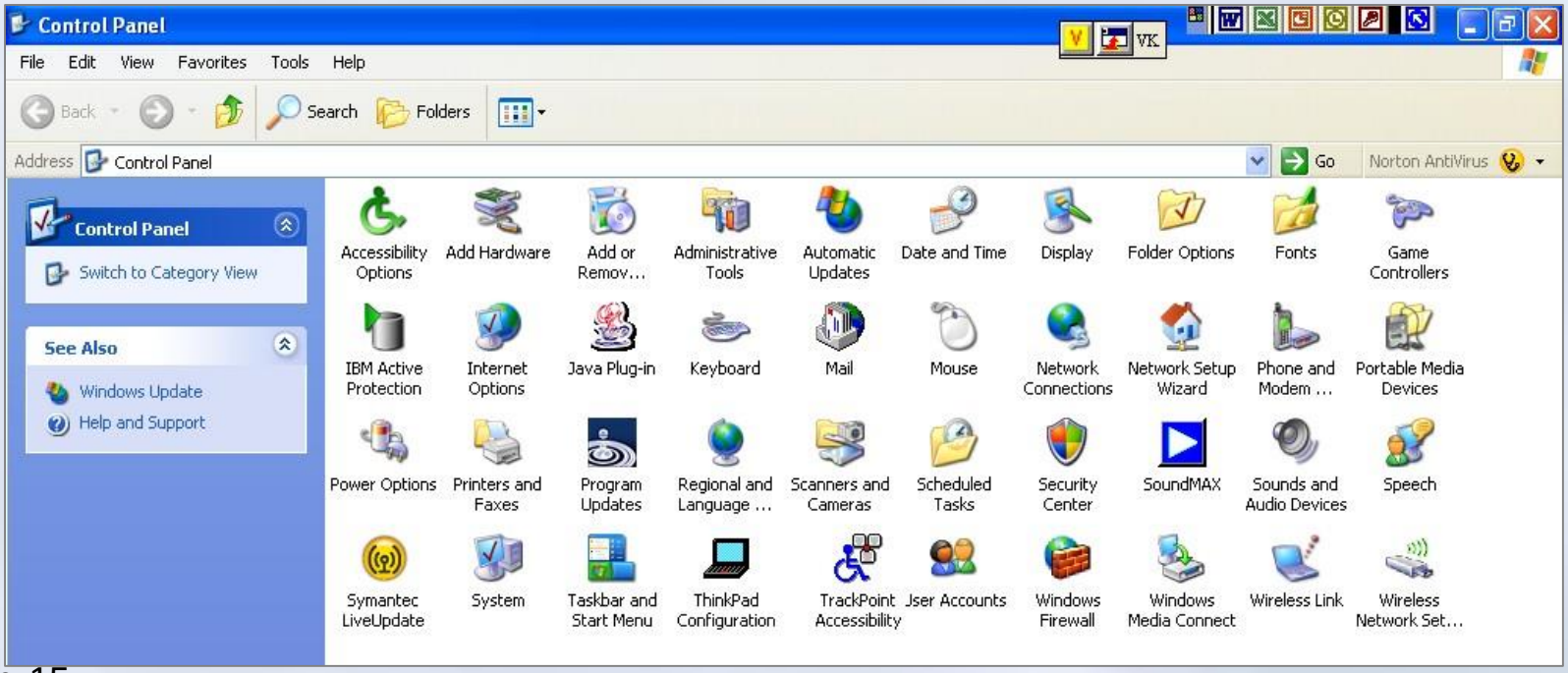
- Windows XP tự động khởi động khi bật máy.
  - Thực hiện đăng nhập (logging on):
    - Windows thông báo nhập tài khoản: Nhập tên (User name) và mật khẩu (Password) của người dùng
  - User profile:
    - Thiết lập chế độ làm viêc riêng cho từng người dùng: Cách bố trí màn hình, các chương trình tự động chạy khi khởi động tài nguyên được phép sử dụng,..
- *Thoát khỏi Windows XP*
  - *Đóng các ứng dụng đang mở*
  - *Chọn Start→Turn of computer/ Reboot/ standby*

Sử dụng chuột (mouse) trong Windows

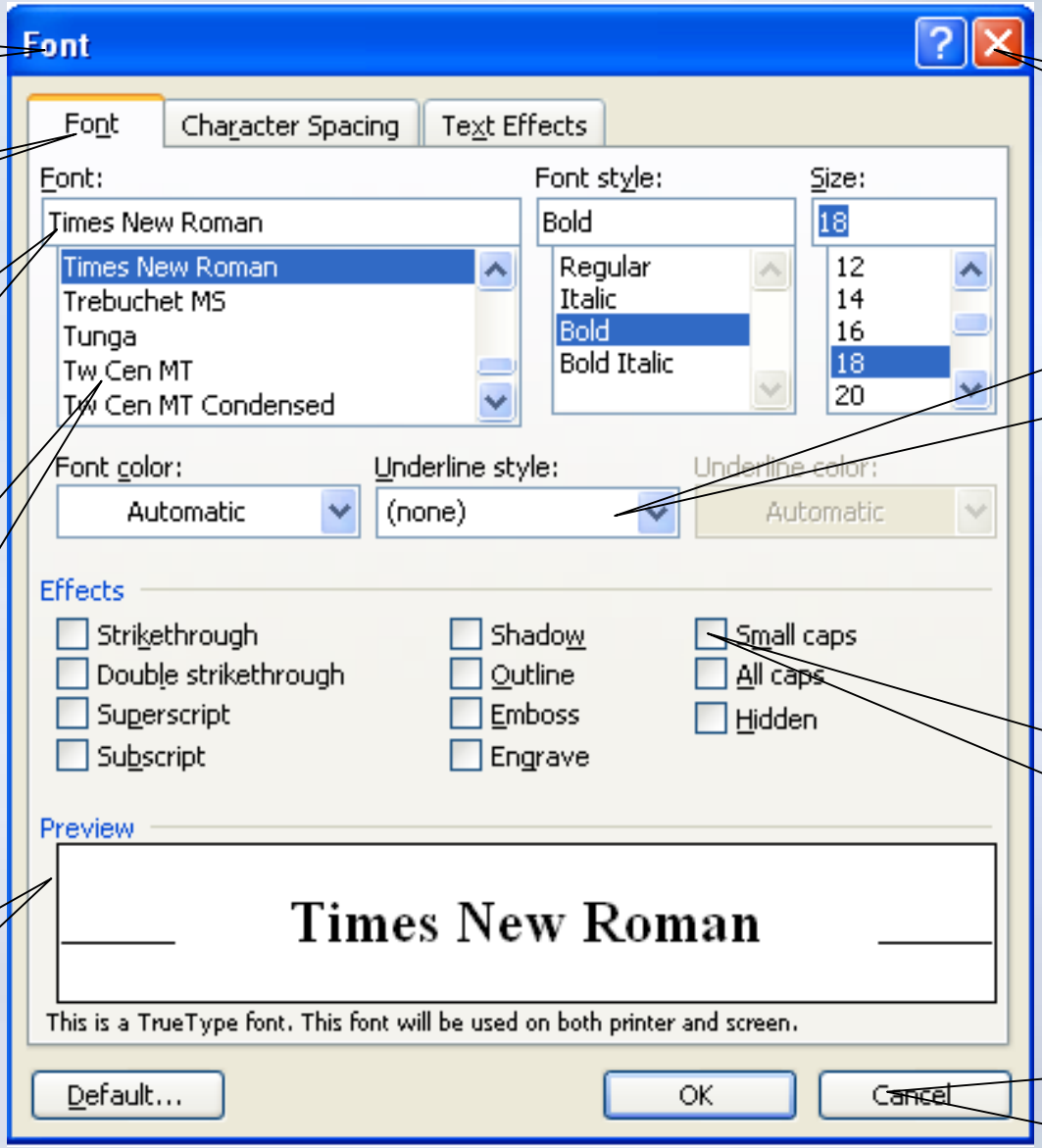
- Chuột là thiết bị cần thiết khi làm việc trong Windows XP.
- Con trỏ chuột (mouse pointer) xác định vị trí tác động của chuột trên màn hình.
- Các thao tác với thiết bị chuột:
  - **Point:** trỏ chuột trên mặt phẳng mà không nhấn nút nào cả.
  - **Click:** nhấn nhanh và thả nút chuột trái. Dùng để lựa chọn thông số, đối tượng hoặc câu lệnh.
  - **Right Click (R_Click):** nhấn nhanh và thả nút chuột phải. Dùng mở menu tương ứng với đối tượng để chọn các lệnh thao tác trên đối tượng đó.
  - **Double Click (D_Click):** nhấn nhanh nút chuột trái hai lần liên tiếp. Dùng khởi động một chương trình hoặc mở thư mục/ tập tin.
  - **Drag** (kéo thả): nhấn và giữ nút chuột trái khi di chuyển đến nơi khác và buông ra. Dùng để chọn một khối văn bản, để di chuyển một đối tượng trên màn hình, mở rộng kích thước của cửa sổ...

Biểu tượng (icon) và cửa sổ (window)

- **Biểu tượng:** các hình vẽ đặc trưng cho
  - Một đối tượng/ứng dụng của Windows
  - Một chức năng nào đó của ứng dụng đang thực hiện
- **Cửa sổ:** khung giao tiếp đồ họa của ứng dụng
  - Thao tác: di chuyển, phóng to, thu nhỏ, đóng, thay đổi,..



Hộp hội thoại (Dialogue box)



Tên hộp thoại

Các lớp

Hộp văn bản (text box)

Hộp liệt kê (List box)

Khung hiển thi (Preview)

Đóng hộp

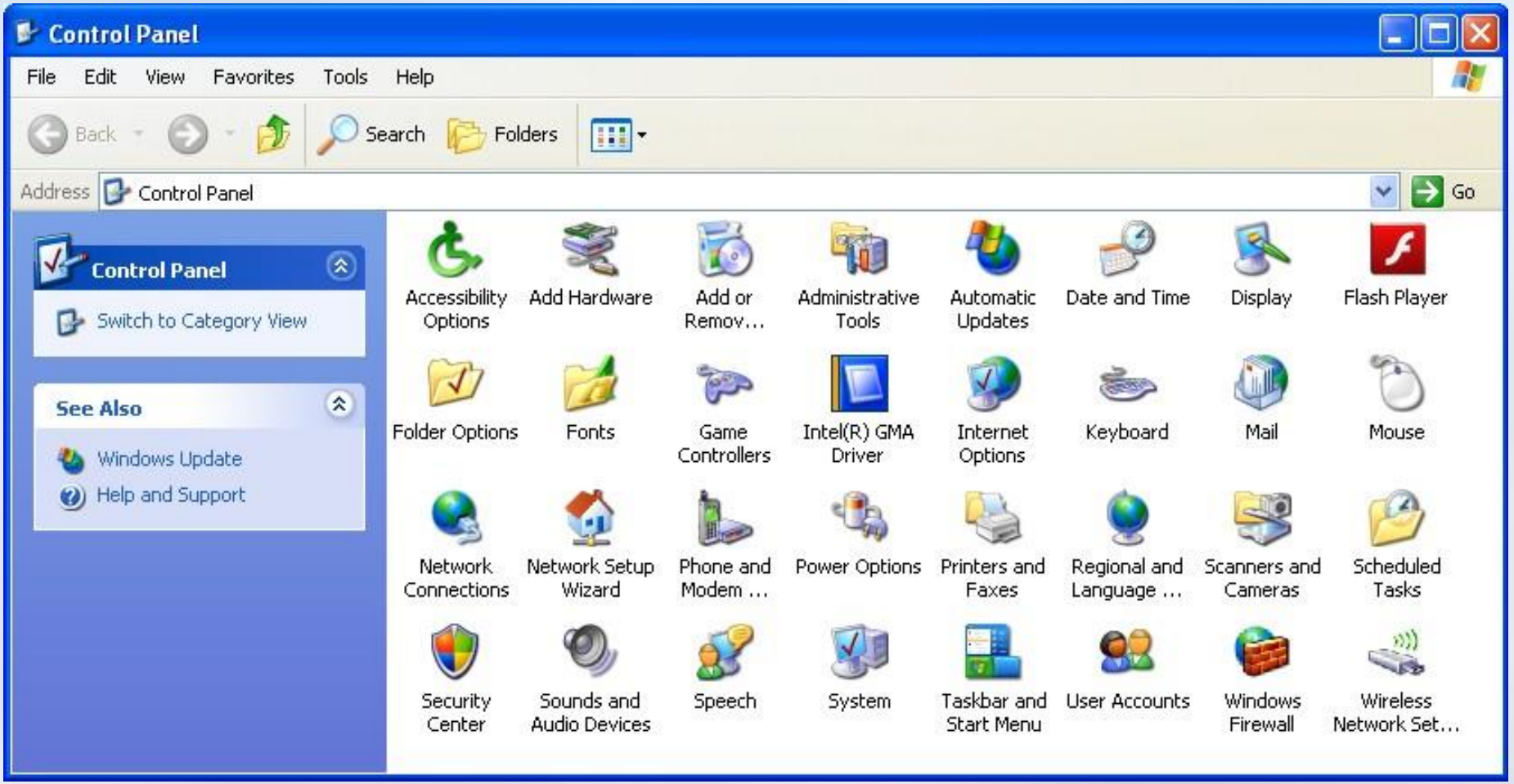
Hộp liệt kê thả (Drop down combo box)

Hộp kiểm tra (Check box)

Nút lệnh (Comman d Button)

Cấu hình Windows

Start [→Settings] → Control Panel

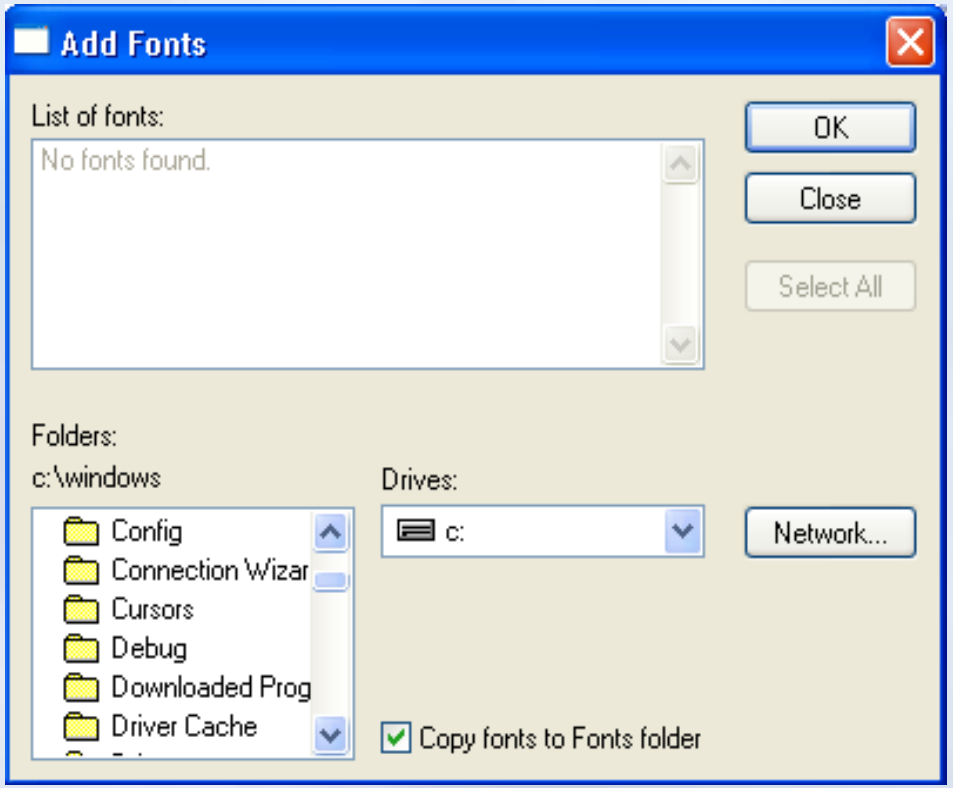


Cấu hình Windows

- Trong Control Panel
  - Cài đặt và loại bỏ Font chữ
  - Thay đổi dạng hiện màn hình Desktop
  - Cài đặt và loại bỏ chương trình
  - Cấu hình ngày, giờ cho hệ thống
  - Thay đổi thuộc tính của bàn phím và chuột
  - Thay đổi thuộc tính vùng (Regional Settings)
  - Cài đặt / loại bỏ máy in
  - ….

Cài đặt và loại bỏ Font chữ

- Control Panel → Fonts
- **Loại bỏ font chữ**
  - Chọn fonts cần bỏ
  - Nhấn phím del hoặc chọn File →Delete
- **Thêm font chữ mới**
  - File → Install New Font
  - Chỉ ra nơi chứa các Font nguồn muốn thêm bằng cách chọn tên ổ đĩa chứa các tập tin Font chữ
  - Chọn các tên Font và Click OK.



Thay đổi dạng hiện màn hình nền (Desktop)

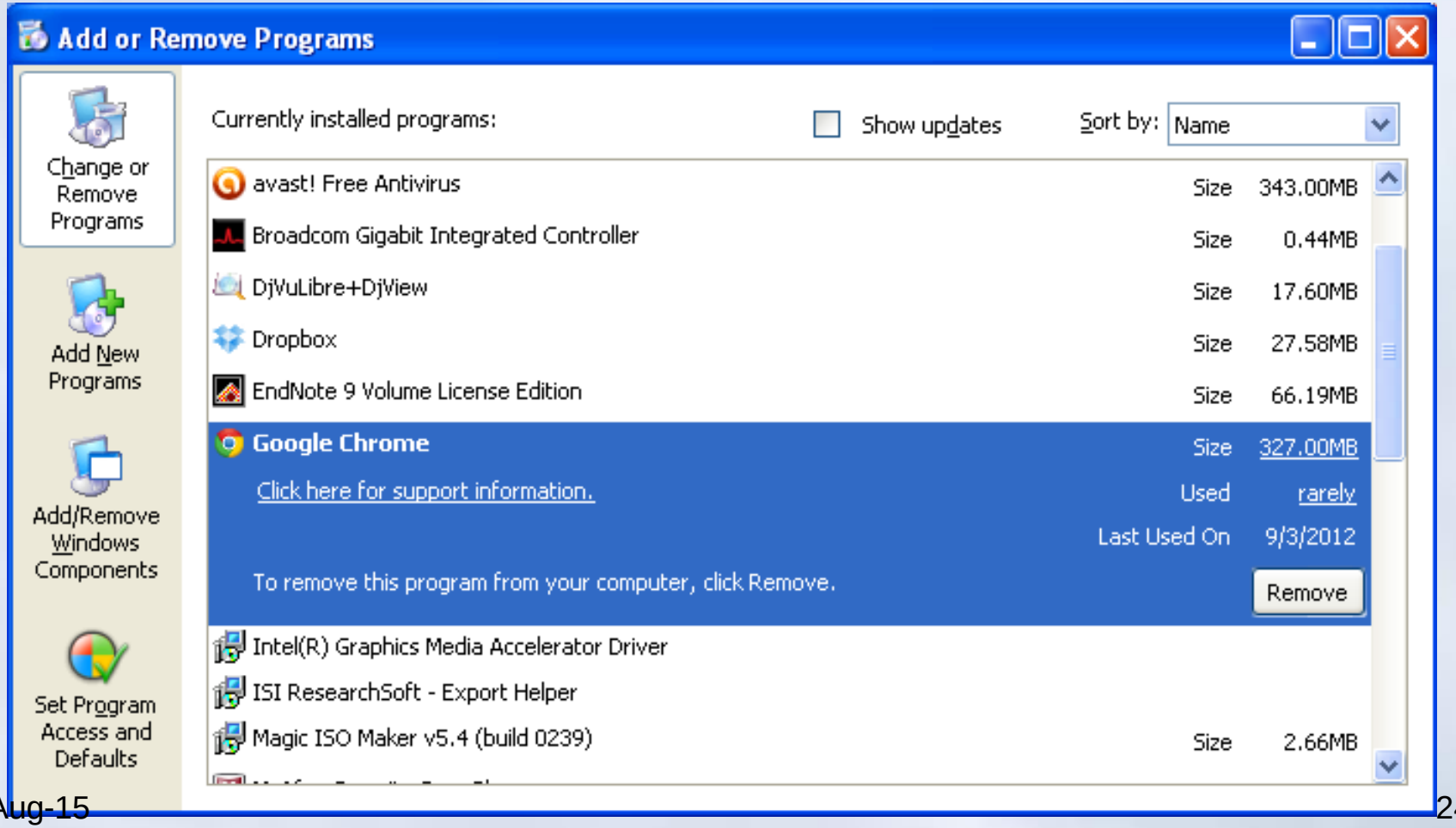
1. Control Panel → Chọn Display
2. R_Click trên màn hình nền →Properties

- Desktop
  - Thay đổi màn hình nền
- Screen Saver
  - Xác lập màn hình nghỉ
- Setting
  - Thay đổi chế độ màu và độ phân giải màn hình

Cài đặt và loại bỏ chương trình

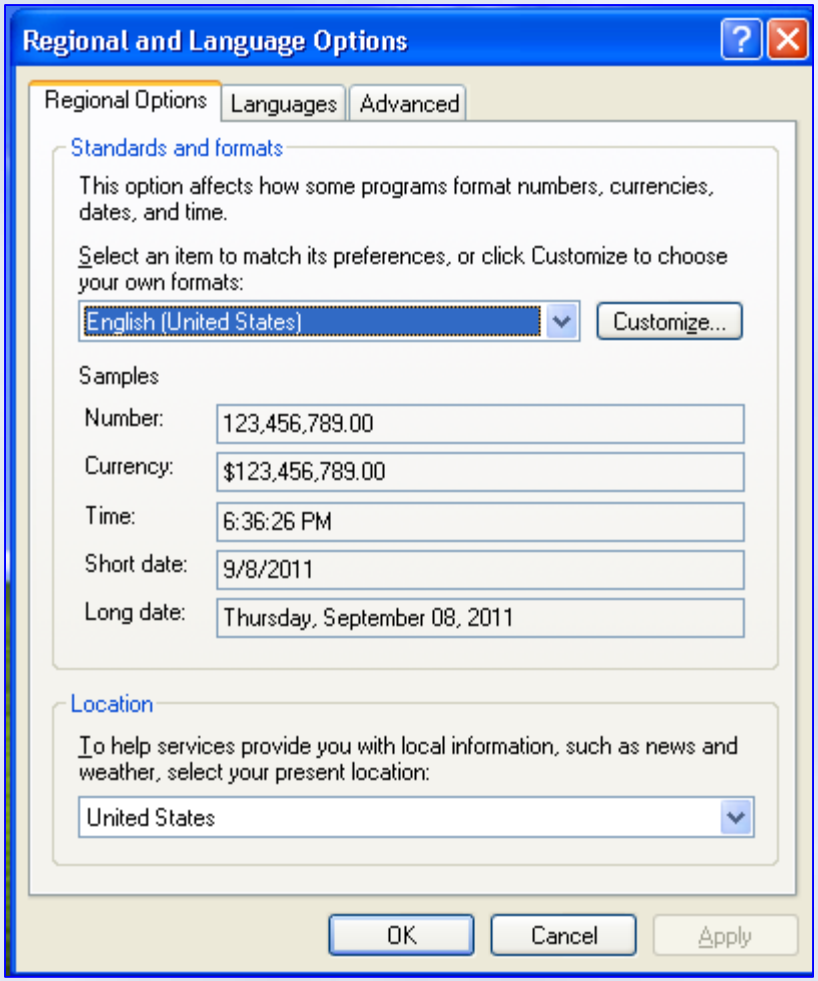
Control Panel → Add or Remove Program

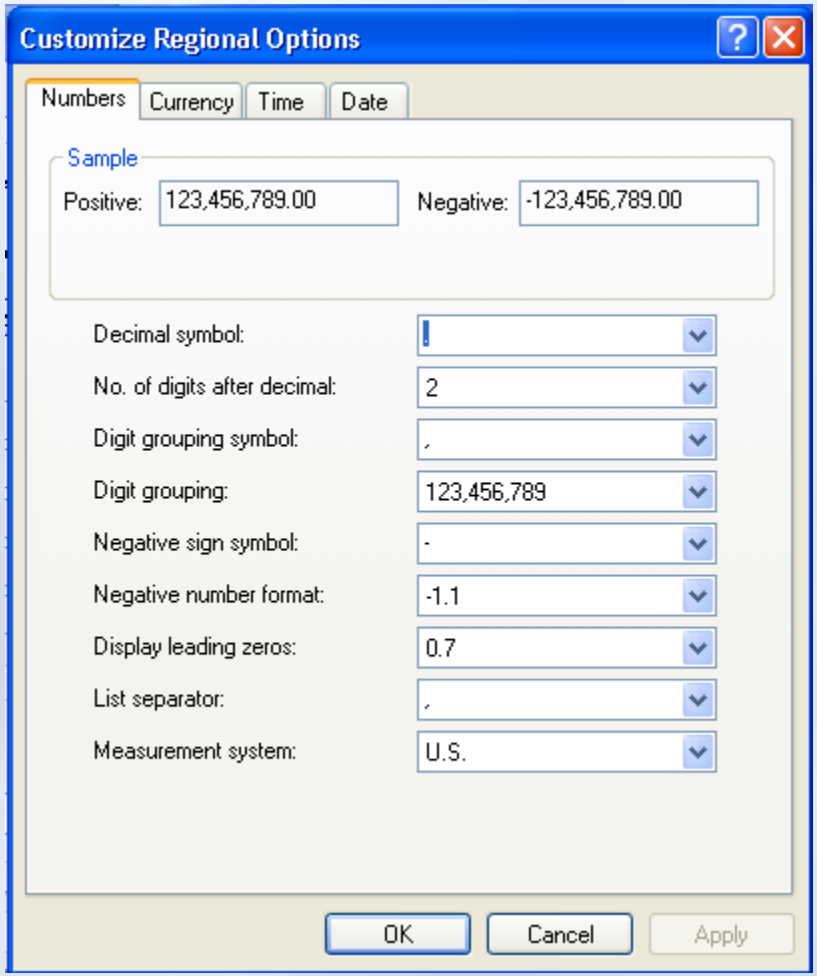
- Lựa chon công và việc làm theo chỉ dẫn
  - **Loại bỏ:** Change or Remove programs → Ứng dụng→Remove



Thay đổi thuộc tính vùng

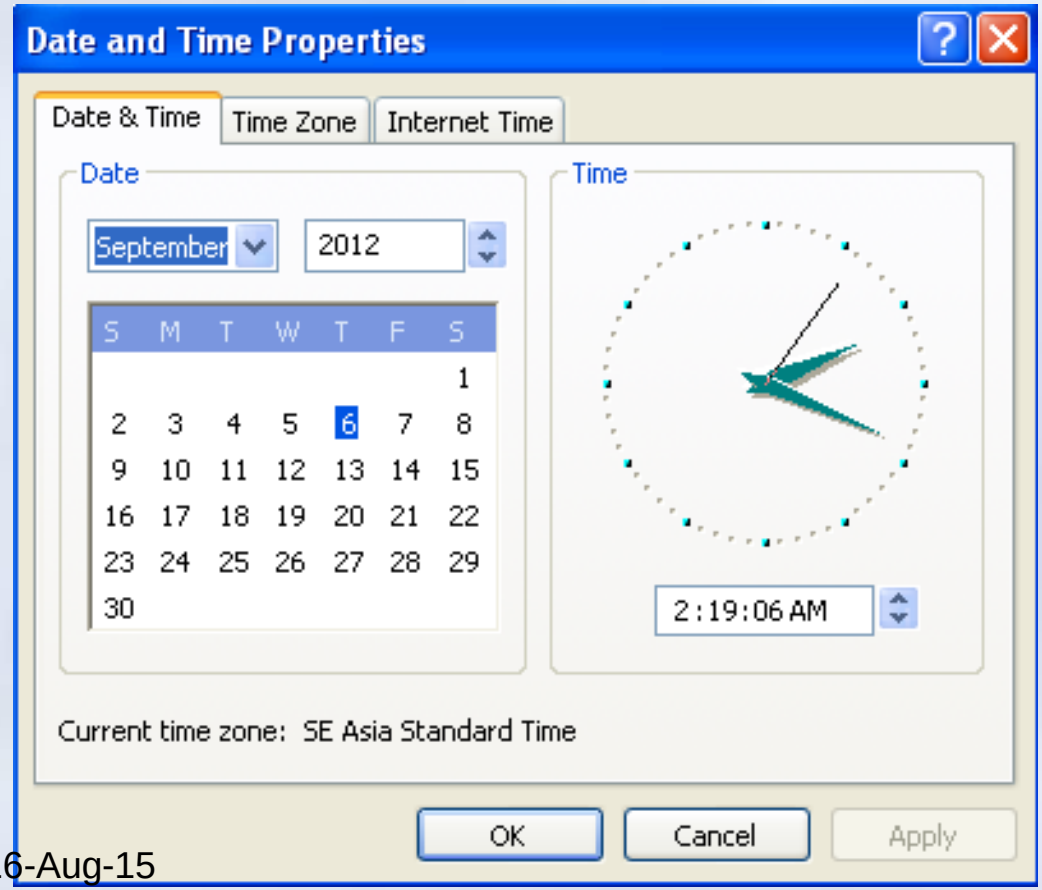
Control Panel → Regional and Language Option





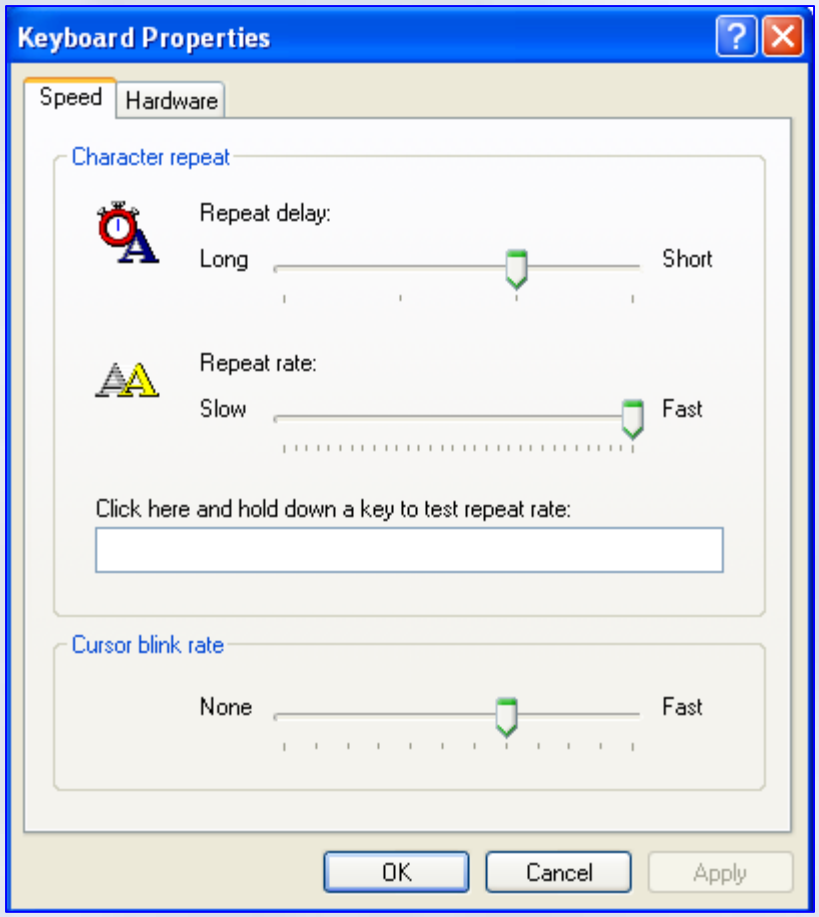
Thay đổi ngày giờ hệ thống

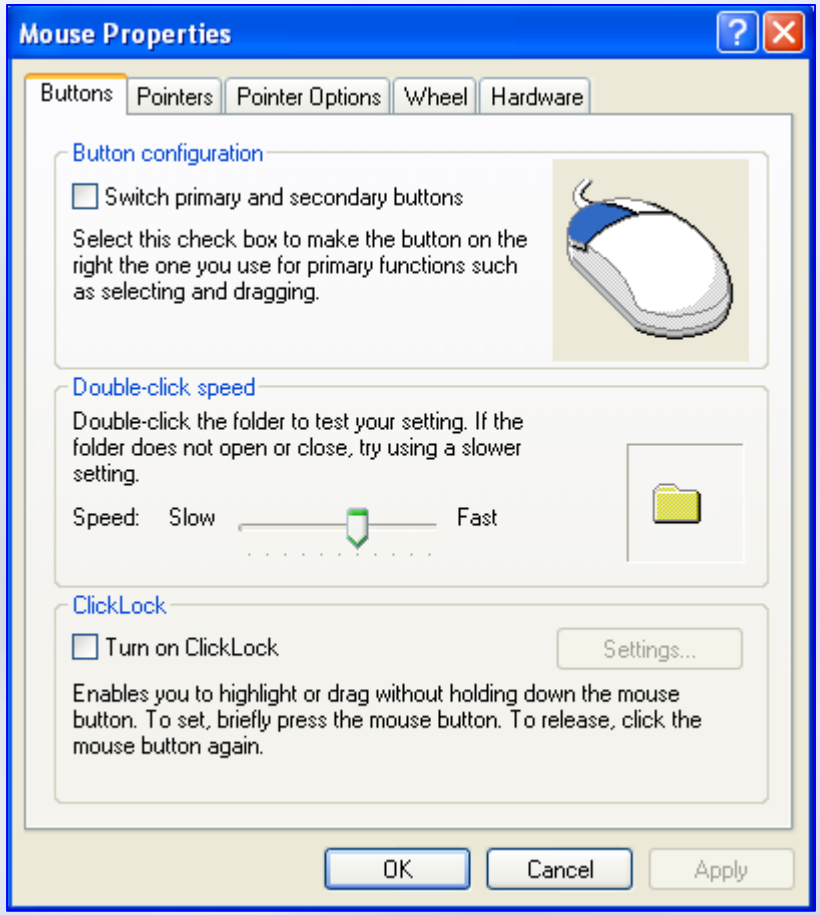
1. Control Panel → Date and Time
2. D_Click lên biểu tượng đồng hồ trên thanh taskbar



Thay đổi thuộc tính bàn phím và chuột

- Control Panel → Keyboard
- Control Panel → Mouse





Cài đặt / loại bỏ máy in

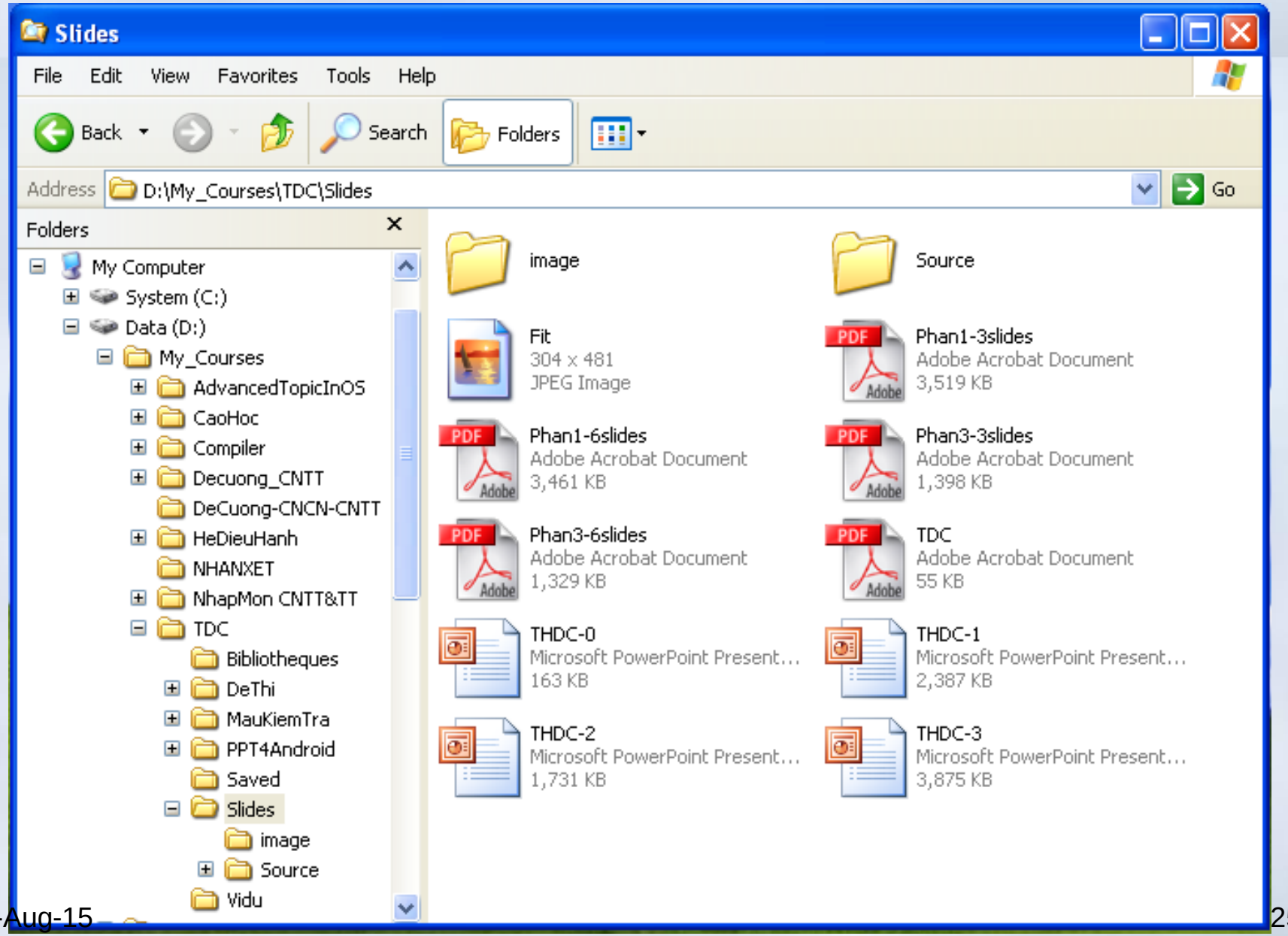
- Windows tích hợp sẵn chương trình điều khiển (driver) của các máy in thông dụng.
- Với máy in mà trong Windows chưa có chương trình điều khiển, muốn sử dụng cần phải cài đặt
- **Các bước cài đặt máy in:**
  - Control Panel → Printers and Faxes
  - Chọn menu File → Add a Printer, xuất hiện hộp thoại
  - Làm theo các bước hướng dẫn của hệ thống

- **Loại bỏ máy in đã cài đặt**
  - Control Panel→ Printers and Faxes
  - Click chuột chọn máy in muốn loại bỏ
  - Nhấn phím Delete, sau đó chọn Yes

Windows Explorer

- Được hỗ trợ từ phiên bản Windows 95
  - *Cho phép thao tác với các tài nguyên* có trong máy tính cũng như các máy tính trong hệ thống mạng
  - Các thao tác với hệ thống như sao chép, xóa file thư mục,..được thực hiện bởi các thao tác kéo/thả…
- Kích hoạt Windows Explorer
  1. Start→All Programs→Accessories → Windows Explorer
  2. R_Click lên nút Start và chọn Explore
  3. R_Click lên biểu tượng My Computer và chon Explore
- Làm việc với Windows Explorer
  - Cửa sổ trái là cây thư mục, phải nội dung tương ứng
  - Các thao tác với thư mục/tệp
    - Mở, tạo, sao chép, di chuyển, xóa, chọn nhóm,...

Windows Explorer



Xóa thư mục và tập tin

- Chọn thư mục/ tập tin cần xóa
  - Nhấn phím Delete
  - Chọn menu File→Delete
  - R_Click → Delete
- Xác nhận có thực sự muốn xóa không (Yes/no)
- Chú ý:
  - Các đối tượng mới chỉ được đưa vào Recycle Bin
    - Muốn xóa hẳn phải xóa trong Recycle Bin
    - R_Click lên Recycle Bin →Empty Recycle Bin để xóa cả
- Khôi phục thư mục/tập tin trong Recycle Bin
  - Mở Recycle Bin
  - Chọn tên đối tượng cần khôi phục
  - Chọn Menu File → Restore hoặc R_Click → Restore

Nội dung chính

Chương 1: Thông tin và biểu diễn thông tin

- Các khái niệm cơ bản về thông tin và tin học
- Biểu diễn dữ liệu trong máy tính

Chương 2: Hệ thống máy tính

- Hệ thống máy tính
- Mạng máy tính
- Hệ điều hành

Chương 3: Các hệ thống ứng dụng

- Hệ thống thông tin quản lý
- Hệ thông tin bảng tính
- Hệ quản trị cơ sở dữ liệu
- Các hệ thống thông minh

Nội dung chính

1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

Khái niệm

- Là hệ thống bao gồm
  - Phần cứng, phần mềm, con người
  - Quy trình thu thập, phân tích, xử lý đánh giá, phân phối và chia sẻ những thông tin cần thiết
- Hệ thống thủ công
  - Quản lý dựa trên giấy tờ, sổ sách
    - Dễ sai sót và nhầm lẫn
    - Lãng phí tài nguyên
      - Diện tích, không gian, thời gian sắp xếp..
- Hệ thống tự động
  - Tận dụng khả năng của máy tính điện tử

Các thành phần cơ bản

- Cơ sở hạ tầng:
  - Phần cứng, phần mềm truyền thông
- Phần mềm
- Cơ sở dữ liệu
- Quy trình
- Nhân sự

Các chức năng chính của hệ thống

- Nhập dữ liệu
  - Thu nhận dữ liệu từ bên ngoài/bên trong để xử lý
- Lưu trữ thông tin
  - Thông tin được lưu trữ để phân tích trong tương lai
  - Thường được lưu trữ trong các hệ quản trị CSDL
- Xử lý thông tin
  - Chuyển đổi từ dữ liệu hỗn hợp thành dạng có ý nghĩa
- Xuất dữ liệu
  - Phân phối thông tin đã xử lý tới đối tượng cần
- Thông tin phản hồi
  - Dùng đánh giá, hoàn thiện lại quá trình thu thập dữ liệu, quá trình xử lý hệ thống

Các dạng của thông tin trong tổ chức

- Thông tin theo quan điểm cá nhân
  - Thông tin được xem xét theo 3 khía cạnh
    - Thời gian: Mức độ cập nhật của thông tin tại thời điểm xem xét
    - Không gian: Địa điểm truy cập thông tin: ở nhà, cơ quan, hay di động
    - Dạng thức: thông tin ở dạng đơn giản, tổng hợp
- Thông tin theo quan điểm tổ chức
  - Dòng thông tin:
    - Trên xuống, dưới lên, ngang hàng từ ngoài vào,..
  - Mức độ chi tiết
    - Cho cấp lạnh đạo (ra quyết định): thông tin tổng hợp
    - Phục vụ cho các hoạt động: Chi tiết, cụ thể

Các đặc tính của thông tin

Chất lượng thông tin xác định qua các đặc tính

- Tính chính xác
  - Tính chính xác thấp, gây hậu quả tồi tệ
- Tính đầy đủ
  - Thể hiện sự bao quát các vấn đề
  - Không đầy đủ có thể dẫn đến kết quả sai lầm
- Tính thống nhất
  - Cần thống nhất giữa thông tin tổng hợp và chi tiết
- Tính thích hợp và dễ hiểu
  - Thông tin không thích hợp với vấn đề xem xét,
  - Thông tin đa nghĩa, tổ chức thông tin rắc rối…
- Tính kịp thời
  - Cần xuất hiện đúng lúc cần thiết

Các giai đoạn phát triển hệ thống thông tin

1. Lập kế hoạch: Mục tiêu là cần xác định
   - Dạng hệ thống thông tin được phát triển
   - Phạm vi làm việc của êệ thống
   - Các nhiệm vụ cụ thể
   - Nguồn lực và khung thời gian cho phất triển
2. Phân tích:
   - Thu thập thông tin
   - Tìm hiểu hệ thống, nhiệm vụ hệ thống
   - Lập tài liệu xác định yêu cầu

Các giai đoạn phát triển hệ thống thông tin

3. Thiết kế
   - Thiết kế kỹ thuật, xây dựng mô hình hệ thống
     - Thiết kế tổng thể, thiết kế chi tiết cho từng phần
   - Kết quả: Tài liệu thiết kế tổng thể & chi tiết
4. Cài đặt
   - Thiết lập CSDL, cơ sở hạ tầng công nghệ
   - Xây dựng theo yêu cầu thiết kế
5. Kiểm định
   - Xem xét toàn bộ để đảm bảo hệ thống đã phát triển đáp ứng các mục tiêu và yêu cầu đặt ra

Các giai đoạn phát triển hệ thống thông tin

6. Vận hành: Đưa hệ thống vào hoạt động theo các chến lược:
   - Song song hệ thống cũ và mới cho tới khi hệ thống mới hoạt động tốt
   - Thí điểm tại một số bộ phận để kiểm định các vấn đề trước khi triển khai toàn bộ
   - Theo giai đoạn: Triển khai từng chức năng của hệ thống→Thích hợp với hệ thống lớn
   - Thay thế toàn bộ hệ thống cũ bằng hệ thống mới
     - Chấp nhận rủi ro cao

6. Bảo trì
   - Thường xuyên kiểm tra, giám sát và hỗ trợ nhưng thay đổi cần thiết trong thời gian vận hành

Nội dung chính

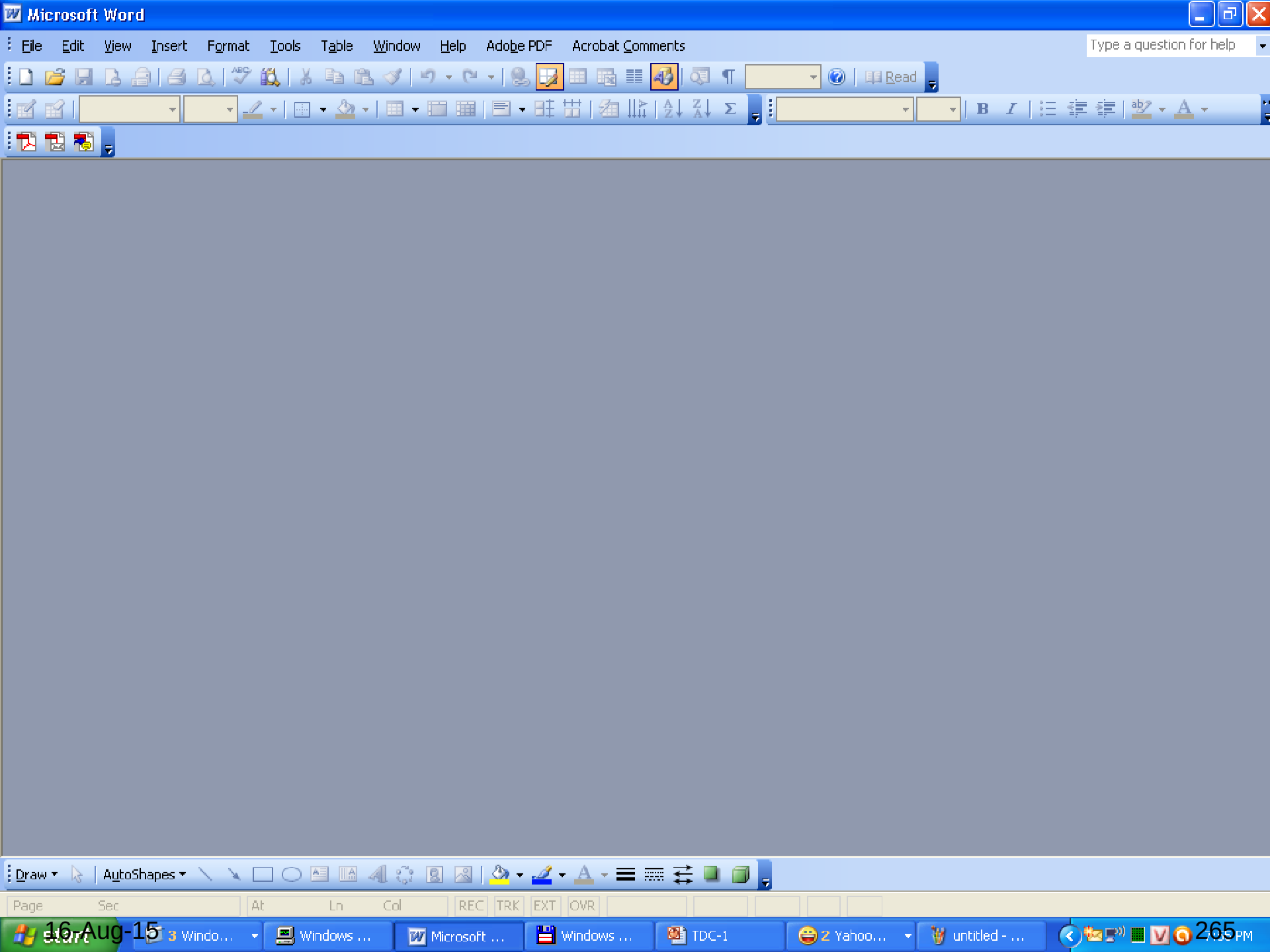
1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

Giới thiệu

- Phần mềm soạn thảo văn bản điện tử, cho phép:
  - Hiện thị nội dung văn bản lên màn hình
  - Dễ dàng sửa đổi, bổ sung tại vị trí bất kỳ
  - Dễ dàng thay đổi kiểu chữ, cỡ chữ,
  - Cho phép kèm theo hình ảnh, bảng, biểu
  - Cho phép lưu trữ văn bản dưới dạng file
  - Hỗ trợ in ấn,..
- Một số phần mềm soạn thảo
  - Microsoft word của Microsofft
  - Word Perfect của hãng Corel
  - Writer trong bộ Open office

Microsoft word

Start → All Programs → Microsoft Office → Microsoft Office Word 2003


Microsoft Word
File Edit View Insert Format Tools Table Window Help Adobe PDF Acrobat Comments
Type a question for help
Read
Draw AutoShapes
Page Sec At Ln Col REC TRK EXT OVR
start
3 Windo...
Windows ...
Microsoft ...
Windows ...
TDC-1
2 Yahoo...
untitled - ...
PM

Bắt đầu soạn thảo một văn bản

Kiểu gõ Telex

a+a → â	e+e→ê	o+o→ô	u+w→ư
a+w →ă	E+E→Ê	o+w→ơ	U+ư→Ư
d+d→đ		D+D→ Đ	

	f	s	j	r	x
a	à	á	ạ	ả	ã
â	ầ	ấ	ậ	ẩ	ẫ
ă	ằ	ắ	ặ	ẳ	ẵ
o	ò	ó	ọ	ỏ	õ
ô	ồ	ố	ộ	ổ	ỗ

Nội dung chính

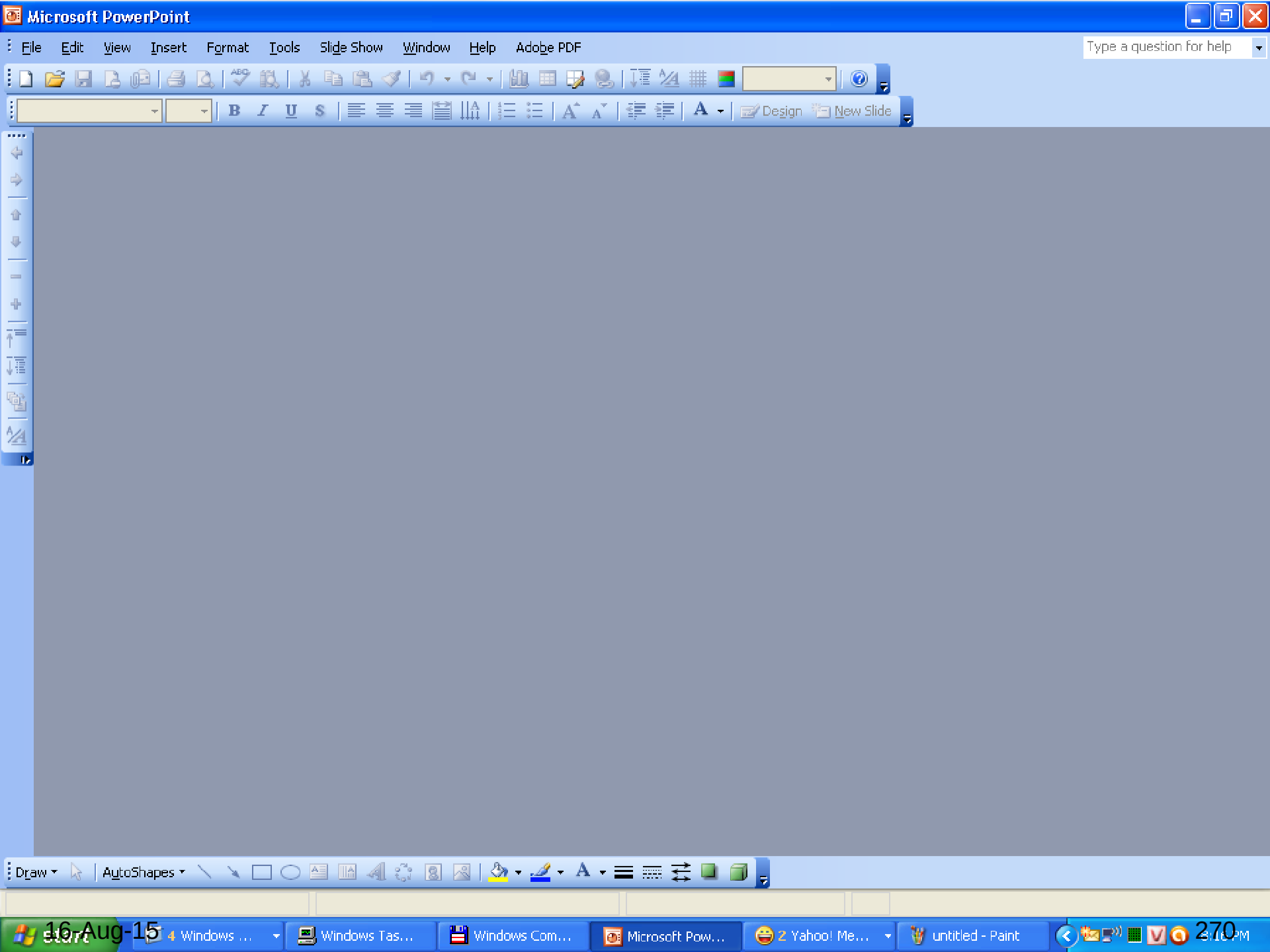
1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

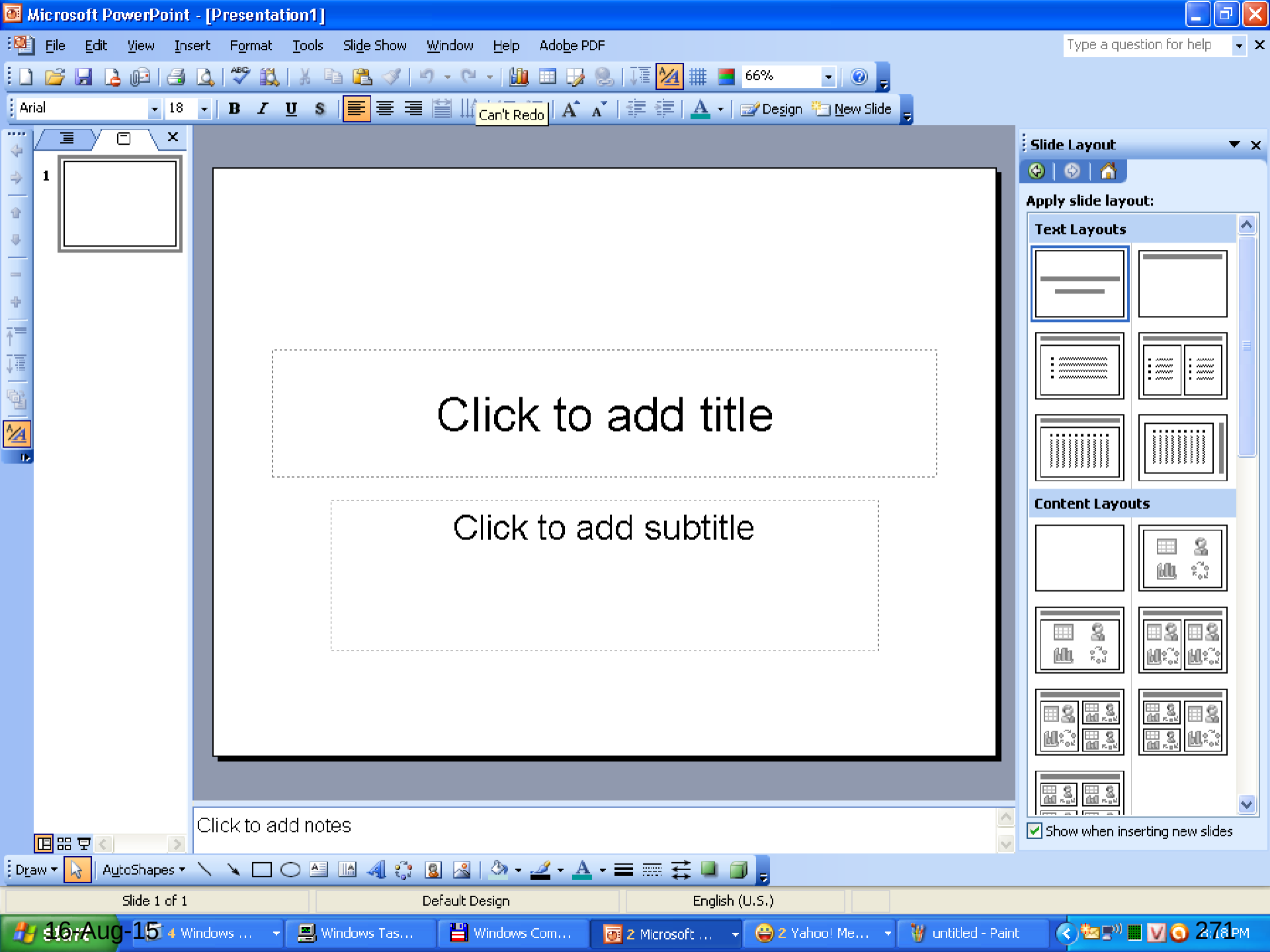
Giới thiệu

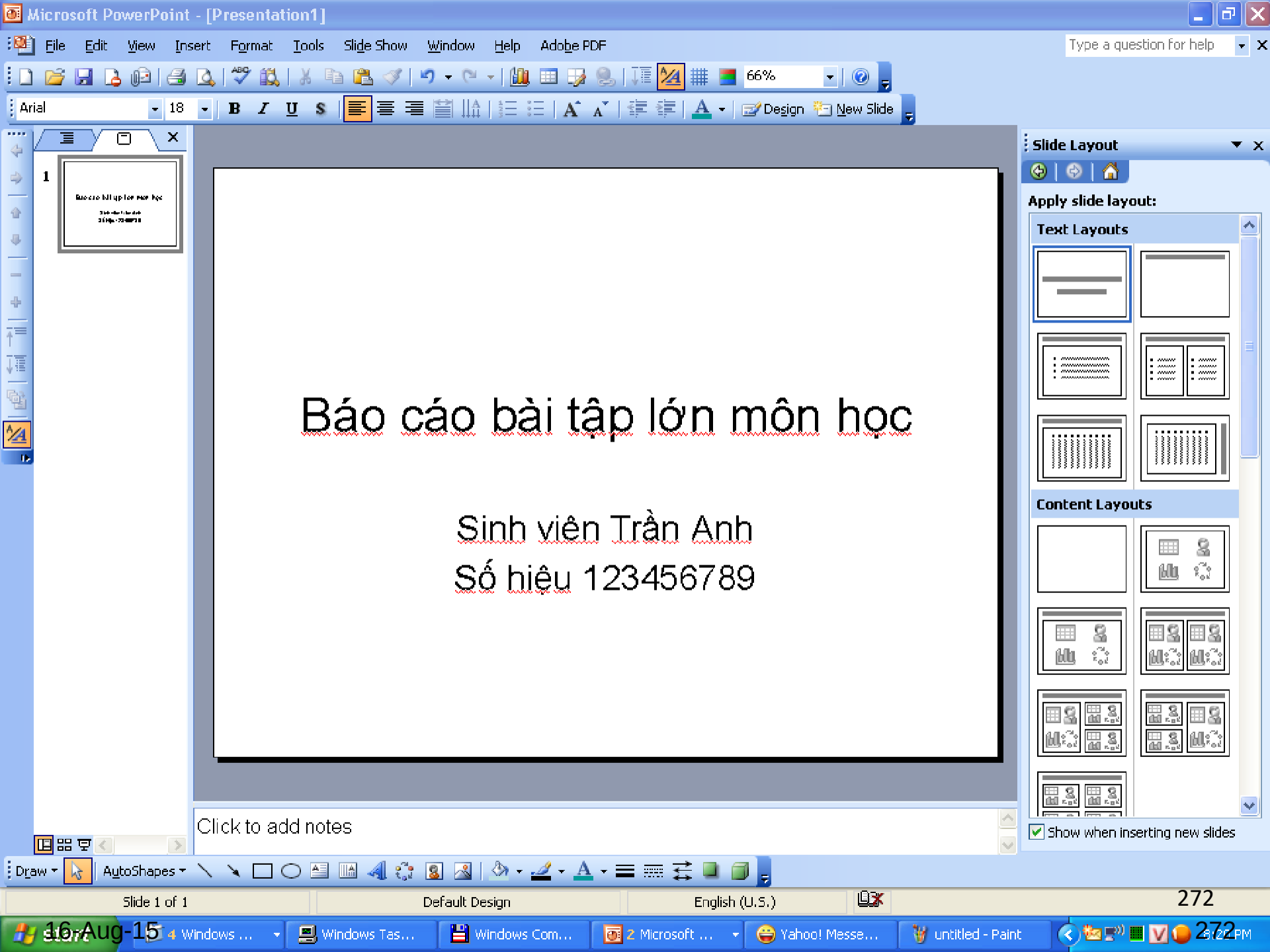
- Là phần mềm trình bày thông tin dạng slides
  - Văn bản, hình ảnh, biểu đồ, âm thanh
- Phần mềm trình chiếu cần
  - Khả năng biên tập, khả năng trình chiếu
  - Khả năng hoạt cảnh (animation)
- Một số phần mềm
  - Microsoft PowerPoint của Microsoft
  - Impress trong bộ OpenOffice
  - Keynote cua Apple

Microsoft PowerPoint

Start → All Programs → Microsoft Office → Microsoft Office PowerPoint 2003

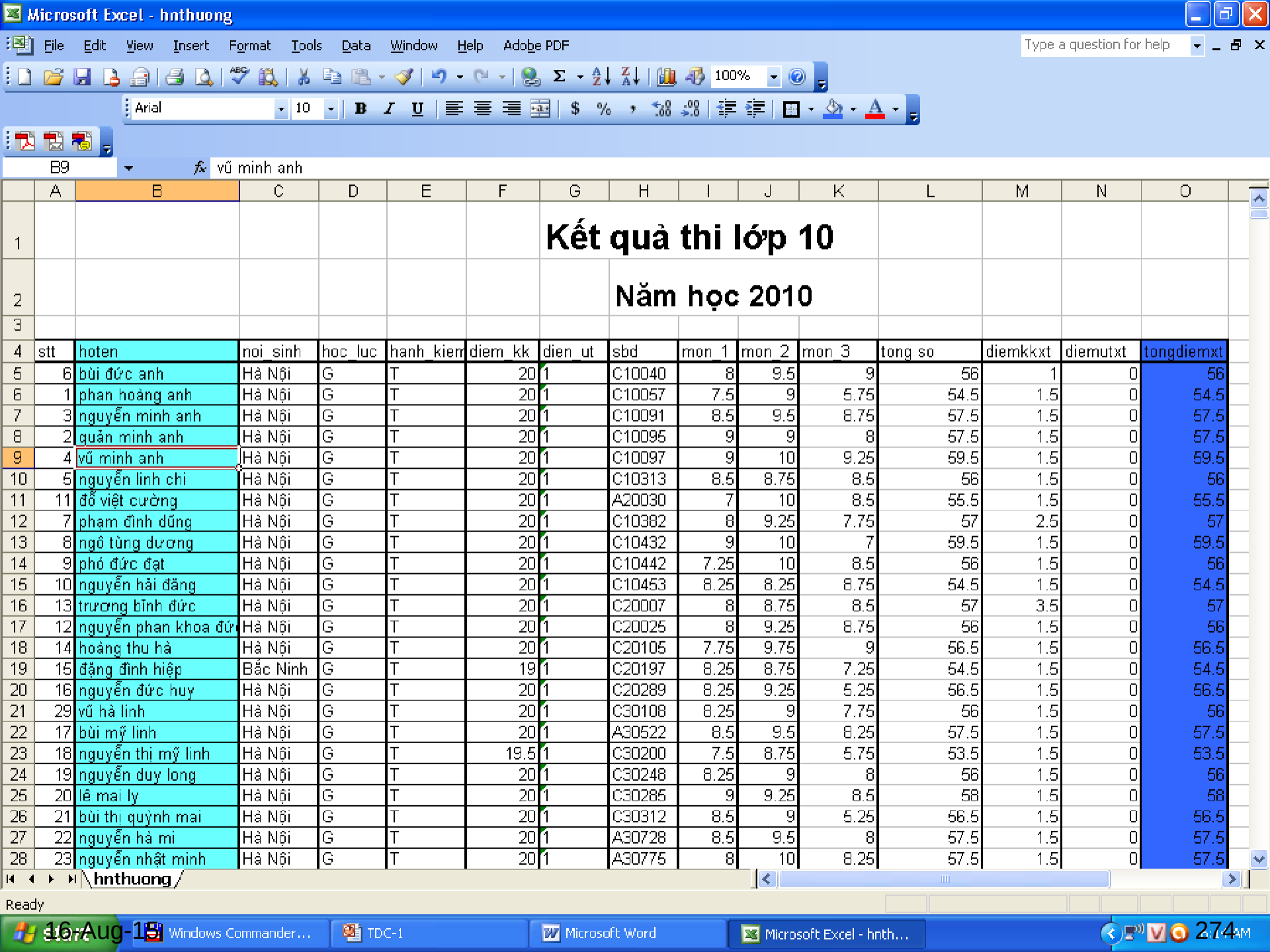

Microsoft PowerPoint
File Edit View Insert Format Tools Slide Show Window Help Adobe PDF
Type a question for help
Design New Slide
Draw AutoShapes
start
4 Windows ...
Windows Tas...
Windows Com...
Microsoft Pow...
2 Yahoo! Me...
untitled - Paint


Microsoft PowerPoint - [Presentation1]
File Edit View Insert Format Tools Slide Show Window Help Adobe PDF
Type a question for help
66%
Arial 18
Can't Redo
Design New Slide
1
Click to add title
Click to add subtitle
Click to add notes
Slide Layout
Apply slide layout:
Text Layouts
Content Layouts
Show when inserting new slides
Draw AutoShapes
Slide 1 of 1
Default Design
English (U.S.)
start
Windows Tas...
Windows Com...
2 Microsoft ...
2 Yahoo! Me...
untitled - Paint


Microsoft PowerPoint - [Presentation1]
File Edit View Insert Format Tools Slide Show Window Help Adobe PDF
Type a question for help
66%
Arial
18
Design
New Slide
Báo cáo bài tập lớn môn học
Sinh viên Trần Anh
Số hiệu 123456789
Click to add notes
Slide Layout
Apply slide layout:
Text Layouts
Content Layouts
Show when inserting new slides
Draw
AutoShapes
Slide 1 of 1
Default Design
English (U.S.)
272
16-Aug-15
272

Nội dung chính

1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử



Microsoft Excel - hnthuong

File Edit View Insert Format Tools Data Window Help Adobe PDF

Type a question for help

Arial 10

B9 vũ minh anh

Kết quả thi lớp 10

Năm học 2010

stt	hoten	noi_sinh	hoc_luc	hanh_kiem	diem_kk	dien_ut	sbd	mon_1	mon_2	mon_3	tong so	diemkkxt	diemutxt	tongdiemxt
6	bùi đức anh	Hà Nội	G	T	20	1	C10040	8	9.5	9	56	1	0	56
1	phan hoàng anh	Hà Nội	G	T	20	1	C10057	7.5	9	5.75	54.5	1.5	0	54.5
3	nguyễn minh anh	Hà Nội	G	T	20	1	C10091	8.5	9.5	8.75	57.5	1.5	0	57.5
2	quản minh anh	Hà Nội	G	T	20	1	C10095	9	9	8	57.5	1.5	0	57.5
4	vũ minh anh	Hà Nội	G	T	20	1	C10097	9	10	9.25	59.5	1.5	0	59.5
5	nguyễn linh chi	Hà Nội	G	T	20	1	C10313	8.5	8.75	8.5	56	1.5	0	56
11	đỗ việt cường	Hà Nội	G	T	20	1	A20030	7	10	8.5	55.5	1.5	0	55.5
7	phạm đình dũng	Hà Nội	G	T	20	1	C10382	8	9.25	7.75	57	2.5	0	57
8	ngô tùng dương	Hà Nội	G	T	20	1	C10432	9	10	7	59.5	1.5	0	59.5
9	phó đức đạt	Hà Nội	G	T	20	1	C10442	7.25	10	8.5	56	1.5	0	56
10	nguyễn hải đăng	Hà Nội	G	T	20	1	C10453	8.25	8.25	8.75	54.5	1.5	0	54.5
13	trương bình đức	Hà Nội	G	T	20	1	C20007	8	8.75	8.5	57	3.5	0	57
12	nguyễn phan khoa đứ	Hà Nội	G	T	20	1	C20025	8	9.25	8.75	56	1.5	0	56
14	hoàng thu hà	Hà Nội	G	T	20	1	C20105	7.75	9.75	9	56.5	1.5	0	56.5
15	đặng đình hiệp	Bắc Ninh	G	T	19	1	C20197	8.25	8.75	7.25	54.5	1.5	0	54.5
16	nguyễn đức huy	Hà Nội	G	T	20	1	C20289	8.25	9.25	5.25	56.5	1.5	0	56.5
29	vũ hà linh	Hà Nội	G	T	20	1	C30108	8.25	9	7.75	56	1.5	0	56
17	bùi mỹ linh	Hà Nội	G	T	20	1	A30522	8.5	9.5	8.25	57.5	1.5	0	57.5
18	nguyễn thị mỹ linh	Hà Nội	G	T	19.5	1	C30200	7.5	8.75	5.75	53.5	1.5	0	53.5
19	nguyễn duy long	Hà Nội	G	T	20	1	C30248	8.25	9	8	56	1.5	0	56
20	lê mai ly	Hà Nội	G	T	20	1	C30285	9	9.25	8.5	58	1.5	0	58
21	bùi thị quỳnh mai	Hà Nội	G	T	20	1	C30312	8.5	9	5.25	56.5	1.5	0	56.5
22	nguyễn hà mi	Hà Nội	G	T	20	1	A30728	8.5	9.5	8	57.5	1.5	0	57.5
23	nguyễn nhật minh	Hà Nội	G	T	20	1	A30775	8	10	8.25	57.5	1.5	0	57.5

hnthuong

Ready

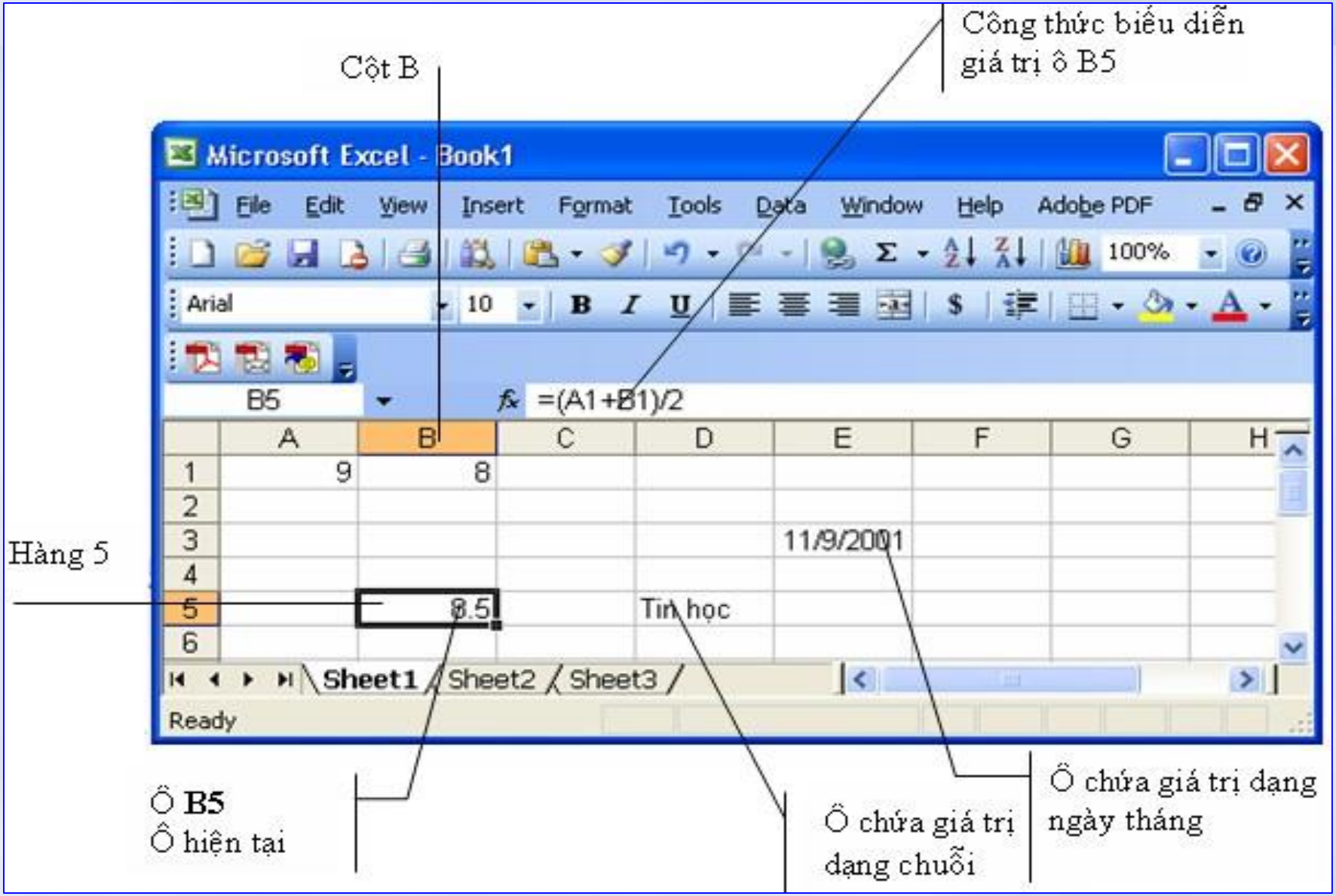
Khái niệm

- Máy tính
  - Hỗ trợ việc tính toán, nhất là kế toán và phân tích thống kê.
- Phần mềm hỗ trợ tính toán thông dụng:
  - Phầm mềm bảng tính (PMBT) spreadsheet software
- Phầm mềm bảng tính
  - Giúp tính toán các số liệu
  - Cho phép xây dựng và làm việc với những tình huống mô phỏng thế giới thực
  - Làm việc dựa trên các bảng tính.

Bảng tính

- Dạng ô lưới gồm
  - Các hàng đánh số từ 1
  - Các cột đánh số từ chữ A
  - Ô là giao của 1 hàng và 1 cột.
    - Ví dụ ô A1 là giao của hàng 1 và cột A.
- Mỗi ô có thể chứa dữ liệu dạng số, chuỗi kí tự, ngày tháng hoặc công thức hiển thị liên hệ giữa các con số.

Bảng tính


Công thức biểu diễn giá trị ô B5
Cột B
Microsoft Excel - Book1
File Edit View Insert Format Tools Data Window Help Adobe PDF
Arial 10
B5
=(A1+B1)/2
A B C D E F G H
1 9 8
2
3 11/9/2001
4
5 8.5 Tin học
6
Sheet1 Sheet2 Sheet3
Ready
Hàng 5
Ô B5
Ô hiện tại
Ô chứa giá trị dạng chuỗi
Ô chứa giá trị dạng ngày tháng

Tính toán trên bảng tính

- Tại ô A1 là điểm toán (giả sử 9)
- Tại ô B1 là điểm Lý (giả sử 8)
- Để kết quả trung bình hiển thị tại ô B5
  - tại B5 điền công thức “=(A1+B1)/2”.
- Chú ý:
  - *Công thức ở ô B5 không hiển thị, mà chỉ thấy kết quả cuối cùng.*
  - *Giá trị tại A1 và B1 thay đổi thì lập tức giá trị ở B5 cũng sẽ được tính toán lại.*

Các chức năng cơ bản của PMBT

- **Tự động lặp các giá trị, tiêu đề và công thức:** Giúp đơn giản hóa việc nhập các dữ liệu lặp.
- **Tự động tính lại:** Khi có một sự thay đổi tại 1 ô thì toàn bộ bảng tính sẽ được tính toán lại.
- **Các hàm thư viện:** thực hiện các công việc tính toán đã định sẵn. Giúp tiết kiệm thời gian và giảm nguy cơ phát sinh lỗi

Các chức năng cơ bản của PMBT

- **Macro**: Giúp "thu" lại các thao tác lặp đi lặp lại và định nghĩa nó là 1 macro. Khi cần thực hiện các thao tác đó thì chỉ việc gọi macro tương ứng.
- **Liên kết:** Cho phép tạo liên kết động giữa các bảng tính.
- **Biểu đồ:** Biểu diễn các con số dưới nhiều dạng biểu đồ khác nhau
- **Cơ sở dữ liệu:** Cho phép thao tác: lưu trữ và truy cập thông tin, tìm kiếm, báo cáo,…

Nội dung chính

1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

Khái niệm cơ sở dữ liệu

- **Khái niệm**:
  - Là một hệ thống các thông tin có cấu trúc
    - Mã số SV, Tên SV, Địa chỉ, Ngày sinh,..
  - Được lưu trữ trên các thiết bị lưu trữ thông tin
- **Ví dụ:**
  - Trang niên giám điện thoại
  - Danh sách sinh viên
  - Hệ thống tài khoản ngân hàng

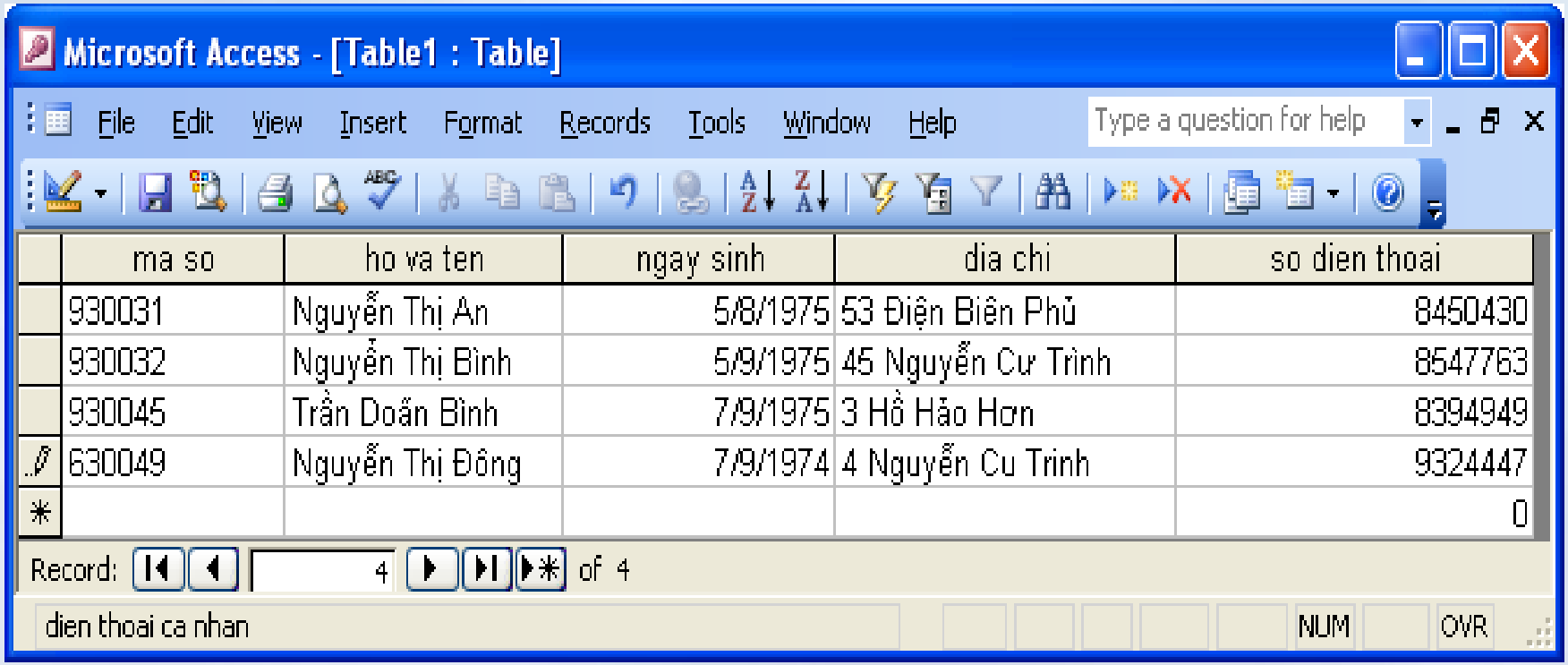
Ưu điểm khi dùng CSDL

- Lưu trữ một lượng thông tin khổng lồ trở nên dễ dàng.
- Dễ dàng sắp xếp và tổ chức thông tin
- Nhanh chóng và mềm dẻo trong việc tra cứu thông tin.
- Cho phép in và phân phối thông tin theo nhiều cách.

Hệ quản trị cơ sở dữ liệu

- Là hệ thống phần mềm cho phép
  - Định nghĩa, tạo lập cơ sở dữ liệu, lưu trữ dữ liệu trên thiết bị nhớ
  - Thực hiện các thao tác như, tìm kiếm, cập nhật, kết xuất… thông tin
- Một số hệ quản trị CSDL phổ biến
  - *MS SQL Server; Oracle; MySQL, MS Access..*

Hệ quản trị cơ sở dữ liệu



Microsoft Access

Các tính năng của một hệ quản trị CSDL

- Quản lý dữ liệu tồn tại lâu dài
  - Do nhu cầu lưu trữ dữ liệu của các tổ chức
    - Lưu trữ nhiều năm
- Truy xuất dữ liệu một cách hiệu quả
  - Dữ liệu được lưu trữ thường rất lớn
  - Yêu cầu thời gian đáp ứng của các thao tác dữ liệu phải được thỏa mãn
- Hỗ trợ một mô hình dữ liệu
  - Thường dùng mô hình quan hệ trong biểu diễn dữ liệu

Các tính năng của một hệ quản trị CSDL

- Đảm bảo tính độc lập dữ liệu
  - Độc lập giữa dữ liệu và chương trình thao tác
    - Các chương trình có thể phát triển theo nhu cầu mà không ảnh hưởng tới kho dữ liệu đã có
- Hỗ trợ ngôn ngữ cấp cao cho phép
  - Định nghĩa cấu trúc, tạo lập CSDL
  - Truy nhập và các thao tác với dữ liệu
    - Thêm mới, cập nhật, tìm kiếm, xóa bỏ,..
  - Ví dụ SQL, QBE

Các tính năng của một hệ quản trị CSDL

- Quản trị giao dịch
  - Cung cấp các truy nhập đồng thời từ nhiều người dùng
  - Hỗ trợ giao dịc tương tranh
- Điều khiển truy cập
  - Phân quyền cho người dùng để dữ liệu được truy nhập hợp pháp
- Sao lưu và phục hồi dữ liệu
  - Chính sách sao lưu dữ liệu (đề phòng sự cố)
  - Có khả năng khôi phục trạng thái DL trc sự cố

Nội dung chính

1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

Khái niệm

- Là các hệ thống xử lý mang đặc tính thông minh, gần với trí tuệ con người
- Xây dựng mô phỏng cơ chế hoạt động và suy nghĩ của con người

Ví dụ

- Hệ chuyên gia
  - Dựa trên tri thức được trang bị hoặc tự hoc và trên hệ luật để trả về kết quả theo yêu cầu của con người
- Hệ thống đa tác tử thông minh
  - Các tác tử có khả năng cảm nhận, thích nghi với môi trường, có khả năng liên kết, phối hợp để hoàn thành nhiệm vụ
- Ngôi nhà thông minh
  - Các thiết bị thân thiện và hiểu được mong muốn của con người

Nội dung chính

1. Hệ thống thông tin quản lý
2. Soạn thảo văn bản
3. Trình chiếu văn bản
4. Hệ thông tin bảng tính
5. Hệ quản trị cơ sở dữ liệu
6. Các hệ thống thông minh
7. Hệ thống thương mại điện tử

Khái niệm

- Là hoạt động thương mại bằng các phương tiện điện tử
  - Hoạt động thương mại truyền thống
  - Các hoạt động nhanh hơn, hiệu quả hơn
- Lợi ích
  - Tiết kiệm chi phí, tạo điều kiện thuận lợi cho các bên tham gia giao dịch
  - Tiến hành giao dịch không bị hạn chế về khoảng cách địa lý

Các loại hình ứng dụng

- Doanh nghiệp - doanh nghiệp
  - **B2B:** Business to Business
- Doanh nghiệp - khách hàng
  - **B2C:** Business to consumer
- Doanh nghiệp - cơ quan nhà nước
  - **B2G:** Business to Government
- Cá nhân – Cá nhân
  - **C2C:** Consumer to Consummer
- Cơ quan nhà nước – Cá nhân
  - **G2C:** Government to Consumer

TIN HỌC ĐẠI CƯƠNG

Phần 2: GIẢI QUYẾT BÀI TOÁN

Nội dung chính

1. Chương 1: Giải quyết bài toán
   - Khái niệm về bài toán
   - Quá trình giải quyết bài toán bằng máy tính
   - Phương pháp giải quyết bài toán bằng MT
2. Chương 2: Thuật toán
   - Khái niệm
   - Biểu diễn thuật toán
   - Thuật toán đệ quy
   - Thuật giải heuristic
   - Một số thuật toán thông dụng

Nội dung chính

1. Khái niệm về bài toán
2. Quá trình giải quyết bài toán bằng máy tính
3. Phương pháp giải quyết bài toán bằng máy tính

Problem – Bài toán hay vấn đề?

- Theo Socrate (470-399 TCN): Vấn đề thường được dùng với ý nghĩa rộng hơn bài toán
- Bài toán là vấn đề mà để giải quyết phải liên quan ít nhiều đến tính toán
  - Bài toán trong vật lý, hóa học, xây dựng, kinh tế,…

Phân loại vấn đề (Pytago)

- **Theorema:**
  - Vấn đề cần khẳng định đúng sai
    - Ví dụ: Chứng minh các định lý trong toán học
- **Problema:**
  - Vấn đề cần tìm giải pháp để đạt mục tiêu xác định từ những điều kiện ban đầu
    - Ví dụ: Bài toán dựng hình, tìm đường đi ngắn nhất, tổng hợp chất hóa học…

Biểu diễn vấn đề (1/3)

$$A \rightarrow B$$

- A: Giả thiết, điều kiện ban đầu
- B: Kết luận, mục tiêu cần thực hiện
- $\rightarrow$: Suy luận, giải pháp cần xác định

Biểu diễn vấn đề (2/3)

- Cho vấn đề/bài toán:

  Cho A và B

- Giải quyết vấn đề/bài toán:

  Từ **A** dùng một số **hữu hạn các bước** suy luận có lý hoặc hành động thích hợp để đạt **B**.

  Cần xác định tập các thao tác cơ bản được dùng trong suy luận và hành động

Biểu diễn vấn đề (3/3)

Trong tin học

$$A \rightarrow B$$

- A: Input
- B: Output
- $\rightarrow$: Chương trình cho phép biến đổi A thành B .

Chương trình

- Chương trình
  - Cách mã hóa lại thuật toán/thuật giải để giải quyết vấn đề/bài toán đã cho
  - Tạo thành từ các lệnh cơ bản của máy tính
- Khó khăn:
  - Tồn tại các yếu tố không xác định
    - A và B không đầy đủ, rõ ràng
- Giải quyết bài toán trên máy tính?
  - Vấn đề tổ chức dữ liệu và thiết kế giải thuật

Cấu trúc dữ liệu + Giải thuật = Chương trình

Thiết kế thuật giải

- **Thực hiện bởi con người**
  - Là cách thức chủ yếu, dựa trên
    - Những thông tin được phản ánh rõ ràng trong A, B hoặc →
    - Các tri thức của con người
- **Tự động hóa xây dựng thuật giải**
  - Lĩnh vực mới, đang được nghiên cứu
  - Cần phải biểu diễn nội dung và các tri thức liên quan dưới dạng tương minh và đầy đủ

Nội dung chính

1. Khái niệm về bài toán

2. Quá trình giải quyết bài toán bằng máy tính

3. Phương pháp giải quyết bài toán bằng máy tính

Máy tính & Lập trình viên

- Máy tính
  - Chỉ làm được những gì được bảo.
  - **Không thông minh:** không thể tự phân tích vấn đề và đưa ra giải pháp.
  - Không thể dùng giải quyết các vấn đề liên quan đến hành động vật lý hoặc biểu thị cảm xúc

- Lập trình viên
  - Phân tích vấn đề
  - Tạo ra các chỉ dẫn để giải quyết vấn đề (xây dựng chương trình).
    - Máy tính sẽ thực hiện các chỉ dẫn này.

Các bước giải quyết bài toán

1. Xác định bài toán
2. Lựa chọn phương pháp giải
3. Xây dựng thuật toán hoặc thuật giải
4. Cài đặt chương trình
5. Hiệu chỉnh chương trình
6. Thực hiện chương trình

Bước 1: Xác định bài toán

- Mô tả bài toán cần giải quyết
  - **Dữ liệu vào:** Danh sách các dữ kiện vào
  - **Điều kiện vào:** Ràng buộc, quan hệ giữa chúng
  - **Dữ liệu ra**: Danh sách các dữ liệu ra
  - **Điều kiện ra:** Ràng buộc, quan hệ giữa chúng
- Đánh giá, nhận định tính khả thi của bài toán
  - Thời gian, kinh phí, nguồn lực,…

Ví dụ: Bài toán tìm ƯSCLN của 2 số nguyên dương

- **Nhập:** 2 số X, Y
- **Điều kiện nhập:** X, Y nguyên dương
- **Dữ liệu ra**: Z
- **Điều kiện ra:** Z là ƯSCLN(X,Y)

Bước 2: Lựa chọn phương pháp giải

- Tồn tại nhiều phương pháp khác nhau
  - Khác nhau về thời gian thực hiện, chi phí lưu trữ dữ liệu, độ chính xác…
- Tùy theo nhu cầu cụ thể và khả năng xử lý tự động được sử dụng để lựa chọn phương pháp thích hợp

Ví dụ: Bài toán sắp xếp dãy số

- Nổi bọt, Vun đống, Sắp xếp nhanh,…

Bước 3: Xây dựng thuật giải

- Xây dựng mô hình chặt chẽ, chính xác và chi tiết cho phương pháp đã lựa chọn
- Lặp liên tiếp các bước sau để thuật toán ngày càng hoàn chỉnh hơn (*quá trình tinh chỉnh từng bước*)
  1. Xác định và chính xác hóa các thao tác
     - Để đạt được kết quả cần làm gì?
  2. Xác định các dữ liệu cần dùng và tính chất của chúng
     - Để thực hiện, thao tác cần gì và sẽ tạo ra gì?
  3. Xác định trình tự các thao tác
     - Thao tác nào cần làm trước
     - Thao tác thực hiện 1 hay nhiều lần, thực hiện trong điều kiện nào..?

Bước 3: Xây dựng thuật giải (tiếp)

- *Quá trình tinh chỉnh từng bước* dừng lại khi
  - Yêu cầu cho biết 1 hay nhiều đại lượng
  - Tính một đại lượng theo công thức đã biết rõ
  - Thông báo một hay nhiều kết quả đã xử lý
- Sau khi tinh chỉnh cần phải diễn tả giải thuật dưới dạng chuẩn
  - Ngôn ngữ liệt kê các hành động
  - Sơ đồ khối…

Bước 4: Cài đặt chương trình

Mã hóa giải thuật bằng một ngôn ngữ lập trình

- Thay thế các thao tác bằng các lệnh tương ứng của ngôn ngữ sử dụng
  - **Thao tác:** In ra một thông báo
  - **Câu lệnh:** printf("….")/ write(".....");
- Lựa chọn ngôn ngữ lập trình, tùy theo bài toán giải quyết
  - NNLT bậc thấp: Hợp ngữ
  - NNLT bậc cao: C, Pascal, Java,..

Bước 5: Hiệu chỉnh chương trình

Chạy thử để phát hiện và điều chỉnh các sai sót có thể có ở bước 4.

- Lỗi cú pháp:
  - Viết sai cú pháp của ngôn ngữ lập trình lựa chọn
- Lỗi ngữ nghĩa
  - Mã hóa sai giải thuật
  - Giải thuật sai

Bước 6: Thực hiện chương trình

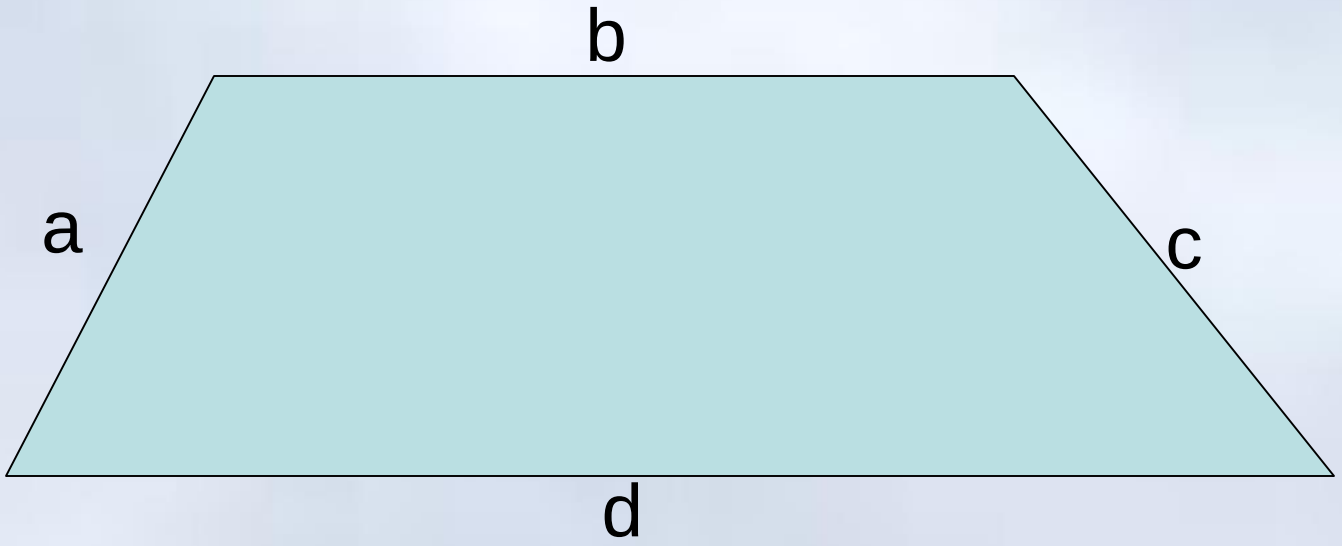
- Cho máy tính thực hiện chương trình.
- Tiến hành phân tích kết quả thu được
  - Kết quả đó có phù hợp hay không.
    - Không phù hợp kiểm tra lại toàn bộ các bước.

Các giai đoạn giải quyết bài toán

1. Giai đoạn quan niệm :
   - Gồm các bước xác định bài toán, lựa chọn mô hình, xây dựng thuật giải, cài đặt chương trình
2. Giai đoạn khai thác và bảo trì
   - Gồm các bước hiệu chỉnh và thực hiện chương trình
   - Nhằm đáp ứng nhu cầu về cải tiến, mở rộng chương trình
     - Do các yếu tố ban đầu của bài toán có thể thay đổi.

Ví dụ

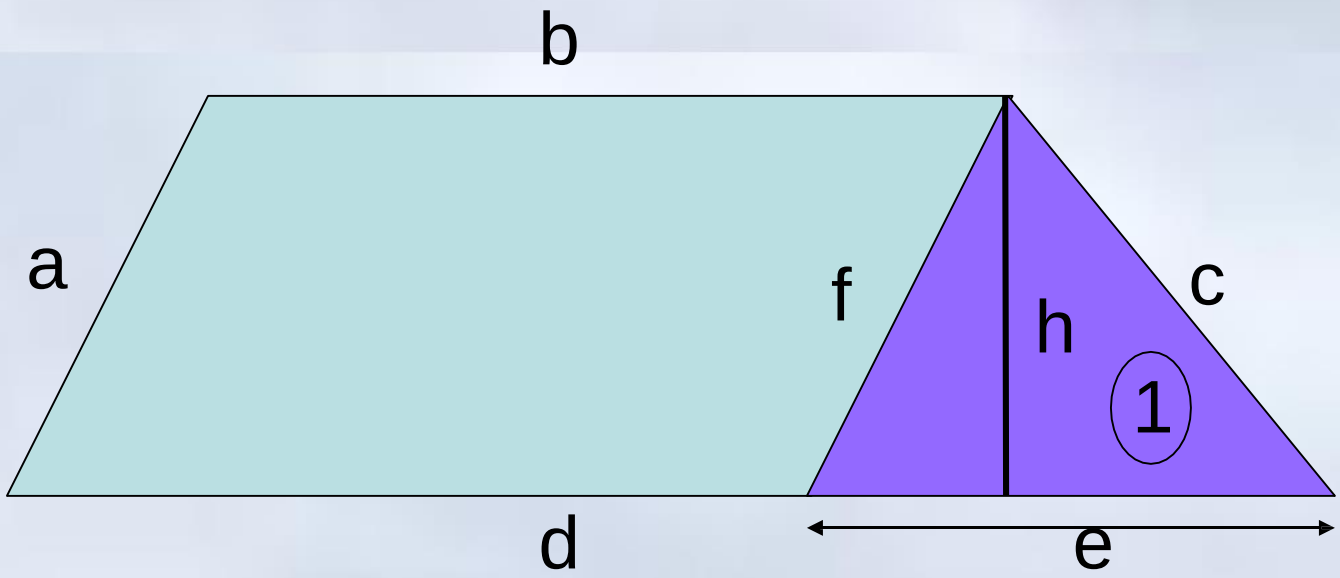
Tính diện tích hình thang khi biết 4 cạnh



Mô tả bài toán

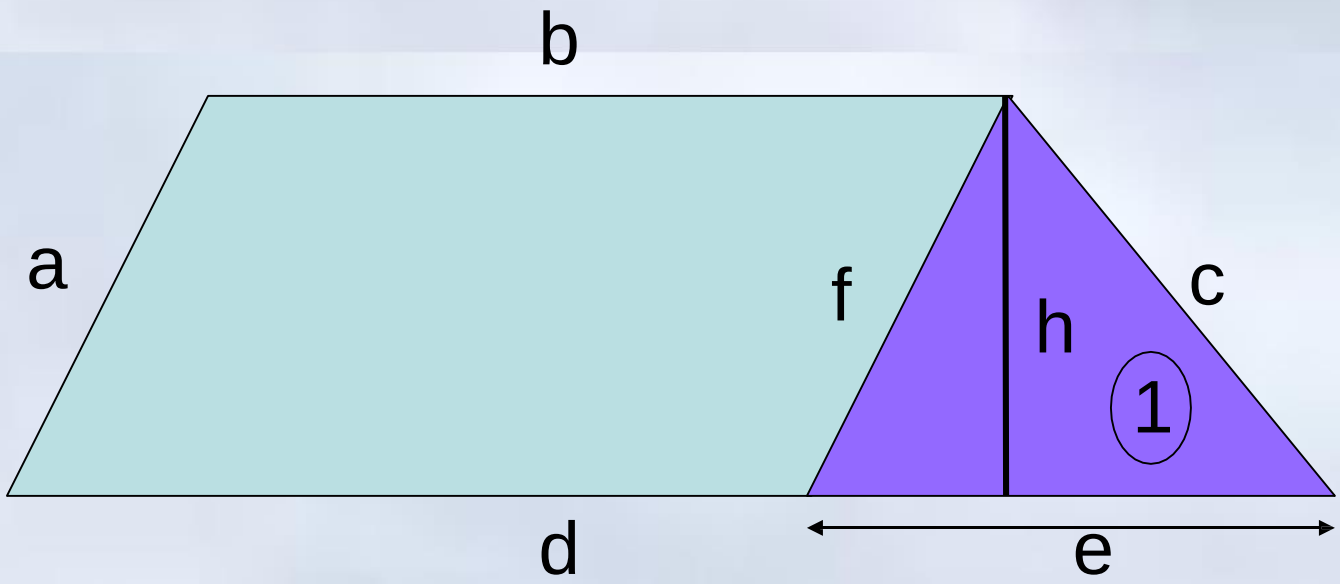
- Nhập: 4 cạnh a, b, c, d
- Điều kiện nhập: a, b, c, d > 0 và d > b
- Xuất: Một giá trị số
- Điều kiện xuất: Diện tích hình thang

Ví dụ→Xây dựng thuật toán



1. Để tính diện tích hình thang, cần tính đường cao (công thức S = h(b+d)/2)
2. Tính đường cao h, cần phải biết 3 cạnh của tam giac (1) $h_c = \frac{2\sqrt{p(p-a)(p-b)(p-c)}}{c}$
3. Cần tính cạnh tam giác (1) trước khi tính đường cao h

Ví dụ→Xây dựng thuật toán



Lặp lại các bước

- Để tính cạnh của tam giác (1) cần biết các cạnh của hình thang
- Các cạnh của hình thang là dữ kiện cho biết của đề bài (*điều kiện dừng*)

Ví dụ→ Chuẩn hóa thuật toán

1. Nhập các số a, b, c, d
2. Tính các cạnh của tam giác (1)
   - f←a
   - e←d-b
   - p ←(f+e+c)/2
3. Tính chiều cao của tam giác (1)

$$h = \frac{2\sqrt{p(p-e)(p-f)(p-c)}}{e}$$

4. Tính diện tích hình thang S= h(d+b)/2
5. In kết quả S
6. Kết thúc

Nội dung chính

1. Khái niệm về bài toán

2. Quá trình giải quyết bài toán bằng máy tính

3. Phương pháp giải quyết bài toán bằng máy tính

Các phương pháp

1. Xác định trực tiếp lời giải

2. Tìm kiếm lời giải

Hướng xác định trực tiếp lời giải

- Thường sử dụng trong quá trình học tập.
  - Ví dụ: Tìm nghiệm phương trình bậc 2 theo định lý Viet.
- Xác định trực tiếp được lời giải qua
  - Các thủ tục tính toán theo công thức, hệ thức, định luật…
  - Các thủ tục bao gồm một số hữu hạn các thao tác sơ cấp, có thể chuyền thành các thuật toán và chương trình chạy trên máy tính.

Hướng xác định trực tiếp lời giải

- Trường hợp dùng các công thức lặp để tính gần đúng nghiệm của bài toán.
  - Lời giải xác định bởi các công thức lặp có thể xấp xỉ lời giải thật sự của bài toán với độ chính xác tăng theo quá trình lặp.
    - Ví dụ: giải phương trình f(x)=0 bằng PP chia đôi
  - Đây là hạn chế khi tính toán thủ công nhưng là thế mạnh của máy tính.
  - Được xem là cách xác định trực tiếp lời giải

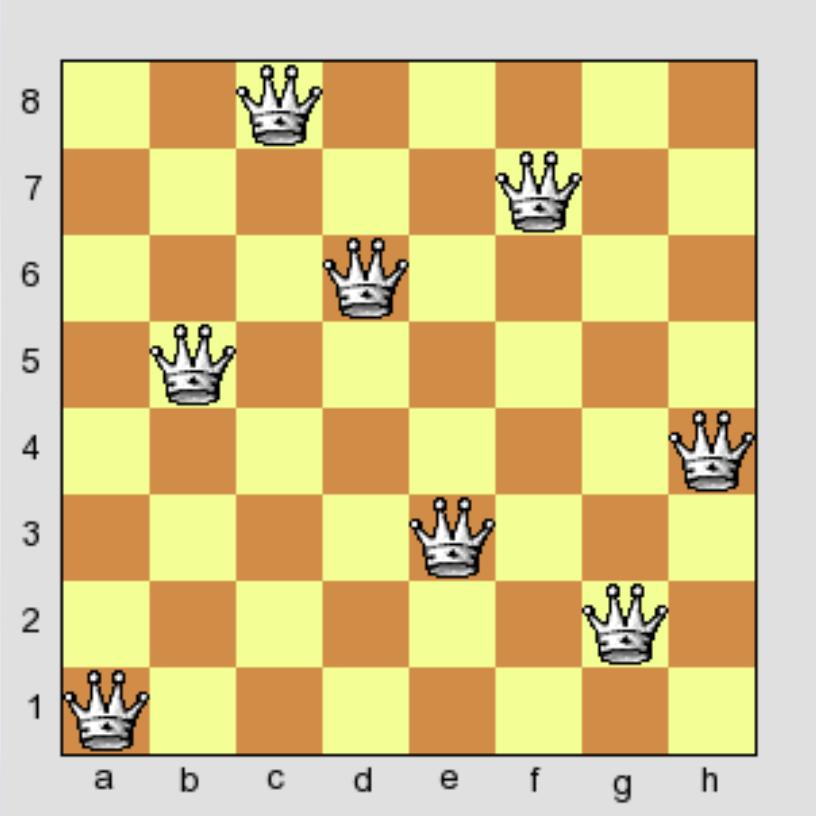
Hướng tìm kiếm lời giải

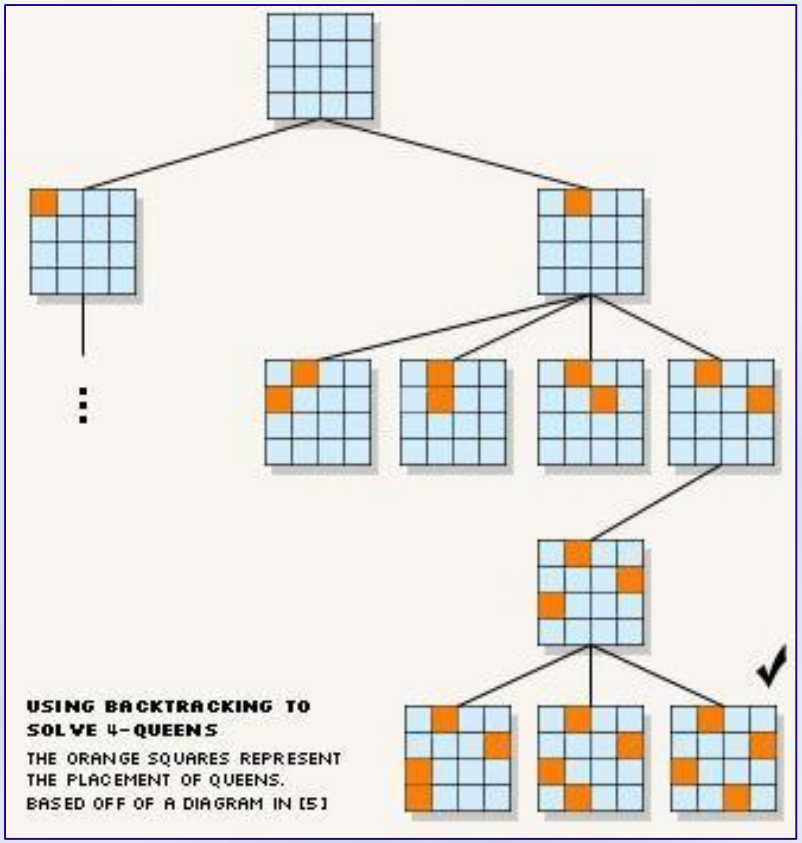
- Cách tiếp cận dựa theo nguyên lý "**thử - sai**".
- Ứng dụng hiệu quả cho một số bài toán - vấn đề phức tạp
- Nhiều phương pháp đề xuất

Hướng tìm kiếm lời giải→Một số phương pháp

- Phương pháp liệt kê hay vét cạn:
  - Xác định tập các khả năng chứa các lời giải và cách thức liệt kê của từng khả năng để thử, không bỏ sót một khả năng nào.
- Phương pháp thử ngẫu nhiên:
  - Thử một số khả năng được chọn ngẫu nhiên trong tập (*rất lớn*) các khả năng.
  - Khả năng thành công tùy theo chiến lược chọn ngẫu nhiên và một số điều kiện cụ thể của bài toán.
- Chia bài toán thành bài toán con:
  - Chia cho tới khi bài toán ban đầu được quy thành bài toán con có lời giải
- Phương pháp quay lui:
  - Đánh dấu các thử nghiệm thất bại và thử khả năng mới (quay lui tìm đường khác).

Hướng tìm kiếm lời giải→Ví dụ bài toán 8 hậu





Nội dung chính

1. Chương 1: Giải quyết bài toán
   - Khái niệm về bài toán
   - Quá trình giải quyết bài toán bằng máy tính
   - Phương pháp giải quyết bài toán bằng MT
2. Chương 2: Thuật toán
   - Khái niệm
   - Biểu diễn thuật toán
   - Thuật toán đệ quy
   - Thuật giải heuristic
   - Một số thuật toán thông dụng

Bạch Tuyết đẹp hơn

Sai

Thỏa mãn

Ngừng

Đúng

Tìm cách hại

Đến nhà 7 chú lùn

Lừa Bạch Tuyết

Về lâu đài

Nội dung chính

1. Khái niệm
2. Biểu diễn thuật toán
3. Thuật toán đệ quy
4. Thuật giải heuritic
5. Một số thuật toán thông dụng

Khái niệm

- Thuật toán (*algorithm*) là khái niệm cơ sở của Toán học và Tin học
- Nghiên cứu thuật toán đóng vai trò quan trọng trong khoa học máy tính
  - Máy tính chỉ có khả năng thực hiện công việc theo một thuật toán.
  - Thuật toán chỉ đạo máy tính từng bước phải làm gì.

Thuật toán là gì?

Khái niệm

- Một tập các lệnh hay chỉ thị nhằm hướng dẫn việc thực hiện một công việc nào đó
- Bao gồm một dãy hữu hạn các chỉ thị rõ ràng và có thể thi hành được, được bố trí theo một trình tự nhất định, cần thực hiện trên những dữ liệu vào sao cho sau một số hữu hạn bước ta thu được kết quả của bài toán cho trước
- Thuật toán là sự thể hiện của một phương pháp để giải quyết một vấn đề

Ví dụ

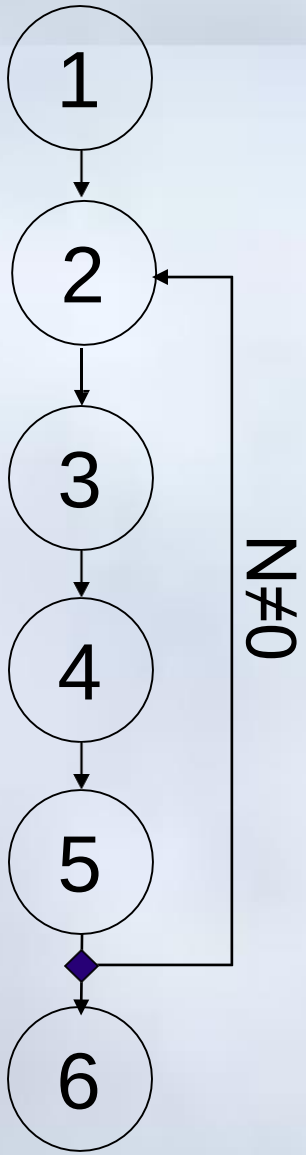
Tìm phần tử lớn nhất trong một dãy hữu hạn các số nguyên

1. Đặt giá trị lớn nhất tạm thời (Max) bằng số nguyên đầu tiên của dãy
   Max là giá trị lớn nhất ở mỗi giai đoạn thực hiện
2. Nếu tất cả số nguyên nào trong dãy đã được xét, thực hiện bước 5
3. So sánh số nguyên kế tiếp trong dãy với Max
   - Nếu lớn hơn Max thì thay Max bằng số nguyên này.
4. Lặp lại bước 2
5. Thông báo: Max là giá trị lớn nhất trong dãy số.

Ví dụ

Đổi số thập phân sang dạng nhị phân

1. Cho biết N
2. Chia N cho 2
3. Ghép phần dư vào bên trái kết quả
4. Lấy phần thương làm N mới
5. Nếu N khác 0, lặp lại Bước 2
6. Xong



Định nghĩa (KHMT)

Thuật toán để giải một bài toán là một dãy hữu hạn các thao tác và trình tự thực hiện các thao tác đó sao cho sau khi thực hiện dãy thao tác này theo trình tự đã chỉ ra, với đầu vào (input) ta thu được kết quả đầu ra (output) mong muốn

Thao tác/lệnh

- Là hành động cần được thực hiện bởi cơ chế của thuật toán
- Các thao tác (*lệnh*) sẽ biến đổi bài toán từ trạng thái trước tới trạng thái sau
- Dãy các thao tác cần thiết sẽ biến đổi bài toán từ trạng thái ban đầu đến kết quả
- Các thao tác có thể phân tích thành thao tác khác nhỏ hơn
- Thứ tự thao tác là quan trọng
  - Cùng tập thao tác, thứ tự khác nhau dẫn đến kết quả khác nhau
- Cơ cấu thể hiện trình tự thực hiện các thao tác gọi là **Cấu trúc điều khiển**
  - Có 3 loại cơ bản: Tuần tự, Lặp, Rẽ nhánh

Các đặc trưng của thuật toán

Khi mô tả thuật toán, cần chú ý các đặc trưng

- Nhập (input)
- Xuất (output)
- Tính xác định (definiteness)
- Tính hữu hạn (finiteness)
- Tính hiệu quả
- Tính tổng quát

Nhập/Xuất

- **Nhập (input):**
  - Các giá trị "*đầu vào*" (input values) từ một tập hợp nhất định nào đó.
- **Xuất (output):**
  - Những giá trị trả về (output values) thuộc một tập hợp nhất định nào đó thể hiện lời giải cho bài toán/vấn đề
  - Tương ứng với tập hợp các giá trị nhập

Tính xác định (definiteness)

- Các bước trong thuật toán phải chính xác rõ ràng, không gây sự nhập nhằng nhầm lẫn
- Cùng một điều kiện nhập, cùng một giải thuật thì 2 bộ VXL (người, máy) phải cho ra cùng một kết quả

Tính hữu hạn (finiteness)

- Trong mọi trường hợp của dữ liệu vào, thuật toán phải cho ra hay kết quả sau một thời gian hữu hạn
  - Thời gian có thể phụ thuộc vào từng bài toán cụ thể hoặc phụ thuộc vào các thuật toán khác nhau cho một bài toán

Tính hiệu quả

- Thực hiện thuật toán cần
  - Thời gian
  - Các công cụ hỗ trợ (giấy, bộ nhớ,..)
    - Để ghi kết quả trung gian
- Độ phức tạp thuật toán: Thời gian và các công cụ hỗ trợ
  - Thuật toán càng hiệu quả độ phức tạp càng bé
  - Trong máy tính, thường quan tâm tới
    - Thời gian thực hiện
      - Số thao tác cơ bản cần thực hiện
    - Độ lớn của bộ nhớ mà thuật toán sử dụng

Tính tổng quát

Thuật toán có tính tổng quát cao nếu có thể giải bất kỳ bài toán nào trong một lớp lớn các bài toán

Ví dụ

Thuật toán giải phương trình $ax^2+bx+c=0$ phổ dụng hơn thuật toán giải phương trình $x^2+5x+6=0$

Nội dung chính

1. Khái niệm
2. Biểu diễn thuật toán
3. Thuật toán đệ quy
4. Thuật giải heuristic
5. Một số thuật toán thông dụng

Đặt vấn đề

- Tại sao:
  - Truyền đạt thuật toán cho người khác
  - "*Truyền đạt*" thuật toán cho máy tính
    - Chuyển thành chương trình điều khiển
- Phương pháp:
  1. Ngôn ngữ tự nhiên
  2. Ngôn ngữ lưu đồ(sơ đồ khối)
  3. Ngôn ngữ tựa ngôn ngữ lập trình (mã giả)
  4. Ngôn ngữ lập trình

1. Ngôn ngữ tự nhiên

- Nguyên tắc:
  - Sử dụng ngôn ngữ tự nhiên để liệt kê các bước của thuật toán
- Đặc điểm
  - Không yêu cầu phải có một số kiến thức đặc biệt
  - Dài dòng
  - Không làm nổi bật cấu trúc của thuật toán

1. Ngôn ngữ tự nhiên→Ví dụ 1

Giải phương trình **ax+ b = 0**

- B1: Nhập a
- B2: Nhập b.
- B3: Nếu a =0 thực hiện B6
- B4: **Thông báo:** Nghiệm –b/a
- B5: Thực hiện B10
- B6: Nếu b = 0, thực hiện B9
- B7: **Thông báo:** Phương trình vô nghiệm.
- B8: Thực hiện B10
- B9: **Thông báo:** Phương trình vô số nghiệm.
- B10: Kết thúc

1. Ngôn ngữ tự nhiên→Ví dụ 2

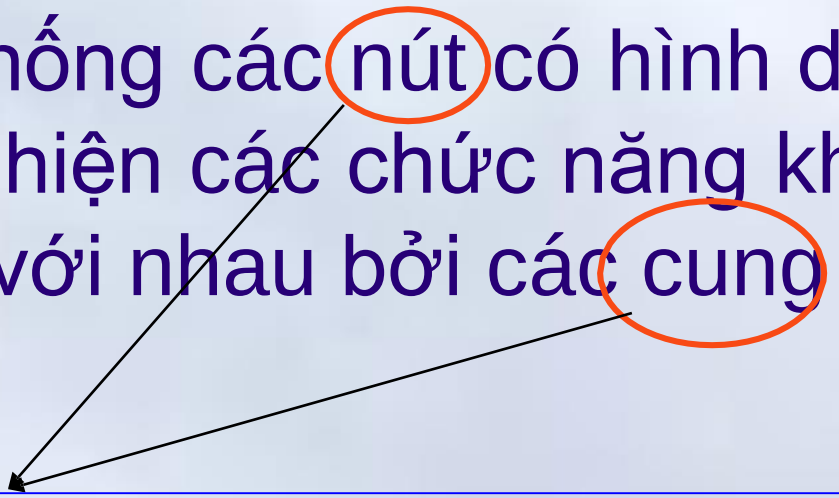
Tìm giá trị lớn nhất của một dãy N số nguyên

- B1:Nhập N
- B2: Nhập dãy số a_i gồm N số.
- B3:Gán giá trị a_1 cho Max, 2 cho biến i (i←2)
- B4:Nếu i > N, thực hiện bước 8
- B5:Nếu a_i > Max, gán giá trị a_i cho Max.
- B6:Tăng i lên 1 đơn vị.
- B7:Quay lên B4.
- B8:Thông báo: Max là giá trị lớn nhất dãy
- B9:Kết thúc.

1. Ngôn ngữ lưu đồ (sơ đồ khối)

Công cụ diễn đạt các thuật toán trực quan

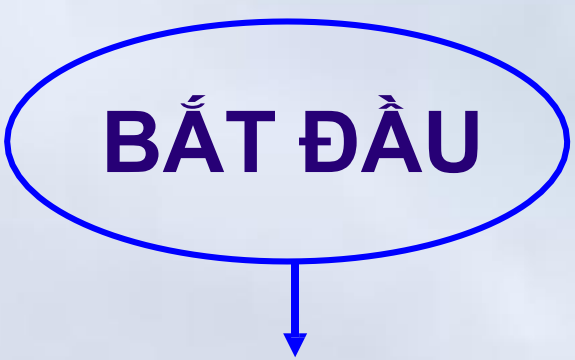
- Đưa ra một cái nhìn tổng quan về quá trình xử lý theo thuật toán
- Gồm hệ thống các nút có hình dạng khác nhau, thể hiện các chức năng khác nhau, được nối với nhau bởi các cung



Thành phần chủ yếu của thuật toán

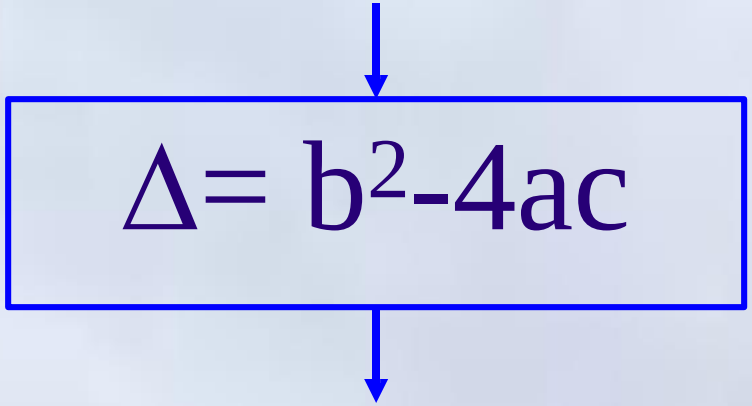
2. Sơ đồ khối → Nút /khối giới hạn

- 2 loại nút giới hạn: nút đầu và nút cuối
- Ghi rõ điểm bắt đầu và kết thúc (dừng) của thuật toán
- Được biểu diễn bởi hình ôvan có ghi chữ bên trong



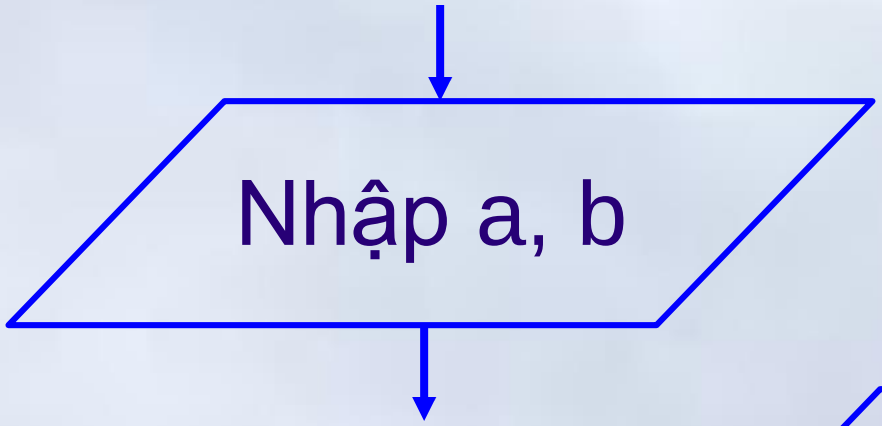
2. Sơ đồ khối → Nút/Khối thao tác

Là một hình chữ nhật chứa dãy các lệnh cần thực hiện như gán, tính toán…



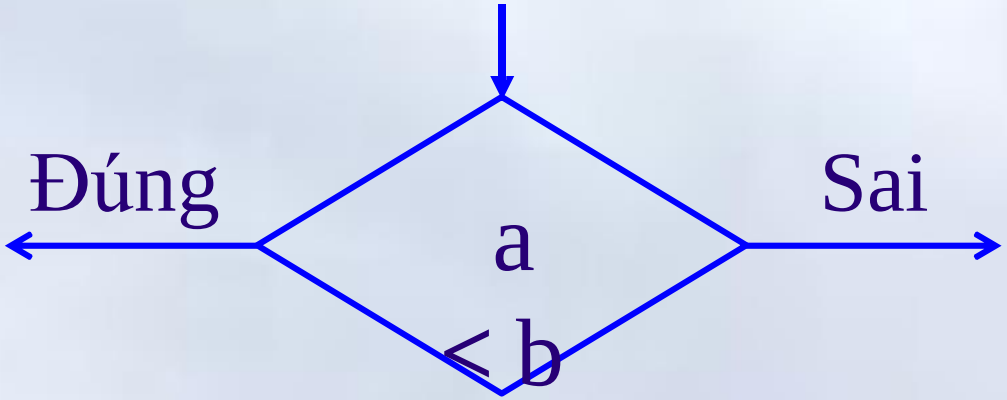
2. Sơ đồ khối → Nút/khối vào/ra dữ liệu

Là một hình bình hành chứa đựng một thao tác nhập/ xuất dữ liệu



2. Sơ đồ khối → Nút/khối điều kiện

Là một hình thoi chứa một điều kiện/biểu thức logic cần kiểm tra.

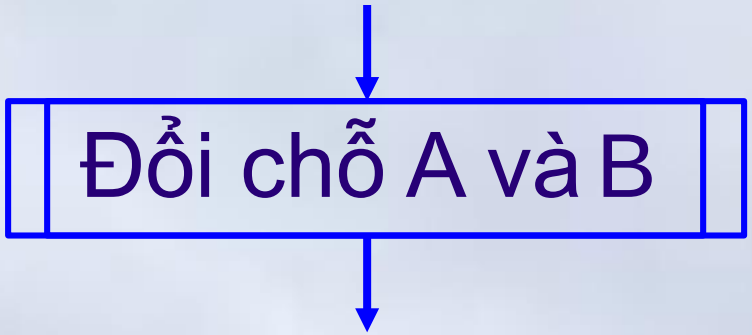


Nút điều kiện có 2 cung ra chỉ hướng ứng với 2 trường hợp: điều kiện đúng và điều kiện sai

2. Sơ đồ khối → Nút/khối gọi chương trình con

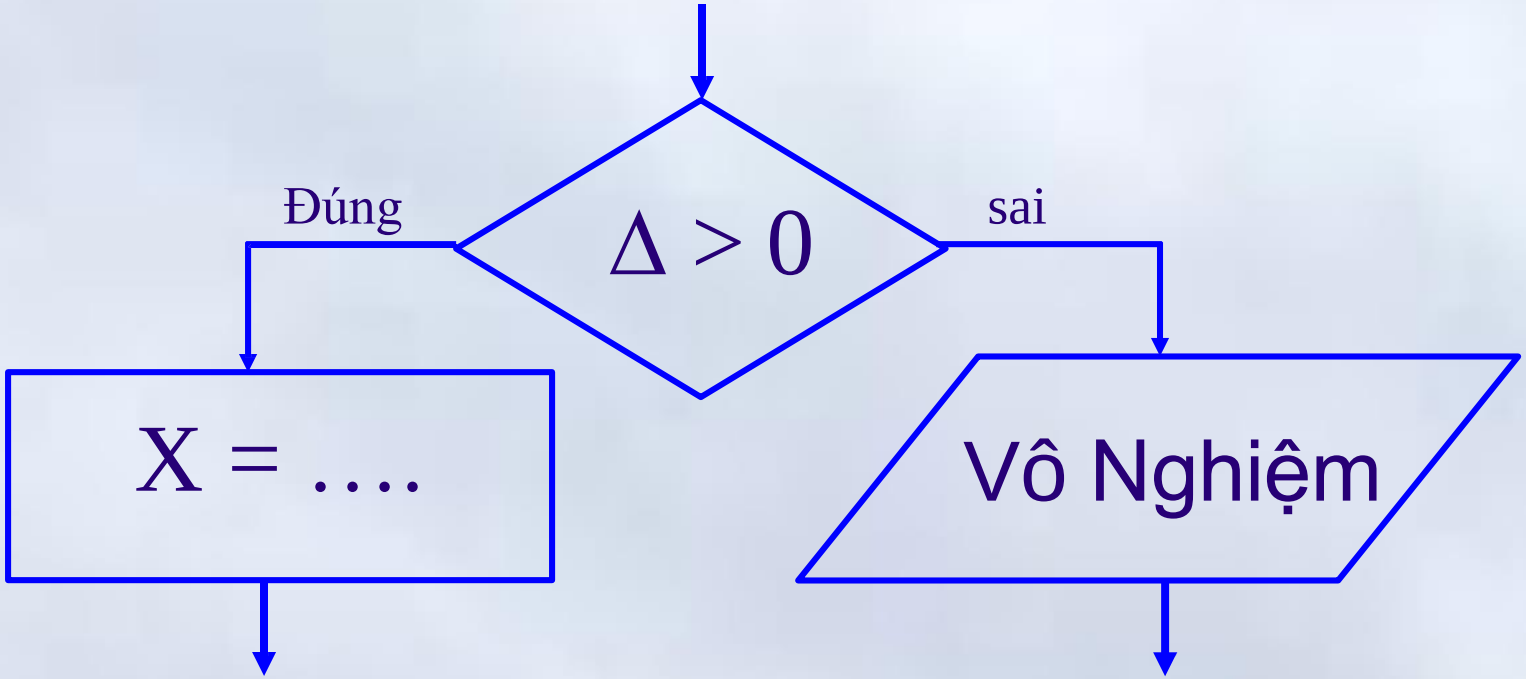
Là một hình chữ nhật, cạnh kép chứa tên một chương trình con cần thực hiện

- Chương trình con: Thuật toán đã biết
- Nhằm cho sơ đồ đỡ rắc rối



2. Sơ đồ khối → Cung

Là các đường nối từ nút này đến nút khác của lưu đồ



2. Sơ đồ khối → Hoạt động

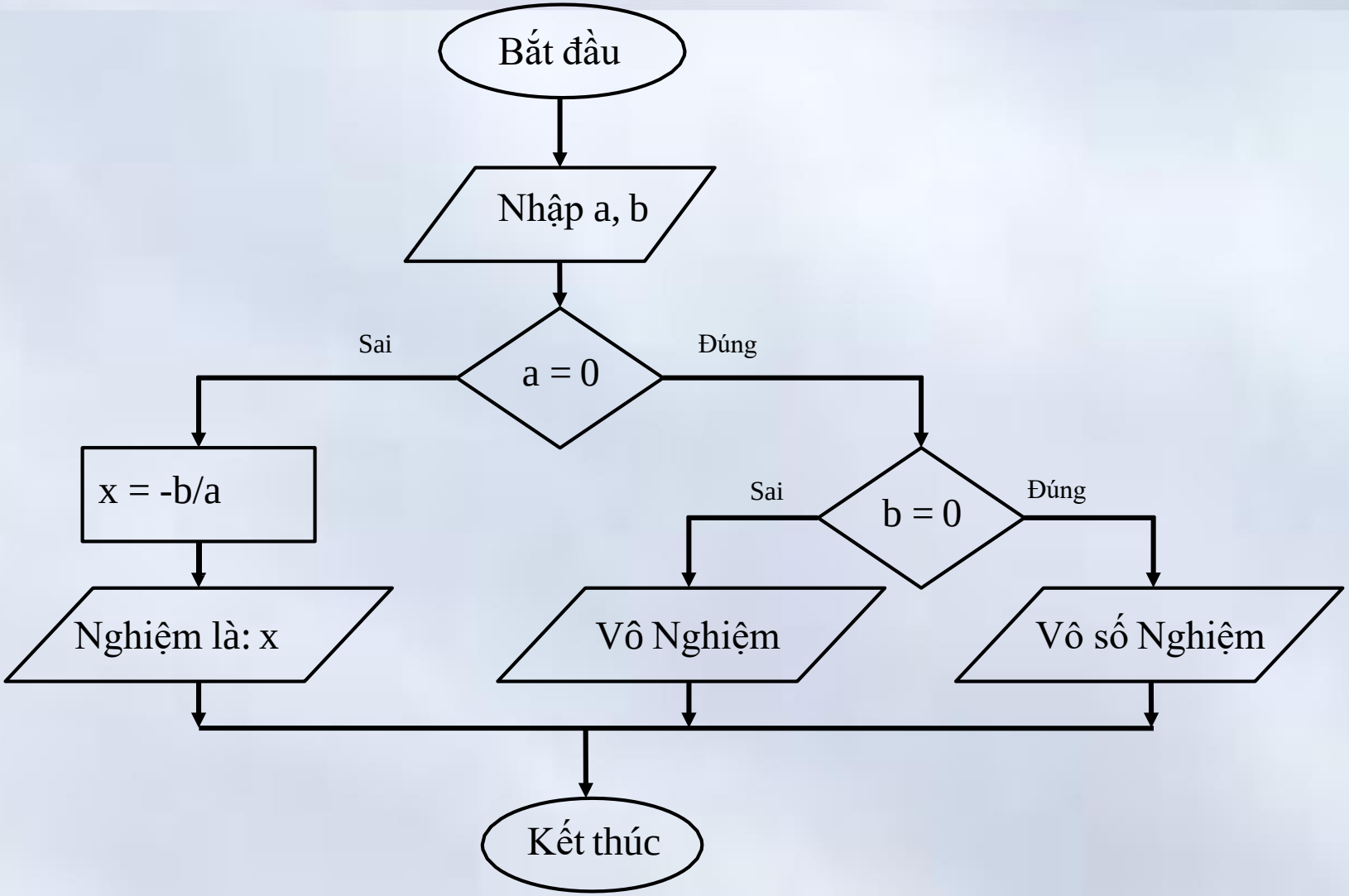
- Bắt đầu từ nút giới hạn đầu tiên (nút bắt đầu).
- Thực hiện các thao tác được ghi trong nút
- Theo một cung đi tới nút khác
- Nếu là nút điều kiện: đi theo cung tương ứng với trạng thái của điều kiện được kiểm tra.
- Thuật toán sẽ dừng khi gặp nút kết thúc

2. Sơ đồ khối → Ví dụ 1 → Biểu diễn bằng lời

Giải phương trình **ax+ b = 0**

- B1: Nhập a
- B2: Nhập b.
- B3: Nếu a =0 thực hiện B6
- B4: **Thông báo:** Nghiệm –b/a
- B5: Thực hiện B10
- B6: Nếu b = 0, thực hiện B9
- B7: **Thông báo:** Phương trình vô nghiệm.
- B8: Thực hiện B10
- B9: **Thông báo:** Phương trình vô số nghiệm.
- B10: Kết thúc

2. Sơ đồ khối → Ví dụ 1 → Biểu diễn sơ đồ khối

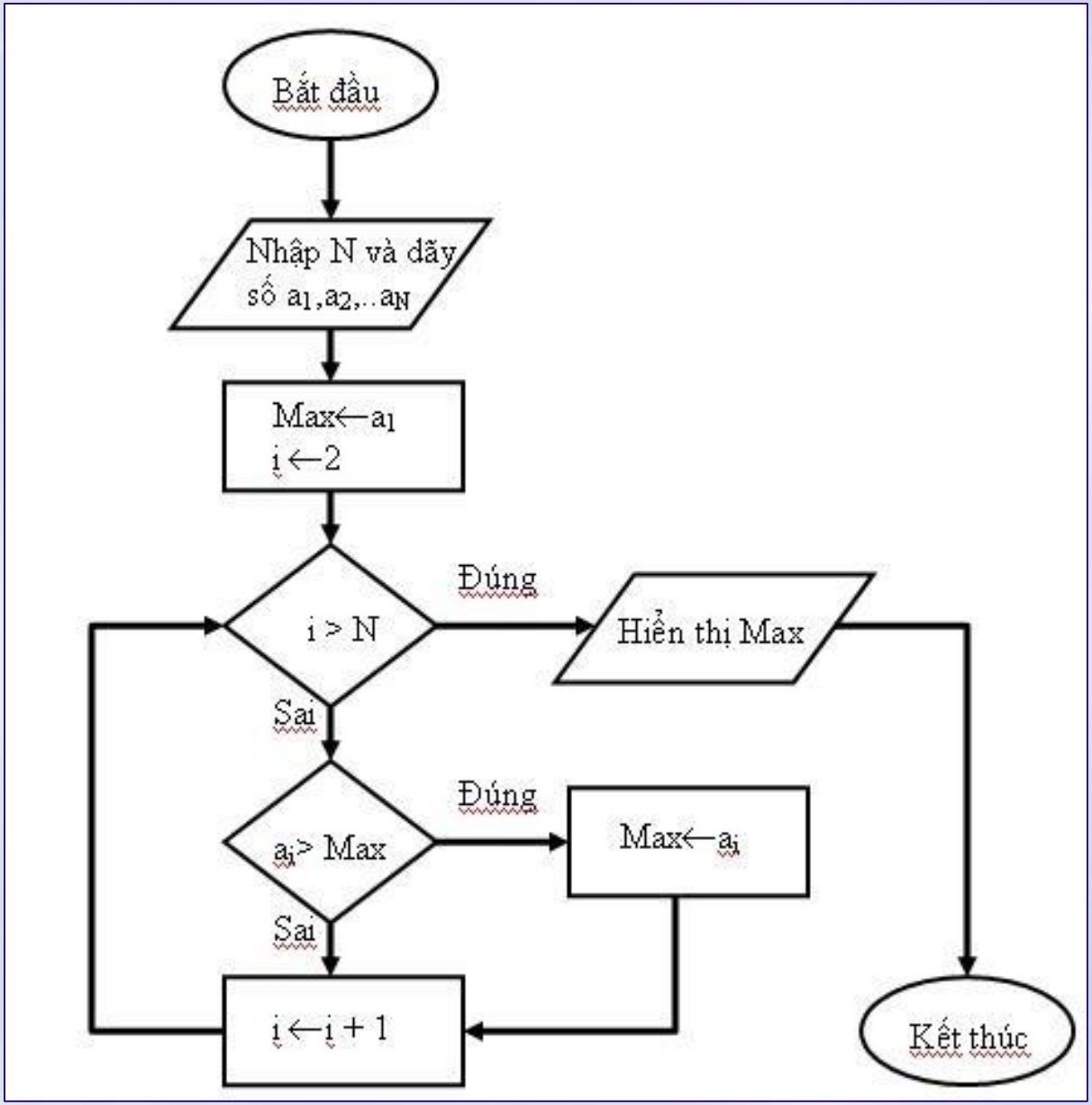


2. Sơ đồ khối → Ví dụ 2 → Biểu diễn bằng lời

Tìm giá trị lớn nhất của một dãy N số nguyên

- B1:Nhập N
- B2: Nhập dãy số a_i gồm N số.
- B3:Gán giá trị a_1 cho Max, i←2.
- B4:Nếu i > N, thực hiện bước 8
- B5:Nếu a_i > Max, gán giá trị a_i cho Max.
- B6:Tăng i lên 1 đơn vị.
- B7:Quay lên B4.
- B8:Thông báo: Max là giá trị lớn nhất dãy
- B9:Kết thúc.

2. Sơ đồ khối → Ví dụ 2 → Biểu diễn sơ đồ khối



3. Ngôn ngữ tựa ngôn ngữ lập trình (mã giả)

- Mô tả thuật toán theo ngôn ngữ tựa ngôn ngữ lập trình
  - Sử dụng các mệnh đề có cấu trúc chuẩn hóa
  - Vẫn dùng ngôn ngữ tự nhiên.
    - Có thể sử dụng các ký hiệu toán học
    - Có thể sử dụng cấu trúc kiểu thủ tục để trình bày thuật toán đệ quy/thuật toán phức tạp..
- Đặc điểm:
  - Tiện lợi, đơn giản, và dễ hiểu.
- Các cấu trúc thường gặp:
  - Gán, lựa chọn, lặp, nhảy, gọi hàm

3. Mã giả → Phát biểu gán

- Mục đích:
  - Đặt giá trị cho một biến
- Biểu diễn:

$Max := a_1$

$Max \leftarrow a_1$

$n \leftarrow n + 1$

3. Mã giả → Lựa chọn/rẽ nhánh

if <điều kiện> **then** <hành động>

endif

Hoặc là

if <điều kiện> **then** <hành động>

else <hành động>

endif

3. Mã giả → Cấu trúc lặp

while <điều kiện> **do**

<hành động>

end while

repeat

<hành động>

until <điều kiện>

for biến←giá trị đầu **to** giá trị cuối **do** <hành động>

end for

for biến ←giá trị đầu **downto** giá trị cuối **do** <h/động>

end for

3. Mã giả → Lệnh nhảy không điều kiện

Nhảy đến vị trí có nhãn là L

goto L

3. Mã giả → Hàm/Thủ tục

- **Khai báo hàm**

  Function <Tên hàm>(<Các tham số>)

  Hành động với các tham số

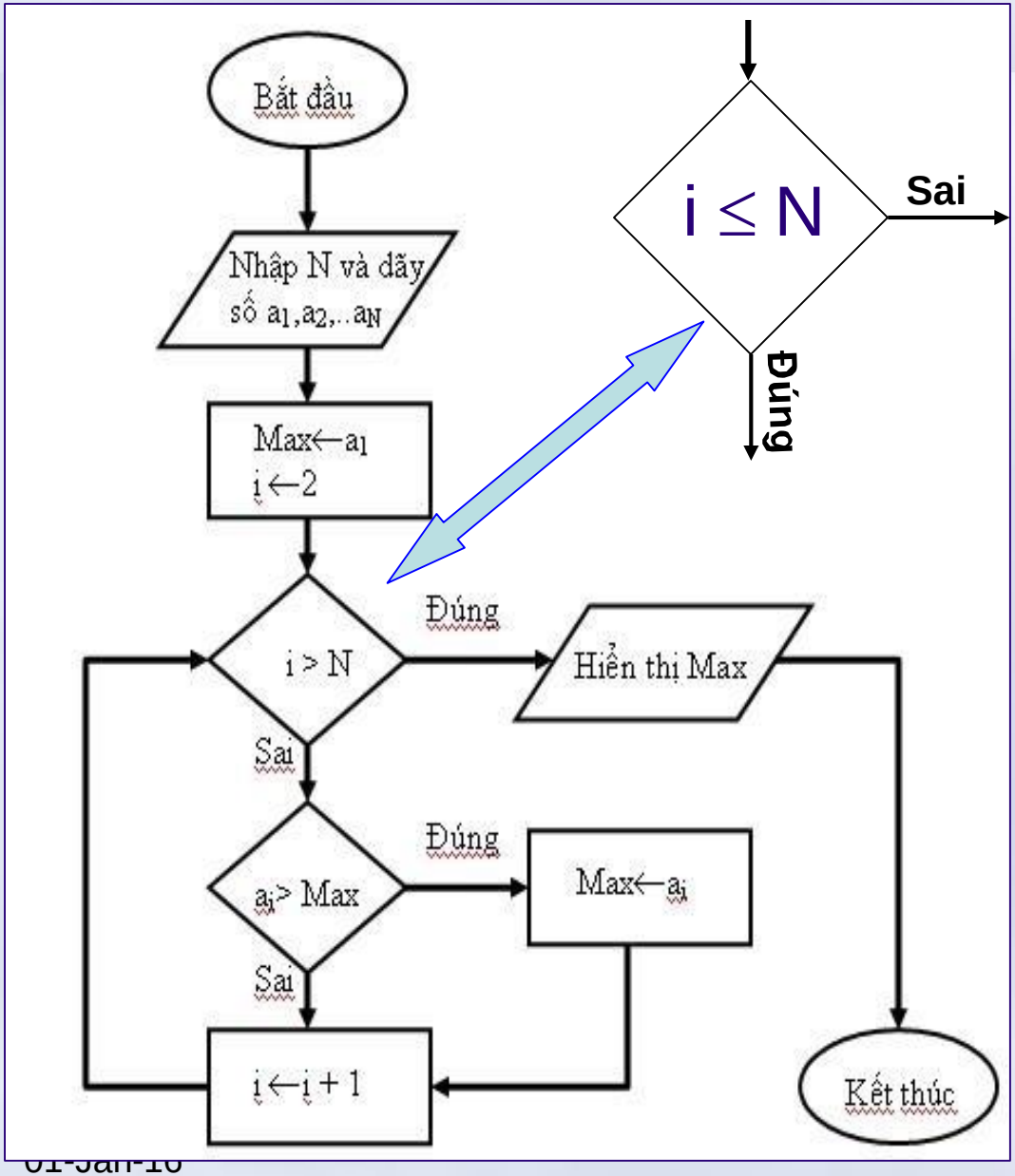
  return <Giá trị>

  End Function

- **Gọi hàm**

  [**Call**] <Tên hàm>**(**<Các tham số>**)**

3. Mã giả → Ví dụ: tìm số lớn nhất của dãy



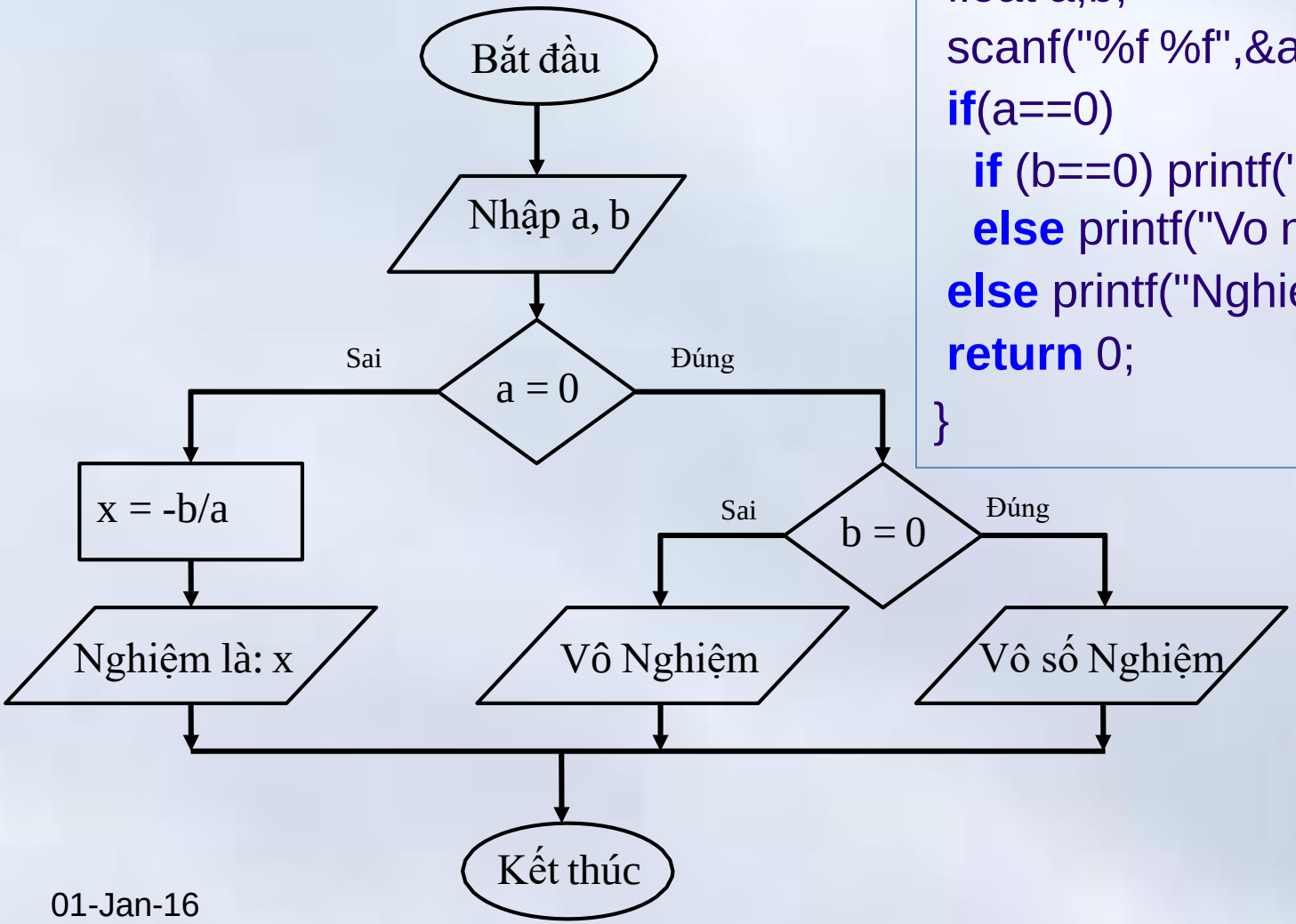
1. **Begin**
2. **Input** N
3. **Input** a_1, a_2,.. a_N
4. Max ←a_1
5. i ← 2
6. **While** i ≤ N **do**
7. **If** a_i > Max **Then**
8. Max ← a_i
9. **End if**
10. i ←i+1
11. **End while**
12. **Output** Max
13. **End**

4. Ngôn ngữ lập trình

- Tuân theo cú pháp của ngôn ngữ lập trình
  - Cấu trúc tuần tự
  - Cấu trúc rẽ nhánh
  - Cấu trúc lặp
  - Chương trình con
- Tồn tại nhiều loại ngôn ngữ lập trình
  - Ngôn ngữ máy /Hợp ngữ
  - Ngôn ngữ bậc cao:
    - Fortran, Pascal, C/C++/C#,Java

4. Ngôn ngữ lập trình → Ví dụ

Giải phương trình ax+b=0

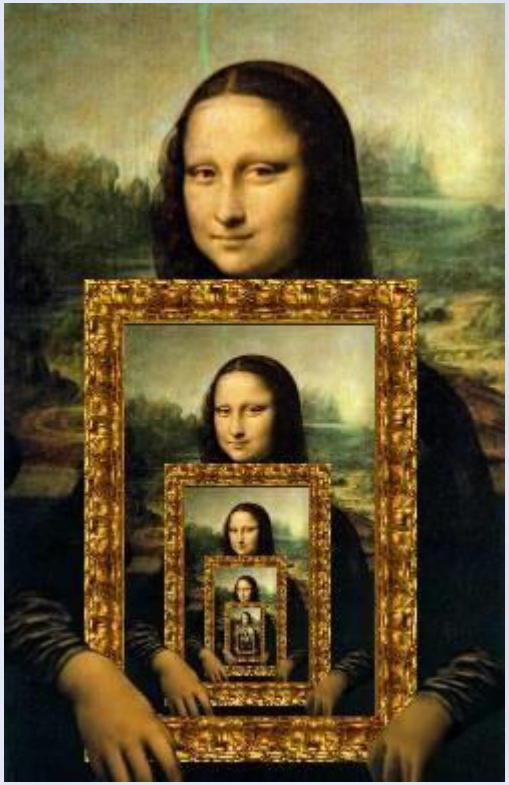


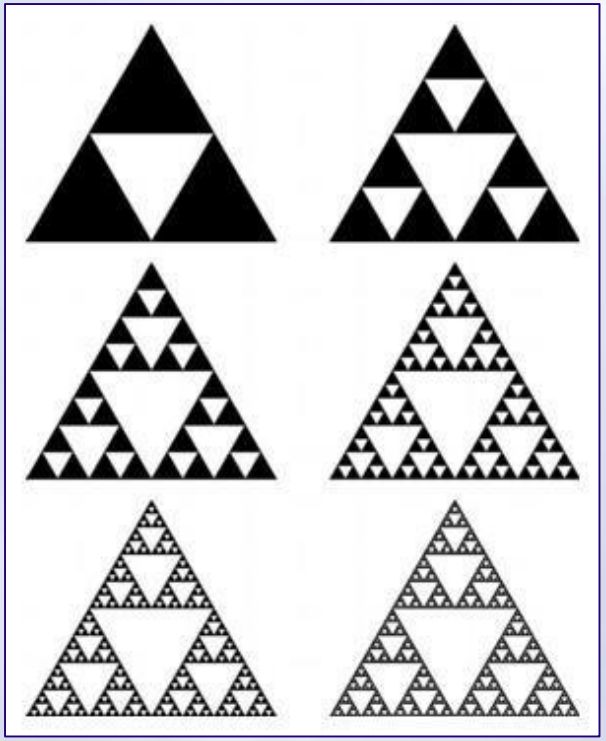

```
#include <stdio.h>
int main(){
 float a,b;
 scanf("%f %f",&a,&b);
 if(a==0)
   if (b==0) printf("Vo so nghiem");
   else printf("Vo nghiem");
 else printf("Nghiem %f",-b/a);
 return 0;
}
```

Nội dung chính

1. Khái niệm
2. Biểu diễn thuật toán
3. Thuật toán đệ quy
4. Thuật giải heuristic
5. Một số thuật toán thông dụng

Ví dụ đệ quy





Khái niệm thuật toán đệ quy

- Bài toán có thể được phân tích và đưa tới việc giải một bài toán cùng loại nhưng cấp độ thấp hơn,
  - Độ lớn dữ liệu vào nhỏ hơn
  - Giá trị cần tính toán nhỏ hơn
- **Ví dụ:** Định nghĩa giai thừa

  Giai thừa của một số tự nhiên n, ký hiệu là n!, được định nghĩa bằng cách quy nạp như sau:
  - 0!=1,
  - n!=(n-1)!*n, với mọi n>0
- Giải một bài toán có thể dựa trên chính nó
  - Mỗi bước của thuật toán hiện lại chính thuật toán đó
    - Dữ liệu vào có độ lớn thấp hơn

  Thuật toán đệ qui.

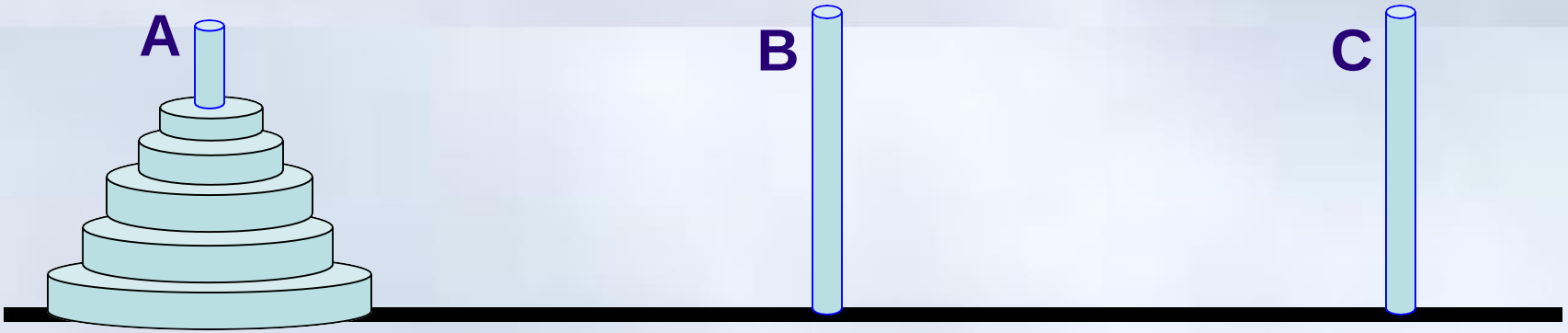
Ví dụ 1: Tính giai thừa của một số tự nhiên

- Input: số tự nhiên n
- Output: F(n)=n!

- Thuật Toán:

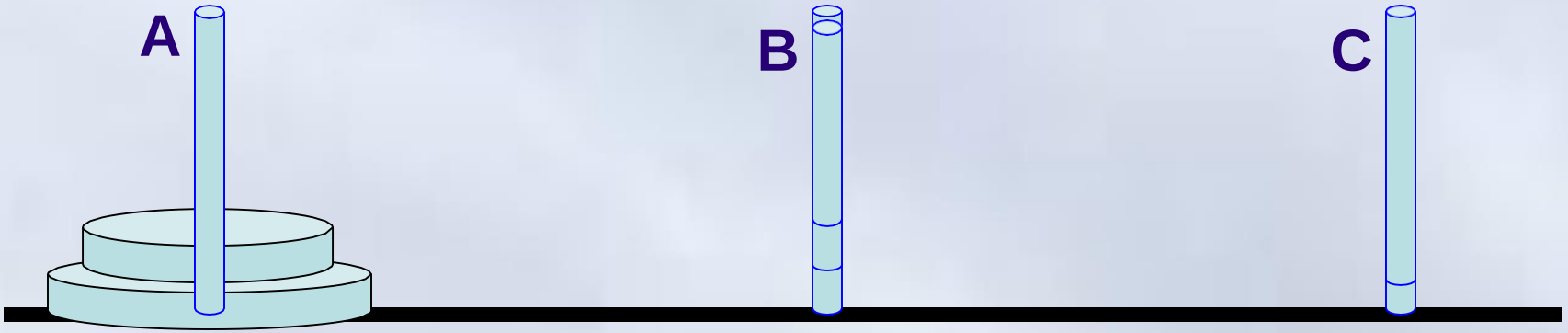
```
Function F(n);
        If n=0 then
                F = 1
        Else
                F:=n * F(n-1)
        End If
        return F
End Function
```

Ví dụ 2: Bài toán tháp Hà nội



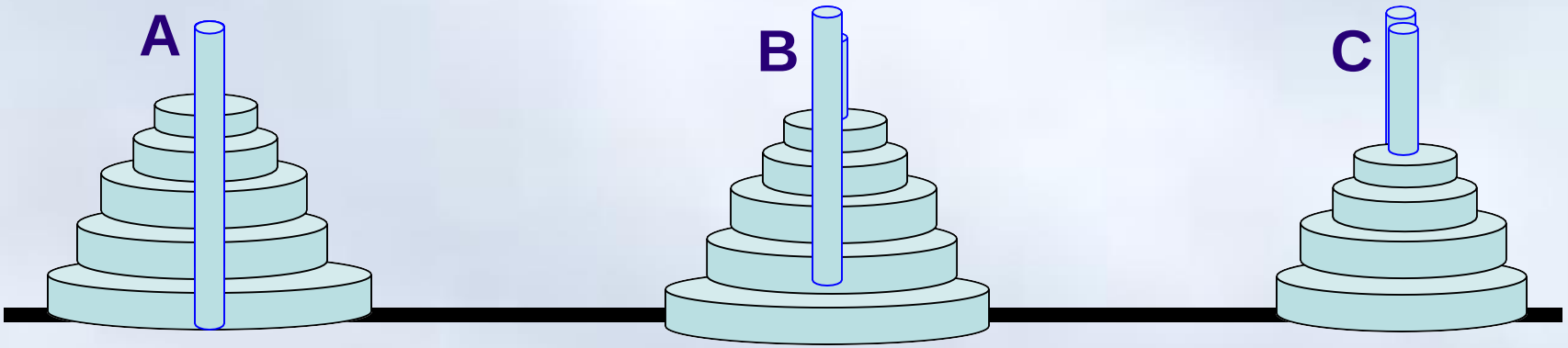
Yêu cầu: Dịch chuyển N đĩa từ cọc A sang B với trung gian là cọc C

Trường hợp với N= 2:



Ví dụ 2: Bài toán tháp Hà nội

Trường hợp tổng quat với N > 2:




```
PROCEDURE ThapHanoi (N,A,B,C)
        IF N= 2 Then Print(A→C, A→B, C→B)
        ELSE
                ThapHanoi(N-1,A,C,B)
                Print(A →B)
                ThapHaNoi(N-1,C,B,A)
        ENDIF
END
```

Lưu ý

- Thuật toán đệ qui gồm 2 phần
  - **Phần cơ sở:** không cần thực hiện lại thuật toán
    - Không có yêu cầu gọi đệ qui.
    - Là điều kiện dừng cử giả thuật đệ quy
  - **Phần đệ qui:** có yêu cầu gọi đệ qui
    - Yêu cầu thực hiện lại thuật toán
    - Đặt trong một điều kiện kiểm tra việc gọi đệ qui.
- Đệ qui dễ gây tình trạng tràn bộ nhớ (stack).
- Nếu có thể, nên viết dưới dạng lặp.

```
P←1
For k ←1 To N Do P ←P*k
Print P
```

Nội dung chính

1. Khái niệm
2. Biểu diễn thuật toán
3. Thuật toán đệ quy
4. Thuật giải heuristic
5. Một số thuật toán thông dụng

Vấn đề mở rộng khái niệm thuật toán

- Có những bài toán đến nay vẫn chưa có một cách giải theo kiểu thuật toán được tìm ra và cũng không biết có tồn tại thuật toán hay không.
- Có những bài toán đã có thuật toán để giải nhưng không chấp nhận được vì thời gian giải theo thuật toán đó quá dài hoặc các điều kiện cho thuật toán khó đáp ứng
- Có những bài toán được giải theo cách giải vi phạm thuật toán nhưng vẫn được chấp nhận.

Cần phải đổi mới cho khái niệm thuật toán

- Mở rộng những t**iêu chuẩn của thuật toán**

Mở rộng tiêu chuẩn của thuật toán

- Tính xác định (*tính đơn trị của mỗi bước*)
  - Các giải thuật đệ qui: bước tiếp gọi đến chính nó
  - Các giải thuật ngẫu nhiên: bước tiếp không xác định rõ
- Tính đúng đắn (*được hiểu cho kết quả đúng*)
  - Không còn bắt buộc với một số cách giải cho các bài toán nhất là các cách giải gần đúng.
  - Trong thực tế có nhiều trường hợp, chấp nhận các cách giải cho kết quả gần đúng nhưng ít phức tạp và hiệu quả
- Ví dụ: trong trí tuệ nhân tạo
  - Cách giải theo kiểu heuristic. Đơn giản, tự nhiên nhưng cho kết quả đúng hoặc gần đúng trong phạm vi cho phép

Thuật giải Heuristic

- **Khái niệm thuật giải:**
  - Các cách giải chấp nhận được nhưng không hoàn toàn đáp ứng đầy đủ các tiêu chuẩn của thuật toán
- **Thuật giải heuritic**
  - Thể hiện cách giải bài toán với các đặc tính sau:
    - Tìm được lời giải tốt (không chắc là tốt nhất)
    - Dễ dàng và nhanh chóng hơn so với giải thuật tối ưu
    - Thể hiện một cách hành động khá tự nhiên, gần gũi với cách suy nghĩ và hành động của con người.

Nguyên lý thiết kế thuật giải heuristic

- **Nguyên lý vét cạn thông minh**:
  - Trong một bài toán tìm kiếm, khi không gian tìm kiếm lớn, thường tìm cách để giới hạn lại không gian hoặc thực hiện một kiểu dò tìm đặc biệt dựa vào đặc thù của bài toán để nhanh chóng tìm ra mục tiêu.
- **Nguyên lý tham lam**:
  - Lấy tiêu chuẩn tối ưu (*trên phạm vi toàn bộ*) của bài toán để làm tiêu chuẩn chọn lựa hành động cho phạm vi cục bộ của từng bước (*hay từng giai đoạn*) trong quá trình tìm kiếm lời giải.
- **Nguyên lý thứ tự**:
  - Thực hiện hành động dựa trên một cấu trúc thứ tự hợp lý của không gian khảo sát nhằm nhanh chóng đạt được một lời giải tốt..

Nội dung chính

1. Khái niệm
2. Biểu diễn thuật toán
3. Thuật toán đệ quy
4. Thuật giải heuristic
5. Một số thuật toán thông dụng

Các bài toán

1. Thuật toán số học
   - Hoán đổ giá trị
   - Số nguyên tố, phân tích ra thừa số nguyên tố…
   - Tìm ước số chung, phân số tối giản
   - Số hoàn hảo
2. Thuật toán về dãy
   - Vào/ra dãy
   - Tìm Max, Min
   - Sắp xếp
   - Tìm phần tử; Đếm phần tử
   - Tính toán trên các phần tử..
     - Trung bình cộng, tính tổng,…
   - Chèn phần tử/Xóa phần tử (*liên quan tới kiểu mảng*)

Hoán đổi giá trị 2 biến X, Y

Nguyên tắc:

Dùng một biến trung gian T

Begin

1. T←X
2. X ←Y
3. Y←T

End

```
Function swap(X,Y)
        T←X
        X ←Y
        Y←T
End Function
```

Kiểm tra số nguyên tố

Begin

1. **Input** P

2. Flag ← FALSE

3. **If** P=1 **Then** **Goto** Bước 6

4. flag ← TRUE (*gán cho cờ hiệu "flag" giá trị true*)

5. **For** k:=2 **to** p-1 **do**

If (k là ước số của P) **Then**

flag ← FALSE;

Goto B6

Endif

End for

6. **If** flag=TRUE **Then Output:** P là số nguyên tố

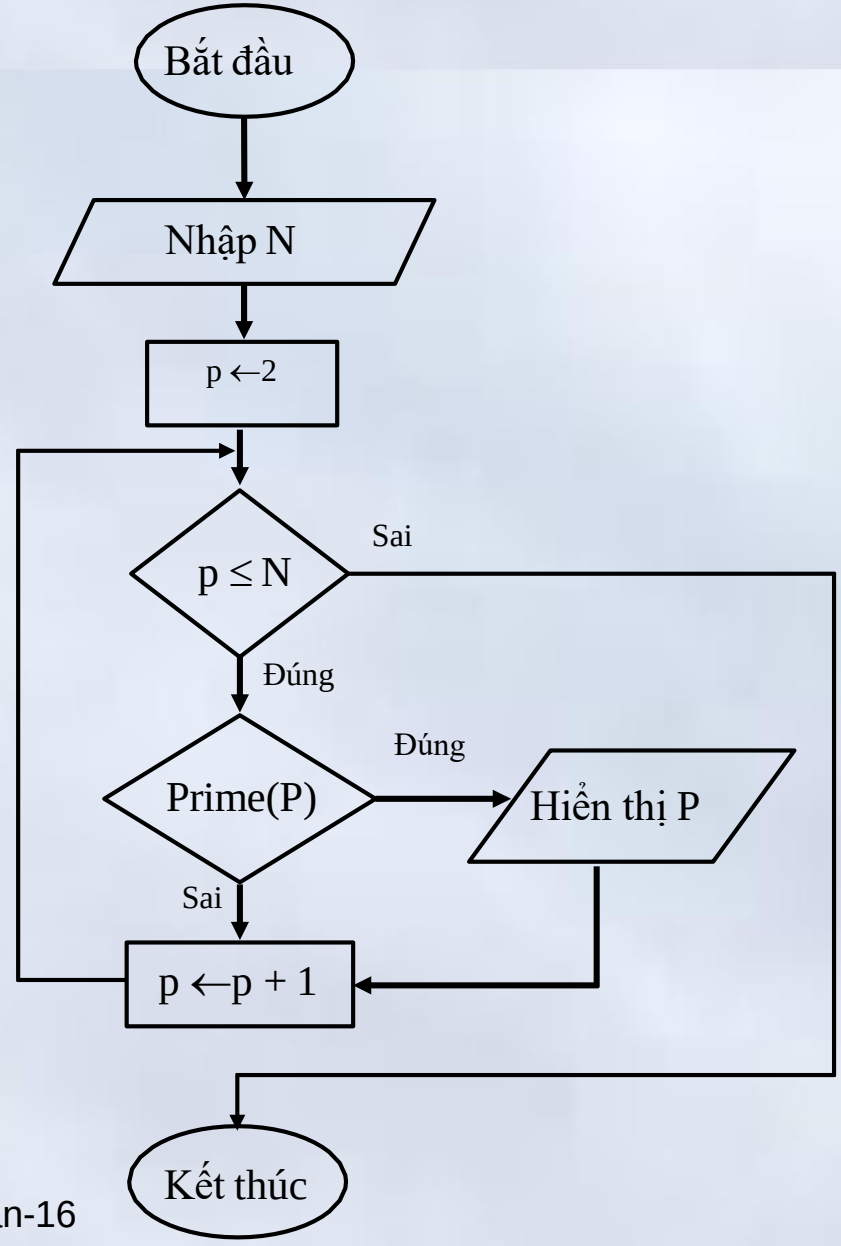
Else Output: P không là số nguyên tố

Endif

End

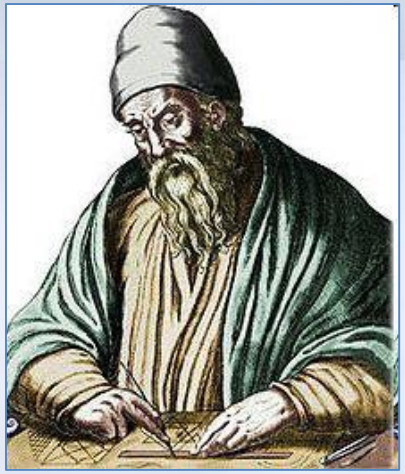
```
Function Prime(P):Bool
        Flag ← FALSE
        …
        return Flag
End Function
```

Liệt kê các số nguyên tố [2..N]



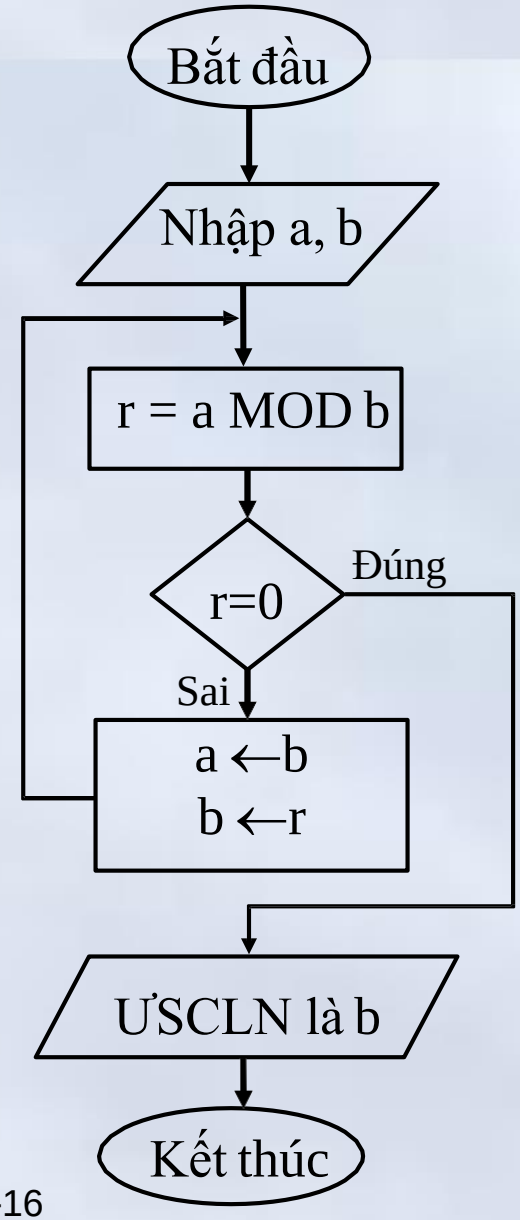

```
Begin
Input N
For p ← 2 to N do
        If  Prime(P) Then
                Output: P
        Endif
End For
End
```

Tìm ƯSCLN của 2 số nguyên dương (1/3)



1. Nhập số a
2. Nhập số b
3. Chia a cho b với số dư là r
4. Nếu r = 0 thực hiện 6
5. Nếu r ≠0 gán giá trị b cho a, giá trị r cho b và quay lại bước 3
6. Thông báo kết quả ƯSCLN là b
7. Kết thúc

Tìm ƯSCLN của 2 số nguyên dương (2/3)

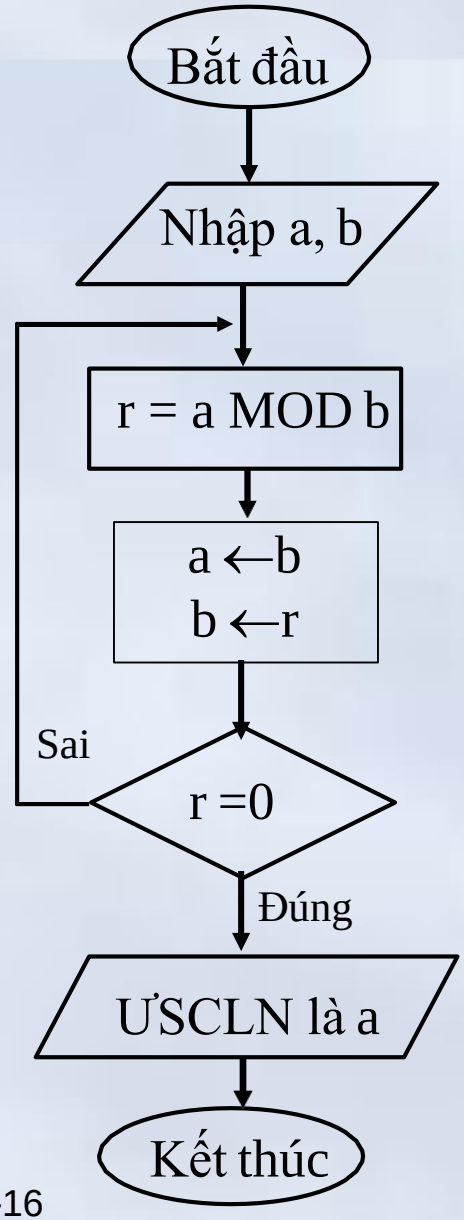


Begin

1. **Input** a, b
2. **Repeat**
3. r ← a MOD b
4. **if** r=0 **then goto** 8
5. a ← b
6. b ← r
7. **Until** r=0
8. **Output** *USCLN là: b*

End

Tìm ƯSCLN của 2 số nguyên dương (3/3)



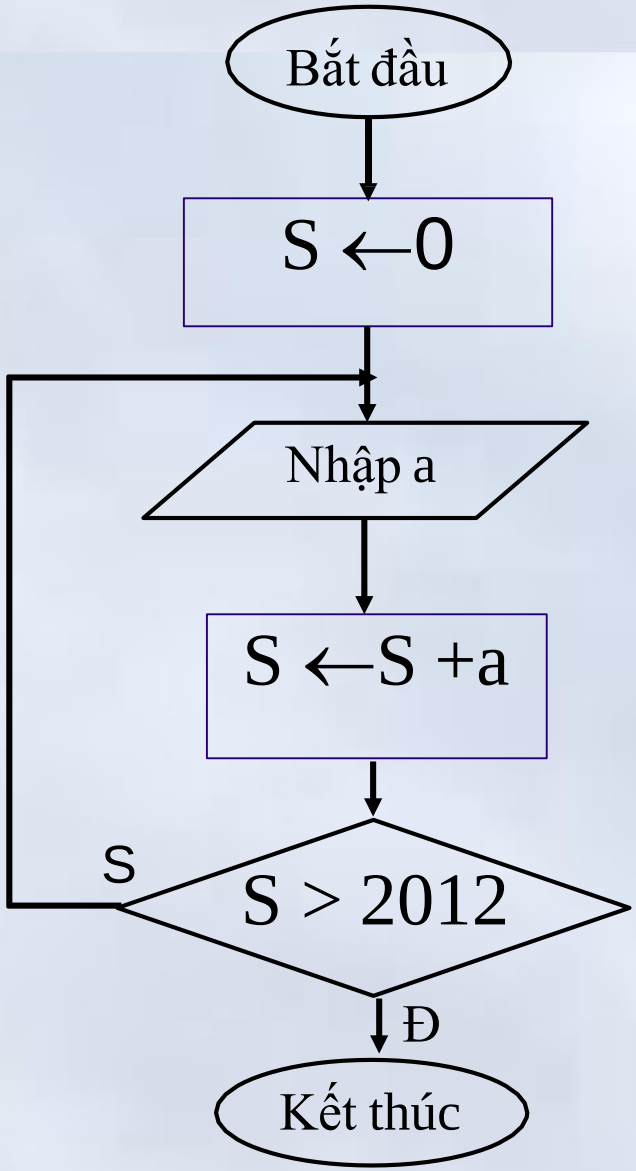
Begin

1. **Input** a, b
2. **Repeat**
3. r ← a MOD b
4. a ← b
5. b ← r
6. **Until** r=0
7. **Output** *USCLN là: a*

End

Giải phương trình $ax^2+bx+c = 0$

Nhập dãy số cho tới khi tổng lớn hơn 2012



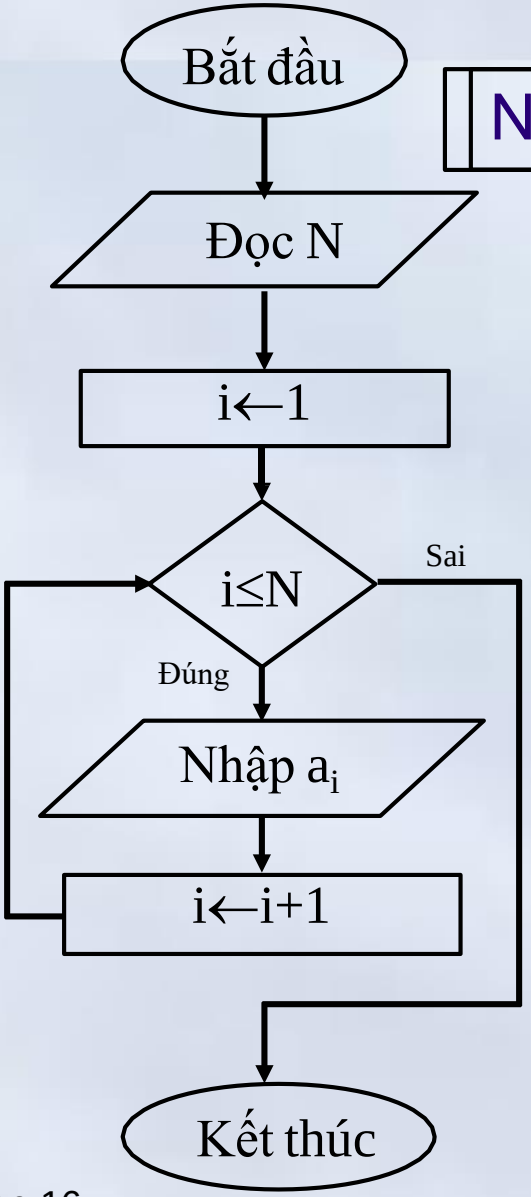

```
Begin
1. S←0
2. Repeat
       Input a
       S←S + a
3. Until S > 2012
End
```

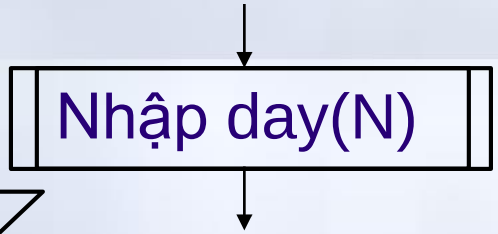
Bài tập tại lớp

Mô tả các thuật toán (sơ đồ khối/mã giả) sau

1. Nhập dãy số nguyên cho tới khi gặp một số âm và đưa ra tổng của các số chia hết cho 5
2. Nhập dãy số nguyên cho tới khi gặp một số chia hết cho 5 và dưa ra trung bình cộng của các số dương trong dãy đã nhập
3. Nhập dãy số cho tới khi gặp số âm và đưa ra số lượng các số nằm trong đoạn [11..22]
4. Nhập dãy số cho tới khi tổng của dãy lớn hơn 1001 hoặc số phần tử đã nhập là 100

Nhập giá trị cho dãy N số



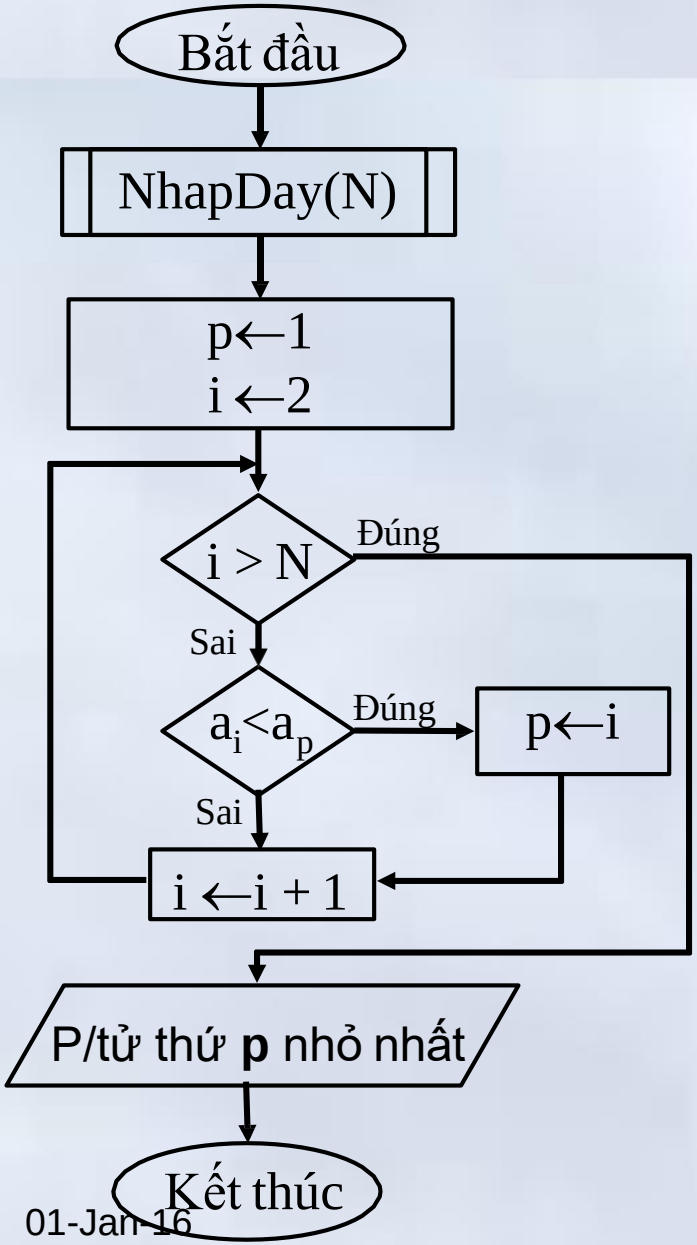


Begin
1. Input N
2. i ← 1
3. While i ≤ N **do**
4. **Input** a_i
5. i ←i+1
6. End while
End

NhapDay(N)

Begin
1. Input N
2. For i ←1 **to** N **do**
Input a_i
3.End For
End

Tìm phần tử nhỏ nhất cho dãy N số

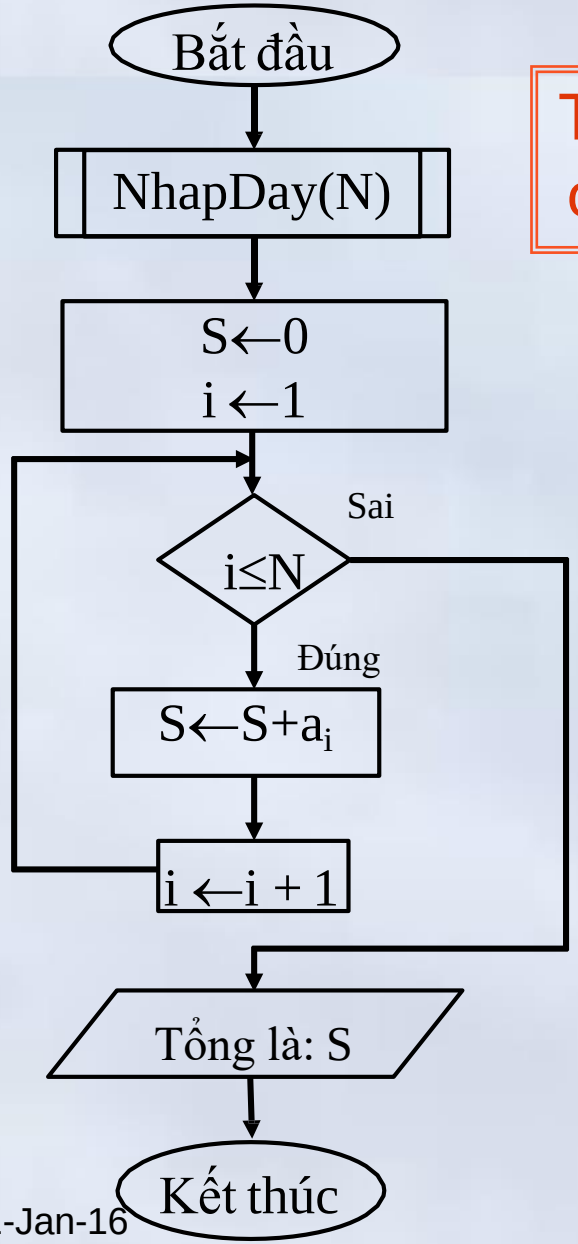


Begin

1. Nhapday(N)
2. p←1
3. i ←2
4. **While** i ≤ N **do**
5. **If** $a_i < a_p$ **then**
6. p←i
7. **Endif**
8. i ←i+1
9. **Endwhile**
10. **Output** P/tử nhỏ nhất: p

End

Tìm tổng của dãy gồm N số



Tổng các số lẻ chia hết cho 5

Begin
1. NhapDay(N)
2. S← 0
3. i ← 1
4. **While** i ≤ N **do**
   S ← S + a_i
   i ←i+1
5. **End while**
6. **Output:** Tổng là S

End

Begin
1. NhapDay(N)
2. S← 0
3. **For** i ←1 **to** N **do**
   S ← S+ a_i
4. **End For**
5. **Output:** Tổng là **S**

End

Kiểm tra phần tử có thuộc dãy (1/3)

Begin

1. Nhập N, dãy số a_1, a_2,…a_N và số cần tìm A
2. i←1
3. Nếu i > N sang bước 7
4. Nếu A = a_i sang bước 8
5. i←i+1
6. Quay về bước 3
7. Thông báo: A không thuộc dãy và kết thúc
8. Thông báo A thuộc dãy và kết thúc

End

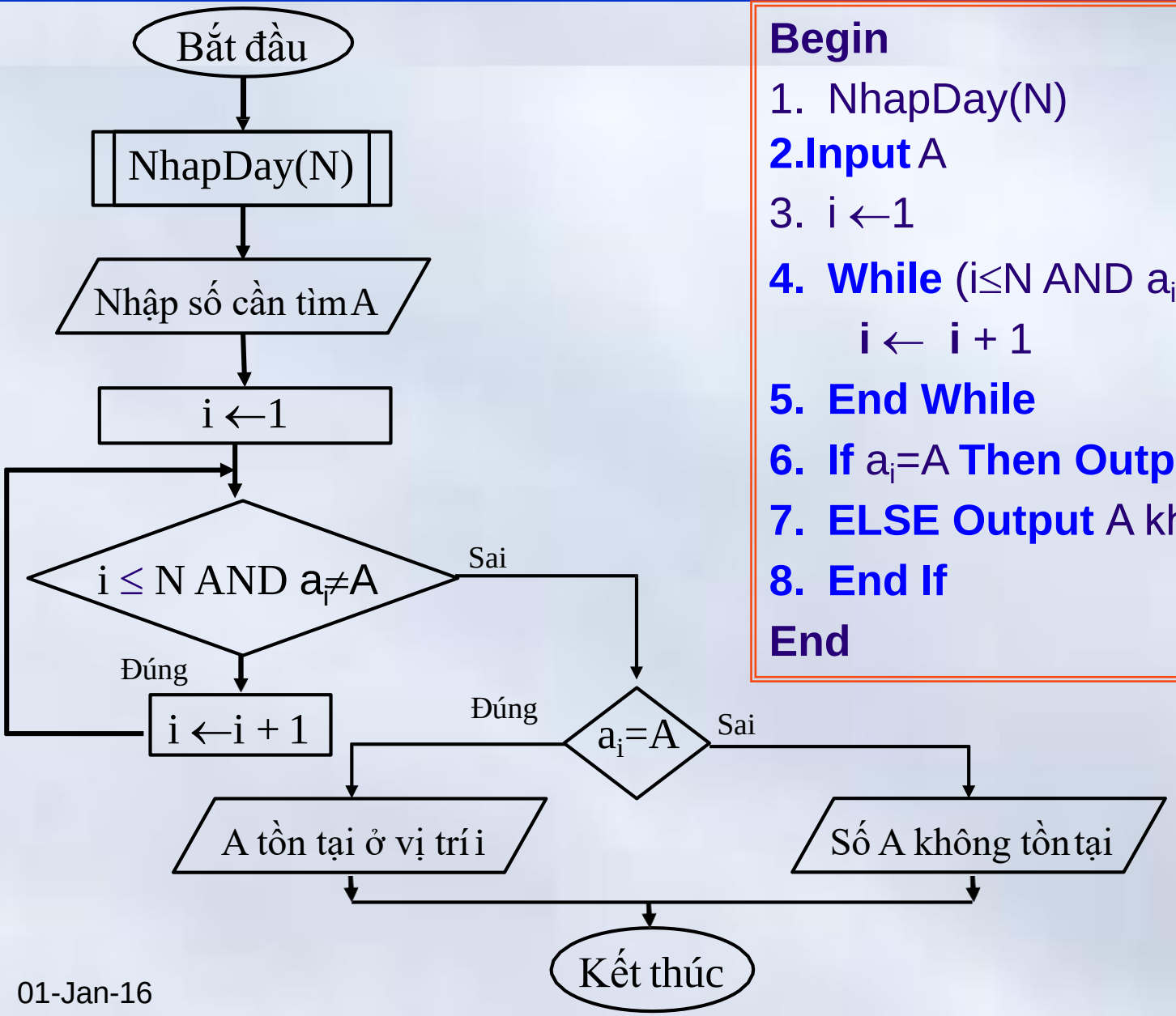
Kiểm tra phần tử có thuộc dãy (2/3)

Hủy bỏ lệnh

Goto ?

```
Begin
1. NhapDay(N)
2.Input A
3. i ←1
4. While (i≤N) Do
      If a_i=A Then
            Output A ở vị trí i
            Goto B7
      End If
      i←i+1
5. End While
6. Output A không tồn tại
7.
End
```

Kiểm tra phần tử có thuộc dãy (3/3)



Begin

1. NhapDay(N)
2. **Input** A
3. i ←1
4. **While** (i≤N AND a_i≠A) **Do**
   i ← **i** + 1
5. **End While**
6. **If** a_i=A **Then Output** A ở vị trí **i**
7. **ELSE Output** A không tồn tại
8. **End If**

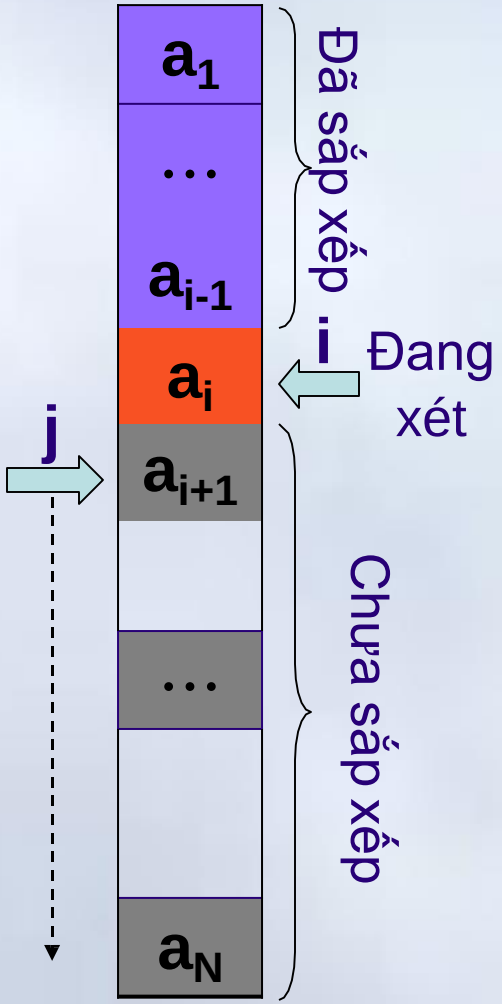
End

Sắp xếp theo thứ tự tăng của dãy gồm N số

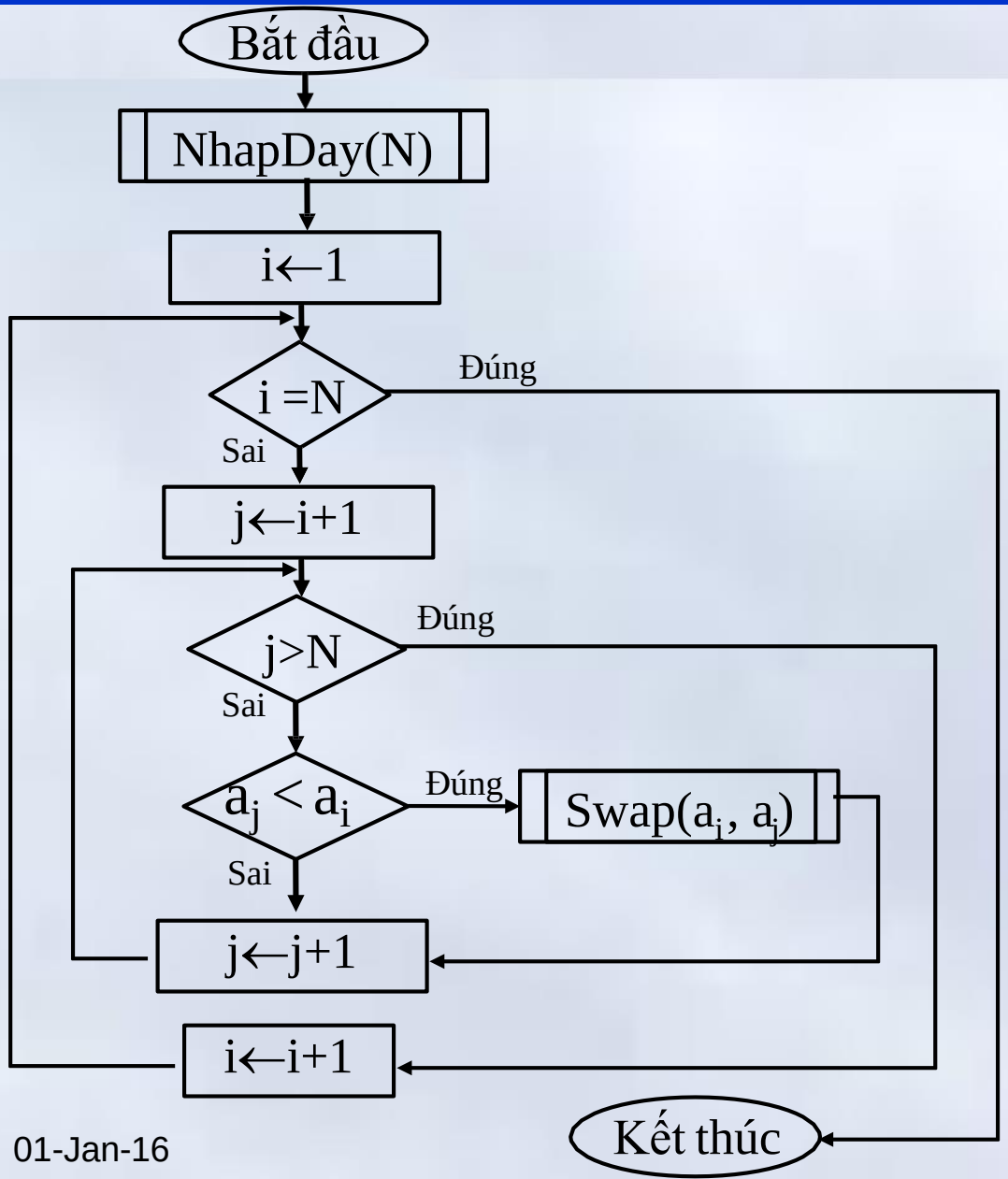
Begin

1. Nhập N và dãy $a_1, a_2, \ldots a_N$
2. $i \leftarrow 1$
3. Nếu i = N, thuật toán kết thúc
4. $j \leftarrow i + 1$
5. Nếu j > N thực hiện bước 9
6. Nếu $a_j < a_i$ thì đổi chỗ a_i và a_j
7. $j \leftarrow j + 1$
8. Quay lại thực hiện bước 5
9. $i \leftarrow i + 1$
10. Quay lại thực hiện bước 3

End



Sắp xếp theo thứ tự tăng của dãy gồm N số



Begin

1. NhapDay(N)
2. i ←1
3. **While** i < N **Do**
4. j ←i+1
5. **While** j ≤ N **Do**
6. **If** $a_j < a_i$ **Then**
   Swap(a_i,a_j)
7. **End If**
8. j ← j+1
9. **End while**
10. i ← i + 1
11. **End while**

End

Bài tập về nhà

Mô tả các thuật toán (sơ đồ khối/mã giả) sau

- Đếm số chẵn của một dãy N số nguyên
- Đưa ra trung bình cộng các số dương của một dãy gồm N số
- Sắp xếp lại dãy N số thực theo nguyên tắc:
  - Các số 0 ở đầu dãy, sau đó là các số âm rồi đến các số dương
- Đếm số hạng đạt giá trị bằng giá trị lớn nhất
- ….

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. Lịch sử phát triển của ngôn ngữ C
2. Các phần tử cơ bản của ngôn ngữ C
3. Cấu trúc cơ bản của chương trình C
4. Biên dịch chương trình C

Sự ra đời của C

- Nhu cầu viết lại HĐH Unix cho các hệ máy tính khác nhau
  - Dùng Assembly
    - Công việc nặng nề, phức tạp
    - Khó chuyển đổi chương trình giữa các hệ máy tính khác nhau
  - Cần ngôn ngữ mới
    - Đơn giản việc lập trình
    - Tính khả chuyển cao
- C ra đời tại Bell Lab thuộc tập đoàn AT&T
  - Tác giả Brian W. Kernighan & Dennis Ritchie
  - Dựa trên nền BCPL& B
  - Phát triển năm 1970, hoàn thành 1972

Ngôn ngữ lập trình C

- Đặc điểm
  - Ngôn ngữ lập trình hệ thống
  - Tính khả chuyển, linh hoạt cao
  - Có thế mạnh trong xử lý dữ liệu số, văn bản, cơ sở dữ liệu,..
- Phạm vi sử dụng
  - Viết các chương trình hệ thống
    - Hệ điều hành Unix có 90% mã C, 10% mã hợp ngữ
  - Các trình điều khiển thiết bị (device driver)
  - Xử lý ảnh

Ngôn ngữ lập trình C

- Các phiên bản
  - **ANSI C**: C chuẩn (1989)
  - Các phiên bản khác xây dựng dựa trên ANSI C
    - Đưa thêm thư viện; Bổ sung cho thư viện chẩn của ANSI C
- Các trình biên dịch phổ biên
  - Turbo C++ và Borland C++ của hãng Borland Inc
  - VC và MSC của Microsoft Corp
  - GCC của GNU project

Nội dung chính

1. Lịch sử phát triển
2. Các phần tử cơ bản của ngôn ngữ C
3. Cấu trúc cơ bản của chương trình C
4. Biên dịch chương trình C

Các phần tử cơ bản

1. Tập ký tự
2. Từ khóa
3. Định danh
4. Các kiểu dữ liệu
5. Hằng
6. Biến
7. Hàm
8. Biểu thức
9. Câu lệnh
10. Chú thích

1. Tập ký tự

Ký tự là các phần tử cơ bản tạo nên chương trình

- **Chương trình:** Tập các **câu lệnh** nhằm giải quyết nhiệm vụ đặt ra
- **Câu lệnh:** là các **từ** (*từ vựng*) liên kết với nhau theo cú pháp của ngôn ngữ lập trình
  - ***Ví dụ:*** while (i < N) do
- **Các từ:** Tổ hợp các ký tự theo nguyên tắc xây dựng *từ vựng*
  - ***Ví dụ:*** TenFile, BaiTap2…

1. Tập ký tự→ Tập ký tự trong C

- 26 chữ cái hoa:A B C ... X Y Z
- 26 chữ cái thường:a b c … x y z.
- 10 chữ số: 0 1 2 3 4 5 6 7 8 9.
- Các kí hiệu toán học:+ - * / = < >
- Các dấu ngăn cách: . ; , : space tab
- Các dấu ngoặc:( ) [ ] { }
- Các kí hiệu đặc biệt:_ ? $ & # ^ \ ! ‘ “ ~ ...

2. Từ khóa (keyword)

- Được định nghĩa sẵn trong mỗi NNLT
- Dành riêng cho các mục đích xác định
  - Đặt tên cho kiểu dữ liệu:
    - int, float, double…
  - Mô tả các lệnh, các cấu trúc lập trình
    - if, else, while, case, for…

2. Từ khóa→Từ khóa hay dùng trong Turbo C

break	case	char	const	continue	default
do	double	else	enum	float	for
goto	if	int	interrupt	long	return
short	signed	sizeof	static	struct	switch
typedef	union	unsigned	void	while	

Lưu ý: *Tất cả từ khóa trong C đều viết bằng chữ cái thường*

3. Định danh (*Identifier*)

- Định danh (*Tên*) là một dãy các kí tự dùng để gọi tên các đối tượng trong chương trình.
  - Các đối tượng trong chương trình
    - Biến
    - Hằng số
    - Hàm
    - Kiểu dữ liệu
- Định danh có thể được đặt bởi
  - Ngôn ngữ lập trình → các từ khóa
  - Người lập trình

3. Định danh→ Quy tắc đặt tên định danh trong C

- Định danh được bắt đầu bởi chữ cái hoặc dấu gạch dưới "_" (*underscore*)
- Các kí tự tiếp theo chỉ có thể là: chữ cái, chữ số hoặc dấu gạch dưới "_"
- Định danh do người lập trình đặt không được trùng với các từ khóa của C
- Độ dài định danh tùy thuộc phiên bản C
  - Turbo C++, không giới hạn độ dài tên, nhưng trình biên dịch chỉ sử dụng 32 ký tự đầu

Chú ý: *C là ngôn ngữ có phân biệt chữ hoa và chữ thường*

3. Định danh→Ví dụ

- Định danh hợp lệ:

  i, x, y, a, b, _function,

  _MY_CONSTANT, PI, gia_tri_1

- Định danh không hợp lệ

  1_a, 3d, 55x (bắt đầu bằng chữ số)

  so luong, sin() (có kí tự không hợp lệ, dấu cách, dấu ngoặc..)

  int, char (trùng với từ khóa của C)

3. Định danh→ Một số quy ước (code convention)

- Định danh nên có tính gợi nhớ
- Nên sử dụng dấu gạch dưới để phân tách các định danh gồm nhiều từ
  - Có thể dùng cách viết hoa chữ cái đầu mỗi từ
    - **Ví dụ:** sinh_vien, sinhVien, SinhVien
- Quy ước thường được sử dụng:
  - Hằng số dùng chữ cái hoa
    - **Ví dụ**: PI, EPSILON,…
  - Các biến, hàm, cấu trúc dùng chữ cái thường
    - *Biến điều khiển vòng lặp*: i, j, k…
    - *Hàm:* NhapDuLieu, TimKiem,…
    - *Cấu trúc:* SinhVien, MatHang,…

4. Các kiểu dữ liệu

- Một kiểu dữ liệu là một tập hợp các giá trị mà một dữ liệu thuộc kiểu dữ liệu đó có thể nhận được.
  - ***Ví dụ:*** Một đối tượng kiểu **int** của C sẽ là
    - Một số nguyên (*Số nguyên có dấu, 2 byte*)
    - Giá trị thuộc khoảng: [-32,768 (-2^{15}) …32,767 ($2^{15}-1$)]
- Trên một kiểu dữ liệu, xác định một số phép toán đối với các dữ liệu thuộc kiểu dữ liệu tương ứng.

4. Các kiểu dữ liệu→Ví dụ kiểu int

Một số phép toán được định nghĩa trên kiểu dữ liệu int của C

Tên phép toán	Ký hiệu	Ví dụ
Đảo dấu	-	
Cộng;Trừ;Nhân	+ ; - ; *	
Chia lấy nguyên	/	17/3→5
Chia lấy phần dư	%	17%3→2
So sánh	>, <, >=, <=, ==, !=	
Logic bit: AND; OR; XOR;NOT, Shift,…	& ; | ; ^ ; ~ ; <<; >>	3^17 →18 ~3 →-4

5. Hằng

- Hằng (*constant*) là đại lượng có ***giá trị không đổi trong chương trình*.**
- Giá trị hằng do người lập trình xác định
- Các loại hằng
  - Hằng số nguyên
  - Hằng số thực
  - Hằng ký tự
  - Hằng chuỗi/xâu kỹ tự

5. Hằng → Hằng số nguyên

- Trong C, hằng số nguyên có thể biểu diễn dưới các dạng
  - Dạng thập phân
  - Dạng thập lục phân
  - Dạng bát phân

Giá trị thập phân	Giá trị thập lục phân	Giá trị bát phân
2011	0x7DB	03733
396	0x18C	0614

5. Hằng → Hằng số thực

- Trong C, hằng số thực có thể biểu diễn dưới các dạng
  - Dạng số thực dấu phẩy tĩnh
  - Dạng số thực dấu phẩy động

Số thực dấu phẩy tĩnh	Số thực dấu phẩy động
3.14159	31.4159 E-1
123.456	12.3456 E+1 hoặc 1.23456 E+2

5. Hằng → Hằng ký tự

- Hằng ký tự có thể biểu diễn theo hai cách
  - Đặt ký hiệu của ký tự giữa hai dấu nháy đơn
  - Dùng mã ASCII của ký tự:
    - Số thứ tự của ký tự đó trong bảng mã ASCII
    - Là số nguyên→tuân thủ quy tắc biểu diễn số nguyên

Ký tự	Dùng nháy đơn	Dùng mã ASCII
Chữ cái A	'A'	65, 0x41, 0101
Dấu nháy đơn	'\''	39, 0x27, 047
Ký tự tab	'\t'	9, 0x09, 011

5. Hằng → Hằng chuỗi/xâu kỹ tự

- Hằng chuỗi/xâu kí tự được biểu diễn bởi đặt dãy các kí tự trong xâu trong cặp dấu nháy kép.
- Ví dụ:
  - “ngon ngu lap trinh C”
  - “Tin hoc dai cuong”
  - “Dai hoc Bach Khoa Ha Noi”

6. Biến (*variable*)

- Biến là đại lượng mà ***giá trị có thể thay đổi trong chương trình*.**
- Tên biến phải được đặt theo quy tắc đặt tên
  - Về thực chất, biến là các ô nhớ trong bộ nhớ máy tính dành cho 1 kiểu dữ liệu nào đó và được đặt tên để tiện tham khảo
    - ***Ví dụ:*** Biến kiểu int chiếm 2 ô nhớ
- **Lưu ý:**
  - Hằng số và biến được sử dụng để lưu trữ dữ liệu trong chương trình và phải thuộc một kiểu dữ liệu nào đó

7. Hàm (function)

- Hàm là chương trình con có chức năng
  - Nhận dữ liệu đầu vào (*các tham số vào*)
  - Thực hiện một công việc nào đó
  - Trả về kết quả ứng với tham số truyền vào
    - Ví dụ: hàm sin(x)
      - sin(3.14/2) →1.000
      - Sin(3.14/6) →0.499770
- Hàm không trả lại một giá trị: Thủ tục
  - Ví dụ: clrscr()

7. Hàm → Một số hàm toán học

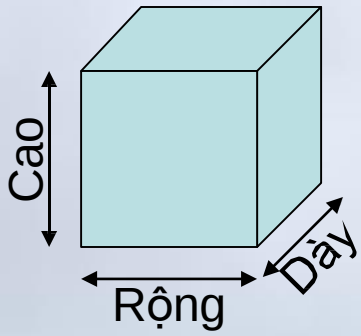
Hàm	Ý nghĩa	Ví dụ
sqrt(x)	Căn bậc 2 của x	sqrt(16.0) → 4.0
pow(x,y)	X mũ y (x^y)	pow(2,3) → 8
fabs(x)	Trị tuyệt đối của x (|x|)	fabs(-5.0) →5.0
exp(x)	E mũ x (e^x)	exp(1.0)→2.71828
log(x)	Logarithm tự nhiên của x (*ln x*)	Log(2.718)→0.999
log10(x)	Logarithm cơ số 10 của x (*logx*)	Log10(100) →2.00
sin(x) cos(x)/ tan(x)	Các hàm lượng giác	
ceil(x)	Số nguyên nhỏ nhất không nhỏ hơn x ($\lceil x \rceil$)	ceil(2.5)=3 ceil(-2.5)=-2
floor(x)	Số nguyên lớn nhất không lớn hơn x ($\lfloor x \rfloor$)	floor(2.5)=2 floor(-2.5)=-3

8. Biểu thức

- Biểu thức là sự kết hợp các các toán hạng (*operand*) bởi các toán tử (*operator*) theo một quy tắc xác định.
- Các toán hạng có thể là biến, hằng, hàm…
- Các toán tử rất đa dạng: cộng, trừ, nhân, chia

Ví dụ

- Thể tích hình hộp: **V=Rộng*Cao*Dày**
  - Phép nhân (*) là toán tử
  - Các toán hạng **Rộng, Cao, Dày**



9. Câu lệnh (*statement*)

- Câu lệnh diễn tả một hoặc một nhóm các thao tác trong giải thuật.
  - Chương trình được tạo thành từ dãy các câu lệnh.
- Các câu lệnh trong C, được kết thúc bởi dấu chấm phẩy (;)
  - Dấu chấm phẩy (;) dùng phân cách các lệnh

9. Câu lệnh → Phân loại

- Câu lệnh đơn:
  - Những câu lệnh không chứa câu lệnh khác.
    - ***Ví dụ:*** Phép gán, gọi hàm, vào/ra dữ liệu
- Các câu lệnh phức:
  - Những câu lệnh chứa câu lệnh khác.
    - ***Ví dụ***: Lệnh khối (Tập các lệnh đơn nhóm lại với nhau và đặt trong cặp ngoặc nhọn « { } »)
  - Các lệnh điều khiển cấu trúc chương trình
    - ***Ví dụ:*** Lệnh rẽ nhánh, lệnh lặp..

10. Chú thích (comment)

- Lời mô tả, giải thích vắn tắt cho một câu lệnh, một đoạn chương trình hoặc cả chương trình
  - Giúp việc đọc hiểu chương trình dễ dàng hơn
  - Chú thích không phải là câu lệnh ⇒ không ảnh hưởng tới chương trình
    - Khi gặp chú thích, trình biên dịch sẽ bỏ qua
- Cách viết chú thích
  - Chú thích một dòng: sử dụng « // »
  - Chú thích nhiều dòng: sử dụng « /* » và « */ »

Nội dung chính

1. Lịch sử phát triển
2. Các phần tử cơ bản của ngôn ngữ C
3. Cấu trúc cơ bản của chương trình C
4. Biên dịch chương trình C

Các phần cơ bản

Khai báo các tệp tiêu đề

#include

Khai báo các đối tượng toàn cục

- Định nghĩa kiểu dữ liệu mới
- Các biến, hằng
- Các hàm nguyên mẫu (prototype)

Định nghĩa hàm main()

{

}

Định nghĩa các hàm đã khai báo nguyên mẫu

1. Khai báo các tệp tiêu đề

- Liệt kê danh sách thư viện sẽ được sử dụng trong chương trình
  - Các hàm của C đều thuộc một thư viện nào đó
  - Không khai báo thư viện, trình biên dịch không hiểu được hàm (*có thể báo lỗi*)
- Cách thức (*cú pháp*) khai báo
  1. **#include<**ThuVien.h**>**
     - Thư viện phải nằm trong thư mục chứa các header file
     - Thường được sử dụng
     - ***Ví dụ:*** #include<stdio.h>
  2. **#include "**ThuVien.h**"**
     - Tìm kiếm thư viện tại thư mục hiện tại

2. Khai báo các đối tượng toàn cục

- Các đối tượng toàn cục có phạm vi sử dụng trong toàn bộ chương trình
  - Các kiểu dữ liệu mới
  - Các hằng, biến
  - Các nguyên hàm
- Tuân theo nguyên tắc khai báo đối tượng

2. Khai báo các đối tượng toàn cục

Định nghĩa kiểu dữ liệu

Cú pháp: typedef <ĐịnhNghĩaKiểu> <Tên kiểu>

Ví dụ: **typedef** unsigned char byte;

typedef struct {float re, im;} complex;

Khai báo hằng

const float PI = 3.1415;

#define Max 50

Khai báo biến

int N;

float Delta, x1, x2;

2. Khai báo các đối tượng toàn cục (tiếp)

Khai báo các hàm nguyên mẫu

- Khai báo thông tin về các hàm của người dùng sẽ được sử dụng trong chương trình
  - Tên hàm
  - Danh sách các **kiểu** tham số sẽ truyền vào
  - Kiểu dữ liệu trả về
- ***Ví dụ***

```
float DienTichTamGiac(float a, float b, float c);
int getMax(int Arr []);
void swap(int * a, int  * b);
    void swap(int *, int  *);
```

Có thể bỏ tên tham số

3. Định nghĩa hàm **main()**

- Bắt buộc phải có
- Là hàm đặc biệt trong C, đánh dấu điểm bắt đầu của mọi chương trình C
  - Khi thực hiện một chương trình C, hệ thống sẽ gọi tới hàm main đầu tiên, sau đó sẽ thực hiện lần lượt các câu lệnh (*bao gồm cả lời gọi tới các hàm khác*) nằm trong hàm main()
- **Cú pháp**

```
void main(){….}
void main(int argc, char * argv[ ]){….}
int main(){….; return 0;}
int main(int argc, char * argv[ ]){….; return 0;}
```

4. Định nghĩa các hàm đã khai báo

- Định nghĩa các hàm đã khai báo ở phần 3 (Phần khai báo nguyên mẫu - prototype)
  - Phần khai báo nguyên mẫu mới chỉ khai báo các thông tin cơ bản về hàm, chưa xác định rõ hàm hoạt động như thế nào
- **Ví dụ**

```
float DienTichTamGiac(float a, float b, float c){
    float p = (a+b+c)/2;
    return sqrt(p*(p-a)*(p-b)*(p-c));
}
```

$$S_\Delta = \sqrt{p(p-a)(p-b)(p-c)}$$

Chú ý

Các phần không bắt buộc phải theo đúng thứ tự

- Khi định nghĩa hàm được đặt trước hàm **main**(), không cần khai báo nguyên hàm
- **Nguyên tắc**:
  - Mọi đối tượng cần phải được khai báo trước khi sử dụng

Chương trình đầu tiên: Hello world!

```
1. #include <stdio.h>
2. int main(){        //Không cần tham số dòng lệnh
3.       printf("Hello world! \n");
4.return 0;           //Trả về giá trị 0
5. }
```

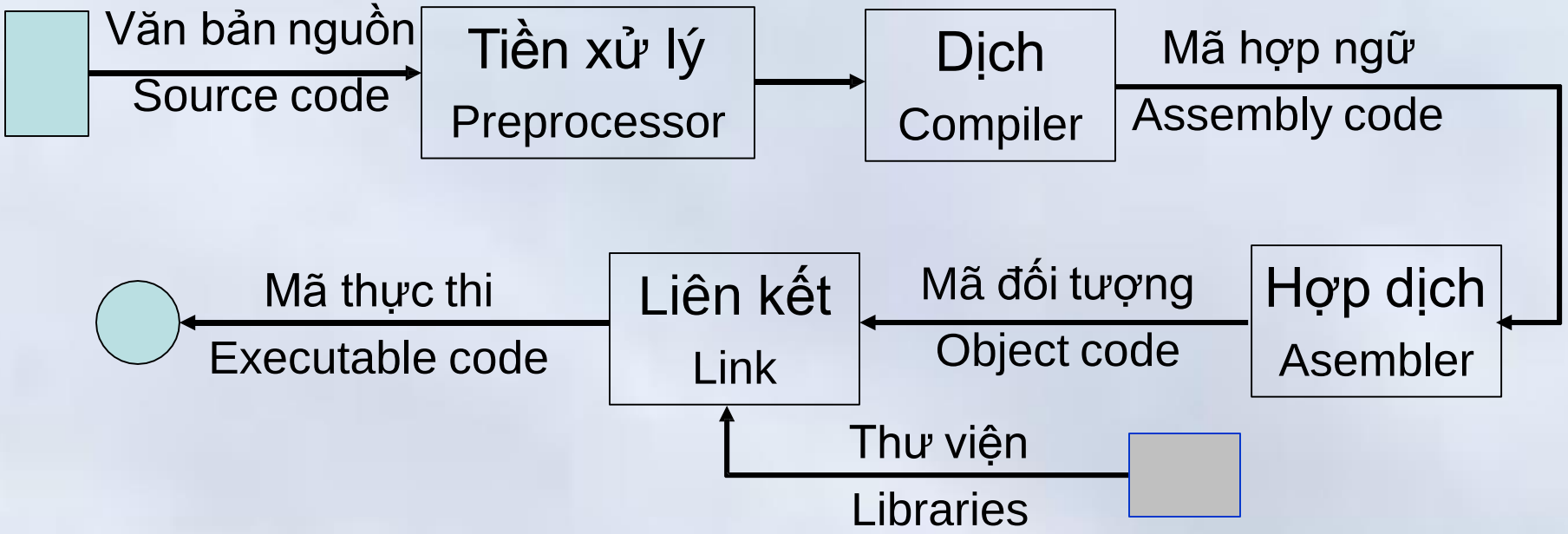
5. Nạp thư viện **stdio.h** vào, đây là thư viện vào ra chuẩn (*standard input output*) chứa khai báo nguyên hàm cho hàm ***printf***
6. Điểm bắt đầu thực hiện của chương trình. Máy tính thực hiện các câu lệnh nằm trong cặp ngoặc {} của main()
7. Hàm **printf** in ra một hằng chuỗi, có kết thuc bởi dấu xuống dòng (\n)
8. Trả về hệ điều hành một giá trị. Giá trị 0 thường dùng để thể hiện chương trình không có lỗi

Nội dung chính

1. Lịch sử phát triển
2. Các phần tử cơ bản của ngôn ngữ C
3. Cấu trúc cơ bản của chương trình C
4. Biên dịch chương trình C

Biên dịch chương trình

- Chương trình được viết bằng ngôn ngữ bậc cao phải được dịch ra mã máy để thực thi
  - Công việc dịch được thực hiện bởi ***trình biên dịch*** (compiler)
- Các giai đoạn dịch chương trình



Trình biên dịch Turbo C++

- Tồn tại nhiều trình biên dịch cho ngôn ngữ C
  - Turbo C++ của Borland Inc
    - Cho phép biên dịch cả C và C++
  - MSC của Microsoft, GCC của GNU
  - Dev-C, C-free,…
- Turbo C++ có nhiều phiên bản khác nhau
  - Sử dụng Turbo C++3.0 (***TC***)
    - Gọn nhẹ, đủ tính năng và dễ sử dụng

Cài đặt Turbo C++ 3.0

B1: Chuẩn bị bộ cài của Turbo C++ 3.0

- Bộ cài tải trên mạng, kích thước khoảng 4M
- Copy bộ cài này vào máy (*giả sử C:\TC_Setup)*

B2: Cài đặt Turbo C

- Tìm đến thư mục chứa bộ cài (*C:\TC_Setup*)
- Kích hoạt file *INSTALL.EXE*
  - Chương trình sẽ yêu cầu chỉ ra ổ đĩa chứa bộ cài TC
- Enter the SOURCE drive to use
  - Nhập tên ổ đĩa (ổ C nếu đặt bộ cài tại *C:\TC_Setup*).
- Enter the SOURCE Path: Nhập đường dẫn tới thư mục chứa các file của bộ cài TC
  - Thông thường chương trình sẽ tự động tìm ra⇒ chỉ cần ấn Enter để chuyển sang bước tiếp theo.

Cài đặt Turbo C++ 3.0

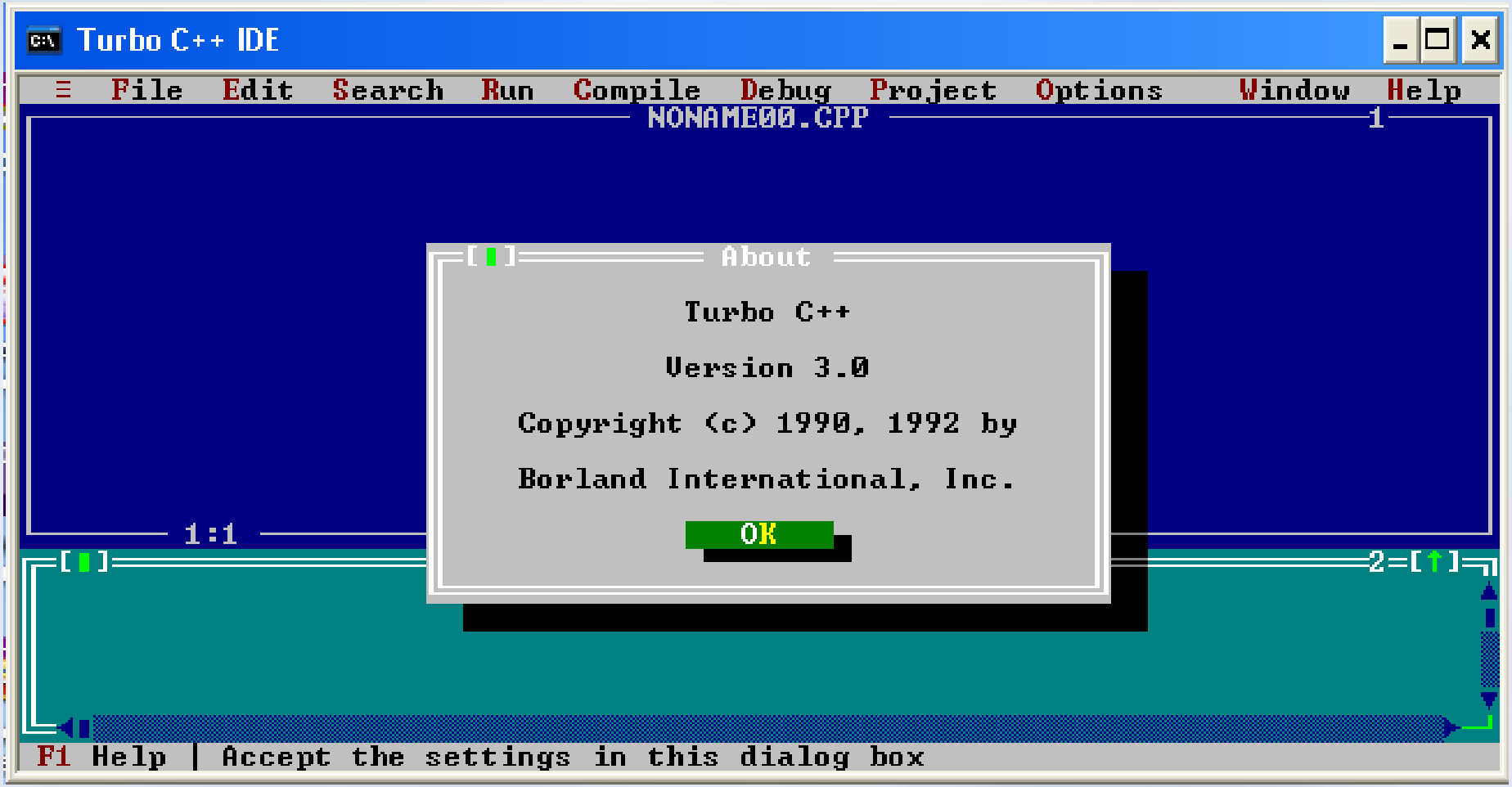
B3: Xác định thư mục cài đặt. Thư mục này sẽ chứa các file của TC được sử dụng về sau.

Dùng các phím ↑ và ↓ để di chuyển hộp sáng đến phần **Start Installation** và ấn **Enter**. Chương trình sẽ tự động thực hiện và hoàn tất quá trình cài đặt

- Thư mục cài đặt mặc định sẽ là *\TC* nằm trên thư mục gốc của ổ đĩa chứa bộ cài.
- Nếu muốn thay đổi thư mục cài đặt, dùng các phím ↑ và ↓ để di chuyển hộp sáng đến **Directories**, gõ Enter và nhập đường dẫn mới, sau đó ấn phím Esc để trở về

Lưu ý: Có thể copy toàn bộ thư mục **TC** để sử dụng

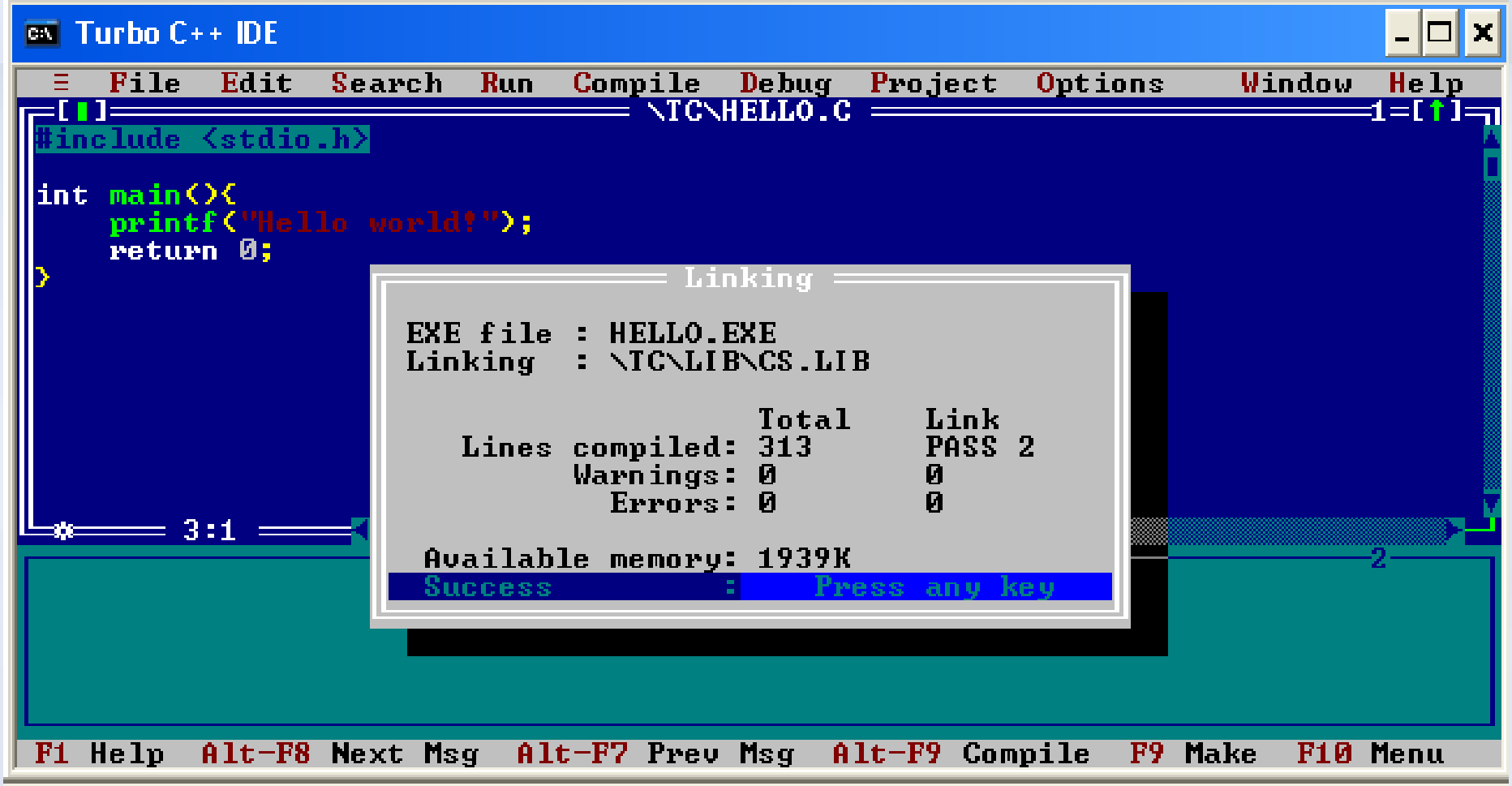
Màn hình giao diện Turbo C++ 3.0

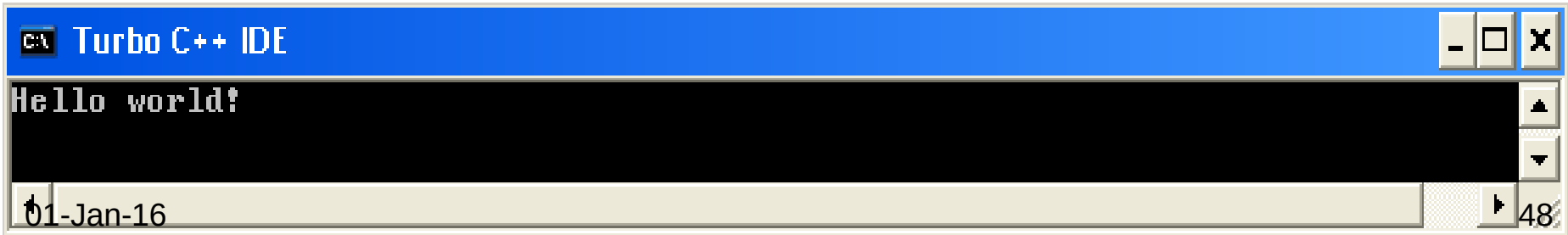


Sử dụng Turbo C++ 3.0

- Khởi động chương trình:
  - Tìm đến thư mục *BIN* trong thư mục cài đặt
  - Chạy file *TC.EXE*
- Tạo cửa sổ soạn thảo mới
  - Chọn menu File (hoặc ấn Alt+F)→ chọn New
- Soạn thảo chương trình
  - Gõ chương trình nguồn vào cửa sổ soạn thảo
- Mở chương trình đã có: Alt+F → Open (**F3**)
- Lưu chương trình: Alt+F → Save (**F2**)
  - Nếu chưa có tên, sẽ được nhắc nhập tên file
- Biên dịch chương trình: Bấm phím **F9**
- Chạy chương trình: **Ctrl + F9**
- Xem lại kết quả thực hiện: **Alt+F5**

Chương trình Hello world!





Tóm tắt

1. Lịch sử phát triển của ngôn ngữ C
2. Các phần tử cơ bản của ngôn ngữ C
   - 10 phần tử cơ bản
3. Cấu trúc cơ bản của chương trình C
   - 4 phần
4. Thực hiện chương trình C với Turbo C

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. Các kiểu dữ liệu chuẩn trong C
2. Biểu thức trong C
3. Các toán tử trong C
4. Một số toán tử đặc trưng

Các kiểu đơn

Kiểu dữ liệu	Ý nghĩa	Kích thước	Miền dữ liệu
char	Kí tự; Số nguyên có dấu	1 byte	-128 ÷ 127
int short int	Số nguyên có dấu	2 byte	-32.768÷32.767
long long int	Số nguyên có dấu	4 byte	-2,147,483,648 ÷ 2,147,483,647
float	Số thực dấu phẩy động, độ chính xác đơn	4 byte	± 3.4E-38 ÷ ± 3.4E+38
double	Số thực dấu phẩy động, độ chính xác kép	8 byte	±1.7E-308 ÷ ± 1.7E+308

Các kiểu kết hợp

Với số nguyên, thêm từ khóa **unsigned** để chỉ ra số không dấu

Kiểu dữ liệu	Ý nghĩa	Kích thước	Miền dữ liệu
unsigned char	Số nguyên không dấu	1 byte	0 ÷ 255
unsigned short unsigned unsigned int	Số nguyên không dấu	2 byte	0÷65.535
unsigned long unsigned long int	Số nguyên không dấu	4 byte	0 ÷ 4,294,967,295
long double	Số thực dấu phẩy động,	10 byte	±3.4E-4932 ÷ ±1.1E+4932
void	Là kiểu rỗng, kích thước không		

Biểu diễn hằng số

Kiểu dữ liệu	Ví dụ	Ý nghĩa
Số nguyên	123, -12	Số thập phân
	012, 03777	Số bát phân
	0x7F, 0x3fe15	Số hệ 16
	39u 0267u, 0xFFu	Số không dấu
Số nguyên lớn	12L, 07723L 0xFFL, -10L 0xFFUL,0xFFLU	
Số thực	3.1415 -12.3, .327 10e-12, -15.3E12 3.1415F, -12.F	

Khai báo biến

- Một biến phải được khai báo trước khi sử dụng
- Cú pháp khai báo:

  KieuDuLieu TenBien;

  KieuDuLieu TenBien1, …, TenBien_N;

- ***Ví dụ***:

```
//Khai báo biến x là một số nguyên 2 byte có dấu
    int x;
//Khai báo các biến y, z là các số thực 4 byte
    float   y,z;
//Sau khi khai báo, có thể sử dụng
    x = 3;          y = x + 1;
```

Khai báo biến

- Sau khi khai báo, biến chưa có giá trị xác định.

  int n; m = 2 * n; ⇒ m=?

  – Biến cần được gán giá trị trước khi sử dụng

- C cho phép kết hợp khai báo và khởi tạo biến

  KieuDuLieu TenBien = GiaTriBanDau;

  KieuDuLieu $Bien_1$=$GiaTri_1$, $Bien_N$=Gia_Tri_N;

- ***Ví dụ:***

```
//Khai báo biến nguyên a và khởi tạo gia tri bằng 3
int a = 3;
//Khai báo biến thực x,y và khởi tạo giá tri bằng 5.0 và 7.6
float x = 5.0, y = 7.6;
```

Khai báo hằng

Dùng chỉ thi **#define**

Không có dấu chấm phây (;)

- Cú pháp:

  # define Tên_hằng Giá_trị

- Ví dụ:

```
#define  MAX_SINH_VIEN  50
#define  CNTT  “Cong nghe thong tin”
#define  DIEM_CHUAN  23.5
```

Khai báo hằng

Dùng từ khóa **const**

- Cú pháp:

  const Kiểu Tên_hằng = giá_trị;

- Ví dụ:

```
const int  MAX_SINH_VIEN  = 50;
const char  CNTT[20] = “Cong nghe thong tin”;
const float  DIEM_CHUAN = 23.5;
```

Khai báo hằng

Chú ý:

- Giá trị của các hằng phải được xác định ngay khi khai báo.
- Trong chương trình, **KHÔNG thể thay đổi** được giá trị của hằng.
- #define là chỉ thị tiền xử lý
  - Dễ đọc, dễ thay đổi
  - Dễ chuyển đổi giữa các nền tảng phần cứng hơn
  - Tốc độ nhanh hơn

Nội dung chính

1. Các kiểu dữ liệu chuẩn trong C
2. Biểu thức trong C
3. Các toán tử trong C
4. Một số toán tử đặc trưng

Mục đích sử dụng

- Làm vế phải của lệnh gán.
- Làm toán hạng trong các biểu thức khác.
- Làm tham số thực sự trong lời gọi hàm.
- Làm biểu thức kiểm tra trong các cấu trúc điều khiển
  - Cấu trúc lặp: **`for, while, do while`**.
  - Cấu trúc rẽ nhánh: **`if, switch`**.

Tính toán giá trị biểu thức

- Các toán hạng được thay thế bởi giá trị tương ứng
- Các phép tính được thực hiên

Ví dụ (alpha = 10, beta = 81)

Biểu thức: alpha + sqrt(beta)

: alpha + sqrt(81)

: alpha + 9.0

: 10 + 9.0

: 19.0

Các loại biểu thức

- Biểu thứ số học
- Biểu thức quan hệ
- Biểu thức logic

Biểu thức số học

- Là biểu thức mà giá trị của nó là các đại lượng số học (*số nguyên, số thực*).
  - Sử dụng các toán tử là các phép toán số học (*cộng, trừ, nhân, chia…*),
  - Các toán hạng là các đại lượng số học (*hằng số, biến, biểu thức khác*).
- ***Ví dụ:*** a, b, c là các biến thuộc kiểu số thực.

```
3 * 3.7
8 + 6/3
a + b – c
```

Biểu thức quan hệ

- Là những biểu thức có sử dụng các **toán tử quan hệ** như lớn hơn, nhỏ hơn, khác nhau…
- Chỉ có thể trả về một trong 2 **giá trị logic** Đúng (***TRUE***) hoặc Sai (***FALSE***)

Ví dụ

```
5 > 7          // có giá trị logic là sai, FALSE
9 != 10        // có giá trị logic là đúng, TRUE
2 >= 2         // có giá trị logic là đúng, TRUE
a > b          // giả sử a, b là 2 biến kiểu int
a+1 > a        // có giá trị đúng, TRUE
```

Biểu thức logic

- Là biểu thức trả về các giá trị logic Đúng/Sai
  - Các phép toán logic gồm có

    AND VÀ logic, sử dụng toán tử **&&** **OR** HOẶC logic, sử dụng toán tử **||** **NOT** PHỦ ĐỊNH, sử dụng toán tử **!**
  - Biểu thức quan hệ là trường hợp riêng của biểu thức logic.
- Ngôn ngữ C coi các giá trị nguyên khác 0 (2, 8, -5,..) là giá trị logic đúng (TRUE), giá trị 0 là giá trị logic sai (FALSE)
  - Biểu thức logic cũng trả về một giá trị số học 0/1

Biểu thức logic →Ví dụ

(5 > 7)&&(9!=10)	// có giá trị logic là sai, FALSE
0 || 1	// có giá trị logic là đúng, TRUE
(5 > 7)||(9!=10)	// có giá trị logic là đúng, TRUE
0	// có giá trị logic là sai, FALSE
!0	// phủ định của 0, có giá trị logic là đúng, TRUE
3	// có giá trị logic là đúng, TRUE
!3	// phủ định của 3, có giá trị logic là sai, FALSE
(a > b)&&(a < b)	// Có giá trị sai, FALSE. Giả sử a, b là 2 biến kiểu int

5 * (12 > 6) →?

Nội dung chính

1. Các kiểu dữ liệu chuẩn trong C
2. Biểu thức trong C
3. Các toán tử trong C
4. Một số toán tử đặc trưng

Các toán tử chính

Các toán tử cho phép tạo nên các biểu thức từ các hằng và biến

- Toán tử số học
- Toán tử quan hệ
- Toán tử logic
- Toán tử logic bit
- Toán tử gán

Các toán tử số học

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ (*int a = 12; float x=3.0*)
-	Đảo dấu	float, double, int, long,.. (*Số nguyên hoặc thực*)	-12, -12.34, - a, - x - -a →12, **--a →?**
+	Cộng	float, double, int, long,..	12 + -x → 9.0
-	Trừ	float, double, int, long,..	12.0 - -3 →15.0
*	Nhân	float, double, int, long,.. (*Số nguyên hoặc thực*)	12 * 3.0 →36.0 12 * 3 → 36
/	Chia	Nếu có ít nhất 1 toán hạng là số thực	17.0/3.0 →5.666667 17/3.0 →5.666667 17.0/3 →5.666667
/	Chia lấy nguyên	**Số nguyên** int, long,..	17/3→5
%	Chia lấy dư	**Số nguyên:** int, long,..	17%3 →2

Các toán tử quan hệ

<, >, <=, >=, ==, !=

- Dùng cho phép so sánh giá trị 2 toán hạng
- Kết quả phép so sánh là một số nguyên
  - **1** nếu quan hệ có kết quả là đúng,
  - **0** nếu quan hệ có kết quả sai

Ví dụ:

6 > 4 → Trả về giá trị 1

6 < 4 → Trả về giá trị 0

int b =(x !=y);

Nếu x và y khác nhau, biểu thức đúng và b mang giá trị 1.
Ngược lại biểu thức sai và b mang giá trị 0

5 * (12 > 6) → 5

Các toán tử logic

Sử dụng để xây dựng các biểu thức logic

- Biểu thức logic có kết quả logic **đúng**

  → Trả về giá trị **1**

- Biểu thức logic có kết quả logic **sai**

  → Trả về giá trị **0**

Các toán tử

Và logic:	Op1 **&&** Op2
Hoặc logic:	Op1 **||** Op2
Phủ định logic:	**!** Op

Oprand: Toán hạng

Các toán tử logic (*tiếp*)

Và logic (&&) :

- Cho kết quả đúng (trả về giá trị 1) khi cả 2 toán hạng đều đúng (**khác 0**)
- ***Ví dụ:*** 3 < 5 && 4 < 6 →1; 3 < 5 && 5 > 6 → 0

Hoặc logic (||):

- Cho kết quả sai (trả về giá trị 0) chỉ khi cả 2 toán hạng đều sai (bằng 0)
- ***Ví dụ:*** 4 || 5 < 3 → 1; 5 < 5 || 2 > 6 → 0

Phủ định logic (!):

- Cho kết quả đúng (1) hoặc sai (0) khi toán hạng là sai (0) hoặc đúng (**khác 0**)
- ***Ví dụ:*** ! 3 → 0; !(2 > 3) → 1;

Toán tử logic bit

Toán tử bit được sử dụng với kiểu số nguyên

Và nhị phân: Op1 **&** Op2

Hoặc nhị phân : Op1 **|** Op2

Hoặc có loại trừ nhị phân: Op1 **^** Op2

Đảo bít nhị phân : **~** Op

Oprand: Toán hạng

Dịch trái: Op **<<** n (nhân với 2^n)

Dịch phải: Op **>>** n (Chia với 2^n)

Op là giá trị được dịch, n là số bit dịch

Toán tử logic bit (*tiếp*)

char Op1 = 83, Op2 = -38, Op = 3;

```
char r = Op1 & Op2;
    0 1 0 1 0 0 1 1
    1 1 0 1 1 0 1 0
  ---------------
r = 0 1 0 1 0 0 1 0 → (82)
```

```
char r = Op1 | Op2;
    0 1 0 1 0 0 1 1
    1 1 0 1 1 0 1 0
  ---------------
r = 1 1 0 1 1 0 1 1 → (-37)
```

```
char r = Op1 ^ Op2;
    0 1 0 1 0 0 1 1
    1 1 0 1 1 0 1 0
  ---------------
r = 1 0 0 0 1 0 0 1 → (-119)
```

```
char r = ~ Op2;
    1 1 0 1 1 0 1 0
  ---------------
r = 0 0 1 0 0 1 0 1 → (37)
```

unsigned char r = Op1 | Op2; r = 1 1 0 1 1 0 1 1 → **219**

Toán tử logic bit (*tiếp*)

char Op1 = 83, Op2 = -38, Op = 3;

char r = Op1 **>>** Op;

0 1 0 1 0 0 1 1

r = **0 0 0** 0 1 0 1 0 → (10)

char r = Op2 **>>** Op;

1 1 0 1 1 0 1 0

r =**1 1 1** 1 1 0 1 1 → (-5)

unsigned char Op =218;
unsigned char r =Op >> 3;

1 1 0 1 1 0 1 0

r =**0 0 0** 1 1 0 1 1 → (27)

char r = Op2 **<<** 2;

r = 0 1 1 0 1 0 0 0 →(104)

(unsigned) int r = Op2<<2 →?

Toán tử gán

Biến = Biểu_thức;

- Ký tự "=" là toán tử gán
  - Biểu thức bên phải dấu bằng tược tính toán
  - Giá trị của *biểu_thức* được gán cho *biến*
- ***Ví dụ:***

```
int a, b, c;
a = 3;
b = a + 5;
c = a * b;
```

Toán tử gán

- Biểu thức gán là biểu thức nên cũng có giá trị.
  - Giá trị của biểu thức gán bằng giá trị của biểu_thức bên phải toán tử
  - Có thể gán giá trị của biểu thức gán cho một biến khác
  - Có thể sử dụng như một biểu thức bình thường
- ***Ví dụ:***

```
int a, b, c;
a = b = 2007;
c = (a = 20) * (b = 30);// c→600
```

Toán tử gán → Dạng kết hợp

Var <op>= Exp ⇔ Var = Var <op> Exp

Toán tử số học:

+= -= *= /= %=

Ví dụ: **a *= b** **// a= a * b**

Toán tử logic bit:

&= |= ^=

Ví dụ: x &= 0x3F // x = x & 0x3F

Toán tử dịch:

<<= >>=

Ví dụ: s <<= 4 // s = s << 4

Nội dung chính

1. Các kiểu dữ liệu chuẩn trong C
2. Biểu thức trong C
3. Các toán tử trong C
4. Một số toán tử đặc trưng

Các toán tử

- Tăng/giảm tự động một đơn vị
- Lấy địa chỉ
- Biểu thức điều kiện
- Toán tử phẩy

Tăng giảm tự động một đơn vị

++	Tăng tự động	++Var, Var++
--	Giảm tự động	--Var, Var--

Variable: Biến

- Tiền tố (hậu tố): biến được tăng(++)/giảm(--) trước (sau) khi sử dụng để tính toán biểu thức

Ví dụ:

```
int     a = 5, b, c, d, e;
        b = a++;        // b = 5 sau đó a = 6
        c = ++a;        // a = 7 rồi tới c = 7
        d = a--;        // d = 7 rồi tới a = 6
        e = --a;        // a = 5 sau đó e= 5
```

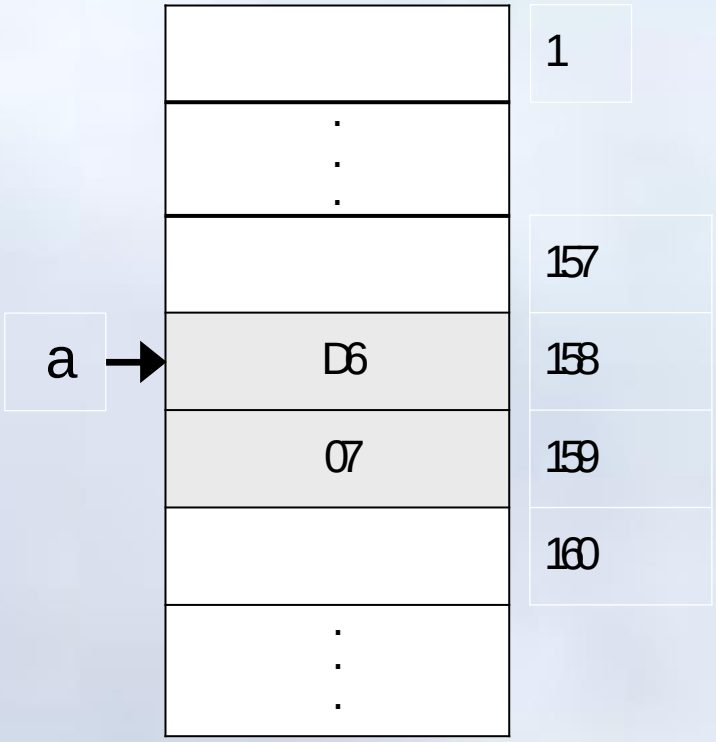
Toán tử lấy địa chỉ

& Tên_biến

Ký tự & là toán tử lấy địa chỉ biến

- Biến thực chất là một vùng nhớ của máy tính được đặt tên → tên của biến
- Mọi ô nhớ trên bộ nhớ máy tính đều được đánh địa chỉ.

  → Mọi biến đều có địa chỉ



Ví dụ:

int a = 2006;

`&a` → Địa chỉ của ô nhớ dùng chứa giá trị biến a

Kiểu địa chỉ?

Toán tử phỏng điều kiện (*biểu thức điều kiện*)

exp1 ? exp 2 : exp3

expression: Biểu thức

- Nếu **exp1 ≠ 0** (*giá trị đúng*), biểu thức điều kiện trả về giá trị của **exp2**
- Nếu **exp1 = 0** (giá trị sai) biểu thức điều kiện trả về giá trị của **exp3**

Ví dụ:

```
float x= 5.2, y = 3.8, z;
z = (x < y) ? x : y;
```

→ `z = 3.8` `// z = min{x, y}`

⇔ if (x < y) z = x; else z = y;

Toán tử phẩy

biểu_thức_1, biểu thức_2,..

- Toán tử phẩy (**,**) cho phép sử dụng nhiều biểu thức tại nơi chỉ cho phép viết một biểu thức
- Các biểu thức được tính toán từ trái qua phải
- Giá trị và kiểu của biểu thức là giá trị và kiểu của biểu thức cuối cùng, bên phải

Ví dụ:

```
if (i = 0, a !=b)…
for(i = 0, j = 0; i < 100; i++, j++)….
```

Chuyển kiểu

(Kiểu) biểu thức

- Chuyển kiểu tự động
  - Chương trình dịch tự động chuyển đổi từ kiểu có phạm vi biểu diễn thấp tới kiểu có phạm vi biểu diễn cao

    `char` → `int` → `long int` → `float` → `double` → `long double`

- Ép kiểu
  - Bằng câu lệnh tường minh trong chương trình
  - Được sử dụng khi muốn chuyển sang kiểu có phạm vi biểu diễn thấp hơn

Chuyển kiểu→Ví dụ

```
#include <stdio.h>
#include <conio.h>

void main(){
    long L = 0xABCDEF; float f = 123.456;
    int i;
    clrscr();
    i = (int) L;
    printf("\n L = %ld; i = %d(%X)", L, i, i);
    i = (int) f; L = (long) f;
    printf("\n f = %f; L = %ld;  i = %d", f, L, i);
}
```

L = 11259375; i = -12817(CDEF)

f = 123.456001; L = 123; i = 123

Thứ tự ưu tiên các toán tử

Mức	Toán tử	Chức năng	Chiều
1	-> . [ ] () ++ hậu tố --hậu tố	Lựa chọn, chỉ số…	→
2	++ -- ~ ! + - * & () sizeof	Toán tử 1 ngôi, ép kiểu,…	←
3	* / %	Toán tử số học lớp nhân	→
4	+ -	Toán tử số học lớp cộng	→
5	>> <<	Dịch bit	→
6	< <= > >=	Toán tử quan hệ	→
7	== !=	Bằng, khác	→
8	&	AND nhị phân	→
9	^	XOR nhị phân	→
10	|	OR nhị phân	→
11	&&	AND logic	→
12	||	OR logic	→
13	? :	Toán tử phỏng điều kiện	←
14	= *= += <<= &= ...	Toán tử gán	←

Chiều kết hợp với các toán hạng

Thứ tự ưu tiên các toán tử

Nguyên tắc

- Biểu thức con trong ngoặc được tính toán trước
- Phép toán một ngôi đứng bên trái toán hạng được kết hợp với toán hạng đi liền nó.
- Toán hạng đứng cạnh hai toán tử
  - Nếu hai toán tử có độ ưu tiên khác nhau thì toán tử nào có độ ưu tiên cao hơn sẽ kết hợp với toán hạng
  - Nếu hai toán tử cùng độ ưu tiên thì dựa vào trật tự kết hợp của các toán tử để xác định toán tử được kết hợp với toán hạng.

Ví dụ

a < 10 && 2 * b < c ≡ (a < 10) && ((2 * b) < c)

Chú ý: int x = 5, a = 5 * x++; → a = 25, x = 6

Ví dụ

```
const int N=10;
float S= 0.0;
int b;
S = N/3 +1;
b=(S>4);
```

S= ? b = ?

```
int a= 3, b=4, c;
c = a++  * ++b;
```

a= ? b= ? c= ?

```
int k ,num=30;
k =num>5 ? (num <=10 ? 100 : 200): 500;
```

k=?

Ví dụ

```
const int N=10;
float S= 0.0;
int b;
S = N/3 +1;
b=(S>4);
```

S= 4 b = 0

```
int a= 3, b=4, c;
c = a++  * ++b;
```

a=4 b= 5 c=15

```
int k ,num=30;
k =num>5 ? (num <=10 ? 100 : 200): 500;
```

k=200

Tóm tắt

- **Kiểu dữ liệu**
  - Nguyên : char, unsigned char, int, long, unsigned int, unsigned long
  - Thực : float, double, long double
- **Giá trị logic**
  - Đúng/TRUE : **1** (*Khác 0*)
  - Sai/FALSE : **0**
- **Toán tử**
  - Một ngôi : + -; ++ --; ~ !; &; ();
  - Hai ngôi : + - * / %; == != < <= > >=; << >>; &, ^, |, && ||; = *= +=…
  - 3 ngôi : **? :**

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. Các hàm vào ra cơ bản:
   - **printf()**
   - **scanf()**
2. Các hàm vào ra khác
   - **gets()**
   - **puts()**
   - **getch()**

Các hàm vào ra cơ bản

- Đưa ra dữ liệu:
  - **printf()**
- Nhập dữ liệu
  - **scanf()**
- Cần nạp thư viện **stdio.h**
  - khai báo tệp tiêu đề :

```
#include <stdio.h>
```

Hàm đưa ra dữ liệu

printf()

Mục đích

- Hiển thị ra màn hình các loại dữ liệu cơ bản
  - Số nguyên, số thực, kí tự, xâu kí tự
- Tạo một số hiệu ứng hiển thị đặc biệt
  - Xuống dòng, sang trang,…

Cú pháp

printf(xau_dinh_dang [, DS_tham_so]);

Cú pháp

printf(xau_dinh_dang [, DS_tham_so]);

- **Xau_dinh_dang**: Là một xâu qui định cách thức hiển thị dữ liệu ra màn hình máy tính.
  - Bao gồm các nhóm kí tự định dạng
  - Nhóm kí tự định dạng thứ *k* xác định quy cách hiển thị tham số thứ *k* trong **`DS_tham_số`**
    - Số lượng tham số trong **`DS_tham_số`** bằng số lượng nhóm các kí tự định dạng trong **`xâu_định_dạng`**.
- **DS_tham_so:** Danh sách các biến/biểu thức sẽ được hiển thị giá trị lên màn hình theo cách thức được qui định trong xau_dinh_dang.

Ví dụ

```
#include <stdio.h>
void main()
{ int a = 5;
  float x = 1.234;
  printf(" Hien thi mot bieu thuc nguyen %d và
   mot so thuc %f ",2 * a, x);
}
```

Kết quả:

Hien thi mot bieu thuc nguyen 10 va mot so thuc 1.234000

Xâu định dạng

- Các kí tự thông thường:
  - Được hiển thị ra màn hình.
- Các kí tự điều khiển:
  - Dùng để tạo các hiệu ứng hiển thị đặc biệt như xuống dòng (**'\n'**)..
- **Các nhóm kí tự định dạng**:
  - Xác định quy cách hiển thị các tham số trong phần danh_sach_tham_so.

Nhóm ký tự định dạng

- Mỗi nhóm kí tự định dạng chỉ dùng cho một kiểu dữ liệu”

  Ví dụ: %d dùng cho kiểu nguyên

  %f dùng cho kiểu thực

- DS_tham_so phải phù hợp với các nhóm kí tự định dạng trong xau_dinh_dang về:
  - Số lượng;
  - Thứ tự
  - Kiểu dữ liệu;

Nếu không phù hợp sẽ hiển thị ra kết quả không như ý

printf(” %d ” ,3.14); →-31457

C-Free →1374389535 !?

Các ký tự định dạng

Ký tự	Kiểu dữ liệu	Kết quả
%i, %d	int, char	Số thập phân
%o	int, char	Số bát phân (không có 0 đằng trước)
%x %X	int, char	Số hexa (chữ thường/chữ hoa)
%u	unsigned int/char	Số thập phân

Các ký tự định dạng

Ký tự	Kiểu dữ liệu	Kết quả
%ld, %li	long	Số thập phân
%lo	long	Số bát phân (không có 0 đằng trước)
%lx, %LX	long	Số hexa (chữ thường/chữ hoa)
%lu	unsigned long	Số thập phân

Nhận xét:

Với kiểu long, thêm ký tự l ngay sau dấu %

Các ký tự định dạng

Ký tự	Kiểu dữ liệu	Kết quả
%f	float/double	Số thực dấu phẩy tĩnh
%e, %E	float/double	Số thực dấu phẩy động
%c	int, char	Kí tự đơn lẻ
%s	char []	Hiển thị xâu kí tự kết thúc bởi '\0'
%%		Hiển thị kí tự %

Độ rộng hiển thị → Số nguyên, Ký tự, Xâu ký tự

- Có dạng "%**m**",
  - **m** là một giá trị nguyên, không âm.
  - **m** cho biết số chỗ trống dành cho hiển thị biểu thức tương ứng

Ví dụ:

int a = 1234;

printf("%5d",a) → □1234

printf("%5d",34)→ □□□34

□ ký hiệu cho dấu trắng (space)

Độ rộng hiển thị→Ví dụ

```
printf("\n%3d %15s %3c", 1, "nguyen van a", 'g');
printf("\n%3d %15s %3c", 2, "tran van b", 'k');
```

```
□□1□□□□nguyen□van□a□□□g
□□2□□□□□□tran□van□b□□□k
```

Độ rộng hiển thị → Số thực

- Có dạng "%**m.n**",
  - **m, n** là 2 giá trị nguyên, không âm.
  - **m** cho biết kích thước để hiển thị số thực
  - **n** cho biết kích thước dành cho phần thập phân, nếu không đủ C sẽ làm tròn khi hiển thị

Ví dụ:

printf("\n%**f**",17.345); → 17.345000

printf("\n%**.2**f",17.345); → 17.35

printf("\n%**8.2f**",17.345); → □□□17.35

Độ rộng hiển thị→ Chú ý

- Nếu số chỗ cần để hiển thị dữ liệu lớn hơn được cung cấp trong định dạng ⇒ Tự động cung cấp thêm chỗ mới để hiển thị đầy đủ, không cắt bớt nội dung của dữ liệu.

Ví dụ:

printf("%2d", 1234); → 1234

printf("%6.3f", 123.456); → 123.456

printf("%12.6e", 123.456); →1.234560e+02

printf("%12.3e", 123.456); →□□□1.235e+02

C-Free →□□1.235e+002

Căn lề trái - căn lề phải

%-

- Khi hiển thị dữ liệu có sử dụng tham số độ rộng, để căn lề trái cần thêm dấu trừ - vào ngay sau dấu %:
  - Ngầm định, căn lề phải

Ví dụ:

printf("%-3d%-10s%-5.2f%-3c",5,"Hello",7.5, 'g')

→5□□Hello□□□□□7.50□g□□

Hàm nhập dữ liệu

scanf()

Mục đích

Dùng để nhập dữ liệu từ bàn phím

- Ký tự đơn lẻ
- Chuỗi ký tự
- Số nguyên
  - Thập phân, Bát phân, Hexa
- Số thực
  - Dấu phẩy tĩnh; Dấu phẩy động

Cú pháp

scanf(xau_dinh_dang[,DS_dia_chi]);

Cú pháp

scanf(xau_dinh_dang [, DS_dia_chi]);

Xau_dinh_dang:

- Gồm các ký tự được qui định cho từng loại dữ liệu được nhập vào.
  - *Ví dụ: dữ liệu định nhập kiểu nguyên thì xâu định dạng là : %d*

DS_dia_chi:

- Bao gồm địa chỉ của các biến (*toán tử* ***&***), phân tách nhau bởi dấu phẩy (**,**)
- Phải phù hợp với các kí tự định dạng trong **xau_dinh_dang** về số lượng, kiểu, thứ tự

Hoạt động

- Đọc các ký tự được gõ vào từ bàn phím
- Căn cứ vào xâu định dạng, chuyển thông tin đã nhập sang kiểu dữ liệu phù hợp
- Gán những giá trị vừa nhập vào các biến tương ứng trong DS_dia_chi

Ví dụ:

int a;

scanf("%d",&a); →1234_ → a = 1234

Ghi chú

Thông tin được gõ vào từ bàn phím, được lưu ở vùng đệm trước khi được xử lý bởi hàm scanf()→Hàm scanf() đọc từ vùng đệm

```
#include <stdio.h>
int main(){
    int a, b;
    scanf("%d",&a);
    scanf("%d",&b);
    printf ("%d %d", a, b);
    return 0;
}
```

```
123     456_
```

```
123 456
```

Các ký tự định dạng

Kí tự	Khuôn dạng dữ liệu nhập
%c	Đọc kí tự đơn lẻ
%d	Đọc số thập phân
%o	Đọc số bát phân
%x	Đọc số hexa
%u	Đọc số thập phân không dấu

Các ký tự định dạng

Kí tự	Chú thích
%s	Đọc xâu kí tự tới khi gặp dấu phân cách
%f	Đọc số thực dấu phẩy tĩnh (float)
%ld	Đọc số nguyên kiểu long
%lf	Đọc số thực dấu phẩy tĩnh (double)
%e	Đọc số thực dấu phẩy động
%%	Đọc ký tự %

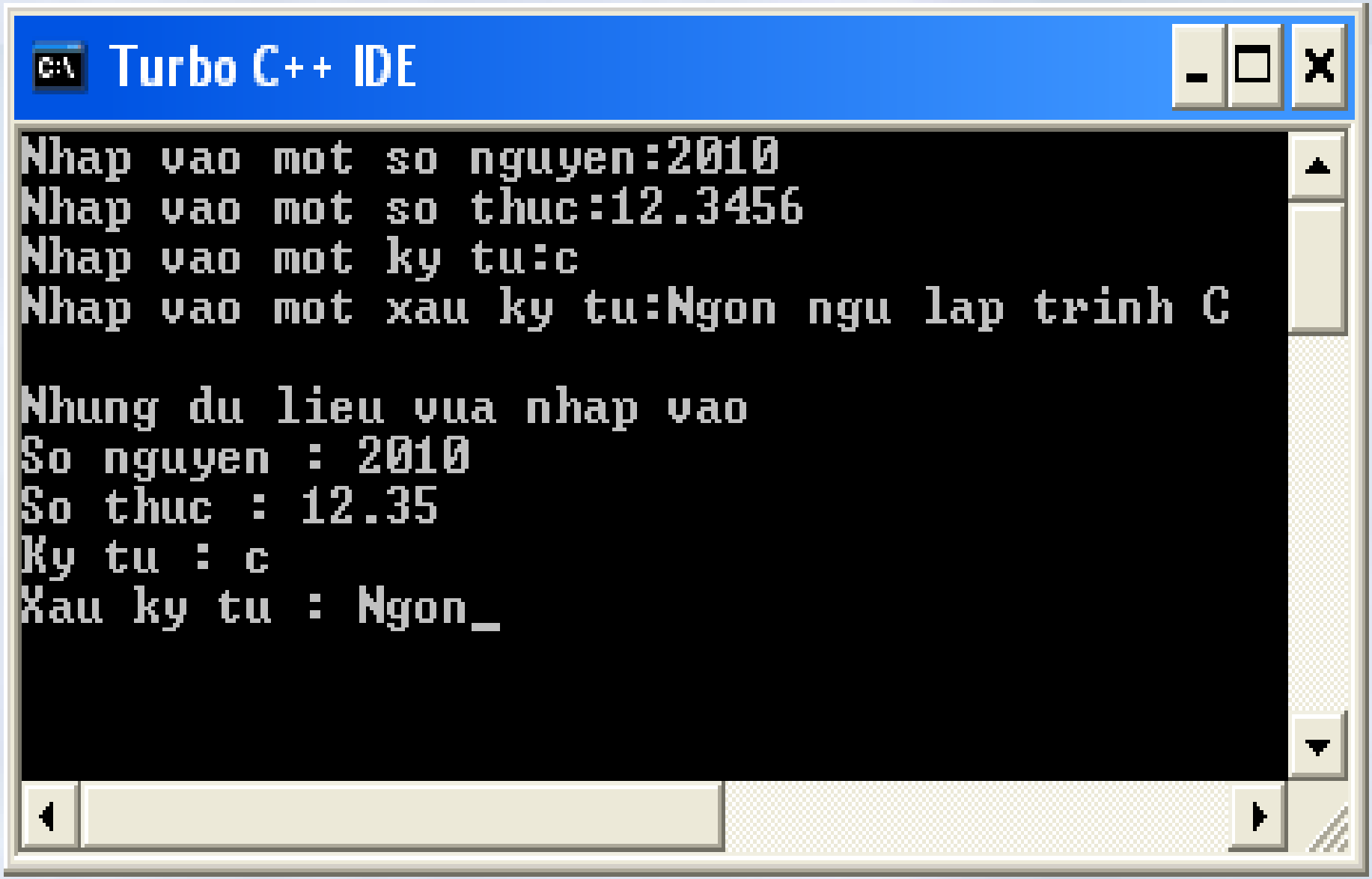
Ví dụ

```
#include <conio.h>
#include  <stdio.h>
void main(){
    // khai bao bien
    int a; float x;
    char ch; char str[30];
    // Nhap du lieu
    printf("Nhap vao mot so nguyen:"); scanf("%d",&a);
    printf("\nNhap vao mot so thuc:"); scanf("%f",&x);
    printf("\n Nhap vao mot ki tu:");
    fflush(stdin); scanf("%c",&ch);
```

Ví dụ

```
    printf("\nNhap vao mot xau ki tu:");
    fflush(stdin); scanf("%s",str);

    // Hien thi du lieu vua nhap vao
    printf("\nNhung du lieu vua nhap vao");
    printf("\nSo nguyen : %d",a);
    printf("\nSo thuc : %5.2f",x);
    printf("\nKy tu : %c",ch);
    printf("\nXau ky tu : %s",str);
    getch();
}
```

Ví dụ→Kết quả thực hiện



Các quy tắc cần lưu ý

Khi đọc số

- Hàm scanf() quan niệm rằng mọi kí tự số, dấu chấm (‘.’) đều là kí tự hợp lệ.
  - Số thực dấu phẩy động, chấp nhận ký tự e/E
- Khi gặp các dấu phân cách như tab, xuống dòng hay dấu cách (space bar), scanf() sẽ hiểu là kết thúc nhập dữ liệu cho một số

Các quy tắc cần lưu ý

Khi đọc kí tự

Hàm **scanf()** cho rằng mọi kí tự có trong bộ đệm của thiết bị vào chuẩn đều là hợp lệ, kể cả các kí tự tab, xuống dòng hay dấu cách

Khi đọc xâu kí tự:

Hàm **scanf()** nếu gặp các kí tự dấu trắng, dấu tab hay dấu xuống dòng thì nó sẽ hiểu là kết thúc nhập dữ liệu cho một xâu kí tự.

Ghi chú:

Trước khi nhập dữ liệu kí tự hay xâu kí tự nên dùng lệnh **fflush(stdin)** để xóa bộ đệm.

Ví dụ: Đọc 2 số nguyên, đưa ra tổng, hiệu, tích…

```
#include <stdio.h>
int main(){
    int A, B;
    printf("Nhap vao 2 so nguyen : "); scanf("%d %d",&A,&B);
    printf("\n");
    printf("Tong %d + %d = %d \n", A, B, A + B);
    printf("Hieu %d - %d = %d\n", A, B, A - B);
    printf("Tich %d x %d = %d\n", A, B, A * B);
    printf("Thuong %d / %d = %.3f\n", A, B, (float)A / B);
    printf("Chia nguyen %d / %d = %d\n", A, B, A / B);
    printf("Chia du %d  %% %d = %d\n", A, B, A % B);
    printf("\n");
    return 0;
}
```

Ví dụ: Đọc 2 số nguyên, đưa ra tổng, hiệu, tích…

```
Nhap vao 2 so nguyen : 17 5

Tong 17 + 5 = 22
Hieu 17 - 5 = 12
Tich 17 x 5 = 85
Thuong 17 / 5 = 3.400
Chia nguyen 17 / 5 = 3
Chia du 17 % 5 = 2
```

Bài tập

1. Viết chương trình nhập vào từ bàn phím chiều dài 3 cạnh của một tam giác, rồi đưa ra diện tích và các đường cao của tam giác
2. Nhập vào từ bàn phím tọa độ 3 điểm A,B,C rồi đưa ra độ dài các cạnh của tam giác ABC và của đường trung tuyến AM
3. Cho hàm số: $f(x) = x^7 + 5\sqrt[3]{x^5 + 3x^3 + 2} + 12$

   Viết chương trình nhập vào 3 số thực a,b,c và đưa ra trung bình cộng của f(a),f(b),f(c)
4. Nhập x vào từ bàn phím và tính giá trị của biểu thức $A = \dfrac{\cos 3a + \sqrt[5]{2x^3 + x + 1}}{\log_7(3^{x^2} + 2.14b)}$ trongđó $a = \sqrt{2^x + \pi}$ và $b = \ln(e^{x+1.23} + 1)$

Ví dụ: Đọc tọa độ 3 điểm A,B,C và đưa ra d/tích ΔABC

```
#include  <stdio.h>
#include <math.h>
int main(){
      float Ax,Ay, Bx, By, Cx, Cy, AB, BC, CA,p;
      printf("Nhap vao toa do diem A  : "); scanf("%f %f",&Ax,&Ay);
      printf("Nhap vao toa do diem B  : "); scanf("%f %f",&Bx,&By);
      printf("Nhap vao toa do diem C  : "); scanf("%f %f",&Cx,&Cy);
      //Tinh do dai cac canh cua tam giac
      AB = sqrt((Ax-Bx)*(Ax-Bx)+(Ay-By)*(Ay-By));
      BC = sqrt((Bx-Cx)*(Bx-Cx)+(By-Cy)*(By-Cy));
      CA = sqrt((Cx-Ax)*(Cx-Ax)+(Cy-Ay)*(Cy-Ay));
      p = (AB +BC +CA)/2;
      printf("Dien tich tam giac ABC la: %f",sqrt(p*(p-AB)*(p-BC)*(p-CA)));
      printf("\n");
      return 0;
}
```

Ví dụ: Đọc tọa độ 3 điểm A,B,C và đưa ra d/tích ΔABC

```
Nhap vao toa do diem A  : 0 0
Nhap vao toa do diem B  : 6 0
Nhap vao toa do diem C  : 0 8
Dien tich tam giac ABC la: 24.000000
```

```
Nhap vao toa do diem A  : -1 -2
Nhap vao toa do diem B  : 5 -2
Nhap vao toa do diem C  : 6 6
Dien tich tam giac ABC la: 24.000000
```

```
Nhap vao toa do diem A  : 1 1
Nhap vao toa do diem B  : 2 2
Nhap vao toa do diem C  : 4 4
Dien tich tam giac ABC la: 0.000000
```

Bài tập tại lớp

1. Viết chương trình nhập vào từ bàn phím bán kính một đường tròn và đưa ra màn hình diện tích và chu vi đường tròn
2. Viết chương trình nhập vào từ bàn phím một giá trị thực. Hãy đưa ra diện tích của các hình tròn, vuông, tam giác đều có chu vi bằng giá trị vừa nhập.

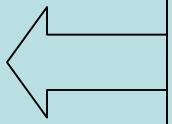
Ghi chú:

1. Giả thiết π = 3.1416. Cần khai báo hằng PI trong chương trình.
2. π là hằng số được khai báo trong tệp tiêu đề **math.h** và có tên là **M_PI**

Nội dung chính

1. Các hàm vào ra cơ bản:
   - **printf()**
   - **scanf()**

2. Các hàm vào ra khác
   - **gets()**
   - **puts()**
   - **getch()**



Cần nạp thư viện conio.h

#include <conio.h>

gets()

- Mục đích:
  - Dùng để nhập vào từ bàn phím một xâu kí tự **bao gồm cả dấu cách**, điều mà hàm **scanf()** không làm được.
- Cú pháp :

  gets (xâu_kí_tự);
- Ví dụ:

```
char str [40];
printf("Nhap vao mot xau ki tu:");
fflush(stdin);
gets(str);
```

puts()

- Mục đích:
  - Hiển thị ra màn hình nội dung xâu_kí_tự và sau đó đưa con trỏ xuống dòng mới
- Cú pháp:

```
puts(xâu_kí_tự);
```

- Ví dụ:

puts(“Nhap vao xau ki tu:”);

Tương đương với lệnh:

printf(“%s\n”,“Nhap vao xau ki tu:“).

getch()

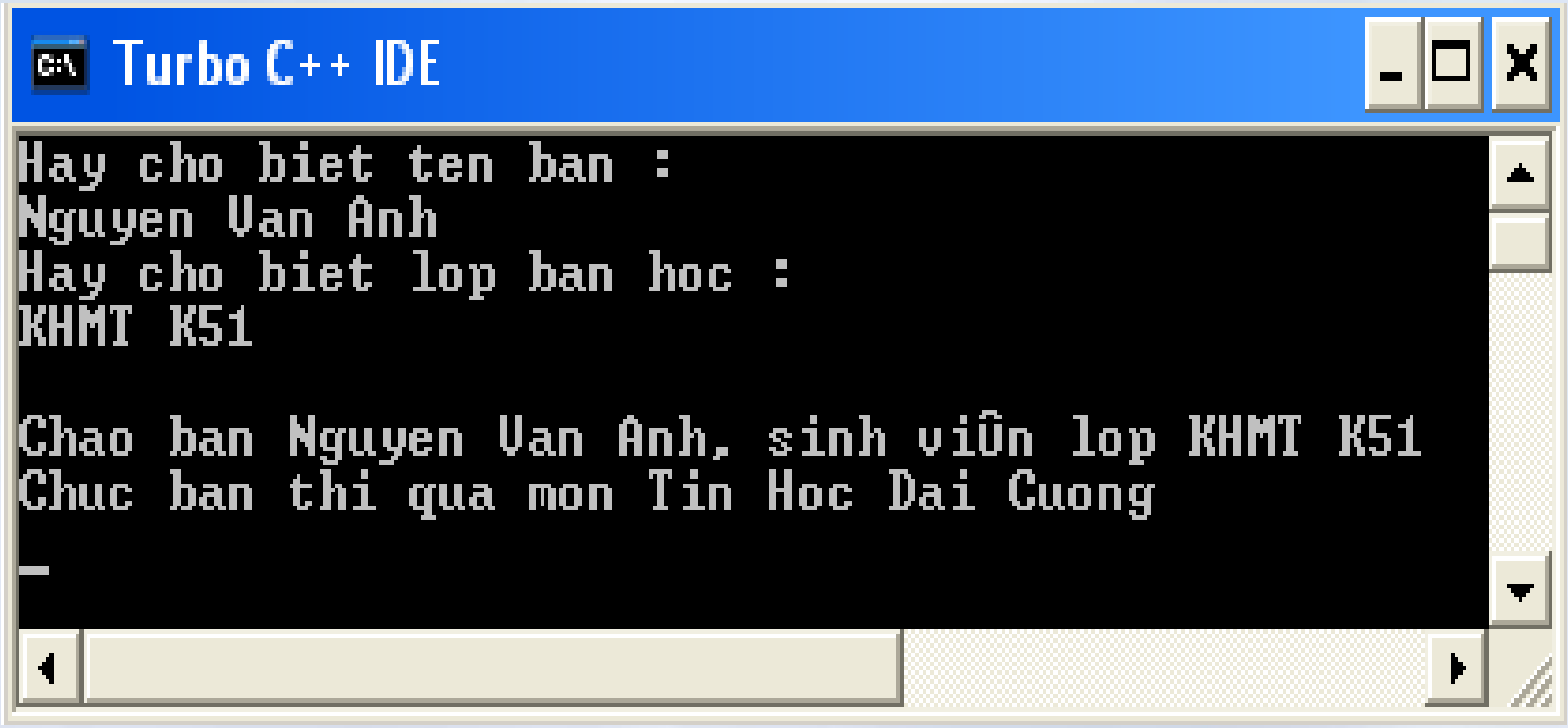
- Mục đích
  - Đợi đọc một ký tự bàn phím
  - Thường dùng để chờ người sử dụng ấn một phím bất kì trước khi kết thúc chương trình.
- Cú pháp

```
getch();
```

Ví dụ

```
#include <conio.h>
#include  <stdio.h>
void main(){
   char ten[30], lop[10]; //Kieu chuoi, mang ky tu
   puts("Hay cho biet ten ban : ");
   fflush(stdin); gets(ten);
   puts("Hay cho biet lop ban hoc : ");
   fflush(stdin); gets(lop);

   printf("\nChao ban %s, sinh viên lop %s\n",ten,lop);
   puts("Chuc ban thi qua mon Tin Hoc Dai Cuong");
   getch();
}
```

Ví dụ → Kết quả thực hiện



Tóm tắt

- Các hàm cơ bản (stdio.h)
  - printf()/ scanf ()
  - Xâu định dạng:
    %[flags][width][.precision][l][t]
    - **flags** (+/-,#): Xác định sự căn lề
    - **l** (l/L): Biến ở dạng **long**
    - **t** (d/i/o/u/x/X/f/e/E/g/G/c/s/%): kiểu hiện thị
- Các hàm khác (conio.h)
  - puts() / gets() / getch()
- Tìm hiểu thêm
  - fprintf() / fscanf() ← Vào/ra từ file
  - sprintf(f) / sscanf() ← Vào ra từ xâu ký tự

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. Cấu trúc lệnh khối
2. Cấu trúc rẽ nhánh
   - Cấu trúc if, if … else
   - Cấu trúc lựa chọn switch
3. Cấu trúc lặp
   - Vòng lặp **for**
   - Vòng lặp **while** và **do while**
4. Các lệnh thay đổi cấu trúc lập trình
   - Câu lệnh **continue**
   - Câu lệnh **break**

Lệnh đơn >< Lệnh ghép

- Lệnh đơn:
  - Là biểu thức theo sau bởi dấu ‘;’
  - ***Ví dụ***: x= 0; i++; printf(“Hello”);
- Lệnh ghép (khối lệnh)
  - Là tập hợp các câu lệnh (*đơn và* ***ghép***) được đặt trong cặp ngoặc nhọn **{ }**
  - **C** cho phép khai báo biến trong một khối lệnh
    - Phần khai báo phải nằm trước các câu lệnh
  - ***Chú ý:***
    - Lệnh ghép có thể đặt tại bất cứ chỗ nào mà cú pháp cho phép đặt 1 câu lệnh đơn
    - Không đặt dấu ‘;’ sau một lệnh khối

Cấu trúc lồng nhau

- Trong lệnh ghép chứa lệnh ghép khác
  - Sự lồng nhau không hạn chế

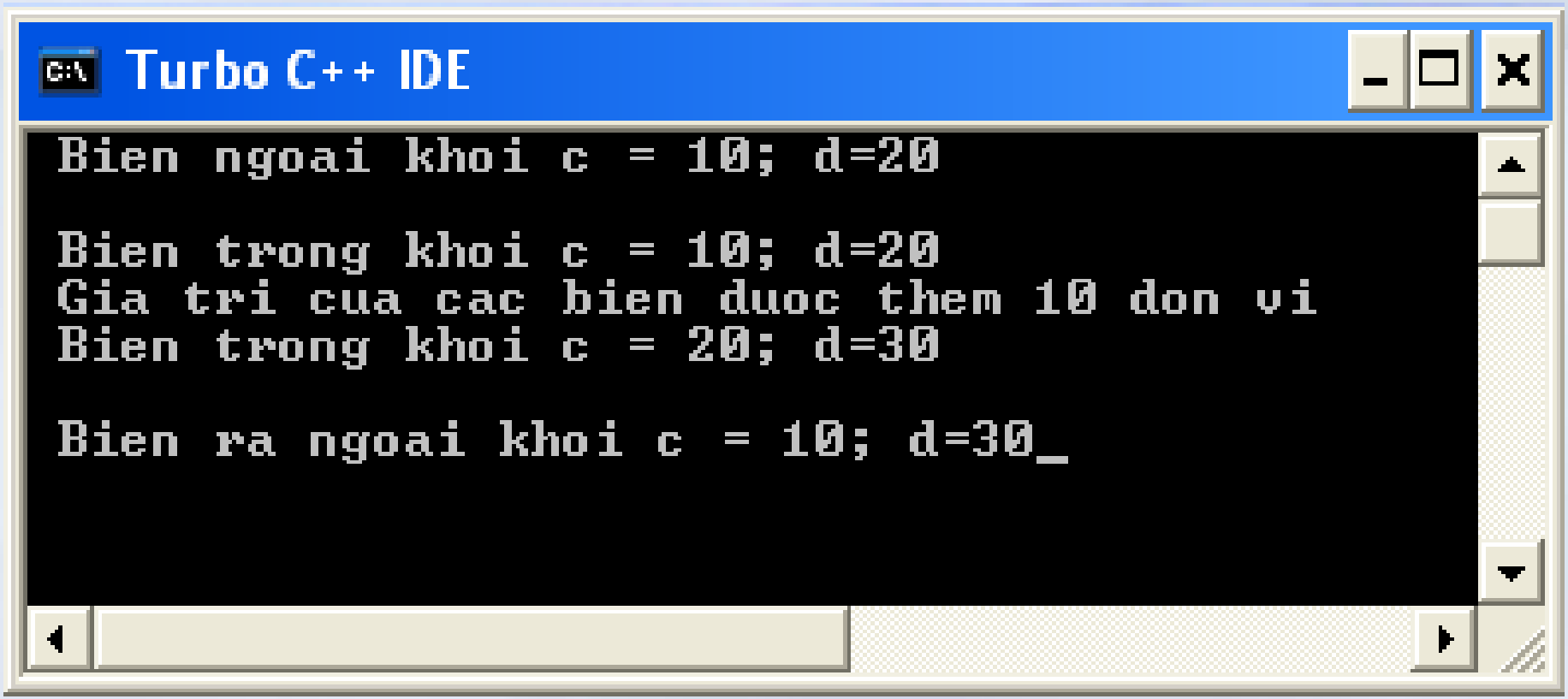
```
{//Khai báo đối tượng cục bộ trong khối
    lenh;
    {//Khai báo đối tượng cục bộ trong khối
        lenh;
        …
    }
    …
}
```

Nếu các đối tượng được khai báo trùng tên nhau ?

Ví dụ

```
#include <conio.h>
#include <stdio.h>
void main(){   // ham main() cung la mot khoi lenh
    int  c = 10,  d= 20;
    printf(" Bien ngoai khoi c = %d; d=%d ",c,d);
    {      int  c = 10; int d=5;
           printf("\n Bien trong khoi c = %d; d=%d",c,d);
           printf("\n Gia tri cua cac bien duoc them 10 don vi");
           c = c + 10; d= d + 10;
           printf("\n Bien trong khoi c = %d; d=%d",c,d);
    }
     printf("\n Bien ra ngoai khoi c = %d; d=%d",c,d);
    getch();
} //ke tthuc khoi lenh cua ham main()
```

Ví dụ→Kết quả thực hiện

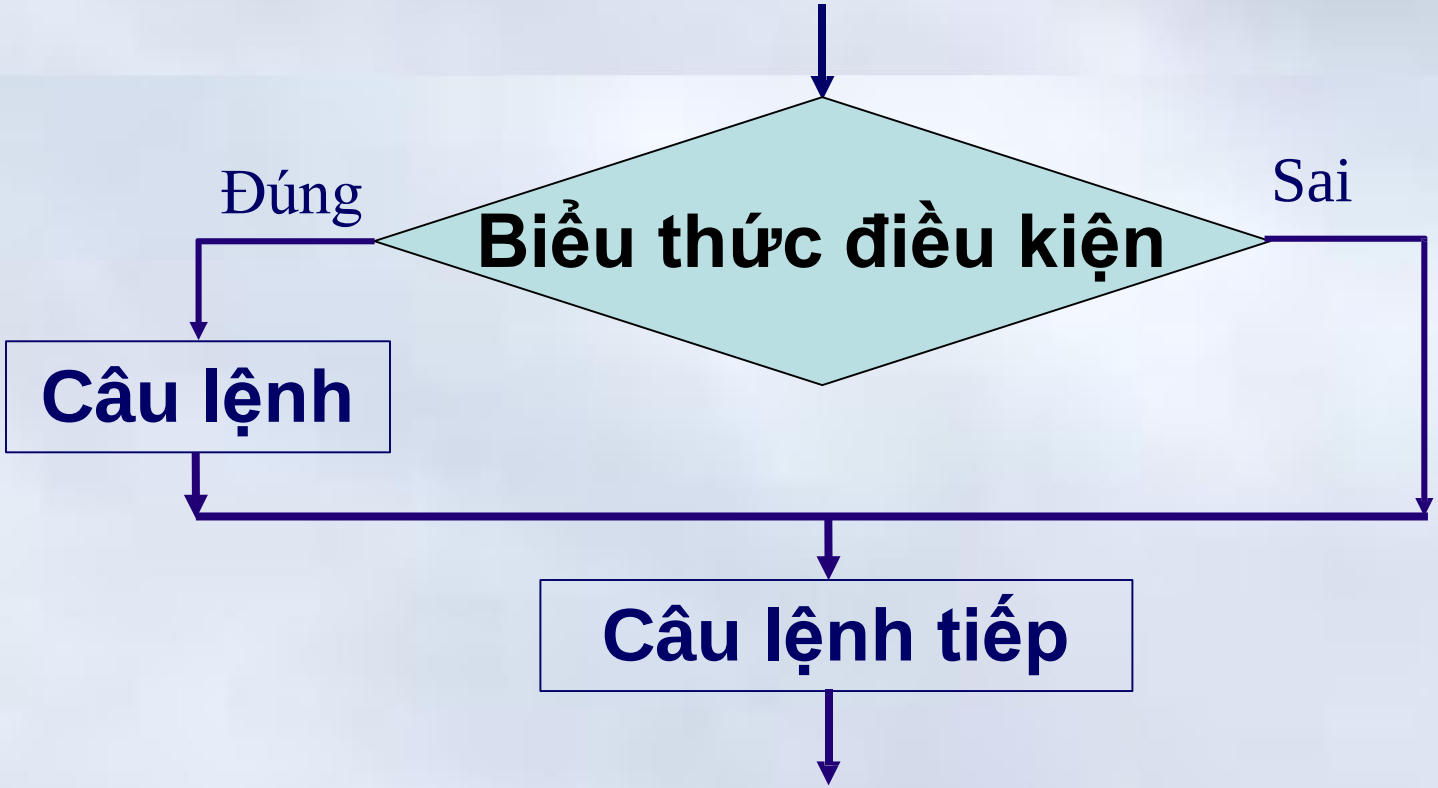


Biến địa phương / Biến toàn cục

Nội dung chính

1. Cấu trúc lệnh khối
2. Cấu trúc rẽ nhánh
   - Cấu trúc if, if … else
   - Cấu trúc lựa chọn switch
3. Cấu trúc lặp
   - Vòng lặp **for**
   - Vòng lặp **while** và **do while**
4. Các lệnh thay đổi cấu trúc lập trình
   - Câu lệnh **continue**
   - Câu lệnh **break**

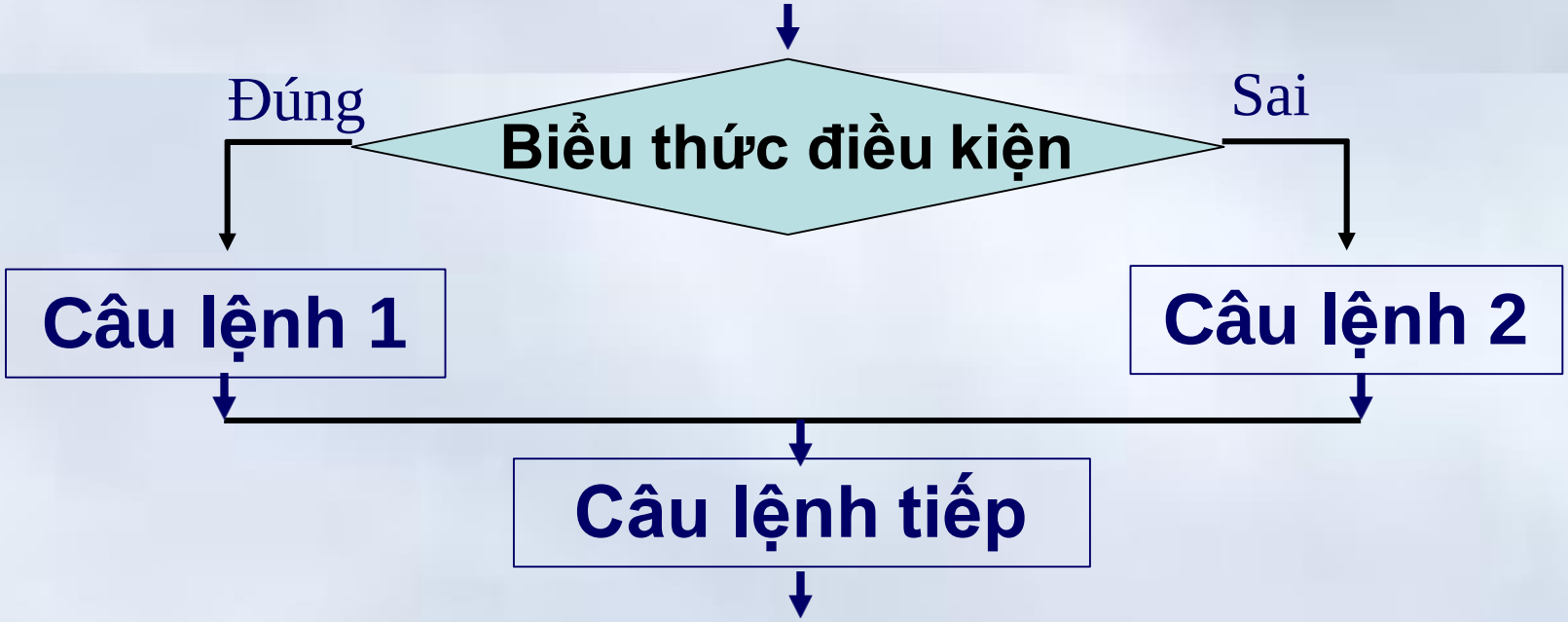
Cấu trúc if …




```
if (Biểu thức điều kiện)
    Câu lệnh;
Câu lệnh kế tiếp;
```

```
if (n % 2 == 0)
    printf("so chan");
```

Cấu trúc **if….else….**




```
if (Biểu thức điều kiện)
    Câu lệnh 1;
else
    Câu lệnh 2;
Câu lệnh kế tiếp;
```

```
if (x > y)
    z = x;
else
    z = y;
printf("max: %d", z);
```

Lưu ý

Biểu thức điều kiện:

- Là biểu thức trả về giá trị logic đúng/sai
  - Giá trị logic đúng/True : khác **0**
- Giá trị logic sai/False: bằng **0**

Ví dụ

if (2+5) printf(“Hello world! ”);→Chấp nhận

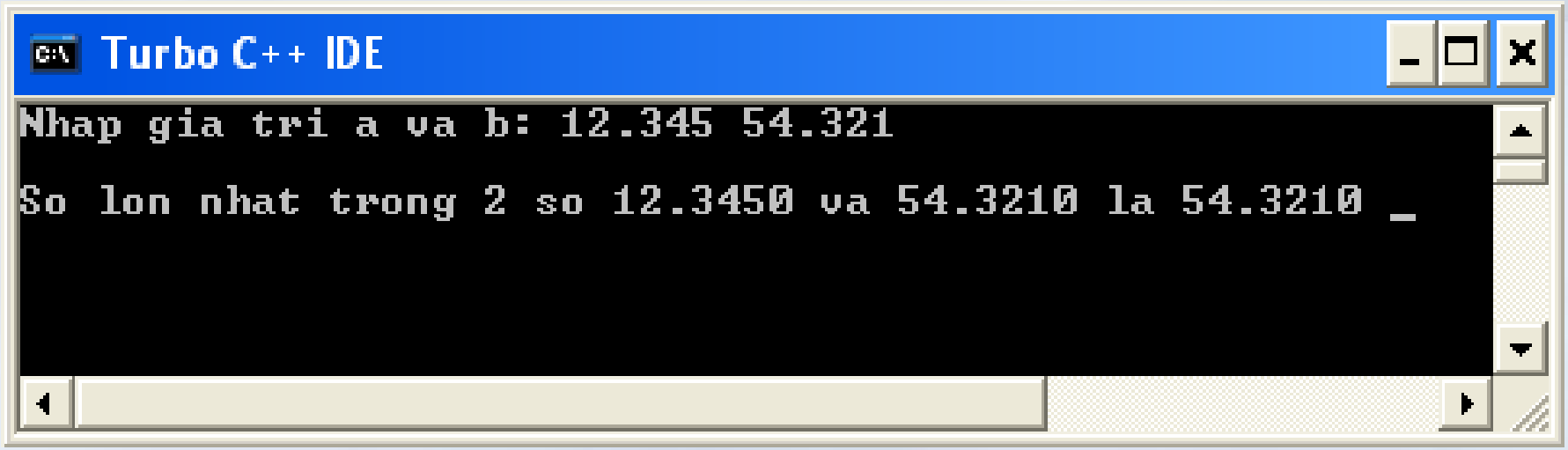
Câu lệnh:

Có thể là một lệnh khối (*Đặt trong cặp { }*)

Ví dụ: So sánh 2 số thực được nhập vào

```
#include <conio.h>
#include  <stdio.h>
void main()
{
    float a, b; float max; // khai bao bien
    printf(" Nhap gia tri a va b: ");
    scanf("%f %f",&a,&b);
    if(a < b)
          max = b;
    else
          max = a;
    printf("\nSo lon nhat trong 2 so %.4f va %.4f la %.4f ",a,b,max);
    getch();
} //ket thuc ham main()
```

Ví dụ: So sánh 2 số thực được nhập vào




```
if(a < b)
  max = b;
else
  max = a;
```

⟺

```
max = a > b ? a : b;
```

Ví dụ: Giải phương trình **ax + b = 0**

```
#include  <stdio.h>
#include <conio.h>
void main()
{ float a, b;
  printf("\nGiai phuong trinh bac nhat ax + b = 0");
  printf("\nCho biet he so a b : ");  scanf("%f%f", &a, &b);
  if (a==0)
        if (b!=0)
              printf("Phuong trinh vo nghiem");
        else
              printf("Phuong trinh vo so nghiem");
  else
        printf("Dap so cua phuong trinh tren = %f", -b/a);
  getch();
} //ket thuc ham main()
```

Giải phương trình **ax + b = 0** → Thực hiện

```
Giai phuong trinh bac nhat ax + b = 0
Cho biet he so a b : 3 17
Dap so cua phuong trinh tren = -5.666667
```

```
Giai phuong trinh bac nhat ax + b = 0
Cho biet he so a b : 0 0
Phuong trinh vo so nghiem
```

```
Giai phuong trinh bac nhat ax + b = 0
Cho biet he so a b : 0 6
Phuong trinh vo nghiem
```

```
Giai phuong trinh bac nhat ax + b = 0
Cho biet he so a b : 5 -123
Dap so cua phuong trinh tren = 24.600000
```

Ví dụ: Nhập x và tính hàm

$$f(x) = \begin{cases} x^2 + \sin^4 2\pi x + 1 & \text{khi } x < 3 \\ 5 & \text{khi } x = 3 \\ \sqrt{x-3} + \log_{10}(x^2 - 3) & \text{khi } x > 3 \end{cases}$$

```
#include <stdio.h>
#include <math.h>
void main()
{
    float x, fx;
    printf("\nNhap x: "); scanf("%f",&x);
    if(x < 3)
          fx = x*x+pow(sin(2*M_PI*x),4)+1;
    else
       if (x==3)
            fx = 5;
       else
            fx = sqrt(x-3) +log10(x*x-3);
    printf("\n Ket quả: %.4f",fx);
}
```

```
Nhap x: 1.0
 Ket qua: 2.0000
```

```
Nhap x: 3
 Ket qua: 5.0000
```

```
Nhap x: 5.0
 Ket qua: 2.7566
```

Cấu trúc **if / if… else** lồng nhau

Cấu trúc **if..** và **if …else** có thể lồng nhau

– Khi đó **else sẽ** tương ứng với **if** (*phía trên, chưa có **else***) gần nhất

```
if (đ/k_1)
   if (đ/k_2)
      lệnh_1;
else
      lệnh_2;
```

```
if (đ/k_1)
  {  if (đ/k_2)
        lệnh_1;
  }
else
      lệnh_2;
```

Cấu trúc **if / if… else** lồng nhau →Ví dụ

int a, b, c = 10;

```
if (a==0)
    if (b == 0)
        c = 20;
else
    c = 30;
```

a ≠ 0, b=? → c = 10

a=0, b=0 →c = 20

a=0, b ≠ 0 →c = 30

```
if (a==0){
    if (b == 0)
        c = 20;
}else
    c = 30;
```

a ≠ 0, b=? → c = 30

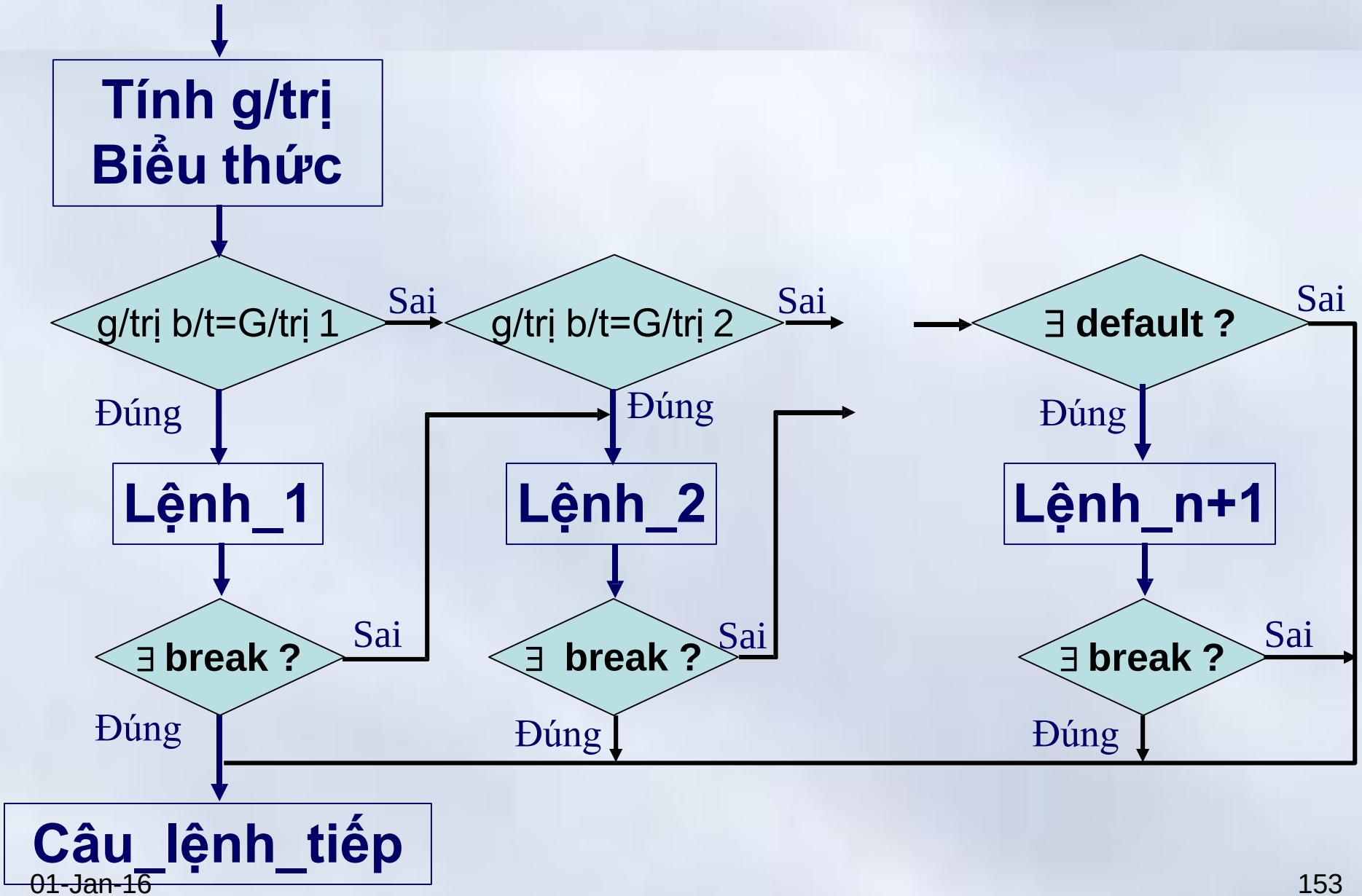
a=0, b=0 →c = 20

a=0, b ≠ 0 → c = 10

Cấu trúc lựa chọn **switch**

```
switch (bieu_thuc)
   {
      case gia_tri_1: lenh_1; [break];
      case gia_tri_2: lenh_2; [break];
      …
      [default: lenh_n+1; [break];]
   }
Câu_lệnh_tiếp
```

Cấu trúc lựa chọn **switch**



Cấu trúc lựa chọn **switch**

Cơ chế hoạt động

- Tính giá trị của **biểu_thức**,
- So sánh giá trị của **biểu_thức** với các **giá_trị_k** (*với k = 1, 2, … n*) nằm sau các từ khóa **case**
  - Xảy ra 2 khả năng

Cấu trúc lựa chọn **switch**→cơ chế hoạt động

- Tồn tại ***giá_trị_i*** bằng giá trị biểu thức.
  - Thực hiện ***lệnh_i***
    - Nếu tồn tại lệnh ***break***,
      - Nhảy tới tiếp tục thực hiện ***Câu_lệnh_tiếp*** nằm sau cấu trúc ***switch***
    - Nếu không tồn tại lệnh ***break***
      - Thực hiện các lệnh sau ***lệnh_i*** cho tới khi gặp ***break*** hoặc tới khi thoát khỏi cấu trúc ***switch***
      - Thực hiện ***Câu_lệnh_tiếp***

Cấu trúc lựa chọn **switch**→cơ chế hoạt động

- Không tồn tại ***giá_trị_i*** (*i = 1, 2, …n*) nào bằng giá trị biểu thức
  - Nếu có nhãn **default**:
    - Chương trình sẽ thực hiện ***lệnh_n+1***
    - Thực hiện ***Câu_lệnh_tiếp*** nằm ngay sau cấu trúc **switch**.
  - Nếu không có nhãn **default**:
    - Chương trình chuyển sang thực hiện lệnh tiếp theo nằm ngay sau cấu trúc **switch**: ***Câu_Lệnh_tiếp***

Cấu trúc lựa chọn **switch**→Ví dụ 1

Lập trình đọc từ bàn phím một số nguyên $1 \leq N \leq 10$ và đưa ra từ tiếng Anh tương ứng

Cấu trúc lựa chọn **switch**→Ví dụ 1

```
#include <stdio.h>
void main(){
    int N;
    printf("\nNhap mot gia tri so nguyen khong am: "); scanf("%d",&N);
    switch(N)  {
        case 1: printf(" %d -> One \n",N); break;
        case 2: printf("%d -> Two \n",N); break;
        case 3: printf("%d -> Three \n",N); break;
        case 4: printf("%d -> Four \n",N); break;
        case 5: printf("%d -> Five \n",N); break;
        case 6: printf("%d -> Six \n",N); break;
        case 7: printf("%d -> Seven \n",N); break;
        case 8: printf("%d -> Eight \n",N); break;
        case 9: printf("%d -> Nine \n",N); break;
        case 10: printf("%d -> Ten \n",N); break;
        default : printf("Khong thoa man dieu kien [1..10] \n");
    }
}
```

Cấu trúc lựa chọn **switch**→ Thực hiện

```
Nhap mot gia tri so nguyen khong am: 7
7 -> Seven
```

```
Nhap mot gia tri so nguyen khong am: 3
3 -> Three
```

```
Nhap mot gia tri so nguyen khong am: -6
Khong thoa man dieu kien [1..10]
```

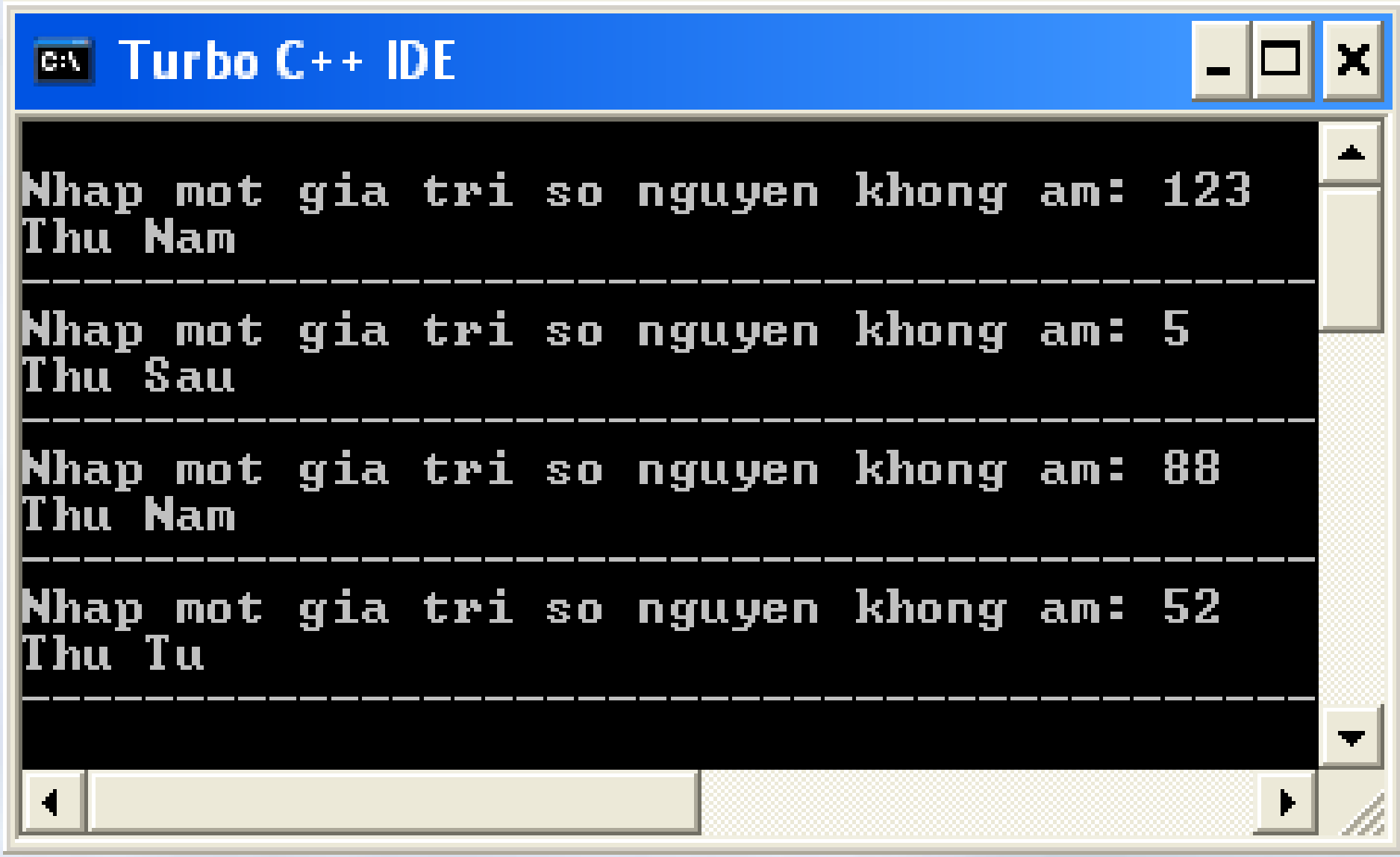
Cấu trúc lựa chọn **switch**→Ví dụ 2

Nhập vào số nguyên không âm, đưa ra ngày trong tuần tương ứng (*theo số dư khi chia cho 7*).

Cấu trúc lựa chọn **switch**→Ví dụ 2

```
#include <conio.h>
#include  <stdio.h>
void main(){
    int a;
    printf("\nNhap mot gia tri so nguyen khong am: "); scanf("%d",&a);
    switch(a % 7) {
          case 0: printf(" Chu nhat"); break;
          case 1: printf(" Thu Hai"); break;
          case 2: printf(" Thu Ba"); break;
          case 3: printf(" Thu Tu"); break;
          case 4: printf(" Thu Nam"); break;
          case 5: printf(" Thu Sau"); break;
          case 6: printf(" Thu Bay"); break;
    }
    getch();
}
```

Cấu trúc lựa chọn **switch** →Thực hiện



Cấu trúc lựa chọn **switch**

Có thể sử dụng đặc điểm Không có lệnh break chương trình sẽ tự động chuyển xuống thực hiên các câu lệnh tiếp sau *để viết chung mã lệnh cho các trường hợp khác nhau nhưng được xử lý như nhau*

Ví dụ: Trong một năm các tháng có 30 ngày là 4, 6, 9, 11 còn các tháng có 31 ngày là 1, 3, 5, 7, 8, 10, 12. Riêng tháng hai có thể có 28 hoặc 29 ngày. Hãy viết chương trình nhập vào 1 tháng, sau đó đưa ra kết luận tháng đó có bao nhiêu ngày

Cấu trúc lựa chọn **switch**→ Ví dụ

```
#include <conio.h>
#include  <stdio.h>
void main () {
    int thang; clrscr();
    printf("\nNhap vao thang trong nam "); sanf("%d",&thang);
    switch(thang) {
        case 1:
         case 3:
         case 5:
         case 7:
         case 8:
         case 10:
         case 12:  printf("\n Thang %d co 31 ngay ",thang);
                    break;
```

Cấu trúc lựa chọn **switch**

```
        case 4:
        case 6:
        case 9:
        case 11: printf("\n Thang %d co 30 ngay ",thang);
                 break;
        case 2:  printf ("\ Thang 2 co 28 hoac 29 ngay");
                 break;
        default :  printf("\n Khong co thang %d", thang);
                 break;
    }
    getch();
}
```

Cấu trúc lựa chọn **switch→Lưu ý**

- Giá trị của **biểu thức** trong cấu trúc **switch** phải là số nguyên (*kiểu đếm được*)
  - Phải có kiểu dữ liệu là **char**, **int**, **long**
- Các giá trị sau từ khóa **case** (*gia_tri_1, gia_tri_2,..*) cũng phải là số nguyên

Điều kiện trong cấu trúc **if / if..else** cho phép làm việc với các kiểu dữ liệu khác số nguyên

Các ví dụ

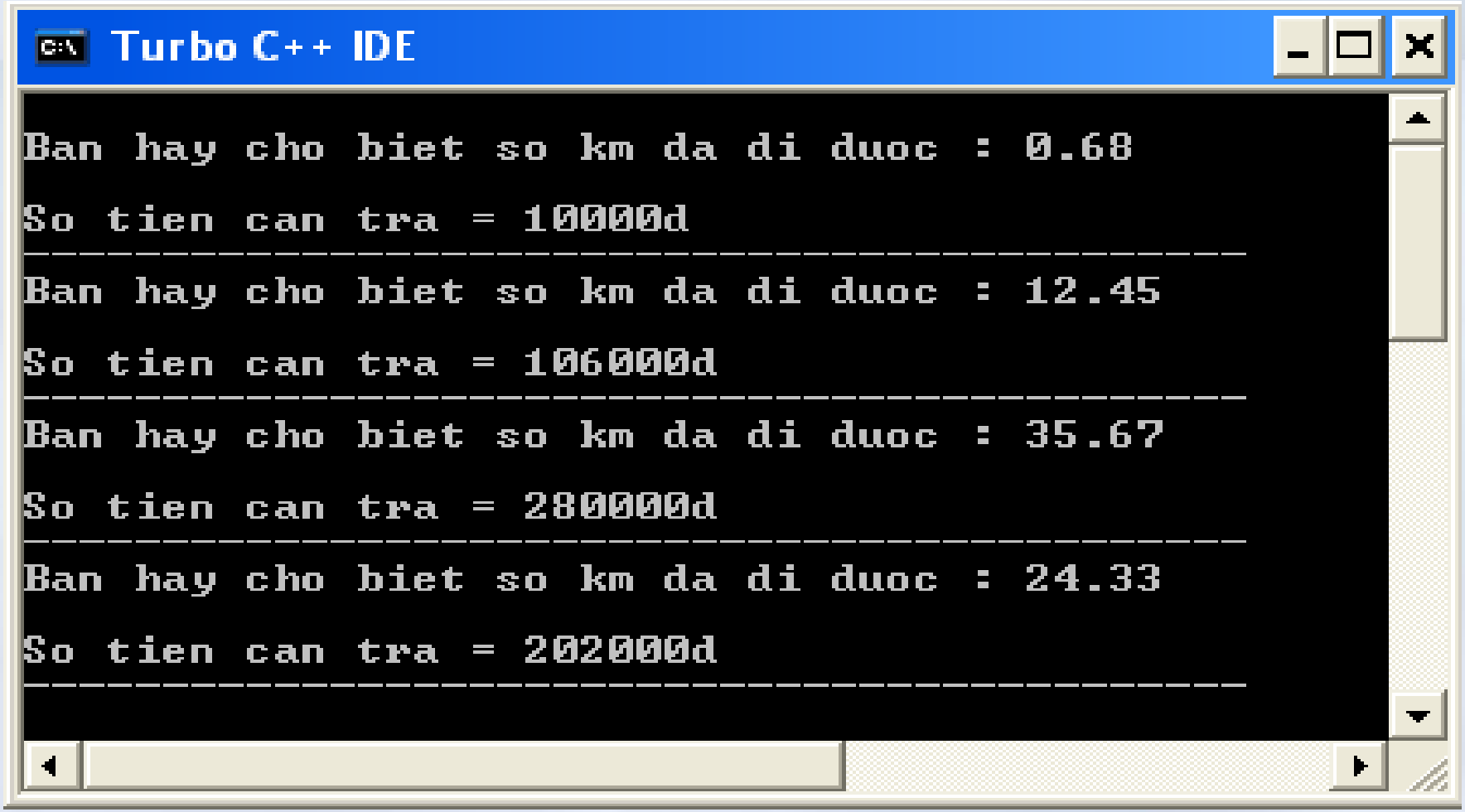
1. Viết chương trình tính cước Taxi theo công thức:
   - 1 km đầu tiên có cước là 10000đ,
   - 30 km tiếp theo có giá là 8000đ/1km
   - Các km sau đó có giá là 6000đ/1km.
2. Viết chương trình giải phương trình bậc hai $ax^2 + bx + c = 0$
3. Viết chương trình giải hệ phương trình bậc nhất

$$\begin{cases} a_1x + b_1y = c_1 \\ a_2x + b_2y = c_2 \end{cases}$$

Ví dụ 1: Tính cước taxi

```
#include <stdio.h>
#include <math.h> //Để sử dụng hàm toán học ceil
void main() {
	unsigned long sotien;
	float sokm;
	printf("\nBan hay cho biet so km da di duoc : ");
	scanf("%f", &sokm);
	if (sokm <= 1.0)
		sotien = 10000;
	else
		if (sokm <= 31.0)
			sotien = 10000 + (ceil(sokm) - 1.0 ) * 8000;
		else
			sotien = 250000 + (ceil(sokm) - 31) * 6000;
	printf("\nSo tien can tra = %lud", sotien);
}
```

Ví dụ 1 → Thực hiện chương trình




```
sotien= sokm <=1.0 ? 10000  : sokm <= 31 ? 10000 + (ceil(sokm) - 1.0 ) * 8000 : 250000+(ceil(sokm) - 31) * 6000;
```

Ví dụ 2: Giải phương trình bậc 2

```
#include <stdio.h>
#include <math.h> //Để sử dụng hàm toán học sqrt
void main(){
   float a, b, c, delta;
   printf("\n\nNhap he so a b c : ");  scanf("%f%f%f", &a, &b, &c);
   delta = b * b - 4 * a * c;
   if( a==0)  printf("P/trinh suy bien thanh p/trinh bac 1 %fx+%f=0",b,c);
   else  if (delta < 0)    printf("Phuong trinh vo nghiem");
   else  if (delta == 0)
       printf("Phuong trinh co nghiem kep x1 = x2 = %f", -b/(2*a));
   else
       printf("Phuong trinh co hai nghiem phan biet\n  x1=%f \n  x2=%f",
           (-b + sqrt(delta))/(2*a), (-b - sqrt(delta))/(2*a) );
}
```

Ví dụ 2→ Thực hiện chương trình

```
Nhap he so a b c : 0 3 2
P/trinh suy bien thanh p/trinh bac 1 3.000000x+2.000000=0
```

```
Nhap he so a b c : 1 2 3
Phuong trinh vo nghiem
```

```
Nhap he so a b c : 1 4 4
Phuong trinh co nghiem kep x1 = x2 = -2.000000
```

```
Nhap he so a b c : 1 -3 2
Phuong trinh co hai nghiem phan biet
  x1 = 2.000000
  x2 = 1.000000_
```

Ví dụ 3: Giải hệ phương trình

```
#include <stdio.h>
void main() {
      float a1,b1,c1,a2,b2,c2,x,y,dx,dy,d;
      printf(“\n\nNhap cac so:\n");
      printf("a1,b1,c1=");scanf("%f%f%f",&a1,&b1,&c1);
      printf("a2,b2,c2=");scanf("%f%f%f",&a2,&b2,&c2);
      d   = a1 * b2 - a2 * b1;
      dx = c1 * b2 - c2 * b1;
      dy = a1 * c2 - a2 * c1;
      if (d != 0) {
          x = dx/d; y = dy/d;
          printf("He PT co nghiem x=%f, y=%f\n",x,y);
      }else
          if (dx==0) printf("He PT co vo so nghiem!\n");
          else printf("He phuong trinh vo nghiem!");
}
```

Ví dụ 3 → Thực hiện chương trình

```
Nhap cac so:
a1,b1,c1 = 3 5 8
a2,b2,c2 = 2 1 9
He PT co nghiem x=5.285714, y=-1.571429
```

```
Nhap cac so:
a1,b1,c1 = 1 2 3
a2,b2,c2 = 1 2 4
He phuong trinh vo nghiem!_
```

```
Nhap cac so:
a1,b1,c1 = 1 2 3
a2,b2,c2 = 2 4 6
He PT co vo so nghiem!
```

Bài tập

1. Viết chương trình nhập vào một ký tự hệ hexa và đưa ra giá trị hệ 10 tương ứng
2. Lập trình đọc tọa độ 4 điểm A,B,C,M rồi kiểm tra xem điểm M nằm trong, nằm trên cạnh hay nằm ngoài tam giác ABC.
3. Lập trình đọc vào từ bàn phím 2 giá trị a, b rồi tính y = 15 x^2+x+7.2 trong đó

$$x = \begin{cases} \dfrac{a+b}{3} & \text{nê'u } a < b \\ 1.5172 & \text{nê'u } a = b \\ \dfrac{a-b}{\sqrt{a^2+b^2}} & \text{nê'u } a > b \end{cases}$$

Nội dung chính

1. Cấu trúc lệnh khối
2. Cấu trúc rẽ nhánh
   - Cấu trúc if, if … else
   - Cấu trúc lựa chọn switch
3. Cấu trúc lặp
   - Vòng lặp **for**
   - Vòng lặp **while** và **do while**
4. Các lệnh thay đổi cấu trúc lập trình
   - Câu lệnh **continue**
   - Câu lệnh **break**

Các cấu trúc lặp

- Vòng lặp **for**
- Vòng lặp **while**
- Vòng lặp **do while**

for

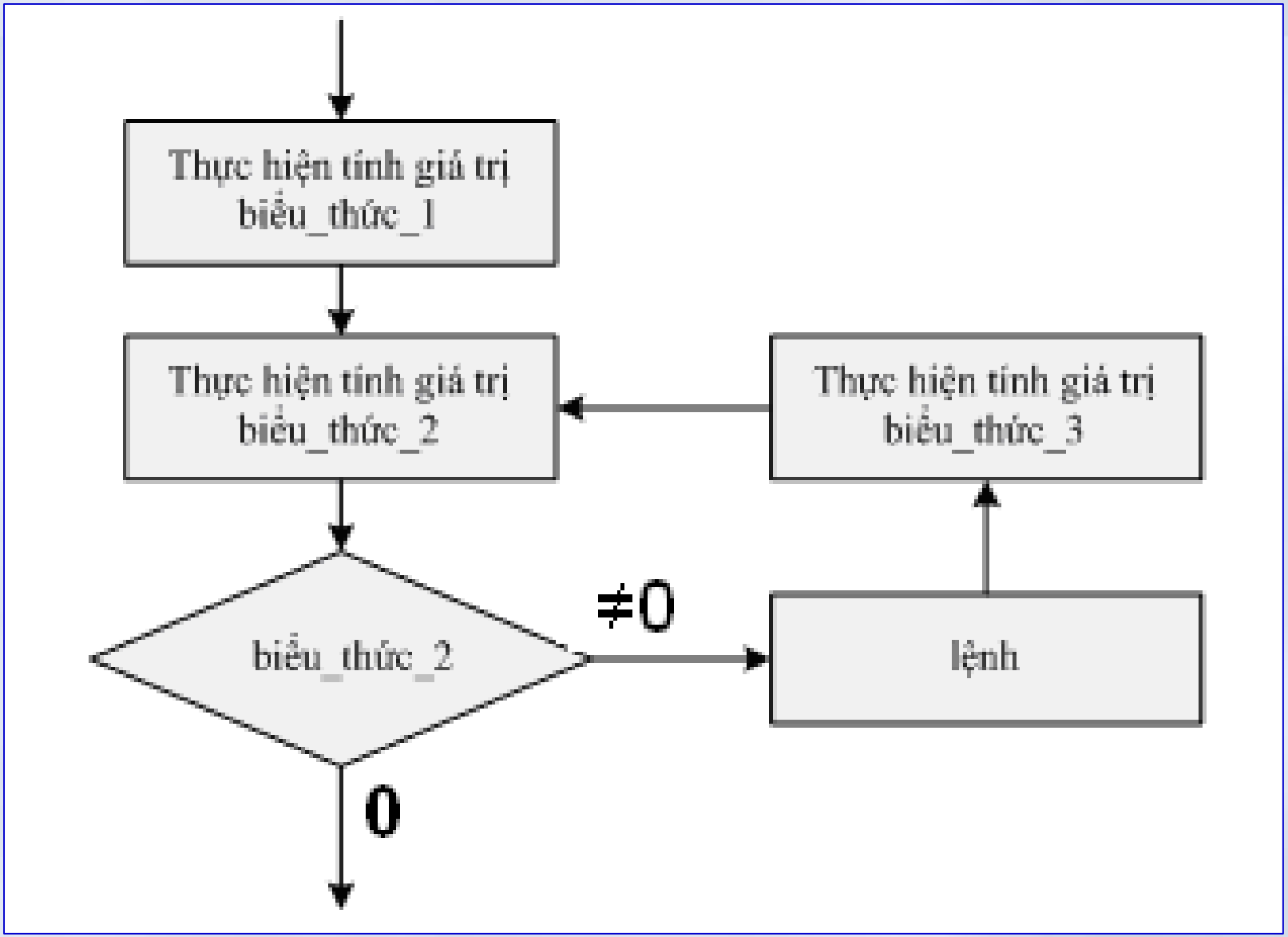
Mục đích và cú pháp

Dùng để lặp công việc một số chính xác lần đã định trước dựa vào sự biến thiên của biến điều khiển

for([b.thuc_1];[b.thuc_2];[b.thuc_3]) Lệnh;

- **b.thuc_1:** Khởi tạo giá trị ban đầu cho vòng lặp
- **b.thuc_2:** Điều kiện tiếp tục vòng lặp
- **b.thuc_3:** Thay đổi biến điều khiển của vòng lặp
- **Lệnh:** Có thể là lệnh đơn lệnh kép hoặc lệnh rỗng

Sơ đồ cú pháp


Thực hiện tính giá trị biểu_thức_1
Thực hiện tính giá trị biểu_thức_2
Thực hiện tính giá trị biểu_thức_3
biểu_thức_2
≠0
lệnh
0

Sử dụng

```
int i;
for(i = 0; i < 100; i++) Câu_lệnh;
```

```
int i;
for(i = 0; i < 100; i+=2)Câu_lệnh;
```

```
int i;
for(i = 100; i > 0; i--) Câu_lệnh;
```

```
for(int i = 0; i < 100; i++) Lệnh;
for(int i = 100; i > 0; i--) Lệnh;
```

Turbo C++ 3.0, văn bản nguồn **.cpp** (**c++**)

Ví dụ 1 : Đưa ra các số nguyên lẻ nhỏ hơn 100

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  void main(){
4.      int i;
5.      for(i = 1;i<100;i++)      {
6.              if(i%2 == 1) printf("%5d",i);
7.              if((i+1)%20 ==0) printf("\n");
8.      }
9.      getch();
10. }
```

```
  1    3    5    7    9   11   13   15   17   19
 21   23   25   27   29   31   33   35   37   39
 41   43   45   47   49   51   53   55   57   59
 61   63   65   67   69   71   73   75   77   79
 81   83   85   87   89   91   93   95   97   99
```

Ví dụ 2 : Đưa ra các số nguyên lẻ nhỏ hơn 100

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  void main(){
4.      int i;
5.      for(i = 99;i > 0;i-=2)      {
6.              printf("%5d",i);
7.              if( (i-1) % 20 == 0) printf("\n");
8.      }
9.      getch();
10. }
```

```
   99   97   95   93   91   89   87   85   83   81
   79   77   75   73   71   69   67   65   63   61
   59   57   55   53   51   49   47   45   43   41
   39   37   35   33   31   29   27   25   23   21
   19   17   15   13   11    9    7    5    3    1
```

Ví dụ 3→ Nhập n và đưa ra n!

```
#include <stdio.h>
#include <conio.h>
void main()
{
  long P = 1;
   int n, i;
   printf("Nhap n : ");scanf("%d",&n);
  for(i = 1;i<=n;i++)
      P = P * i;
   printf("Ket qua là %ld ",P);
  getch();
}
```

```
Nhap n : 6
Ket qua là  720
```

Ví dụ 4 → Nhập n và tính tổng 1 +1/2+..+1/n

```
#include <stdio.h>
#include <conio.h>
void main()
{
  float S = 0.0;
   int n, i;
   printf("Nhap n : ");scanf("%d",&n);
  for(i = 1;i <= n;i++)
     S = S + (float)1/i; //S+=1.0/i;
   printf("Ket qua là %7.4f ",S);
  getch();
}
```

Nhap n : 10
Ket qua là 2.9290

Ví dụ 5→ Tìm số 3 chữ số thỏa mãn $abc=a^3+b^3+c^3$

```
#include  <stdio.h>
#include <conio.h>
void main()
{int i, a, b, c;
    for(i = 100;i<1000;i++){
          a = i / 100;
          b = i % 100 / 10;
          c = i % 100 % 10;
          if(a*a*a+b*b*b+c*c*c ==  i)
               printf("%d \n",i);
    }//for
    getch();
}
```

153
370
371
407

Ví dụ 5→ Tìm số 3 chữ số thỏa mãn $abc=a^3+b^3+c^3$

```
#include  <stdio.h>
#include <conio.h>
void main()
{ int a, b, c;
   for(a  =  1;a<=9;a++)
      for(b = 0;b<=9;b++)
         for(c                =                0;c<=9;c++)
            if(a*a*a+b*b*b+c*c*c==100*a+10*b+c)
               printf("%d \n",100*a+10*b+c);
   getch();
}
```

Chú ý

Không nhất thiết phải có đầy đủ các thành phần trong vòng lặp **for**

int getchar(): đọc ký tự từ vùng đệm bàn phím. Nếu vùng đệm rỗng, đợi người dùng gõ dãy ký tự (*cho tới khi ấn phím Enter*), sẽ trả về ký tự đầu

putchar(int c): đưa ký tự ra màn hình

Chú ý

1. Biểu thức khởi tạo

```
char c; int i=0;
for( ; (c=getchar())! = '\n' ; i++)
    putchar(c);
printf("\nSo ky tu: %d",i);
```

```
Hello world
Hello world
So ky tu: 11
```

2. Biểu thức điều khiển

```
for(i=0 ; ; c=getchar(), i++)
    if(c=='\n') break;
printf("\nSo ky tu: %d",i);
```

```
Hello world

So ky tu: 12
```

3. Thân vòng lặp

```
for(i=0 ; getchar() != '\n', i++);
printf("\nSo ky tu: %d",i);
```

```
Hello world
So ky tu: 11
```

while

Mục đích & Cú pháp

Dùng để thực hiện lặp đi lặp lại một công việc nào đó với số lần lặp **không** xác định.

Cú pháp:

```
while (bieu_thuc_dieu_kien)
    lenh;
```

- Chương trình kiểm tra điều kiện **trước** khi lặp
  - Giá trị của biểu thức điều kiện là đúng ⇒ thực hiện lệnh
- Các **lenh** của vòng lặp có thể không được thực hiện lần nào ⇐Biểu_thức_điều_kiện sai ngay từ đầu
- Biểu_thức_điều_kiện luôn đúng ⇒ lặp vô hạn lần

Sơ đồ cú pháp


Tính toán giá trị của
Biểu thức điều kiên
Lệnh
Biểu thức != 0
Đúng
Sai

Nhập n và đưa tổng của n số nguyên đầu tiên

```
#include <stdio.h>
#include <conio.h>
void main(){
  long S = 0;
   int n;
   printf("Nhap n : ");scanf("%d",&n);
  while (n > 0){
     S = S + n;
     n = n - 1;
   }
   printf("Ket qua là %ld ",S);
  getch();
}
```

```
while (n> 0)
        S += n--;
```

```
Nhap n : 96
Ket qua la 4656
```

Tìm số nguyên lớn nhất thỏa mãn $3n^5-317n < 5$

```
#include  <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
   clrscr();
   int n=0;
   while (3* pow(n,5) - 317*n < 5) n++;
   printf("%4d",n-1);
   getch();
}
```

```
n =   10
while (3*pow(n,5)-317*n >= 5)
        n--;
```

n = 3

Cho biết kết quả thực hiện chương trình

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int i=3;
  while (i > 1){
    if(i % 2==0) i = i / 2;
    else i = i * 3 + 1;
    printf("%4d",i);
   }
  getch();
}
```

10 5 16 8 4 2 1

Nhập chuỗi và đếm số nguyên âm, phụ âm, khoảng trắng

```
>Hello world
Chuoi co :3 nguyen am :7 phu am va 1 khoang trang_
```

```
#include  <stdio.h>
#include <conio.h>
void main()
{   int na, pa, kt;          char c;
    na = pa = kt = 0;
    clrscr(); printf(“>”);
    while( (c=getchar()) !='\n'){
        switch(c){
            case 'a':   case ‘e':  case ‘i':  case ‘o':  case ‘u‘ :
            case ‘A':   case‘E': case ‘I':  case‘0':   case ‘U‘ :  na++;  break;
            case ' ': kt++;  break;
            default : pa++;
        }
    }
    printf("Chuoi co :%d nguyen am :%d phu am va %d khoang  trang",na,pa,kt);
}
```

do..while

Mục đích & Cú pháp

- Dùng để thực hiện lặp đi lặp lại một công việc nào đó với số lần lặp **không** xác định.

- **Cú pháp:**

```
do{
    lenh;
}while(bieu_thuc_dieu_kien);
```

- Chương trình kiểm tra điều kiện **sau** khi lặp
- Các **lenh** được thực hiện ít nhất một lần
- Biểu thức luôn đúng, lặp vô hạn lần

Sơ đồ cú pháp


Lệnh
Tính toán giá trị của
Biểu thức điều kiên
Biểu thức != 0
Đúng
Sai

Nhập n và đưa tổng của n số nguyên đầu tiên

```
#include <stdio.h>
#include <conio.h>
void main(){
  long S = 0;
   int n;
   printf("Nhap n : ");scanf("%d",&n);
  do {
     S = S + n;
      n = n - 1;
   }while (n > 0);
   printf("Ket qua là %ld ",S);
  getch();
}
```

```
do
        S += n--;
while (n> 0);
```

```
Nhap n : 96
Ket qua là 4656
```

Ví dụ

- Nhập vào điểm của một sinh viên, nếu điểm đó không $\in$ [0, 10] thì thông báo cho người dùng nhập lại.
- **Thực hiện**:
  - Nếu dùng lệnh **`if`**

    → Chỉ kiểm tra được 1 lần
  - Sử dụng for

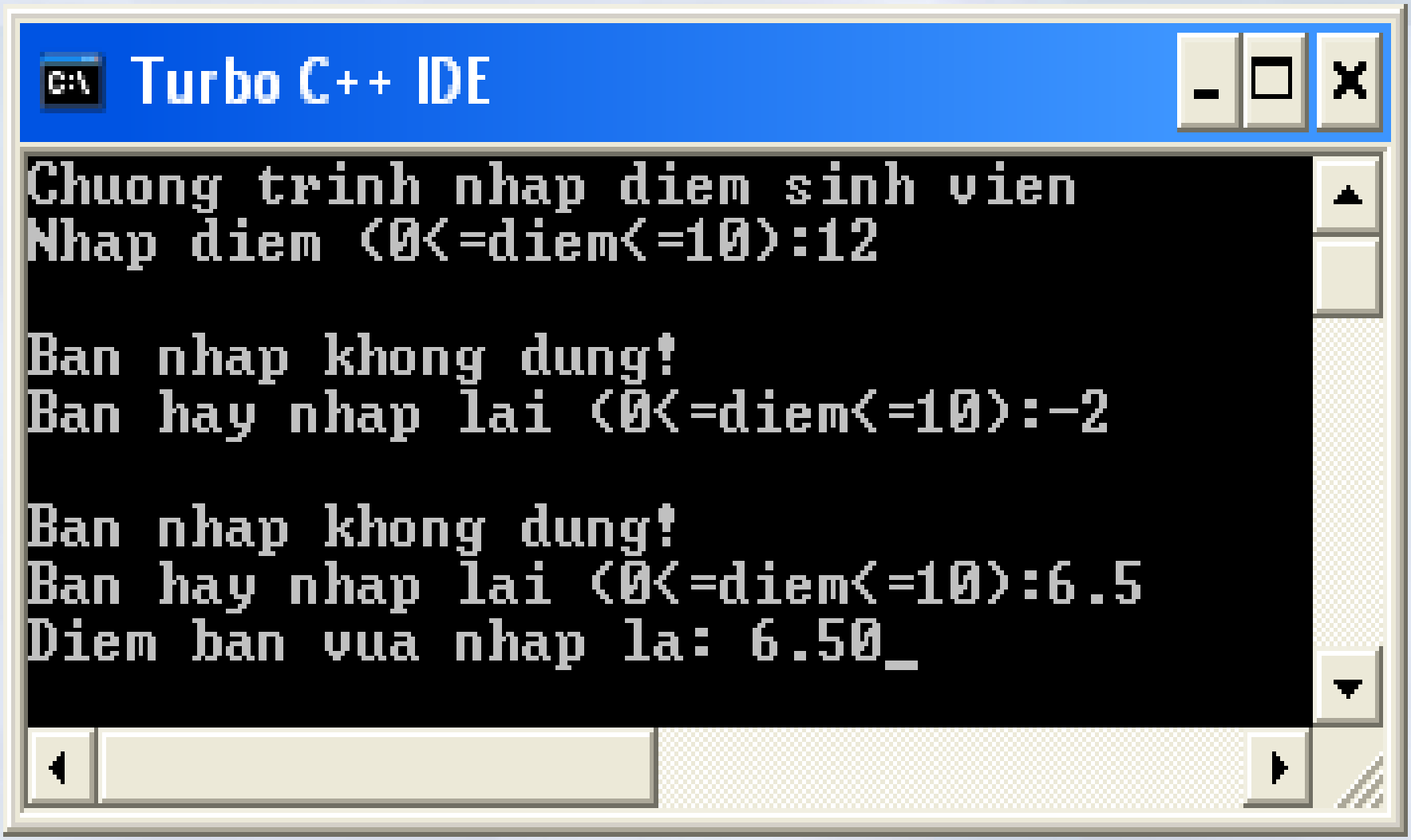
    Chưa biết trước số lần lặp.

  → Sử dụng vòng lặp không cần xác định trước số lần lặp: **while / do while**

Dùng vòng lặp **while**

```
#include <stdio.h>
void main(){
   float diem; clrscr();
   printf(“Chuong trinh nhap diem sinh vien\n");
   printf("Nhap diem (0<=diem<=10):");  scanf("%f",&diem);
   while (diem < 0 || diem > 10)   {
       printf("\nBan nhap khong dung!\n");
       printf("Ban hay nhap lai (0<=diem<=10):");
       scanf("%f",&diem);
   }
   printf("\nDiem ban vua nhap la: %.2f“, diem);
}
```

Dùng vòng lặp **while** → Kết quả



Dùng vòng lặp **do…while**

```
#include <stdio.h>
void main() {
   float diem; clrscr();
   printf(“Chuong trinh nhap diem sinh vien\n");
   do {
       printf("Nhap diem (0<=diem<=10):");
       scanf("%f",&diem);
       if (diem < 0 || diem > 10)
            printf("\nBan nhap khong dung!\n");
   } while (diem < 0 || diem > 10);
   printf("\nDiem ban vua nhap la: %.2f“, diem);
}
```

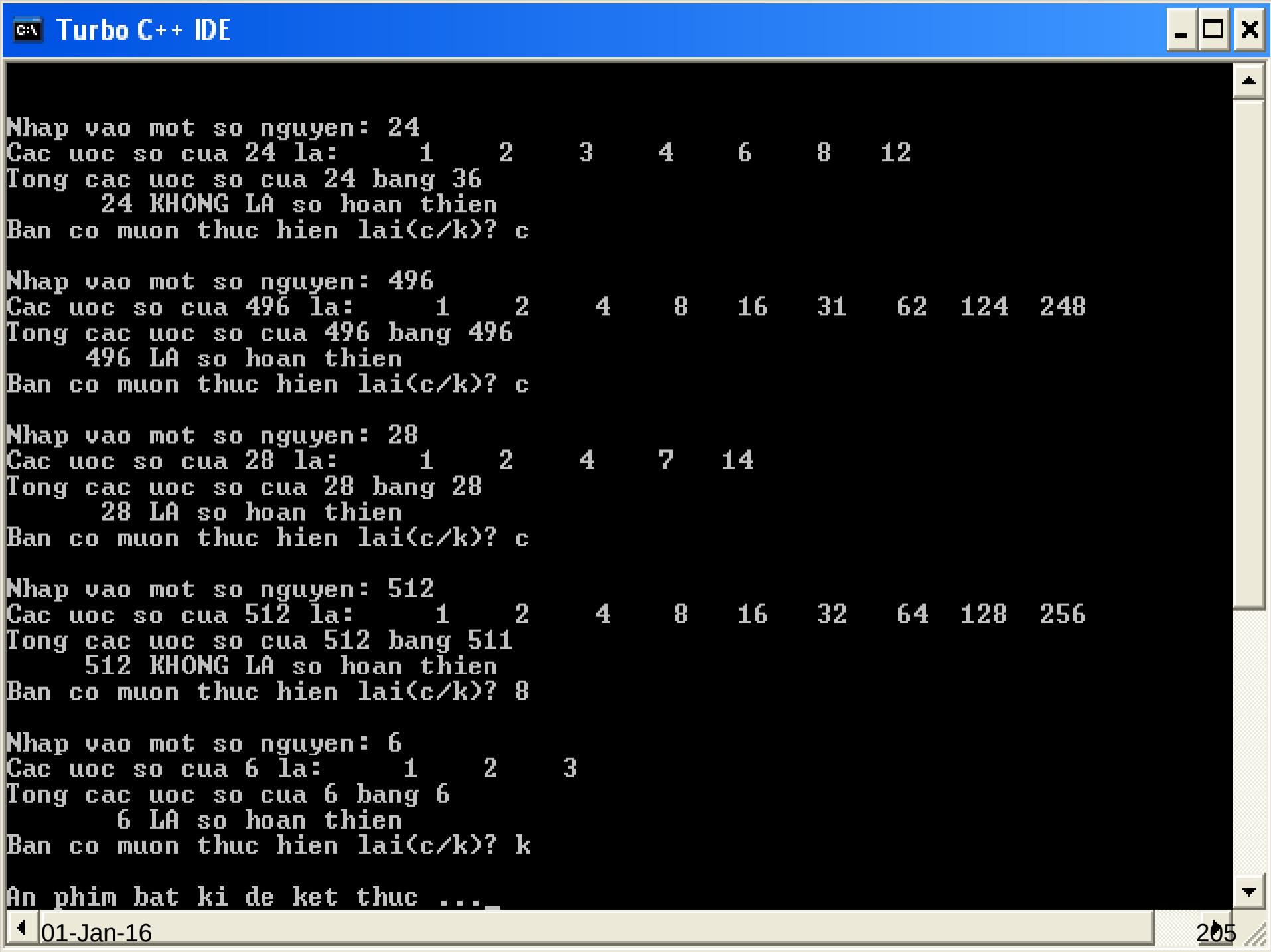
Ví dụ: Nhập số và kiểm tra số hoàn hảo

fflush(): xóa vùng đệm bàn phím

toupper(): chuyển sang chữ hoa

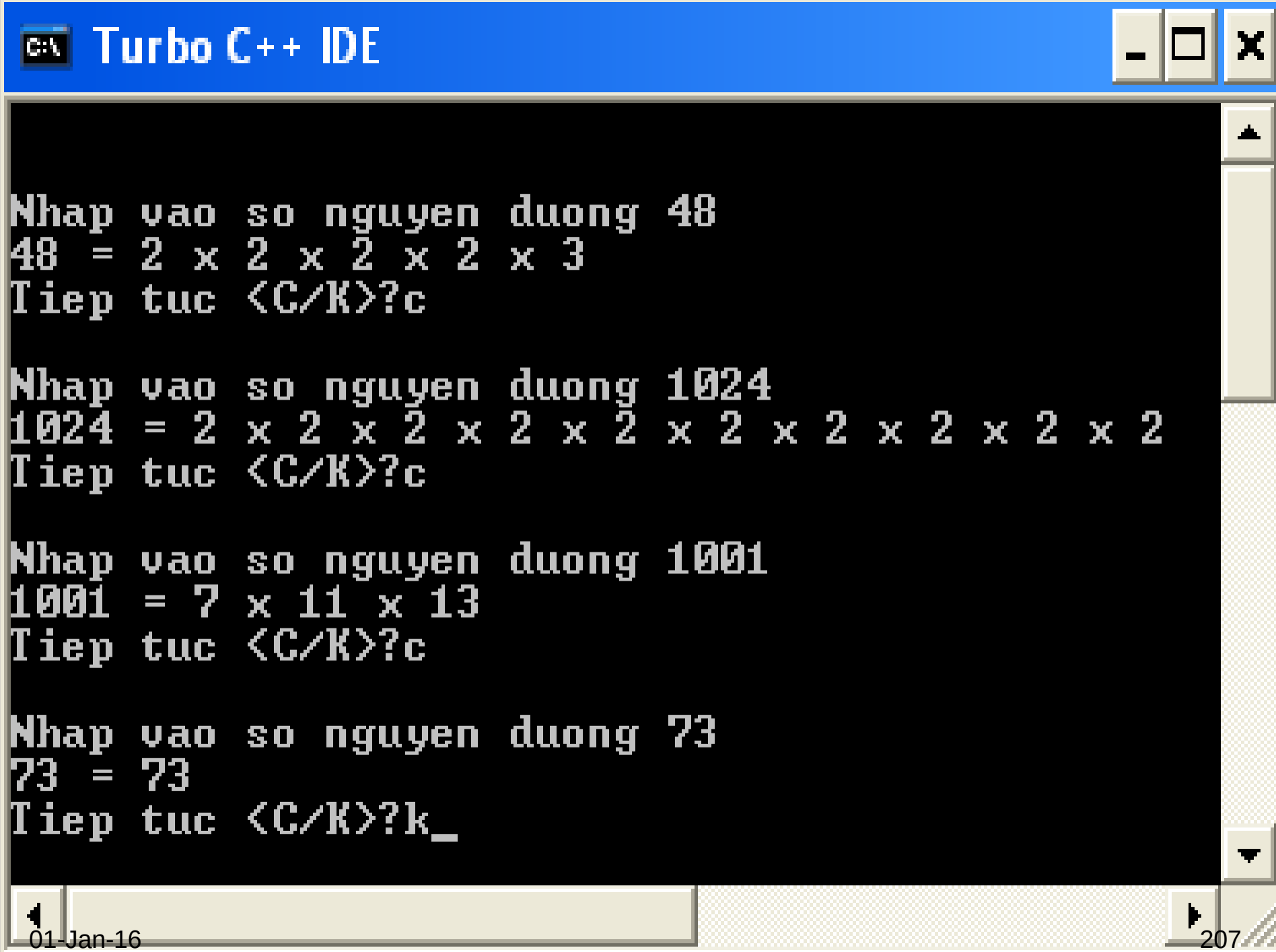
getche(): Đọc ký tự tự vùng đệm & hiện thị lên màn hình (**getch()** thì không)

```
1.      #include <stdio.h>
2.      #include <ctype.h>
3.      #include <conio.h>
4.      void main(){
5.          long int n, tong, i;
6.char ch;  7.
8.          do          {
9.                      tong = 0;
10.                     printf("\n\nNhap vao mot so nguyen: "); scanf("%ld",&n);
11.                     printf("Cac uoc so cua %ld la: ",n);
12.                     for(i = 1;i<n;i++)
13.                       if(n % i == 0){
14.                                             printf("%5d",i);
15.                                             tong = tong + i;
16.                                 }//if & for
17.                     printf("\nTong cac uoc so cua %ld bang %ld",n,tong);
18.                     if(tong == n)  printf("\n   %5ld LA so hoan hao“,n);
19.                     else       printf("\n   %5ld KHONG LA so hoan hao“,n);
20.                     printf("\nBan co muon thuc hien lai(c/k)? ");
21.                     fflush(stdin);
22.         }while(toupper(getche()) !='K');
            printf("\n\nAn phim bat ki de ket thuc ..."); getch();
23.     }
```


Turbo C++ IDE
Nhap vao mot so nguyen: 24
Cac uoc so cua 24 la: 1 2 3 4 6 8 12
Tong cac uoc so cua 24 bang 36
24 KHONG LA so hoan thien
Ban co muon thuc hien lai(c/k)? c
Nhap vao mot so nguyen: 496
Cac uoc so cua 496 la: 1 2 4 8 16 31 62 124 248
Tong cac uoc so cua 496 bang 496
496 LA so hoan thien
Ban co muon thuc hien lai(c/k)? c
Nhap vao mot so nguyen: 28
Cac uoc so cua 28 la: 1 2 4 7 14
Tong cac uoc so cua 28 bang 28
28 LA so hoan thien
Ban co muon thuc hien lai(c/k)? c
Nhap vao mot so nguyen: 512
Cac uoc so cua 512 la: 1 2 4 8 16 32 64 128 256
Tong cac uoc so cua 512 bang 511
512 KHONG LA so hoan thien
Ban co muon thuc hien lai(c/k)? 8
Nhap vao mot so nguyen: 6
Cac uoc so cua 6 la: 1 2 3
Tong cac uoc so cua 6 bang 6
6 LA so hoan thien
Ban co muon thuc hien lai(c/k)? k
An phim bat ki de ket thuc ...
01-Jan-16
205

Nhập số và phân tích số nguyên ra thừa số nguyên tố

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  #include <ctype.h>
4.  void main(){
5.    int N, i;
6.    do{  printf("\n\nNhap vao so nguyen duong "); scanf("%d",&N);
7.                 printf("%d = ",N);
8.                 i = 2;
9.                 while (i < N ){
10.                          if (N % i == 0){
11.                                   printf("%d x ",i);
12.                                   N = N/i;
13.                          } else   i++;
14.                }
15.                printf("%d \n",N);
16.                printf("Tiep tuc <C/K>?");  fflush(stdin);
17.  }while(toupper(getche()) != 'K');
18. }
```


Turbo C++ IDE
Nhap vao so nguyen duong 48
48 = 2 x 2 x 2 x 2 x 3
Tiep tuc <C/K>?c
Nhap vao so nguyen duong 1024
1024 = 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2
Tiep tuc <C/K>?c
Nhap vao so nguyen duong 1001
1001 = 7 x 11 x 13
Tiep tuc <C/K>?c
Nhap vao so nguyen duong 73
73 = 73
Tiep tuc <C/K>?k_

Ví dụ

Viết chương trình thực hiện công việc

- Nhập vào từ bàn phím 2 số nguyên
- Nhập vào từ bàn phím một ký tự bất kỳ;
  - Nếu đây là một toán tử số học thì đưa ra giá trị tương ứng với toán tử.
  - Nếu không phải thì đưa ra thông báo sai
- Chương trình thực hiện cho tới khi ký tự nhập vào là ‘q’ hoặc ‘Q’

```
#include  <stdio.h>
#include <conio.h>
void main() {
   int a, b;
   char ch;
   int Fin =  0;
   clrscr();
   printf("Nhap cac so a, b "); scanf("%d%d",&a,&b);
   do{
     printf("\nToan tu (+ ; - ; * ; / ; %) "); ch=getche();
     switch(ch){
         case '+': printf(" Co ket qua: %d\n",a+b); break;
         case '-': printf(" Co ket qua: %d\n",a-b); break;
         case '*': printf(" Co ket qua: %d\n",a*b); break;
```

```
        case '%':       if   (b==0) printf(" Chia cho 0\n");
                        else  printf(" Co ket qua: %d\n",a%b);
                        break;
        case '/':        if  (b==0) printf(" Chia cho 0\n");
                        else  printf(" Co ket qua: %d\n",a/b);
                        break;
        case 'q':
        case 'Q':       Fin=1;          break;
        default: printf(" khong co toan tu nay\n");
     }
   }while(Fin==0);
   printf("\nKet thuc, an mot phim...");
   getch();
}
```

```
Nhap cac so a, b 145 60

Toan tu (+ ; - ; * ; / ; %) + Co ket qua: 205

Toan tu (+ ; - ; * ; / ; %) - Co ket qua: 85

Toan tu (+ ; - ; * ; / ; %) * Co ket qua: 8700

Toan tu (+ ; - ; * ; / ; %) / Co ket qua: 2

Toan tu (+ ; - ; * ; / ; %) % Co ket qua: 25

Toan tu (+ ; - ; * ; / ; %) g khong co toan tu nay

Toan tu (+ ; - ; * ; / ; %) q
Ket thuc, an mot phim...
```

Ví dụ: Nhập xâu, đếm số ký tự của xâu

- **Khai báo:** *int n=0; char c;*
- **Dùng vòng for**
  - *for(n=0;;c=getchar(),n++) if(c==‘\n’) break;*
  - *for(n=0 ; getchar() != '\n' ; n++);*
- **Dùng vòng lặp while**

```
c = getchar();
while (c !=‘\n’){
        c= getchar();
        n++;
}
```

- **Dùng vòng lặp do… while**

```
do{
        c = getchar();
        n++;
}while(c!=‘\n’);
```

- ***Đưa kết quả ra:*** *printf(«Chuoi chua %d ky tu »,n);*

Nhập chuỗi ký tự cho đến khi gặp ký tự '*'
Tính tần suất xuất hiện nguyên âm 'a'

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  #include <ctype.h>
4.  void main(){
5.        char c; int n, d;
6.        do{
7.          printf("\n\n");
8.          d=0; n=0;
9.          while( (c=getche()) !='*'){
10.                 n++;
11.                 if (c=='a') d++;
12.         }
13.       if(n==0)
14.         printf("\nChuoi ky tu rong\n");
15.       else
16.         printf("\ntan suat xuat hien ky tu 'a' la %5.2f%%\n",(float)100*d/n);
17.       printf("Tiep tuc <C/K>?: ");
18.     }while(toupper(getche()) != 'K');
19. }
```

```
Madam*
tan suat xuat hien ky tu 'a' la 40.00%
Tiep tuc <C/K>?: c

Hello world*
tan suat xuat hien ky tu 'a' la  0.00%
Tiep tuc <C/K>?: c

*
Chuoi ky tu rong
Tiep tuc <C/K>?: c

ABBA*
tan suat xuat hien ky tu 'a' la  0.00%
Tiep tuc <C/K>?: k
```

Ví dụ: Nhập một số nguyên, kiểm tra là số nguyên tố không?

```
#includ <stdio.h>
#include <math.h>
void main()
{ int N, i, OK = 1;
  printf("\nNhap gia tri N : "); scanf("%d", &N);
  if (n<2) printf("\nSo %d khong phai so nguyen to",N);
  else {
    for (i=2; i<sqrt(N); i++)
    for (i=2; i<N; i++)
      if (N%i == 0) {
            OK = 0;
            break;
      }
    if (OK)  printf("\nSo %d la so nguyen to.", N);
    else   printf("\nSo %d la hop so.", N);
  }
  getch();
}
```

i <= (int)sqrt(N)

```
i= 2;
while (N % i != 0) i++;
if (i == N) printf(« so ng to »)
```

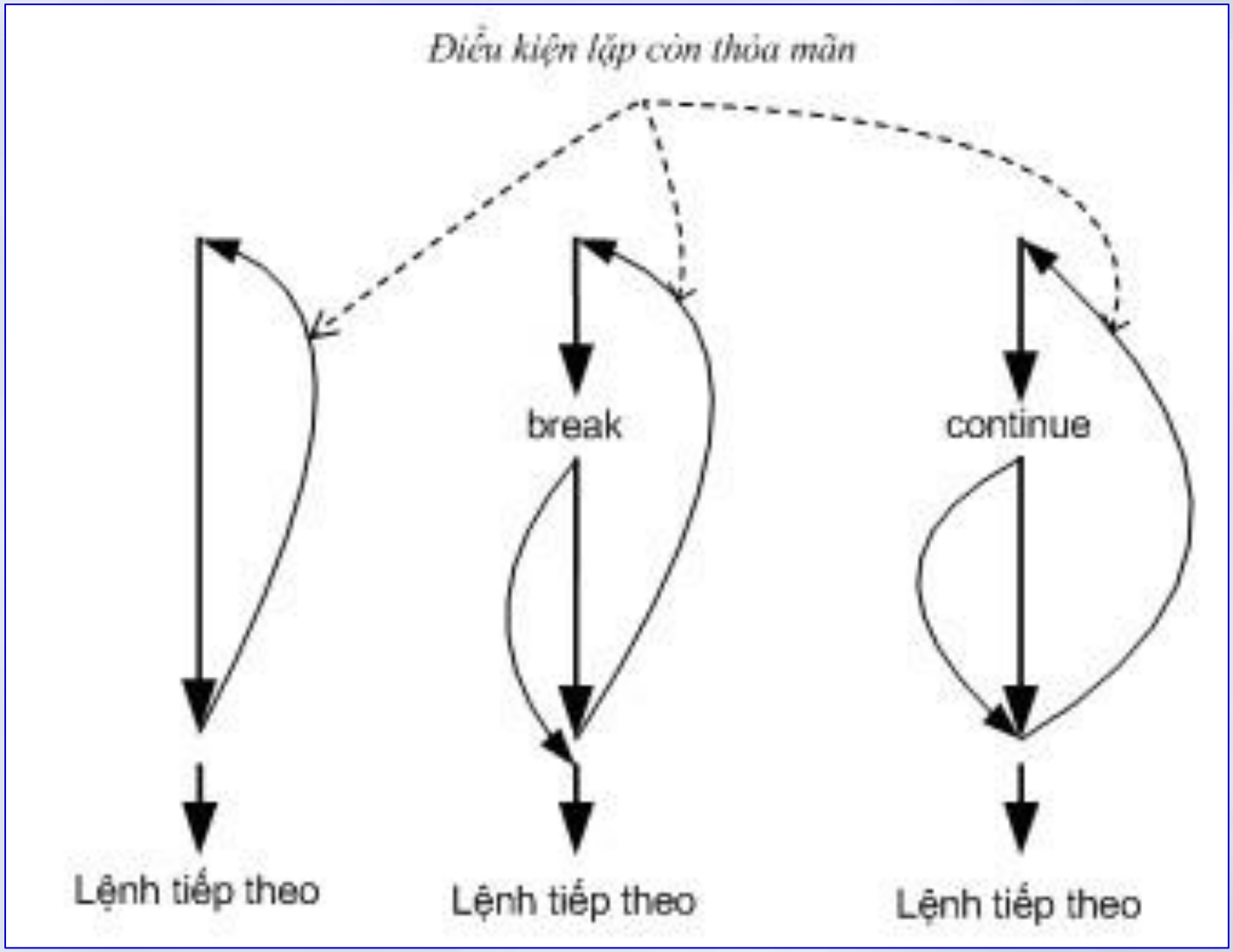
Nội dung chính

1. Cấu trúc lệnh khối
2. Cấu trúc rẽ nhánh
   - Cấu trúc if, if … else
   - Cấu trúc lựa chọn switch
3. Cấu trúc lặp
   - Vòng lặp **for**
   - Vòng lặp **while** và **do while**
4. Các lệnh thay đổi cấu trúc lập trình
   - Câu lệnh **continue**
   - Câu lệnh **break**

Mục đích

- Các vòng lặp **while/ do ... while/ for** sẽ kết thúc quá trình lặp khi biểu thức điều kiện của vòng lặp không còn được thỏa mãn.
- Tuy nhiên trong lập trình đôi khi ta cũng cần thoát khỏi vòng lặp ngay cả khi biểu thức điều kiện của vòng lặp vẫn còn được thỏa mãn.
- Để hỗ trợ người lập trình làm việc đó, ngôn ngữ C cung cấp 2 câu lệnh là **continue** và **break**

Continue >< break



continue

- Bỏ qua việc thực hiện các câu lệnh nằm sau lệnh **continue** trong thân vòng lặp.
- Chuyển sang thực hiện một vòng lặp mới

Ví dụ 1

```
#include <stdio.h>
#include <conio.h>
void main()
{
  int i;
   int sum = 0;
  for(i = 1;i<=100;i++)
  {
     if(i % 5 == 0)
          continue;
     sum += i;
  }
}
```

Tính tổng 100 số nguyên đầu tiên ngoại trừ các số chia hết cho 5

```
for(i=1;i<=100;i++)
  if (i % 5 != 0)
     sum += i;
```

break

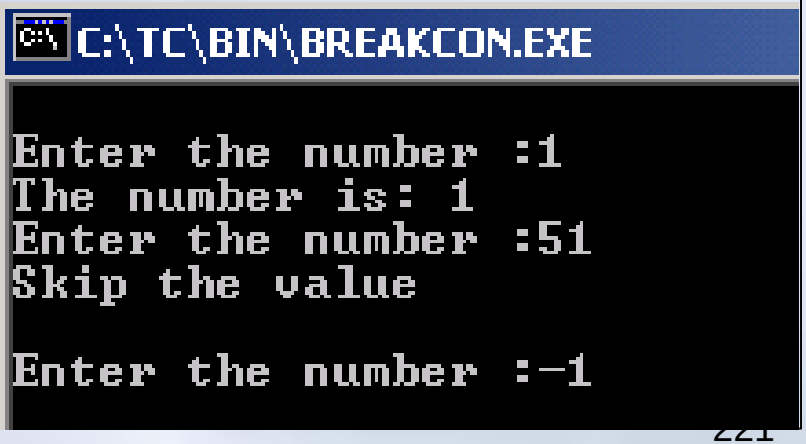
Thoát khỏi vòng lặp ngay cả khi biểu thức điều kiện của vòng lặp vẫn còn được thỏa mãn.

Chú ý:

- **break** dùng để thoát ra khỏi khối lặp hiện tại
- **break** cũng dùng để thoát ra khỏi lệnh rẽ nhánh **switch**

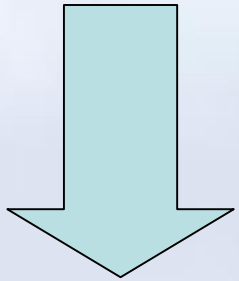
Ví dụ 2

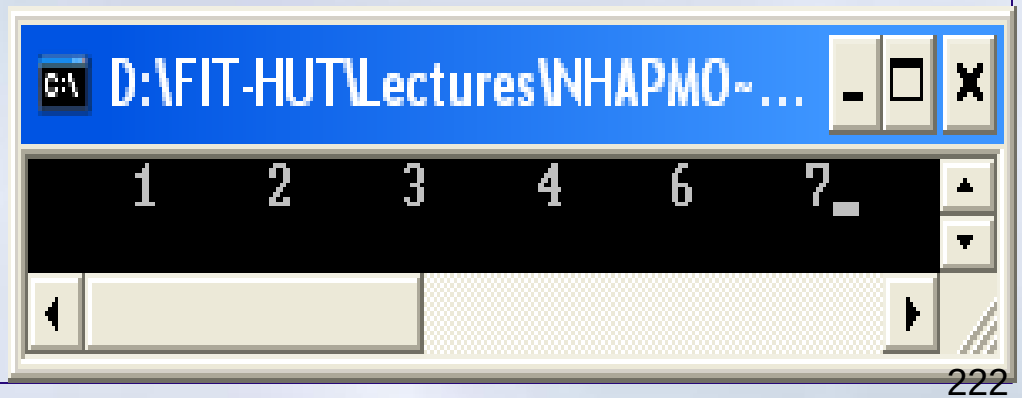
```
#include  <stdio.h>
#include <conio.h>
void main() {
    int n;
   do {
        printf(" \nEnter the number :"); scanf("%d", &n);
        if (n < 0) {
           break;
        }
        if (n >10) {
            printf("Skip the value\n");
           continue;
        }
        printf("The number is: %d", n);
  } while (n!= 0);
}
```



Ví dụ 3

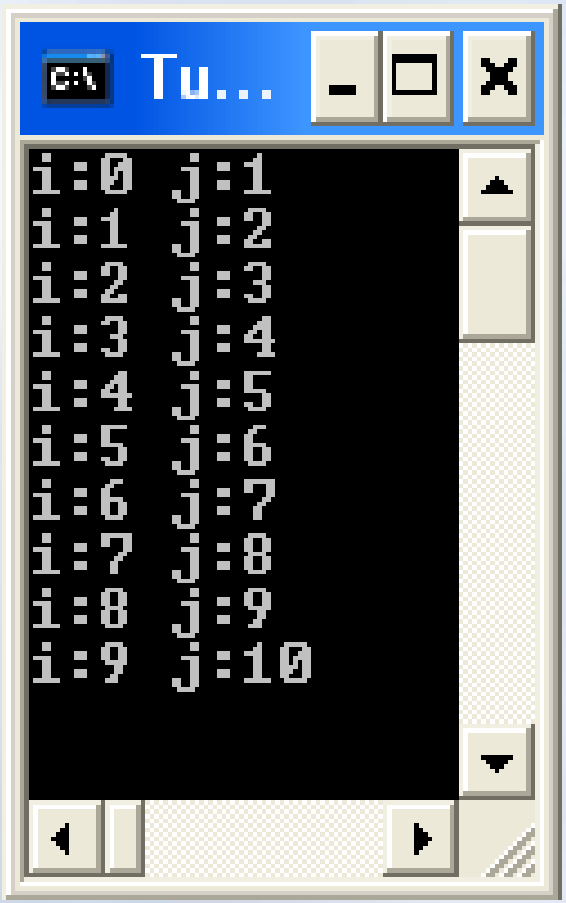
```
#include  <stdio.h>
#include <conio.h>
void main()
{
    int i;
    for(i = 1;i<=10;i++)
    {
          if(i == 5) continue;
       printf("%5d",i);
          if(i==7) break;
    }
    getch();
}
```





Ví dụ 4

```
#include <stdio.h>
#include <conio.h>
void main()
{ int i,j;
  clrscr();
  for(i = 0;i<10;i++) {
    for (j=0; j < 10; j ++) {
      if(j > i){
          break;
      }//if
    }//for _ j
    printf("i:%d j:%d\n",i,j);
  }//for i
    getch();
}
```

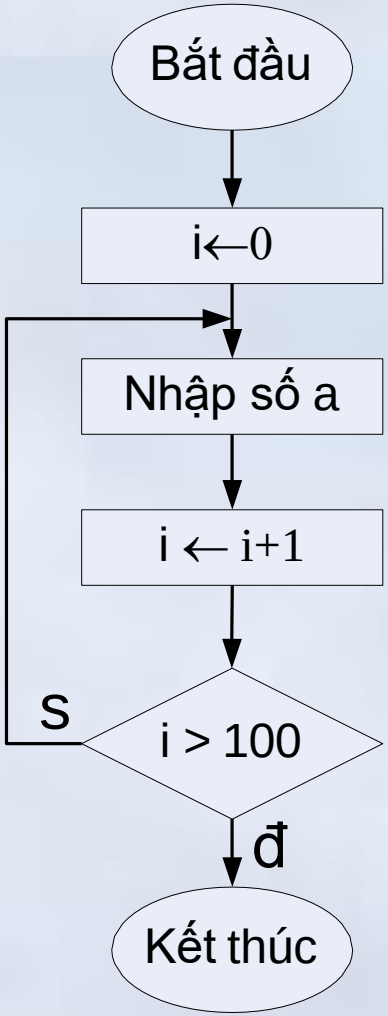


Ví dụ tổng hợp

Viết chương trình thực hiện các công việc sau

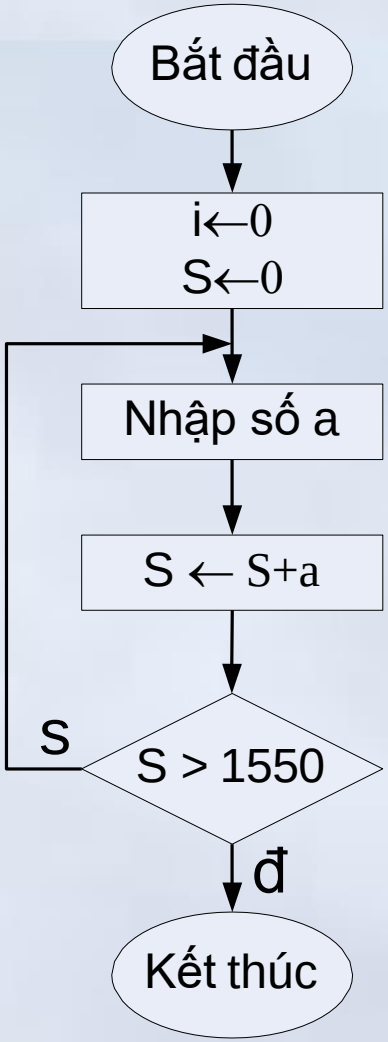
- Nhập vào một dãy số cho tới khi
  - Tổng của dãy lớn hơn 1550 hoặc là
  - Số phần tử trong dãy lớn hơn 100
- Đưa ra số lượng phần tử nằm trong khoảng (35, 70)
- Đưa ra trung bình cộng của các phần tử chia hết cho 7

Nhập một dãy số cho tới khi số phần tử trong dãy lớn hơn 100



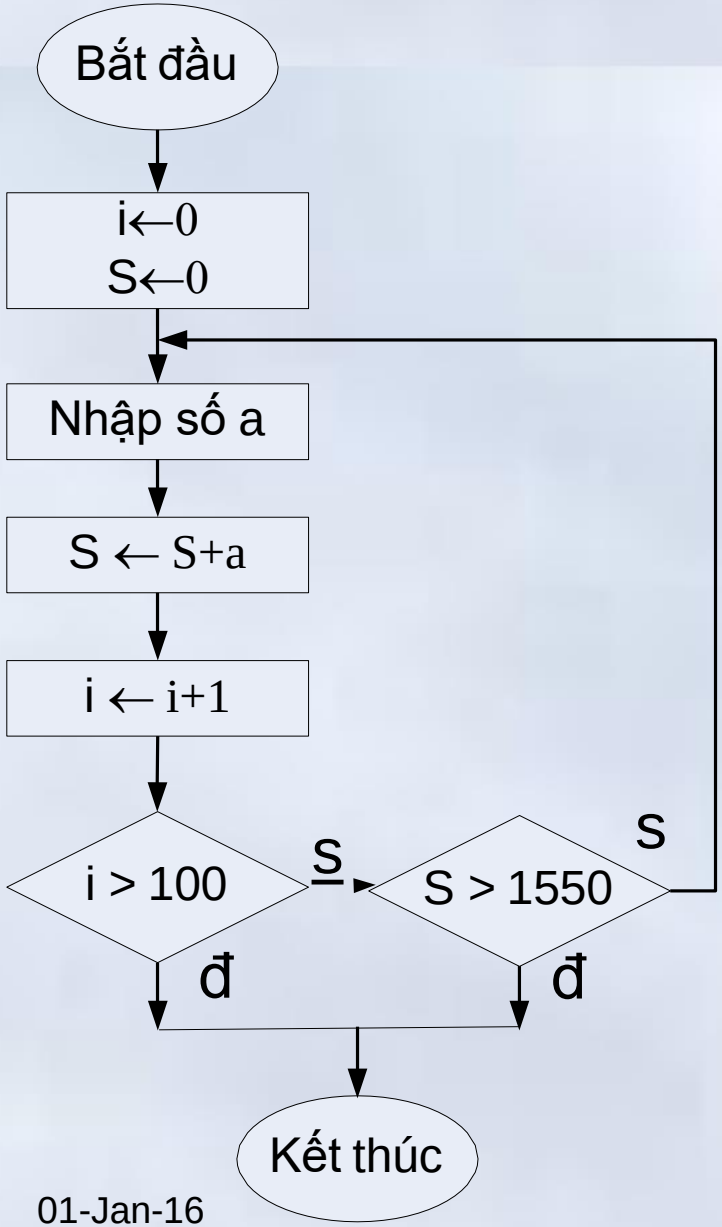

```
#include <stdio.h>
void main(){
    int a, i;
    i = 0;
    do{
        printf("Nhap vao so nguyen:");
        scanf("%d",&a);
        i++;
   }while (i <= 100);
}
```

Nhập một dãy số cho tới khi tổng của dãy lớn hơn 1550




```
#include <stdio.h>
void main(){
    int a, S;
    S = 0;
    do{
        printf("Nhap vao so nguyen:");
        scanf("%d",&a);
        S+=a;
   }while (S <= 1550);
}
```

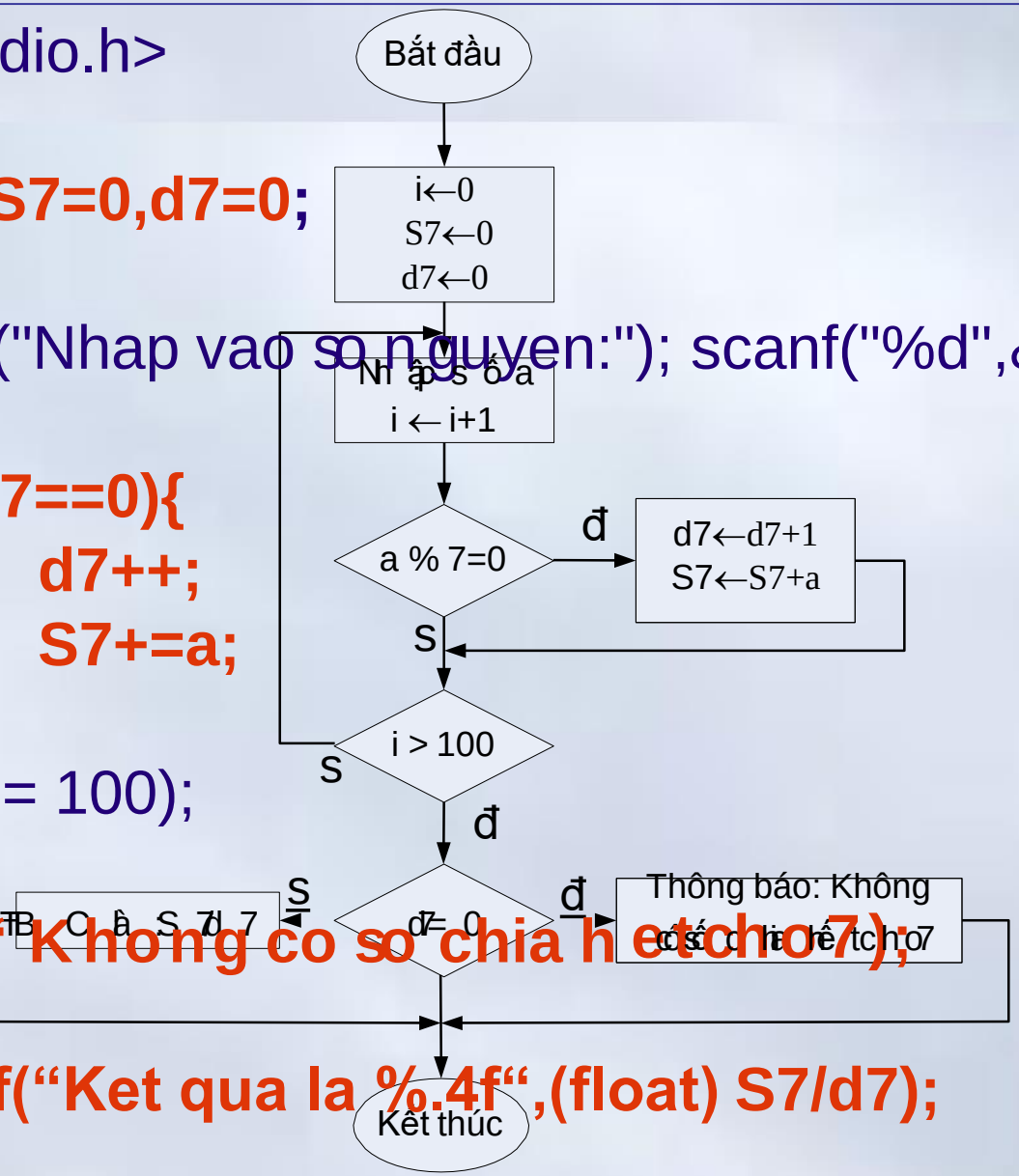
Nhập một dãy số cho tới khi thỏa mãn….




```
#include <stdio.h>
void main(){
    int a, i, S;
    S = 0; i=0;
    do{
        printf("Nhap vao so nguyen:");
        scanf("%d",&a);
        S+=a;
        i++;
    }while ( (i <=100)&&(S <= 1550) );
}
```

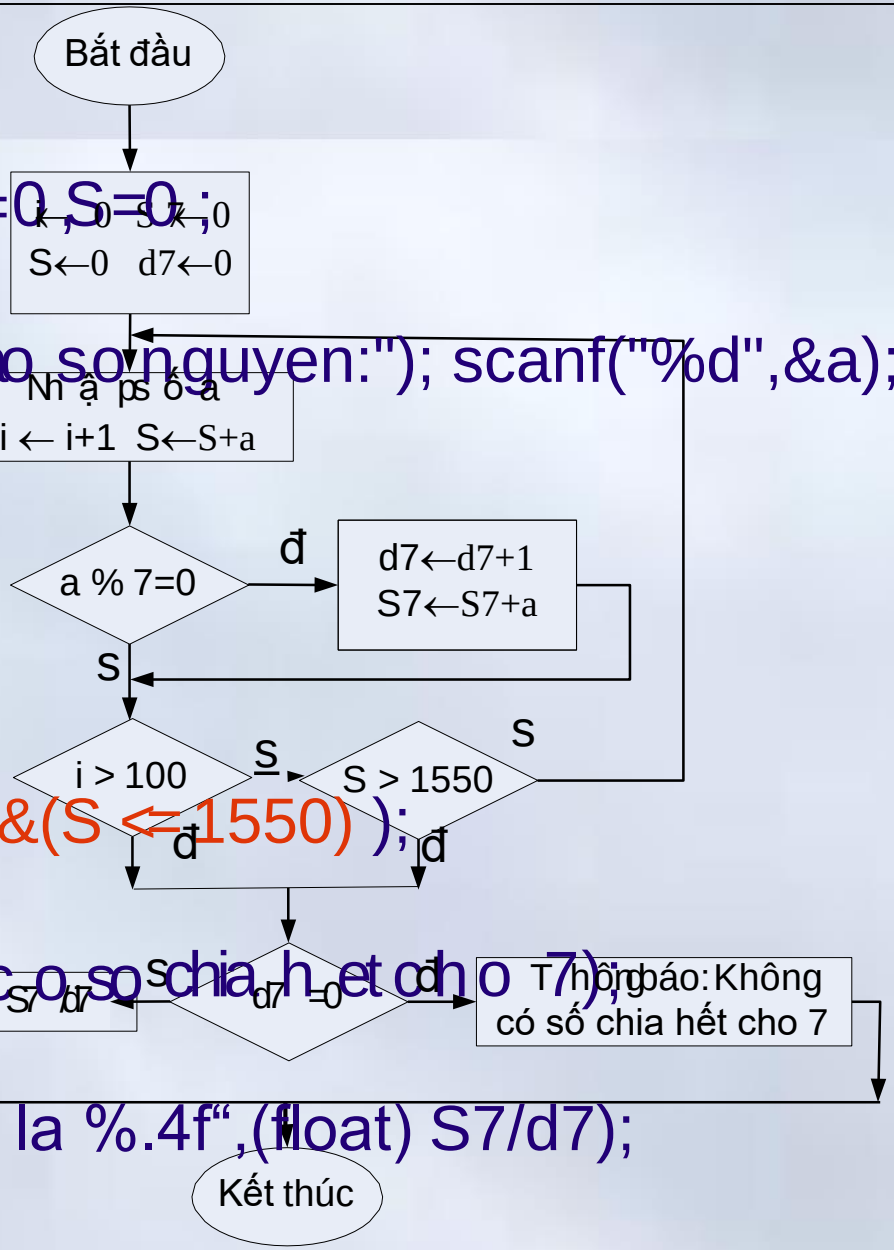
Đưa ra TBC của các phần tử chia hết cho 7(1)

```
#include <stdio.h>
void main(){
    int a, i=0,S7=0,d7=0;
    do{
        printf("Nhap vao so nguyen:"); scanf("%d",&a);
        i++;
        if(a%7==0){
                d7++;
                S7+=a;
        }
    }while (i <= 100);
    if(d7==0)
        printf("Khong co so chia het cho 7");
    else
        printf("Ket qua la %.4f",(float) S7/d7);
}
```



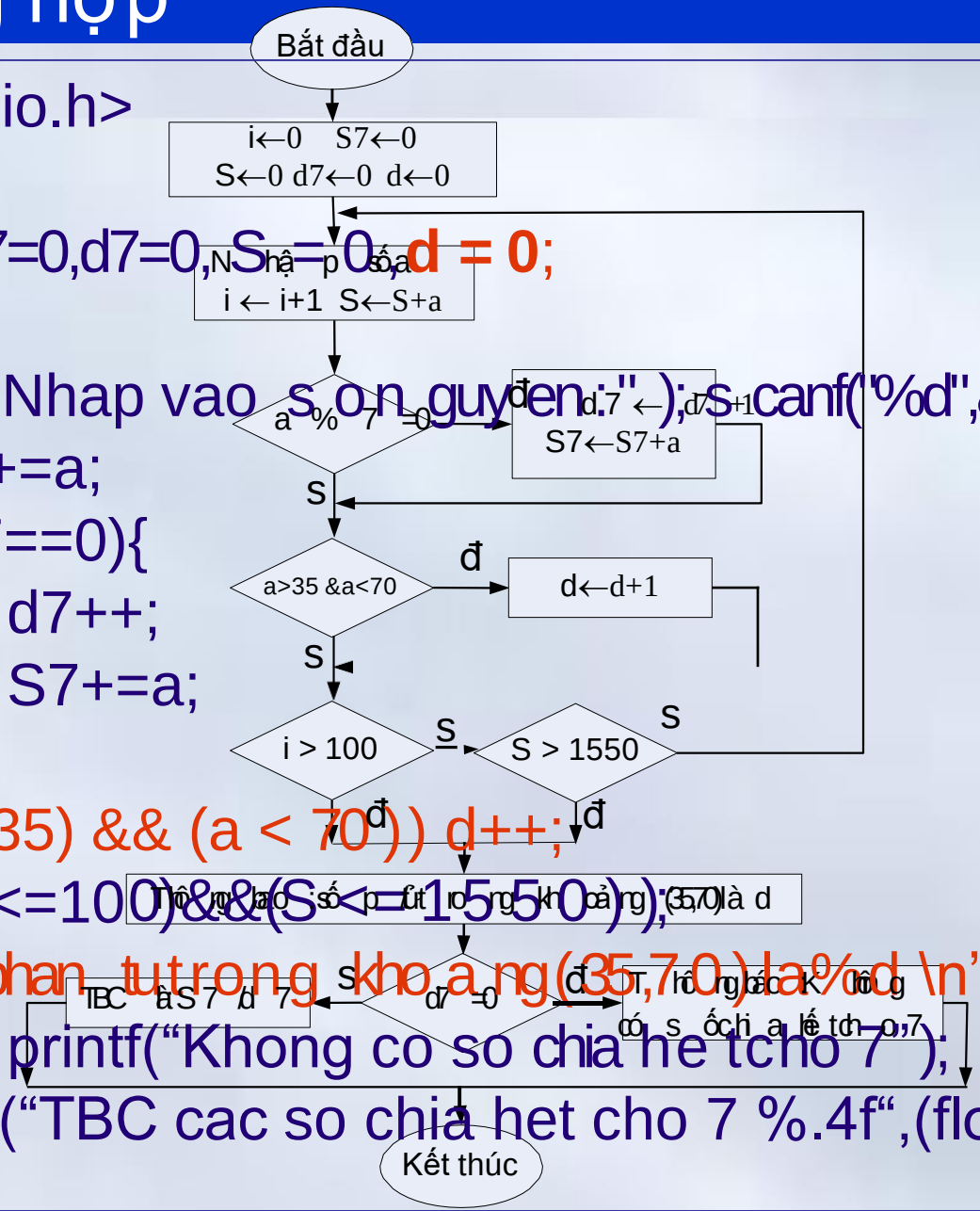
Đưa ra TBC của các phần tử chia hết cho 7(2)

```
#include <stdio.h>
void main(){
    int a, i=0,S7=0,d7=0,S=0 ;
    do{
        printf("Nhap vao so nguyen:"); scanf("%d",&a);
        i++; S+=a;
        if(a%7==0){
                d7++;
                S7+=a;
        }
    }while ( (i <=100)&&(S <=1550) );
    if(d7==0)
        printf("Khong co so chia het cho 7");
    else
        printf("Ket qua la %.4f",(float) S7/d7);
}
```



Ví dụ tổng hợp

```
#include <stdio.h>
void main(){
    int a, i=0,S7=0,d7=0,S=0,d = 0;
    do{
        printf("Nhap vao so nguyen:"); scanf("%d",&a);
        i++; S+=a;
        if(a%7==0){
                d7++;
                S7+=a;
        }
        if( (a>35) && (a < 70)) d++;
    }while ( (i <=100)&&(S<=1550));
    printf("So phan tu trong khoang (35,70) la %d \n",d);
    if(d7==0)   printf("Khong co so chia het cho 7");
    else  printf("TBC cac so chia het cho 7 %.4f",(float) S7/d7);
}
```



Bài tập tại lớp

1. Cho hàm số $f(x) = x^5 + \sqrt[5]{x}$ Lập trình tính và đưa ra màn hình các cặp giá trị <x,f(x)> với x lấy dãy giá trị -10;-9.9;-9.8; …..; 4.9;5.0.
2. Đọc vào dãy số cho tới khi gặp một số dương chia hết cho 5; Tìm số lớn nhất của dãy và số lần xuất hiện các giá trị đó
3. Nhập vào dãy cho tới khi gặp số 0. Tính trung bình cộng các số chẵn đã đọc (không tính số 0)
4. Lập trình thực hiện trò chơi bốc sỏi với máy tính
   - Có N viên sỏi, 2 người chơi lần lượt bốc từ 1 đến 5 viên. Người bốc vên sỏi cuối luôn thắng/thua. Hãy viết chương trình thực hiện nhập N (N $\in$ [30..50)) và máy tính là một người chơi. Máy tính đi trước và luôn thắng nếu được (*N=6,12,.. người đi trước thua*)

Tính hàm $f(x) = x^5 + \sqrt[5]{x}$

```
1.  #include <stdio.h>
2.  #include <math.h>
3.  void main(){
4.      float x, fx;

5.      for(x=-10.0; x<=5.0; x+=0.1){
6.          if(x==0)
7.              fx = 0.0;
8.          else
9.              fx = pow(x,5)+x/fabs(x) * pow(fabs(x), 0.2);
10.       printf("<%4.1f,%7.2f>\n",x,fx);
11.     }
12. }
```

- **pow(x,y)** sinh ra lỗi khi x âm và y không là số nguyên
- **fabs(x)** trả về trị tuyệt đối của x khi là số thực

Có sai số khi x=0.0

```
for(int i=-100;i<50;i++){
        x=(float)i/10;
        .......
```

Đọc dãy số tới khi gặp số 0, tìm và đếm số max

```
1.  #include <stdio.h>
2.  #include <limits.h>

3.  void main(){
4.      int a, d=0, max = INT_MIN;
5.      do {
6.          printf("Nhap mot so : "); scanf("%d",&a);
7.          if( a > max){
8.              max = a;
9.              d = 1;
10.         }else
11.if (a == max) d++;  12.   }while ( a
< 0 || a%5 !=0);
13.     printf("Max: %d; Co %d gia tri",max,d);
14. }
```

Bài tập

Viết chương trình đọc x và n vào từ bàn phím rồi tính

$$S = \sqrt{x + \sqrt{x + \sqrt{x + \cdots + \sqrt{x}}}} \quad n \text{ dâ'u căn}$$

$$S = 1 + x + \frac{x^2}{2} + \frac{x^3}{3} + \cdots + \frac{x^n}{n}$$

$$S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$

$$S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots + \frac{(-1)^n x^n}{n!}$$

Viết chương trình đọc x và n vào từ bàn phím rồi tính

$$S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  void main(){
4.      int n, i;
5.      float x, u = 1.0,S=1.0;
6.      clrscr();
7.      printf("Nhap vao so nguyen n : "); scanf("%d",&n);
8.      printf("Nhap vao so thuc x : ");      scanf("%f",&x);
9.      for(i = 1; i <= n; i ++){
10.             u *= x/i;
11.             S += u;
12.     }
13.     printf("Ket qua la %.8f",S);
14.     getch();
15. }
```

```
Nhap vao so nguyen n : 20
Nhap vao so thuc x : 1
Ket qua la 2.71828198_
```

```
Nhap vao so nguyen n : 50
Nhap vao so thuc x : 2.31
Ket qua la 10.07442379_
```

Bài tập (*Tính tổng vô hạn*)

Đọc x và eps từ bàn phím và tính biểu thức sau với độ chính xác nhỏ hơn eps

$$S_1 = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots + \frac{(-1)^n x^n}{n!} + \cdots$$

$$S_2 = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + \frac{(-1)^n}{(2n+1)!} x^{2n+1} + \cdots // \sin(x)$$

$$S_3 = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \cdots + \frac{(-1)^n}{(2n)!} x^{2n} + \cdots // \cos(x)$$

Nhập x và ε và tính với độ chính xác nhỏ hơn ε

$$S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots + \frac{(-1)^n x^n}{n!} + \cdots$$

Tổng kết

1. **Câu lệnh khối**

   Đặt trong cặp ngoặc nhọn **{ }**

2. **Cấu trúc rẽ nhánh**
   - **if** (bieu_thuc)**, if** (bieu_thuc) **… else**
   - **switch** (bieu_thuc) {(**case**/**break**/**default**)}

3. **Cấu trúc lặp**
   - **for**(bieu_thuc_1; bieu_thuc_2; bieu_thuc_3) CauLenh;
   - **while** (bieu_thuc) CauLenh;
   - **do** Cau_Lenh **while** (bieu_thuc);

4. **Các lệnh thay đổi cấu trúc lập trình**
   - **continue/ break**

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. **Mảng**
   - Khái niệm
   - Khai báo và sử dụng
   - Các thao tác thường gặp
2. **Con trỏ**
   - Khái niệm và cách khai báo
   - Toán tử địa chỉ (&), toán tử nội dung (*)
   - Phép toán trên con trỏ
   - Con trỏ và mảng
3. **Xâu ký tự**
   - Khái niệm, khai báo và sử dụng
   - Các hàm xử lý ký tự và xâu ký tự
   - Mảng xâu ký tự

Giới thiệu

Bài toán:

- Nhập điểm thi (số nguyên) môn Tin đại cương cho lớp gồm 50 sinh viên rồi đưa ra số lượng sinh viên phải học lại

Phương pháp: Điểm của mỗi sinh viên là 1 biến

- Tên biến là tên sinh viên

  Ví dụ: int An, Anh, Binh1, Binh2, Cuong,…..
  Van, Viet;

- Tên biến dạng "***dx***" với ***x*** là chỉ số thứ tự của SV trong lớp

  Ví dụ: int d1, d2, d3,……,d50;

Nhận xét 1: Không hợp lý

- Có quá nhiều biến (*Điểm thi cho toàn trường.. !?*)
- Khó khăn cho các thao tác duyệt toàn bộ danh sách

  Số SV học lại: if(d1 <5) d++; if(d2 <5) d++; ~ …if(d50 <5) d++;

Nhận xét 2: Các biến có chung ý nghĩa, tính chất

Giới thiệu

- Trong thực tế, thường gặp các đối tượng có tính chất chung
  - Tháng trong năm
  - Điểm trung bình của sinh viên trong lớp
- Các đối tượng được nhóm lại dưới một tên
- Đối tượng được đặc trưng bởi **tên nhóm** và **thứ tự** trong nhóm
  - Tháng thứ 3 trong năm: Tháng 3
  - Sinh viên thứ 17 trong lớp:…
- Số thứ tự của đối tượng trong nhóm là **chỉ phần tử**

Khái niệm mảng

- Kiểu mảng là một kiểu dữ liệu gồm
  - Một số hữu hạn thành phần.
  - Các thành phần có cùng một kiểu: **kiểu cơ sở** hay là **kiểu thành phần**.
- Mỗi phần tử của mảng được tham khảo thông qua
  - Tên mảng và
  - Chỉ số của phần tử trong mảng

  Ví dụ:
  - <d7>: Điểm thi tin của sinh viên thứ tự 7 trong lớp

Khai báo mảng

Kiểu_dữ_liệu Tên_Mảng[Kích thước];

- **Kiểu_dữ_liệu:** kiểu của các phần tử trong mảng (*nguyên, thực, ký tự, chuỗi, mảng,…*)
- **Tên_mảng:** tên của mảng
- **Kích_thước_mảng:** số phần tử trong mảng

Ví dụ

```
// khai báo mảng 50 phần tử có kiểu dữ liệu int
int DiemTin[50];
float A[10]; //⇐ Mảng 10 phần tử kiểu số thực
```

Cấp phát bộ nhớ cho mảng

- Các phần tử trong mảng được cấp phát các ô nhớ kế tiếp nhau trong bộ nhớ
- Kích thước của mảng bằng kích thước một phần tử nhân với số phần tử

Ví dụ:

int A[10];//Mảng A gồm 10 phần tử nguyên

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]

Kích thước của mảng A: 10 x 2 = 20 bytes

Truy nhập đến thành phần của mảng

- Biến mảng lưu trữ địa chỉ ô nhớ đầu tiên trong vùng nhớ được cấp phát
- Ngôn ngữ C đánh chỉ số các phần tử trong mảng bắt đầu từ **0**
- Các phần tử của mảng được truy nhập thông qua
  - Tên mảng và
  - Chỉ số của phần tử của phần tử trong mảng

Tên_Mang[Chỉ_số_phần_tử];

Truy nhập đến thành phần của mảng→Ví dụ

Mảng A gồm 10 phần tử nguyên

A[0] = 7;

A[1] = 5;

A[4] = 7;

int N = A[1] + A[4]; → N = 12

Ví dụ

```
int A[10];
for(int i = 0; i < 10; i++) A[i]= 2* i;
```

0	2	4	6	8	10	12	14	16	18

i : 1 9 0 1 2 3 4 5 6 7 8 9 0

Chú ý: C không kiểm tra vượt quá giới hạn của mảng khi truy nhập

```
int A[3], B[4], C[3];
```

A[0]	A[1]	A[2]	B[0]	B[1]	B[2]	B[3]	C[0]	C[1]	C[2]

A[5] ↔ B[2] ↔ C[-2] ← nếu c/cấp liên

Mảng nhiều chiều

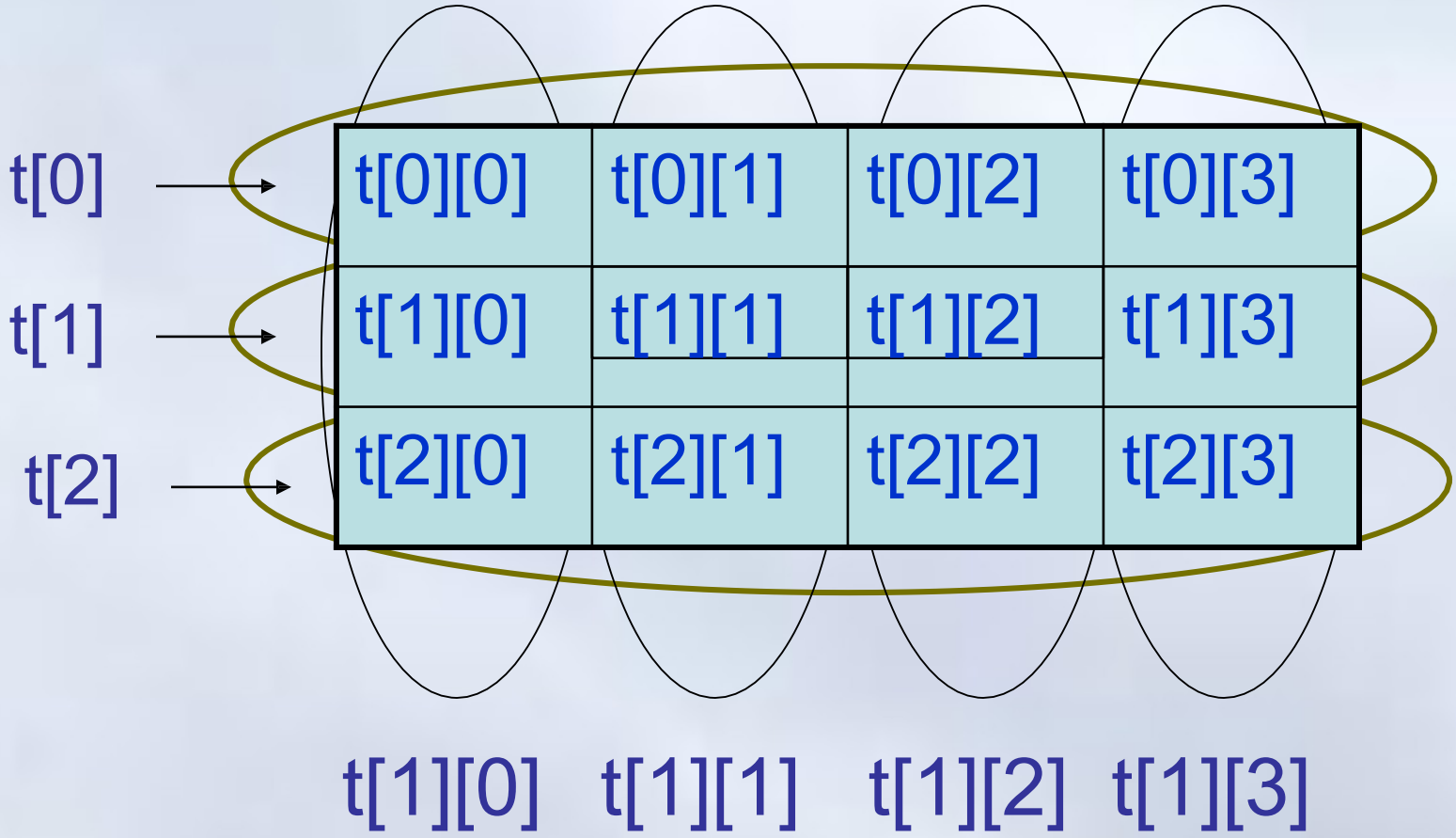
- Mỗi phần tử của mảng có thể là một mảng
  Mảng nhiều chiều

```
Kiểu Tên[Chiều_1] [Chiều_2]… [Chiều_N];
```

- **Kiểu**: Kiểu của mỗi phần tử trong mảng
- **Chiều_1, Chiều_2,…Chiều_N:** Các hằng số nguyên, cho biết kích thước (*số phần tử*) của mỗi chiều
- Mảng gồm: Chiều_1 x Chiều_2 x...x Chiều_N phần tử được lưu trữ trong vùng nhớ liên tục. Các phần tử thuộc kiểu **Kiểu**

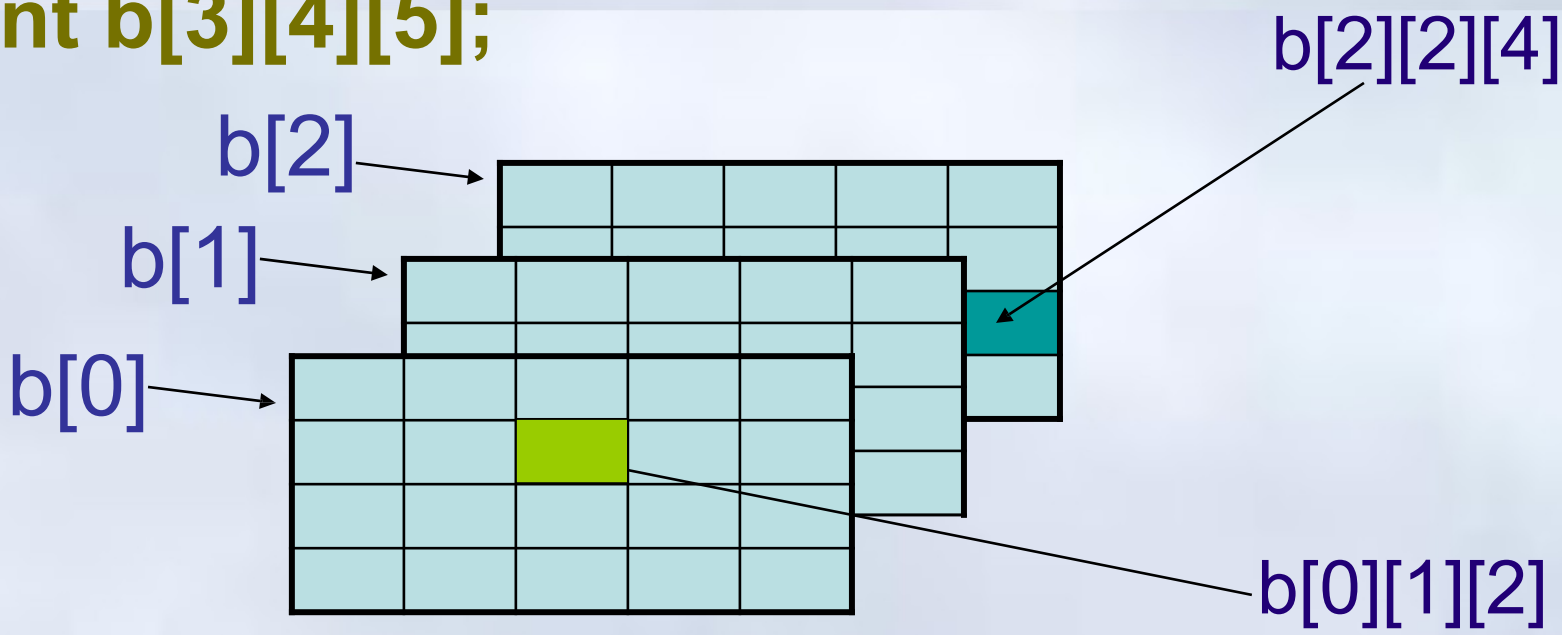
Mảng nhiều chiều

int t[3][4]



Mảng nhiều chiều→Ví dụ

int b[3][4][5];



- Mảng b gồm 3 phần tử b[0], b[1], b[2]
- Mỗi phần tử là mảng hai chiều gồm 4 hàng (hàng 0, 1, 2, 3) và 5 cột (0, 1, 2, 3, 4)
- Mỗi phần tử có kiểu nguyên có dấu, 2 byte

Khởi tạo giá trị cho mảng

Các phần tử của mảng có thể được khởi tạo giá trị ngay khi khai báo

Ví dụ

```
int a[4] = {1,4,6,2};
int b[2][3]={ {1,2,3}, {4,5,6} };
int t[3][4] = {
                {1, 2, 3, 4},
                {5, 6, 7, 8},
                {9, 10, 11, 12},
                };
```

Khởi tạo giá trị cho mảng → Chú ý

- Số lượng giá trị khởi tạo không được lớn hơn số lượng phần tử trong mảng
  - Nếu số lượng này nhỏ hơn, các phần tử còn lại được khởi tạo giá trị 0

```
int A[3][4] = { {1}, {4,5} };
int A[3][4] = { };←Tất cả đều mang giá trị 0
```

- Có thể xác định kích thước mảng thông qua số giá trị khởi tạo nếu để trống kích thước mảng

```
int A1 [8] = {2, 4, 6, 8, 10, 12, 14, 16};
int A2 [] = {2, 4, 6, 8, 10, 12, 14, 16};
```

Các thao tác thường gặp

- Nhập/Xuất dữ liệu cho mảng
  - Mảng 1 chiều, ma trận
- Bài toán đếm
  - Đếm số phần tử
  - Tính toán trên các phần tử..
- Tìm kiếm phần tử
  - Lớn nhất/nhỏ nhất/bất kỳ
- Sắp xếp phần tử trong mảng
  - Theo thứ tự, theo nguyên tắc
- Chèn thêm phần tử, xóa phần tử

Nhập dữ liệu

Dùng hàm scanf()

Ví dụ: int Table[10];

- Nhập dữ liệu cho một phần tử

  scanf(“%d”,&Table[2]);←phần tử thứ 3 của mảng

- Nhập dữ liệu cho cả mảng
  - Dùng vòng lặp for

    for(i = 0; i < 10; i++)

    scanf(“%d”,&Table[i]);

  - Nên in ra chỉ số phần tử khi nhập printf(“Table[%d] : ”,i); scanf(“%d”,&Table[i])

Nhập dữ liệu→Ví dụ 1

Nhập vào lượng mưa (mm) trong năm

```
#include <stdio.h>
#define   MONTHS
12  int main(){
    int
    rainfall[MONTHS],
    i;
    for ( i=0; i <
    MONTHS; i++ ){
        printf(“Nhap luong mưa tháng %d: “, i+1);
         scanf("%d", &rainfall[i] );
}
```

Nhập dữ liệu→Lưu ý

- Nếu số phần tử của mảng chỉ được biết tại thời điểm thực hiện chương trình (nhưng *biết số phần tử tối đa*)
  - Khai báo mảng với kích thước tối đa
  - Sử dụng biến nguyên lưu số phần tử thực sự của mảng.

Ví dụ:

- Nhập vào mảng không quá 100 số thực
  - Khai báo mảng thực Table có tối đa 100 phần tử.
  - Nhập số phần tử thực sự của mảng
  - Nhập giá trị cho từng phần phần tử (dùng for)

Nhập dữ liệu→Ví dụ 2

```
#include<stdio.h>
 void main(){
   float
   A[100];  int
   n, i;
   do{
        printf("\n Cho biet so phan tu cua mang: ");
         scanf("%d",&n);
   }while (n>100 || n<=0);
   for(i = 0; i < n; i++){
        printf("A[%d] = ", i); scanf("%f",&A[i]);
   }
```

Xuất dữ liệu trong mảng

Dùng hàm printf()

Ví dụ: int Table[10];

Hiện thị phần tử thứ 5:

```
printf("%d",Table[4]);
```

- Để hiển thị tất cả các phần tử:

```
for(i = 0; i < 10; i++)
    printf("%4d",Table[i])
    ;
```

Các kiểu xuất dữ liệu

- Hiển thị tất cả/một phần theo dòng/cột..

Xuất dữ liệu trong mảng→Ví dụ 1

```
#include <stdio.h>
 #define MAX 12
void main(){
    int A[MAX], i;
    for ( i=0; i < MAX; i++ )      //Nhập dữ liệu
          {  printf("A[%d]: ",       scanf("%d", &A [i] );
          i+1);
}

ffoorr((iii==00;;iii << MMAAXX;; iii+
          if( (i+1)%4==0) printf("\n");
     }
}
          pprriinntf"%%4dd\
```

Xuất dữ liệu trong mảng→Ví dụ 1→Thực hiện

```
A[1]: 12
A[2]: 14
A[3]: 15
A[4]: 26
A[5]: 27
A[6]: 48
A[7]: 56
A[8]: 68
A[9]: 50
A[10]: 30
A[11]: 19
A[12]: 14
```

```
12   14   15   26
27   48   56   68
50   30   19   14
```

```
12 14 15 26 27 48 56 68 50 30 19 14
```

```
12
14
15
26
27
48
56
68
50
30
19
14
```

Ví dụ 2: Nhập và đưa ra màn hình một ma trận

```
1.  #include <stdio.h>
2.  void main(){
3.      int A[20][20], n, m, i,j;
4.      printf("Nhap so hang : ");    scanf("%d",&n);
5.      printf("Nhap so cot : ");     scanf("%d",&m);
6.      printf("\n");
7.      for ( i=0; i < n; i+
+ )  8.    for(j=0; j < m; j++) {
9.              printf("Nhap phan tu A[%d,%d]: ", i+1,j+1);
10.              scanf("%d", &A[i][j] );
11.     }
12.     printf("\n\n MA TRAN DA NHAP \n\
n");  13.    for ( i=0; i < n; i++ ){
14.        for(j=0; j < m; j++)
15.             printf( "%4d" ,A[i][j]);
16.        printf("\n");
17.     }
1-
```

Ví dụ 2→Kết quả thực hiện

```
Nhap so hang : 2
Nhap so cot : 4

Nhap phan tu [1,1]: 12
Nhap phan tu [1,2]: 34
Nhap phan tu [1,3]: 54
Nhap phan tu [1,4]: 3
Nhap phan tu [2,1]: 123
Nhap phan tu [2,2]: 872
Nhap phan tu [2,3]: 12
Nhap phan tu [2,4]: 34


 MA TRAN DA NHAP

  12   34   54    3
 123  872   12   34
```

Đếm số phần tử thỏa mãn điều kiện

- Duyệt từng phần tử của dãy (*dùng **for***)
- Nếu phần tử xét thỏa mãn điều kiện
  - Ghi nhận
- Chuyển sang xem xét phần tử tiếp theo

Ví dụ: Đếm số tháng có lượng mưa lớn hơn 50mm

```
int dem = 0;
for(i = 0; i < MONTHS; i++)
    if(rainfall[i] > 50)
        dem++;
printf("\nThang mua nhieu hon 50mm: %d", dem);
```

Ví dụ: Nhập mảng, đưa ra TBC các số chia hết cho 7

```
#include<stdio.h>
void main(){
    int
    A[100];
int n, i, d =
    0, S=0;
printf("\n So
    phan tu
    cua mang
    (<100) :
    ");
    scanf("%
    d",&n);
for(i = 0; i <
    n; i++){
       printf("A
```

Tìm kiếm phần tử

Tìm phần tử lớn nhất (*nhỏ nhất*)

- Giả sử phần tử đó là phần tử đầu tiên
- Lần lượt so sánh với các phần tử còn lại
  - Nếu phần tử mới của dãy lớn hơn⇒ coi đây là phần tử lớn nhất và tiếp tục so sánh với phần tử kế
  - Nếu không đúng, so sánh tiếp với phần tử kế

Ví dụ: Tìm tháng có lượng mưa nhiều nhất trong năm

```
max = rainfall[0];
for(i = 1; i < MONTHS; i++)
    if(rainfall[i] > max)
        max = rainfall[i];
printf("\n Luong mua nhieu nhat la: %d", max);
```

Tìm kiếm phần tử

- Tìm kiếm các phần tử thỏa mãn điều kiện (*giống bài toán đếm*)
  - Dùng for duyệt toàn bộ
  - Nếu cần thiết, dùng thêm mảng ghi lại chỉ số

Ví dụ: Đưa ra danh sách các tháng có lượng mưa nhiều hơn 50mm

```
printf(“Thang co luong mua lon hon 500mm”)
   for(i = 0; i < MONTHS; i++)
        if(rainfall[i] > 50)  printf("\
            nThang %d", i+1);
```

Tìm kiếm phần tử (tiếp)

- Tìm phần tử đầu tiên của danh sách
  - Dùng vòng lặp **for** kết hợp với **break**;
  - Dùng vòng lặp **while**

Ví dụ

Đưa ra phần tử đầu của mảng có giá trị bằng k;

Tìm kiếm phần tử →Ví dụ

```
int Table[100]
int N, i, k, f;//N: số phần tử, k phần tử cần tìm
```

Dùng for

```
    for(i = 0; i < N; i++)
        if(Table[i] == k) break;
if(i< N) printf("Tim thay tai
    vi tri %d",i);
```

Dùng while

```
i=0; f =0;                    //f:
    found. f = 1 ⇔ k is found
    while(i < N && f==0){
       if(Table[i] == k) f = 1;
```

Bài tập 1

1. Nhập vào dãy (<100) số, tính và đưa ra màn hình
   - Tổng và tích của dãy số
   - Các số chia hết cho 3 và lớn hơn 10
   - Đếm các số nằm trong đoạn [100,1000)
2. Nhập vào một dãy số; tìm số chẵn nhỏ nhất dãy
3. Nhập dãy số; đếm xem có bao nhiêu bộ 3 số thỏa mãn điều kiện $x_i=(x_{i-1}+x_{i+1})/2$
4. Viết chương trình nhập vào từ bàn một dãy số (<100 phần tử). Đưa ra số bé nhất và vị trí những số bằng số bé nhất
5. Nhập vào n và dãy số $(x_1,x_2,\ldots x_n)$;$(y_1,y_2,..y_n)$ rồi tính

$a)\ \sum_{i=1}^{n} \cos x_i \sin x_i$ $b)\ \sqrt{\sum_{i}^{n}(x_i - y_i)^2}$ $\sum_{1}^{n-1} x_{i+1} y_{i+1}$

Bài 1

Bài toán sắp xếp theo thứ tự

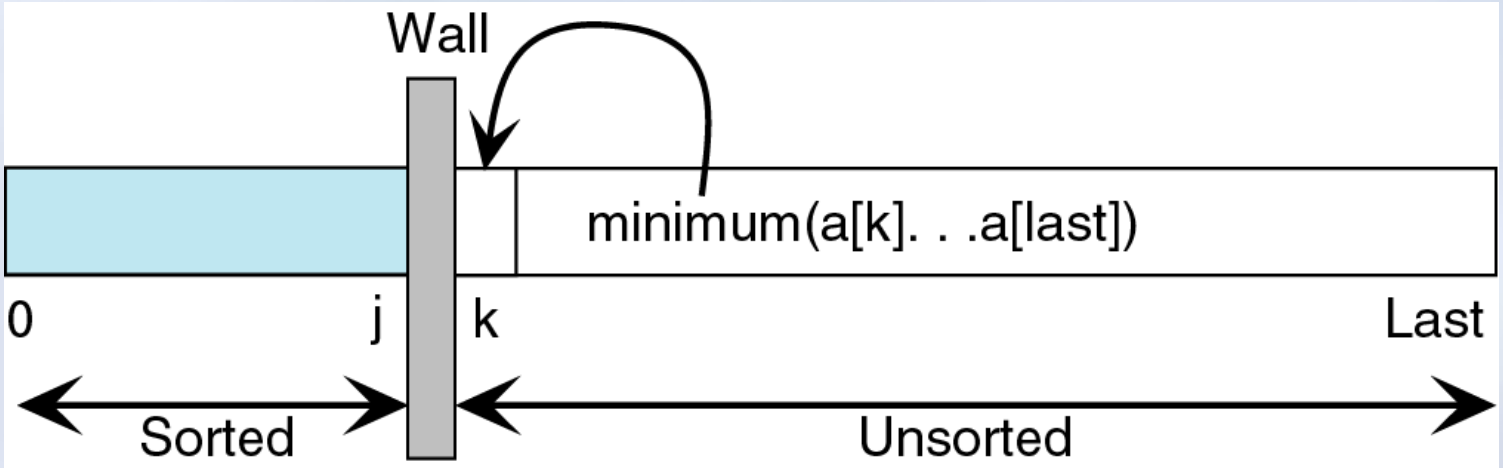
- Cho mảng phần tử, sắp xếp theo thứ tự tăng/giảm
- Các thuật toán
  - Sắp xếp thêm dần (insertion sort)
  - Sắp xếp lựa chọn (selection sort)
  - Sắp xếp nổi bọt (bubble sort)
  - Sắp xếp vun đống (heap sort)
  - Sắp xếp nhanh (quick sort)
  - Sắp xếp trộn (merge sort)
  - ….

Bài toán sắp xếp tăng →Thuật toán lựa chọn

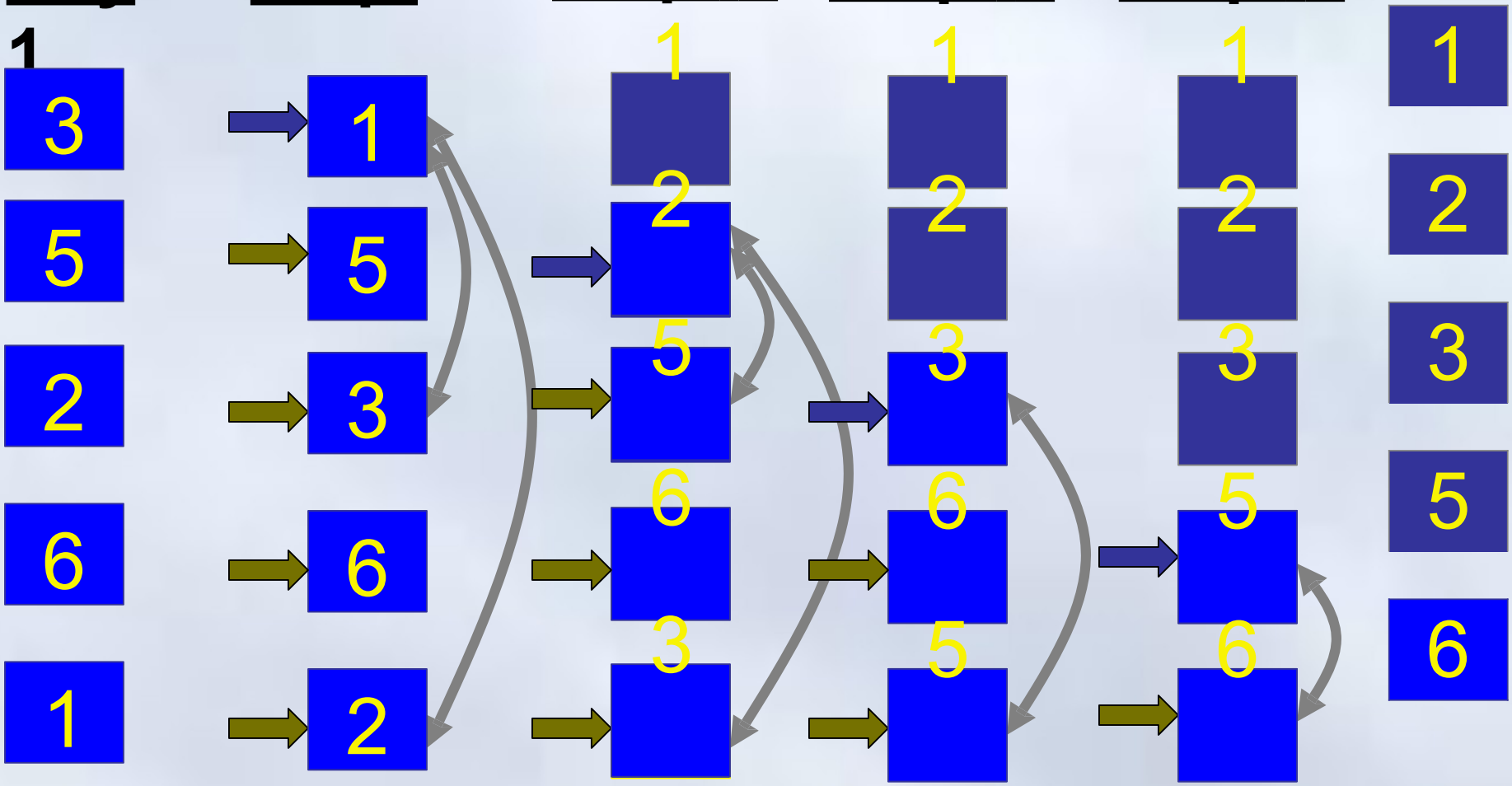
Nguyên tắc

Tại lượt sắp thứ k, tìm phần tử nhỏ nhất trong số các phần tử chưa được sắp xếp ([k..last]) và đổi chỗ cho phần tử thứ k (*có chỉ số k-1*)

- Khi k = 1, phần tử thứ nhất (chỉ số 0) đúng vị trí
- Khi k= 2, phần tử thứ hai (chỉ số 1) đúng vị trí…



Bài toán sắp xếp tăng →Thuật toán lựa chọn



Bài toán sắp xếp tăng →Thuật toán lựa chọn

Khai báo các biến

```
    int A[100];        //Mảng chứa dữ liệu
    int N, i, j, tmp;
//Sắp xếp
    for(i = 0; i < N – 1; i++)
          for(j = i + 1; j < N; j++)
              if(A[i] > A[j]) {
                        tmp = A[i];
                              A[i] = A[j];
                        A[j] = tmp;
              }
```

Ví dụ

Nhập vào từ bàn phím một mảng các số nguyên không quá 100 phần tử

Hiển thị dãy số vừa nhập

Sắp xếp dãy theo thứ tự **giảm** dần

Hiện thị dãy tại mỗi lượt sắp xếp

Ví dụ

```
1.  #include<stdio.h>
2.void main(){  3.
        int A[100] ;
4.      int N, i, j ,
        t;
5.      printf("So
        phan tu [<
        100]: ");
        scanf("%d
        ",&N);
6.      printf("Hay
        nhap day
        so...\n");
7.      for(i=0; i <
N; i++){
```

Ví dụ

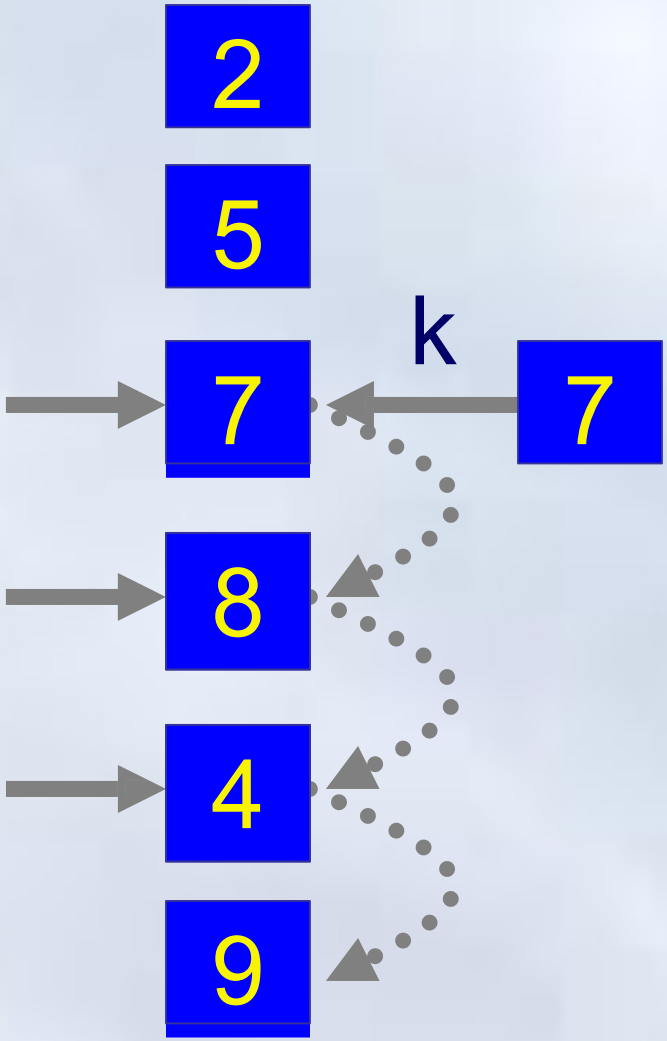
```
14.     printf("Sap xep day theo thuat toan lua chon");
15.     for(i=0; i < N-1; i++){
16.             for(j=i+1; j < N; j++)
17.                     if(A[i] < A[j]) {
18.                             t = A[i];
19.                             A[i] = A[j];
20.                             A[j] = t;
21.                     }//if & for_j
22.             printf("\nLuot %d : ",i+1);
23.             for(j=0; j < N; j++)
24.                     printf("%4d", A[j]);
25.     }//for_i
26. }//main
```

Ví dụ→Kết quả

```
So phan tu [< 100]: 6
Hay nhap day so...
A[1] = 5
A[2] = 32
A[3] = 67
A[4] = 12
A[5] = 24
A[6] = 16
```

```
Day vua nhap...
    5   32   67   12   24   16

Sap xep day theo thuat toan lua chon
Luot 1 :    67    5   32   12   24   16
Luot 2 :    67   32    5   12   24   16
Luot 3 :    67   32   24    5   12   16
Luot 4 :    67   32   24   16    5   12
Luot 5 :    67   32   24   16   12    5
```

Bài toán chèn phần tử **a** vào vị trí **k**




```
for(i = N; i > k; i--)
   A[i] = A[i-
1];  A[k] = a;
   N= N + 1;
```

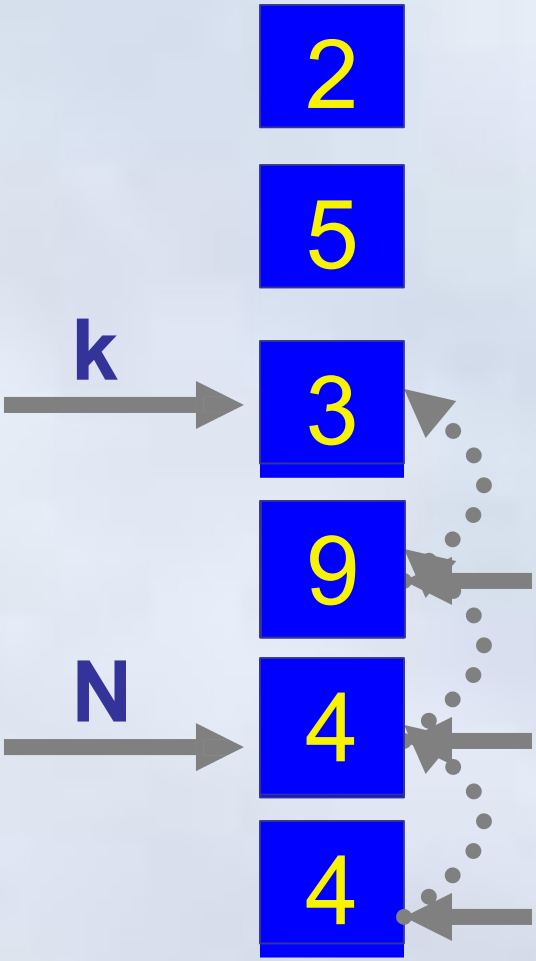
Chú ý:

N = MAX: không chèn được

k > N → Chèn vào vị trí N;

k< 0 → Chèn vào vị trí 0

Bài toán xóa phần tử ở vị trí **k** (0≤k<N)




```
for(i = k+1; i < N; i++)
    A[i-1] = A[i];
N= N - 1;
```

Bài tập 2

1. Nhập vào từ bàn phím một dãy số nguyên (<100 phần tử). Sắp xếp dãy theo nguyên tắc: Bên trên là số chẵn chia hết cho 3. Bên dưới là số lẻ chia hết cho 3. Giữa là các số còn lại. Đưa cả 2 dãy ra màn hình.
2. Đọc vào dãy số có n phần từ (n<100). Đọc số x và số k nguyên. Chèn x vào vị trí k của dãy. Nếu k>n, chèn x vào vị trí n+1.
3. Nhập vào một dãy số (<100 phần tử) và sắp xếp theo thứ tự tăng dần. Nhập thêm vào một số và chèn số mới nhập vào đúng vị trí
4. Nhập vào một dãy (<100 phần tử); xóa đi các phần tử chia hết cho 5 và đưa kết quả ra màn hình

Bài chữa

```
#include<stdio.h>
 void main(){
        int A[100];
        int N, i;
        //Nhập dữ
        liệu
        printf("So phan tu : "); scanf("%d",&N);
         for(i=0; i < N; i ++){
            printf("A[%d] =
            ",i);scanf("%d",&A[i]);
        }
        //Các thao tác xử lý mảng: chèn, xóa,
        sắp xếp,…
        //Đưa Dữ liệu ra
        for(i=0; i < N; i ++)
```

Bài chữa → Sắp xếp số chẵn chia hết 3 lên đầu dãy..

```
1.  { int d = 0, t;
2.    for(i=0;i < N; i++)
3.        if(A[i]%6==0){
4.            t=A[i];  A[i]=A[d]; A[d] = t;
5.            d++;
6.        }
7.    for(i=d; i < N; i++)
8.        if(A[i]%3 != 0){
9.            t=A[i];   A[i]=A[d];  A[d] = t;
10.           d++;
11.       }
12. }
```

Bài chữa → Sắp xếp tăng dần và chèn đúng vị trí

```
1.   {  int k = 0,i, j, t;
2.       for(i=0;i < N - 1; i++)//Sắp xếp lựa chọn
3.                  for(j=i+1;j < N ; j++)
4.         if(A[i] > A[j]){
5.                  t = A[j];
6.                           A[j] = A[i];
7.                           A[i]=t;
8.         }
9.    printf("Phan tu moi:");
10.      scanf("%d",&k);//Nhập p/tử mới
11.  i = N;          //Chèn đúng vị trí
12.      while( (i > 0) &&(A[i-1] > k) ){
13.      A[i] = A[i-1];  i--;
14.  }
15.  A[i] = k;
     }
     N++;
```

Bài chữa → Xóa các phần tử chia hết cho 5..

```
1.  { // PP: Giữ lại các phần tử không chia hết cho 5
2.          int d = 0, i;
3.          for(i=0;i < N; i++)
4.                  if(A[i] % 5 != 0){
5.                          A[d] = A[i];
6.                          d++;
7.                  }
8.          N = d;
9.  }
```

Xóa các phần tử chia hết cho 5 →Kết quả

```
So phan tu : 12
A[0] = 5
A[1] = 1
A[2] = 21
A[3] = 15
A[4] = 10
A[5] = 34
A[6] = 23
A[7] = 45
A[8] = 54
A[9] = 32
A[10] = 12
A[11] = 50
```

```
Day ban dau :  5  1  21  15  10  34  23  45  54  32  12  50
```

```
Day sau khi xoa :  1  21  34  23  54  32  12
```

Bài tập 3 : Ma trận

1. Viết chương trình nhập vào một ma trận vuông, các phần tử nguyên, sau đó
   - Đưa ra ma trận tam giác dưới
   - Đưa ra ma trận tam giác trên
2. Nhập M, N (M, N < 30) và một ma trận MxN. Đưa ma trận ra màn hình
   - Tìm hàng/cột có tổng các phần tử lớn nhất
   - Tìm số lớn nhất/nhỏ nhất và vị trí trong ma trận
   - Đưa ra ma trận S cùng kích thước thỏa mãn

$$s_{i,j} = \begin{cases} 1 & \text{nê'u } u_{i,j} > 0 \\ 0 & \text{nê'u } u_{i,j} = 0 \\ -1 & \text{nê'u } u_{i,j} < 0 \end{cases}$$

Nhập vào một ma trận vuông,..

```
#include <stdio.h>
void main(){
      int A[20][20], N,i,j;
      printf("Nhap kich thuoc : "); scanf("%d",&N);
       printf("\n");
      for ( i=0; i < N; i++ )
         for(j=0; j < N; j++)    {
           printf("Nhap phan tu [%d,%d]:", i+1,j+1);
              scanf("%d", &A[i][j] );
      }
      printf("\n\n MA TRAN DA NHAP \n\n");
       for ( i=0; i < N; i++ ){
         for(j=0; j < N; j++)
               printf( "%4d" ,A[i][j]);
          printf("\n");

}
```

Đưa ra ma trân tam giác trên, dưới

```
    printf("\n\n MA TRAN TAM GIAC TREN \n\n");
    for ( i=0; i < N; i++ ){
        for(j=0; j < N; j++)
        if(j >= i)
                printf( "%4d"
                   ,A[i][j]);
        else
             printf("%4c",32); //32 là mã ASCII của dấu cách
    }  printf("\n");
    printf("\n\n MA TRAN TAM GIAC DUOI \n\n");
     for ( i=0; i < N; i++ ){
       for(j=0; j <= i; j++)
             printf( "%4d" ,A[i][j]);
   printf("\n");
}
}//main
```

Thực hiện

```
Nhap kich thuoc : 4

Nhap phan tu [1,1]: 12
Nhap phan tu [1,2]: 34
Nhap phan tu [1,3]: 52
Nhap phan tu [1,4]: 9
Nhap phan tu [2,1]: 10
Nhap phan tu [2,2]: 52
Nhap phan tu [2,3]: 54
Nhap phan tu [2,4]: 75
Nhap phan tu [3,1]: 8
Nhap phan tu [3,2]: 12
Nhap phan tu [3,3]: 45
Nhap phan tu [3,4]: 7
Nhap phan tu [4,1]: 11
Nhap phan tu [4,2]: 42
Nhap phan tu [4,3]: 22
Nhap phan tu [4,4]: 34
```

```
MA TRAN DA NHAP

 12   34   52    9
 10   52   54   75
  8   12   45    7
 11   42   22   34
```

```
MA TRAN TAM GIAC TREN

 12   34   52    9
      52   54   75
           45    7
                34
```

```
MA TRAN TAM GIAC DUOI

 12
 10   52
  8   12   45
 11   42   22   34
```

Nội dung chính

1. Mảng

- Khái niệm
- Khai báo và sử dụng
- Các thao tác thường gặp

2. Con trỏ

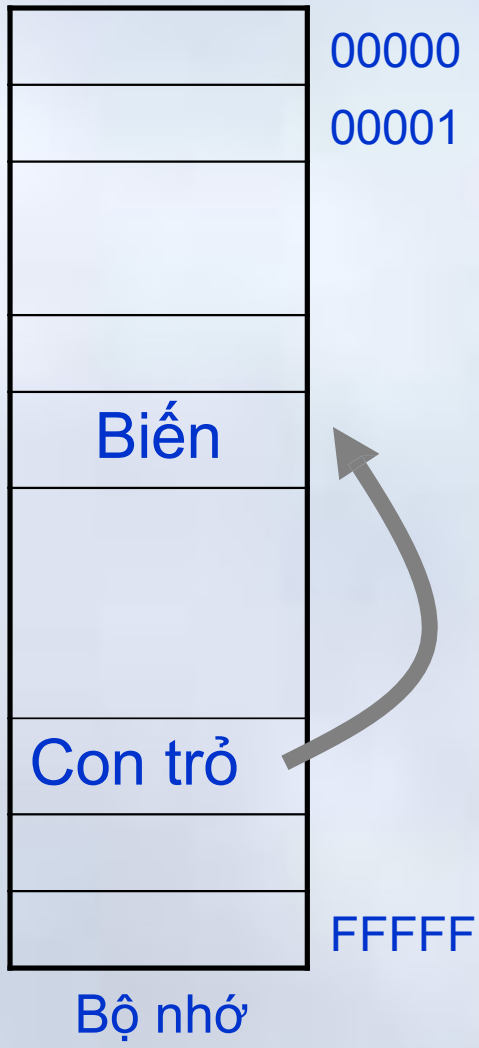
- Khái niệm và cách khai báo
- Toán tử địa chỉ (&), toán tử nội dung (*)
- Phép toán trên con trỏ
- Con trỏ và mảng

3. Xâu ký tự

- Khái niệm, khai báo và sử dụng
- Các hàm xử lý ký tự và xâu ký tự
- Mảng xâu ký tự

Giới thiệu

- Là một khái niệm "*mạnh*" trong C
  - Cho phép tính toán trên con trỏ
  - Sử dụng con trỏ hàm
- Cho phép truy nhập gián tiếp tới một đối tượng có địa chỉ (*biến, hàm*)
  - Truy nhập trực tiếp→ thông qua tên

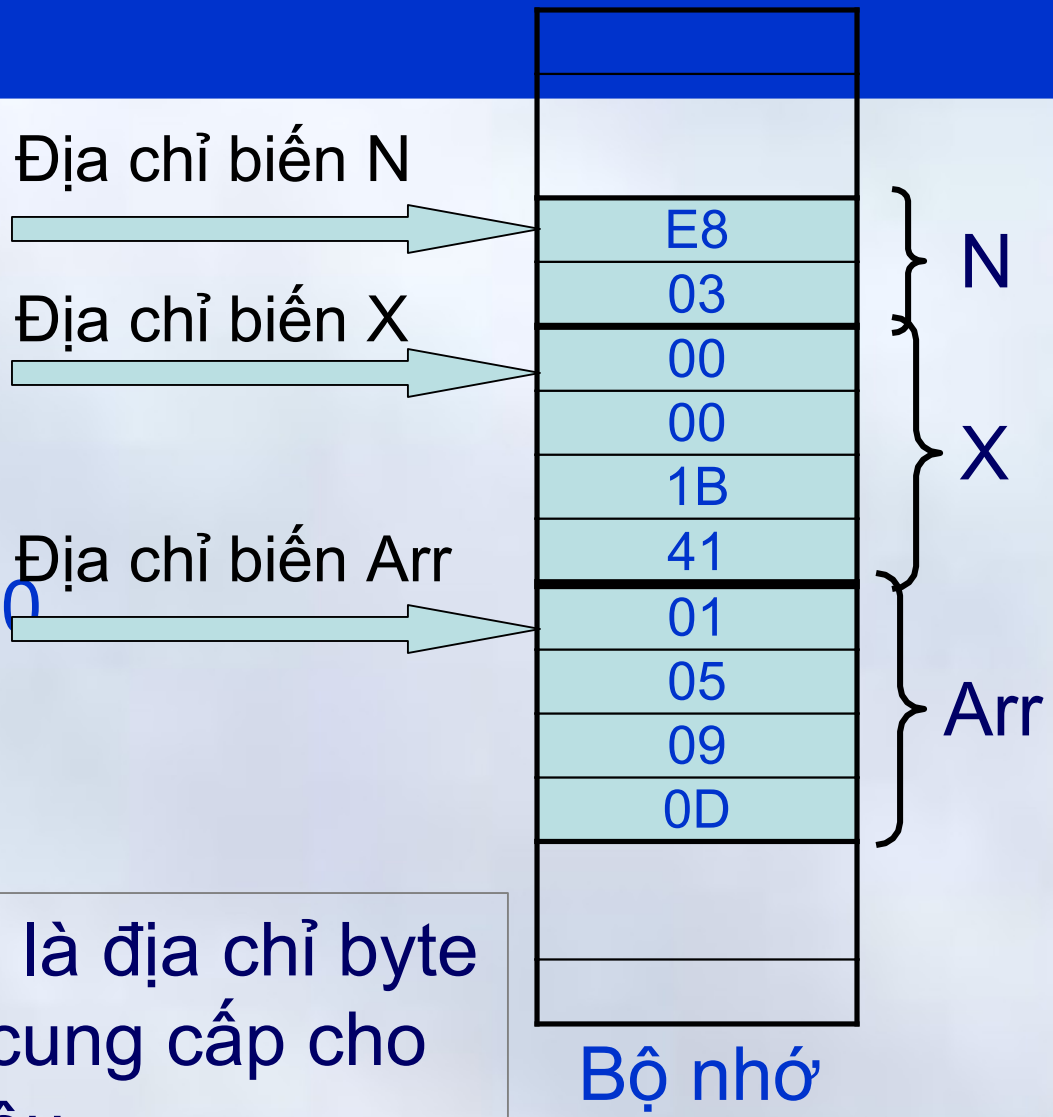


Địa chỉ

- Bộ nhớ gồm dãy các ô nhớ
  - Mỗi ô nhớ là một byte
  - Mỗi ô nhớ có một địa chỉ riêng
- Các biến trong chương trình được lưu tại vùng nhớ nào đó trong bộ nhớ
- Khi khai báo biến, tùy thuộc vào kiểu, biến sẽ được cấp một số ô nhớ liên tục nhau
  - Biến int được cấp 2 bytes, float được cấp 4 bytes,..
  - Địa chỉ của biến, là địa chỉ của byte đầu tiên trong số các byte được cấp
  - Khi gán giá trị cho biến, nội dung các byte cung cấp cho

Địa chỉ →Ví dụ

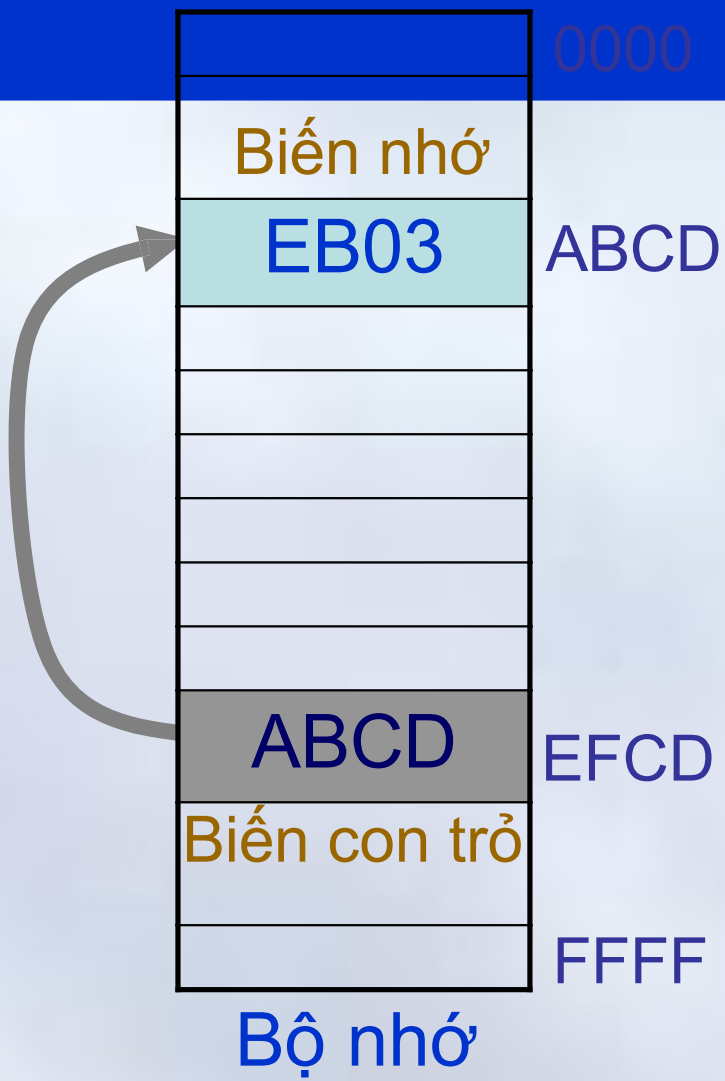
```
float x;
char
Arr[4];
N = 1000;//03E8
X=9.6875;//411B000
0
for(i=0;i<4;i++)
    Arr[i]=
    4*i+1;
```



Địa chỉ của một biến là địa chỉ byte nhớ đầu tiên được cung cấp cho biến để lưu trữ dữ liệu

Con trỏ

- Con trỏ là **một biến** mà **giá trị của nó là địa chỉ** của một vùng nhớ
  - Vùng nhớ này có thể dùng để chứa các biến có kiểu cơ bản (*nguyên, thực, ký tự, …*) hay có cấu trúc (*mảng, bản ghi,..*)
- Con trỏ dùng "***trỏ tới***" một biến nhớ
  - Có thể trỏ tới một hàm
  - Có thể trỏ tới con trỏ khác



Con trỏ →Khai báo

Kiểu * Tên;

- **Tên:** Tên của một biến con trỏ
- **Kiểu:** Kiểu của biến mà con trỏ "Tên" trỏ tới
  - Giá trị của con trỏ có thể thay đổi được
    - Trỏ tới các biến khác nhau, có cùng kiểu
  - Kiểu biến mà con trỏ trỏ tới không thay đổi được
    - Muốn thay đổi phải thực hiện "***ép kiểu***"

Ví dụ:

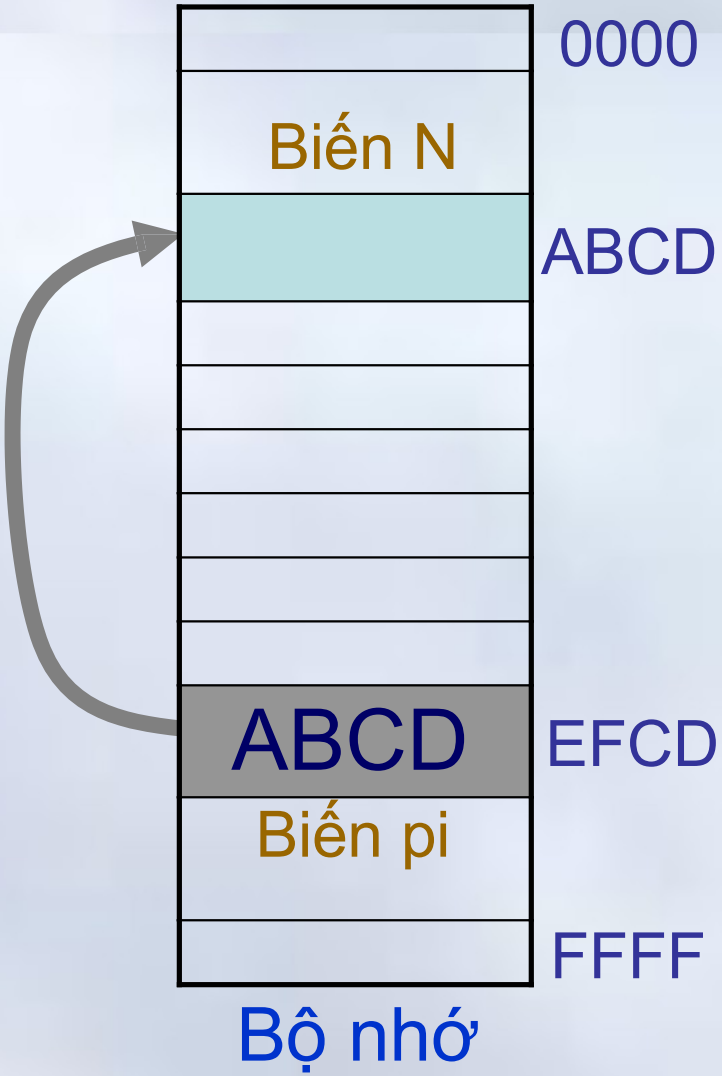
```
int * pi;          //Con trỏ, trỏ tới một biến kiểu nguyên
char * pc; //Con trỏ, trỏ tới một biến kiểu ký tự
```

Toán tử địa chỉ (&)

- Ký hiệu: **&**
- Là toán tử một ngôi, trả về địa chỉ của biến
  - Địa chỉ biến có thể được gán cho một con trỏ, trỏ tới đối tượng cùng kiểu

Ví dụ

```
int N; // &N→ ABCD
   int * pi;
pi = &N; // pi←ABCD
```



Toán tử nội dung (*)

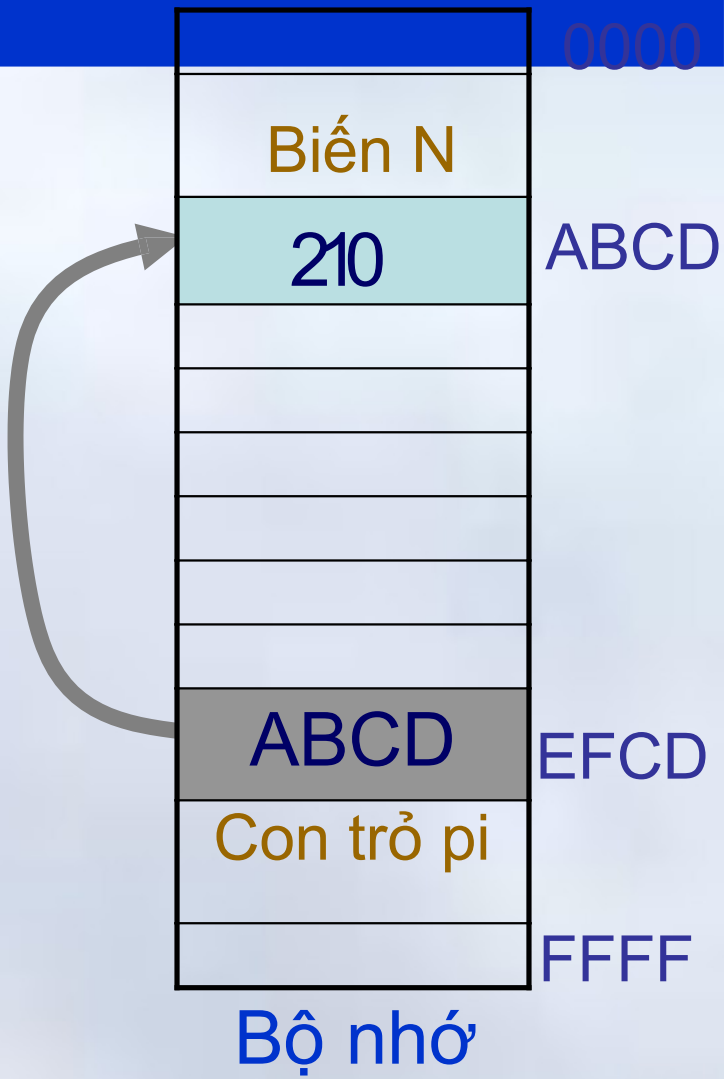
- Là toán tử một ngôi, trả về giá trị (nội dung) của vùng nhớ mà con trỏ đang trỏ tới

Ví dụ

```
int N;
int * pi;
pi = &N;
```

⇔ `N= 10;` ⇔ `*pi = 10;`

```
N = 10;//Vùng nhớ mà pi trỏ tới
    mang giá trị 10; Vậy *pi=10
*pi = 20;// Vùng nhớ pi trỏ tới được
    gán giá trị 20; Vậy N= 20
```



Gán giá trị cho con trỏ

- Con trỏ được gán địa chỉ của một biến
  - Biến cùng kiểu với kiểu mà con trỏ trỏ tới
    - Nếu không, cần phải ép kiểu
- Con trỏ được gán giá trị của con trỏ khác
  - Hai con trỏ sẽ trỏ tới cùng một biến(do cùng địa chỉ)
  - Hai con trỏ nên cùng kiểu trỏ đến
    - Nếu không, phải ép kiểu
- Con trỏ được gán giá trị NULL

  Ví dụ: int *p; p = 0;
- Gán nội dung vùng nhớ 2 con trỏ trỏ

Ví dụ

```
void main(){
    int N=5, M=10;
    int *p1 = &N;
     int *p2 =
    &M;
    *p1 = *p2;
    printf("%d
    %d",*p1,*p2)
    ;
}
```

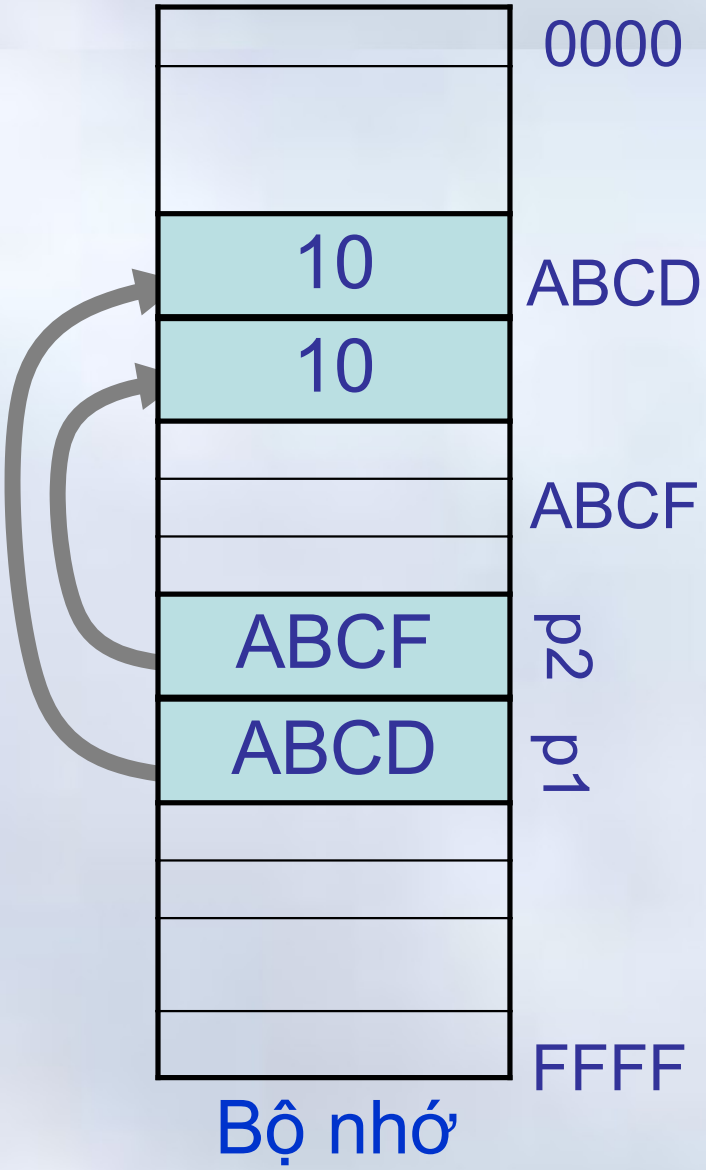
10 10

```
#include <stdio.h>
void main(){
    int N=5, M=10;
    int *p1 = &N;
       int   *p2   =
    &M;     p1   =
    p2;
    printf("%d
    %d",*p1,*p2)
    ;
}
```

10 10

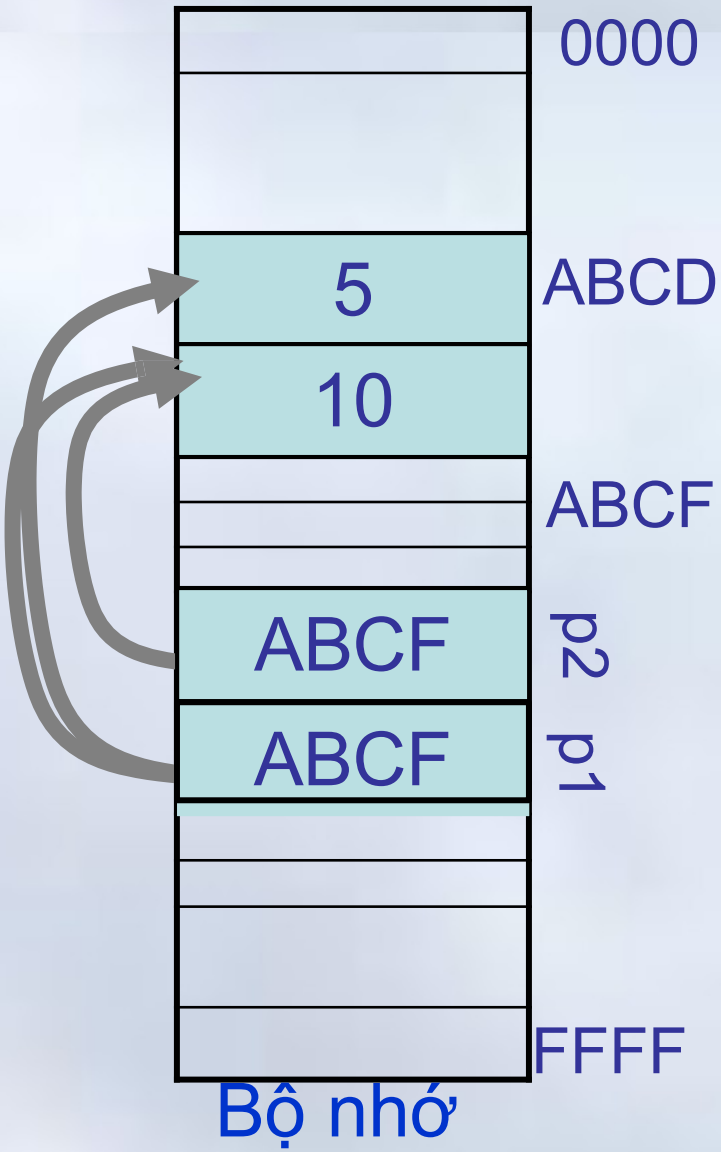
Ví dụ→Trường hợp 1

```
1. #include <stdio.h>
2. void main(){
3.       int N=5, M=10;
4.       int *p1 = &N;
5.       int *p2 = &M;
6.       *p1 = *p2;
7.       printf("%d %d",*p1,*p2);
8. }
```



Ví dụ→Trường hợp 2

```
1. #include <stdio.h>
2. void main(){
3.       int N=5, M=10;
4.       int *p1 = &N;
5.       int *p2 = &M;
6.       p1 = p2;
7.       printf("%d %d",*p1,*p2);
8. }
```



Các phép toán trên con trỏ

- Cộng con trỏ với một số nguyên
  - Kết quả: Con trỏ cùng kiểu
- Trừ con trỏ với một số nguyên
  - Kết quả: Con trỏ cùng kiểu
- Trừ 2 con trỏ cùng kiểu cho nhau
  - Kết quả: Một số nguyên
    - Khoảng cách giữa 2 con trỏ được đo bằng số phần tử thuộc kiểu dữ liệu mà con trỏ trỏ tới

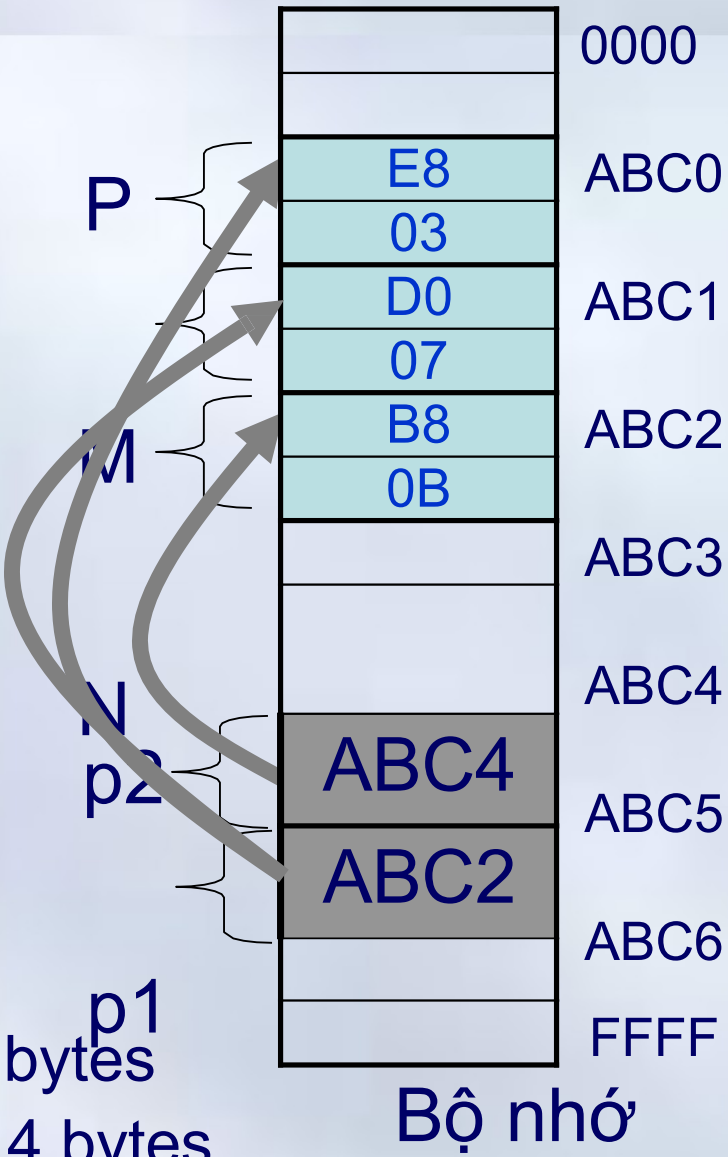
Các phép toán trên con trỏ→Ví dụ

```
int N=1000, M=2000,P=3000;
int * p1 = &P, *p2 = &N;
```

p1 - p2 → **-2**

* (p2-1) → 2000

* ++ p1 → 2000



Ghi chú:

- Kiểu **int**, các phần tử cách nhau 2 bytes
- Kiểu **float**, các phần tử cách nhau 4 bytes

Mối quan hệ giữa con trỏ và mảng một chiều

- Nếu **T** là tên một mảng ⇒ **T** là một **con trỏ hằng** chứa địa chỉ của phần tử đầu tiên của mảng **T** (&T [0])
  - Không tồn tại phép tính trên tên mảng, hoặc gán giá trị cho tên mảng (*VD: T=…; T++*)
- Có thể sử dụng một con trỏ để duyệt mảng nếu nó được gán giá trị bằng địa chỉ của mảng (*địa chỉ của phần tử đầu tiên*)

Ví dụ

```
int A[10];
int * p = A;  // int *p = &A[0]
```

```
for(i = 0; i < 10; i ++)
    printf("%d ", *(p + i) );
```

```
for(i = 0; i < 10; i ++)
    printf("%d ", p[i]);
```

```
for(i = 0; i < 10; i ++)
    printf("%d ", *(p++) );
```

Con trỏ void

void * Tên_con_trỏ

- Là một con trỏ đặc biệt: con trỏ không có kiểu
- Có thể nhận giá trị là địa chỉ của một biến có kiểu dữ liệu bất kỳ
  - Thường dùng làm đối số trong lời gọi hàm để có thể nhận bất kỳ kiểu địa chỉ nào của tham số được truyền

Ví dụ:

```
void * p, *q;
int n;      float
x;
p= & n;
```

Câu hỏi 1

```
#include<stdio.h>
 int main()
{
    int a=3, *p;
     p = &a;
    printf("%d\n", a * *p * a + *p);
     return 0;
}
```

30

Câu hỏi 2

```
#include<stdio.h>
 int main(){
    int arr[2][2][2]
    = {10, 2, 3, 4,
    5, 6, 7, 8};

    int *p, *q;
    q = (int *) arr;
    p = &arr[1][1]
    [1];
    printf("%d, %d\n", *p, *(q+4) );
     return 0;
}
```

8, 5

Nội dung chính

1. Mảng

- Khái niệm
- Khai báo và sử dụng
- Các thao tác thường gặp

2. Con trỏ

- Khái niệm và cách khai báo
- Toán tử địa chỉ (&), toán tử nội dung (*)
- Phép toán trên con trỏ
- Con trỏ và mảng

3. Xâu ký tự

- Khái niệm. khai báo và sử dụng
- Các hàm xử lý ký tự và xâu ký tự
- Mảng xâu ký tự

Khái niệm xâu ký tự

- Xâu kí tự (string) là một dãy các kí tự viết liên tiếp nhau
  - Độ dài xâu là số kí tự có trong xâu
  - Xâu không có kí tự nào: Xâu rỗng
- Ví dụ: “Tin hoc”, “String”
- Lưu trữ: kết thúc xâu bằng kí tự ‘\0’ hay NULL (mã ASCII là 0)

‘T’	‘i’	‘ n ‘	‘ ‘	‘h’	‘o’	‘c’	‘\0’

Khái niệm xâu ký tự→Lưu ý

- Xâu kí tự >< mảng kí tự
  - Tập hợp các kí tự viết liên tiếp nhau
    - Truy nhập một phần tử của xâu ký tự (*là một ký tự*) giống như truy nhập vào một phần tử của mảng: Tên[Chỉ_số]
  - Xâu kí tự có kí tự kết thúc xâu, mảng kí tự không có kí tự kết thúc xâu
- Xâu kí tự độ dài 1 >< kí tự *("A" ='A' ?)*
  - 'A' là 1 kí tự, được lưu trữ trong 1 byte
  - "A" là 1 xâu kí tự, ngoài kí tự 'A' còn có kí tự '\0'
    => được lưu trữ trong 2 byte

Khai báo

```
char    tên_xâu [số_kí_tự_tối_đa];
```

- Để lưu trữ một xâu có **n** kí tự chúng ta cần một mảng có kích thước **n+1**
  - Phần tử cuối cùng chứa ký tự NULL

Ví dụ

- Để lưu trữ xâu “Tin hoc” chúng ta phải khai báo xâu có số phần tử tối đa ít nhất là 8

char str[8] =“Tin hoc”;	‘T’	‘i’	‘n’	‘ ’	‘h’	‘o’	‘c’	‘\0’

Truy nhập phần tử của xâu

Giống như truy nhập tới một phần tử của mảng ký tự

tên_xâu [chỉ_số_của_kí_tự]

Ví dụ: char Str[10] = “Tin hoc”;

T	i	n	-	h	o	c	\0	?1	\?0

Str[0] → ‘T’
Str[3] → ‘ ’
Str[7] → ‘\0 ‘
Str[8] → ?

Str[3] = ‘-’;
Str[7] = ‘ ’;
Str[8] = ‘1 ‘ ;
Str[9] = ‘\0’;

Str: Tin-hoc 1

Ví dụ: Nhập xâu và đếm số ký tự '*'

```
#include <stdio.h>
void main(){
  char Str[100];

  int d=0, i=0;
  printf("Nhap xau ky tu: "); gets(Str);
   while(Str[i] != '\0'){
    if(Str[i]=='*')
      d++;
     i++;
    }
     printf(
   "Ket
```

Tính chiều dài của xâu

```
  d=0;
while(Str[d] != '\0') d+
  +;
```

```
Nhap xau ky tu: **  abc** dd*e**e*dd
Ket qua : 8_
```

```
Nhap xau ky tu: ******  ******  **
Ket qua : 12
```

Ví dụ: Nhập câu và đưa ra dưới dạng cột

```
1.  #include <stdio.h>
2.  void main(){
3.      char S[100];
4.      int i=0;
5.      printf("Nhap xau: "); gets(S);
6.      while(S[i] != '\0'){ //32 là mã ASCII của phím space
7.          if(S[i] != 32 && S[i+1]==32) printf("%c\n",S[i]);
8.          else  if(S[i] != 32)
printf("%c",S[i]);  9.  i++;
10.     }
11.     printf("\n\n");
12.
}
```

```
Nhap xau:   Dai hoc   Bach Khoa
Dai
hoc
Bach
Khoa
```

Đếm số từ, nếu các từ được cách nhau bởi dấu phân cách

Các hàm xử lý ký tự

Tệp tiêu đề : ctype.h

#include <ctype.h>

Các hàm xử lý ký tự→Chuyển đổi chữ hoa/thường

- int **toupper**(char ch):
  - Chuyển kí tự thường thành kí tự hoa

    toupper('a') => 'A'
- int **tolower**(char ch)
  - Chuyển kí tự hoa thành kí tự thường

    tolower('B') => 'b'

Ví dụ

```
do{
  ……
  printf("Tiep tuc <C/K>? :");
  fflush(stdin);
}while(toupper(getche()) !='K');
```

Các hàm xử lý ký tự →Kiểm tra chữ hoa/thường

- int **islower**(char ch)
  - Kiểm tra chữ thường:
    - Hàm trả về giá trị khác 0 nếu ch là chữ thường, ngược lại trả về 0
    - Ví dụ: printf("%d ",islower('A')); ⇒ 0
- int **isupper**(char ch):
  - Kiểm tra chữ hoa:
    - Hàm trả về giá trị khác 0 nếu ch là chữ hoa, ngược lại trả về 0
    - Ví dụ: printf("%d ",isupper('A')); ⇒ ≠ 0 (1 !?)

Các hàm xử lý ký tự →Kiểm tra chữ cái/chữ số

- int **isalpha**(char ch):
  - Kiểm tra kí tự trong tham số có phải chữ cái không (‘a’…’z’,’A’,..’Z’). Hàm trả về khác 0 nếu đúng, ngược lại trả về giá trị bằng 0
  - Ví dụ: printf("%d ",isalpha('A')); ⇒ ≠ 0 (1 !?)
- int **isdigit**(char ch):
  - Kiểm tra kí tự trong tham số có phải chữ số (‘0‘,‘1‘,..‘9‘) không. Hàm trả về khác 0 nếu đúng, ngược lại trả về giá trị bằng 0
  - Ví dụ: printf("%d ",isdigit('A')); ⇒ 0

Khái niệm xâu ký tự→Kiểm tra ký tự đặc biệt

- int **iscntrl**(char ch)
  - Kiểm tra kí tự điều khiển (0-31).
  - Hàm trả về khác 0 nếu đúng, ngược lại trả về giá trị bằng 0
- int **isspace**(char ch)
  - Kiểm tra kí tự dấu cách (mã 32), xuống dòng (‘\n’ 10), đầu dòng (‘\r’ 13), tab ngang (‘\t’ 9), tab dọc (‘\v’ 11).
  - Hàm trả về khác 0 nếu đúng, ngược lại trả về giá trị bằng 0

Ví dụ: Nhập xâu, chuyển thành xâu chữ hoa

```
1.  #include <stdio.h>
2.  #include <ctype.h>
3.  void main(){
4.      int i =0;
5.      char S[50];
6.      printf("Nhap mot xau: ");
        gets(S);
7.printf("\n\nXau ban dau : %s. ",S);  8.
while(S[i] != '\0'){
9.              S[i] = toupper(S[i]);
10.             i = i + 1;
11.     }
12.     printf("Xau ket qua : %s",S);
13. }
```

```
Nhap mot xau: Study, Stydy more,..


Xau ban dau : Study, Stydy more,..
Xau ket qua : STUDY, STYDY MORE,..
```

Ví dụ: Nhập xâu và đếm từ, phân cách bởi dấu trắng

```
#include <stdio.h>
#include <conio.h>
            #include
<ctype.h>          int
main(){
    char Str[100]; int d=0, i=0;
    printf("Nhap xau ky tu: "); gets(Str);
     if(Str[0] == '\0') printf(“ Xau rong
     else{
          if( ! isspace(Str[0]) ) d=1;
           i=1;
          while(Str[i] != '\0'){
              if(
              isspace(Str[i-1] ) &&
              (! isspace(Str[i])) )
                                  d++;
              i++;
}
```

```
Nhap xau ky tu: Hello   world
Ket qua : 2_
```

```
Nhap xau ky tu:          Hello
Ket qua : 1
```

```
Nhap xau ky tu:    Hello
Ket qua : 1_
```

Các hàm xử lý xâu ký tự

Vào/ra xâu ký tự

- Tệp tiêu đề: **stdio.h**
- Nhập xâu kí tự
  - gets(tên_xâu);
  - scanf(“%s”,&tên_xâu);
- Hiển thị xâu kí tự
  - puts(tên_xâu);
  - printf(“%s”,tên_xâu);

Sự khác nhau giữa gets và scanf?

Các hàm xử lý xâu ký tự

Tệp tiêu đề: string.h

#include <string.h >

Chú ý:

char str[100] = "Hello world";

*char * p = str;*

- **p** là con trỏ, trỏ tới mảng các ký tự/xâu ký tự

  p+6 : (*Phép tính toán trên con trỏ*), cũng là xâu ký tự. p+6 trỏ tới xâu "world"

Các hàm xử lý xâu ký tự

size_t strlen(char * xâu)

- Trả về độ dài xâu

```
printf("%d ",strlen("Hello world")); ⇒ 11
```

char * strcpy(char * đích, char * nguồn)

- sao chép nội dung xâu ***nguồn*** vào xâu ***đích***, trả về giá trị xâu nguồn

```
char Str[20];
printf("%s ",strcpy(Str,"Hello")); ⇒ Hello
printf("%s", Str); ⇒ Hello
```

Chú ý: Phép gán Str = “Hello” là không hợp lệ

Các hàm xử lý xâu ký tự

int strcmp(char * xâu_1, char * xâu_2)

- So sánh hai xâu.
- Trả về giá trị 0 nếu hai xâu giống nhau;
- Giá trị < 0: xâu_1 < xâu_2
- Giá trị >0: xâu_1 > xâu_2

Ví dụ

```
char Str[20];
strcpy(Str,“hello“)
;
printf(“%d”, strcmp(Str,”hello”));→0
printf(“%d”, strcmp(Str,”hello!”); →-1 (!?)
printf(“%d”, strcmp(Str,”Hello”); → 1 (!?)
```

Các hàm xử lý xâu ký tự

char * strcat(char * xđích, char * nguồn)

– Ghép nối xâu nguồn vào ngay sau xâu đích, trả lại xâu kết quả

Ví dụ

```
char Str[20];
   strcpy(Str,"Hello ");
printf("%s ",strcat(Str,"world")); ⇒ Hello world
   printf("\n%s",Str); ⇒ Hello world
```

Các hàm xử lý xâu ký tự

char * strchr (char * s, int c)

– Trả về con trỏ trỏ tới vị trí xuất hiện đầu tiên của ký tự c trong s. Nếu không có trả về con trỏ null

strcpy(Str,"Hello world");

printf("%s ",strchr(Str,'o')); ⇒ o world

char* strstr(char * s1, char * s2)

– Trả về con trỏ trỏ tới vị trí xuất hiện đầu tiên của chuỗi s2 trong s1. Nếu không tồn tại, trả về con trỏ null

printf("%s ",strstr(Str,"llo")); ⇒ llo world

Các hàm xử lý xâu ký tự (tiếp)

Tệp tiêu đề: stdlib.h

- int **atoi**(char * str):
  - Chuyển một xâu kí tự thành một số nguyên tương ứng
  - ***Ví dụ:*** atoi(“1234”) →1234
- int **atol**(char * str):
  - Chuyển xâu kí tự thành số long int
- float **atof**(char * str):
  - Chuyển xâu kí tự thành số thực
  - ***Ví dụ:*** atof(“123.456E-2”) →1.23456
- **Thất bại cả 3 hàm: trả về 0**

Ví dụ 1: Nhập 2 xâu cho biết số lần xuất hiện xâu 1 trong xâu 2

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  void main(){
4.      int d =0;
5.      char S1[50],S2[20], *p;
6.          printf("Nhap xau thu nhat: ");          gets(S1);
7.          printf("Nhap xau thu hai: ");          gets(S2);
8.          p = strstr(S1,S2);
9.while(p !=NULL ){  10.
            d = d + 1;
11.                 p =
strstr(p+1,S2);//vi tri tim
kiem ke tiep
12.     }
```

```
Nhap xau thu nhat: abcabcabca
Nhap xau thu hai: bc
Xau "bc" x/hien trong xau "abcabcabca" 3 lan
```

```
Nhap xau thu nhat: aaaaa
Nhap xau thu hai: aa
Xau "aa" x/hien trong xau "aaaaa" 4 lan
```

Ví dụ 2: Kiểm tra xâu đối xứng

```
#include<stdio.h>
#include<string.h>
 main(){
    char s[20];
     int i,n;
    printf("Nhap vao xau ki tu: ");gets(s);
     n=strlen(s);
    for(i=0;i<n/2;i++)
        if(s[i]!=s[n-1-i])
            break;
     if(i==n/2)
        printf("Xau doi xung");
     else
            printf("Xau khong
        doi xung");
}
```

Ví dụ 3: Đảo ngược xâu ký tự

```
#include<stdio.h>
#include<string.h>
 main(){
    char s[100],c;
     int i, n;
    printf("Nhap xau: ");gets(s);
     n =strlen(s);
    for(i=0;i <n/2;i++){
          c = s[i];
          s[i] = s[n-i-1];
          s[n-i-1]=c;
    }
    printf("%s",s);
}
```

Ví dụ 4: Đếm số lần xuất hiện chữ cái trong xâu

```
#include<stdio.h>
#include<ctype.h>

#include<string.h>
 main(){
    char s[20];
    int dem[26] = {};
     int i,n;
    puts("Nhap vao xau ki tu:");gets(s);
     n=strlen(s);
    for(i=0;i<n;i++)
        if(isalpha(s[i]))
            dem[ tolower(s[i ]) - 'a‘ ]++;
     for(i=0;i<26;i++)
              if(dem[i]!=0)
                  printf("Ki tu %c xuat
```

```
Nhap vao xau ki tu:
study, study more, study forever
Ki tu d xuat hien 3 lan
Ki tu e xuat hien 3 lan
Ki tu f xuat hien 1 lan
Ki tu m xuat hien 1 lan
Ki tu o xuat hien 2 lan
Ki tu r xuat hien 3 lan
Ki tu s xuat hien 3 lan
Ki tu t xuat hien 3 lan
Ki tu u xuat hien 3 lan
Ki tu v xuat hien 1 lan
Ki tu y xuat hien 3 lan
```

Mảng xâu ký tự

- Xâu ký tự có thể là kiểu phần tử của mảng
- Khai báo

```
char DS[100][30];
```

  Mảng có tối đa 100 phần tử, các phần tử là xâu có độ dài tối đa 30

- Sử dụng
  - Như một mảng bình thường
  - Mỗi phần tử mảng được sử dụng như một xâu ký tự

Ví dụ: Nhập vào DSSV cho tới khi gặp tên rỗng, in DS

```
#include <stdio.h>
#include <string.h>
 void main(){
   int i, n;
   char DS[100][30];
   printf("Nhap DSSV (<100), go Enter de thoat..\n");
      n =0;
   do{
    printf("Ten sinh vien[%d]: ",n+1);      gets(DS[n]);
    if(DS[n][0] !='\x0') n++;
      else break;
      if(n==100) break;
     }while(1);
     printf("\n\nDS sinh vien vua nhap \n");
      for(i=0;i<n;i++) printf("%s\n",DS[i]);
}
```

```
for(n = 0; n <100; n++){
    printf("Ten sinh vien[%d]: ",n+1);
    gets(DS[n]);  if(DS[n][0] ='\x0') break;
    //n++;
}
```

```
if(strcmp(DS[n],""))n++
 if(strlen(DS[n])>0)  n+
```

Ví dụ→Kết quả thực hiện

```
Nhap DSSV (<100), go Enter de thoat..
Ten sinh vien[1]: Pham Viet Linh
Ten sinh vien[2]: Ha Minh Duc
Ten sinh vien[3]: Nguyen Binh An
Ten sinh vien[4]: Bui Thanh Trung
Ten sinh vien[5]: Tran Van Truong
Ten sinh vien[6]: Vu Tu Anh
Ten sinh vien[7]: Phung Nhat Huy
Ten sinh vien[8]: Le Quang Nam
Ten sinh vien[9]:


DS sinh vien vua nhap
Pham Viet Linh
Ha Minh Duc
Nguyen Binh An
Bui Thanh Trung
Tran Van Truong
Vu Tu Anh
Phung Nhat Huy
Le Quang Nam
```

Nhập dãy (<100) xâu cho tới khi gặp xâu "***"
Đưa ra màn hình xâu có độ dài lớn nhất

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  void main(){
4.      int i, n = 0, d=0;
5.      char DS[100][30], s[30]="";
6.      do{
7.          printf("Nhap xau thu [%d]: ",n+1);
            gets(DS[n]);
8.          if( strcmp(DS[n],"***")) n= n + 1;//Không tính xâu
            "***"
9.          else break;
10.     }while(1);
11.      for(i = 0; i < n; i++)
12.          if(strlen(DS[i]) > d){
13.                  d = strlen(DS[i]);
14.                  strcpy(s,DS[i]);
15.          }
17.  }
```

```
Nhap xau thu [1]: Paris
Nhap xau thu [2]: Rome
Nhap xau thu [3]: Luxembourg
Nhap xau thu [4]: Santiago
Nhap xau thu [5]: Moscow
Nhap xau thu [6]: New Delhi
Nhap xau thu [7]: Washington
Nhap xau thu [8]: Madrid
Nhap xau thu [9]: ***

Xau dai nhat la: Luxembourg, co do dai :10
```

```
Nhap xau thu [1]: ab
Nhap xau thu [2]: a
Nhap xau thu [3]:
Nhap xau thu [4]: aa
Nhap xau thu [5]: ***

Xau dai nhat la: ab, co do dai :2
```

Ví dụ: Nhập vào DS sinh viên, in ra DS đã sắp xếp

```
#include <stdio.h>
#include <string.h>
 void main(){
   int i, j, N;
   char     DS[100]
   [30], str[30];
   //Nhap DS sinh viên                scanf("%d",&N);
   printf("So sinh vien : ");
    fflush(stdin);
   for(i=0;i<N;i++){                printf("Ten sinh vien[%d]: ",i);
       gets(DS[i]);
    }
```

Ví dụ: Nhập vào DS sinh viên, in ra DS đã sắp xếp

```
//So sánh theo Họ+đệm+tên
 for(i = 0; i < N - 1; i ++)
    for(j = i +1; j < N; j ++)
      if(strcmp(DS[i],DS[j]) > 0){
            strcpy(str,DS[i]);
            strcpy(DS[i],DS[j]);
            strcpy(DS[j],str);
    }
//In danh sách đã sắp xếp  printf("\
 nDS sinh vien vua nhap \n");
 for(i=0;i < N;i++)
  printf("%s\n",DS[i]);
}//main
```

Ví dụ→Kết quả thực hiện

```
So sinh vien : 10
Ten sinh vien[0]: Nguyen Hoai Nam
Ten sinh vien[1]: Le Tu Anh
Ten sinh vien[2]: Tran Thai Ha
Ten sinh vien[3]: Nguyen Van Tuan
Ten sinh vien[4]: Tran Thanh Son
Ten sinh vien[5]: Tran Hoai Nam
Ten sinh vien[6]: Bui Trong Son
Ten sinh vien[7]: Le Thanh Tuan
Ten sinh vien[8]: Le Thanh Son
Ten sinh vien[9]: Nguyen Nam Giang

DS sinh vien vua nhap
Bui Trong Son
Le Thanh Son
Le Thanh Tuan
Le Tu Anh
Nguyen Hoai Nam
Nguyen Nam Giang
Nguyen Van Tuan
Tran Hoai Nam
Tran Thai Ha
Tran Thanh Son
```

Ví dụ : Sắp xếp theo tên

```
//Sap xep theo tên
 char ten_i[30],ten_j[30];
   for(i = 0; i < N - 1; i +
  +)
   for(j = i +1; j < N; j ++)
    {
      strcpy(ten_i,strrchr(DS[i],32));//trích ra từ cuối
       strcpy(ten_j,strrchr(DS[j],32));//của xâu họ&tên
        if(strcmp(ten_i,ten_j) > 0){
              strcpy(str,DS[i]);
              strcpy(DS[i],DS[j]);
              strcpy(DS[j],str);
```

Ví dụ

```
So sinh vien : 10
Ten sinh vien[0]: Nguyen Hoai Nam
Ten sinh vien[1]: Le Tu Anh
Ten sinh vien[2]: Tran Thai Ha
Ten sinh vien[3]: Nguyen Van Tuan
Ten sinh vien[4]: Tran Thanh Son
Ten sinh vien[5]: Tran Hoai Nam
Ten sinh vien[6]: Bui Trong Son
Ten sinh vien[7]: Le Thanh Tuan
Ten sinh vien[8]: Le Thanh Son
Ten sinh vien[9]: Nguyen Nam Giang

DS sinh vien vua nhap
Le Tu Anh
Nguyen Nam Giang
Tran Thai Ha
Tran Hoai Nam
Nguyen Hoai Nam
Bui Trong Son
Le Thanh Son
Tran Thanh Son
Nguyen Van Tuan
Le Thanh Tuan
```

Bài tập

1. Nhập vào 2 xâu S1, S2 và một số nguyên k. Hãy chèn xâu S1 vào S2 và đưa ra màn hình (*giả thiết xâu S2 được khai báo đủ lớn*)
2. Một văn bản gồm không quá 60 dòng, mỗi dòng không quá 80 ký tự. Hãy viết chương trình thực hiện nhập vào một văn bản, sau đó
    1. Nhập vào xâu s và chỉ ra vị trí xuất hiệncủa xâu S trong văn bản nếu có.
    2. Thay tất cả các chuỗi « hanoi » (nếu có) bằng chuỗi « HANOI »
    3. Đếm xem trong văn bản có bao nhiêu từ (*các từ phân cách bởi dấu cách*)
    4. Tính tần xuất xuất hiện của các từ trong văn bản

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- *Chương 8: Tệp dữ liệu*

Nội dung chính

1. **Khái niệm cấu trúc**
   - Khái niệm
2. **Khai báo cấu trúc**
   - Khái báo kiểu cấu trúc
   - Khai báo biến cấu trúc
   - Định nghĩa kiểu dữ liệu với **typedef**
3. **Xử lý dữ liệu cấu trúc**
   - Truy nhập các trường dữ liệu
   - Phép gán giữa các biến cấu trúc
4. **Một số ví dụ**

Ví dụ → Bài toán quản lý thí sinh thi đại học

Để quản lý cần lưu trữ các thông tin

- Số báo danh: Số nguyên không dấu
- Họ tên sinh viên: Chuỗi ký tự không quá 30
- Khối thi: Ký tự (A,B,C..)
- Tổng điểm 3 môn thi: kiểu thực

Do vậy với mỗi sinh viên cần các biến

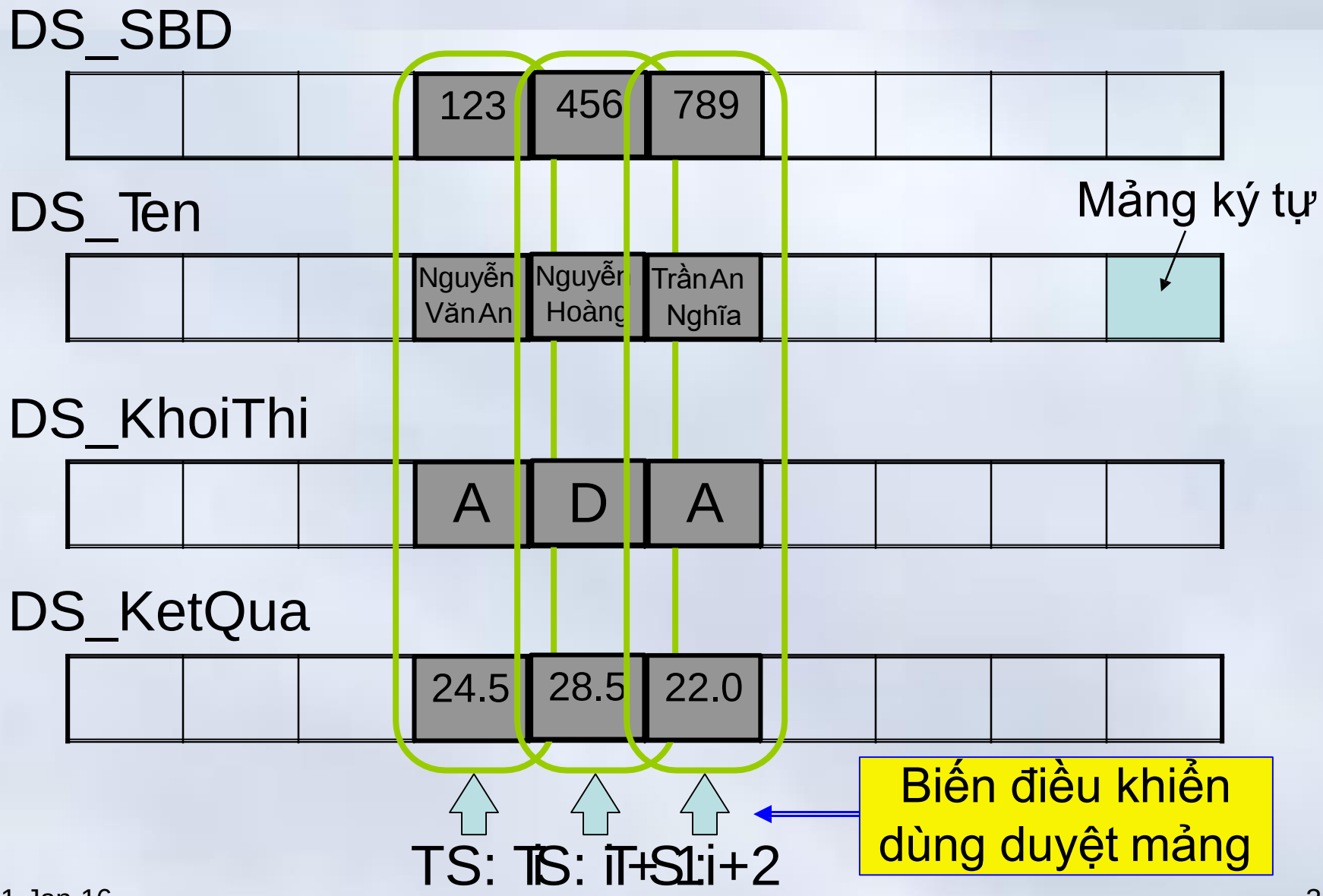
```
unsigned    SBD;
char        Ten[30];
char        KhoiThi;
float       KetQua;
```

Ví dụ → Bài toán quản lý thí sinh thi đại học (tiếp)

Để quản lý danh sách (dưới 1000) thí sinh dự thi, cần nhiều mảng rời rạc

```
#define MAX 1000
unsigned    DS_SBD[MAX];
char        DS_Ten[MAX][30];
char        DS_KhoiThi[MAX];
float       DS_KetQua[MAX];
```

Ví dụ → Bài toán quản lý thí sinh thi đại học (tiếp)



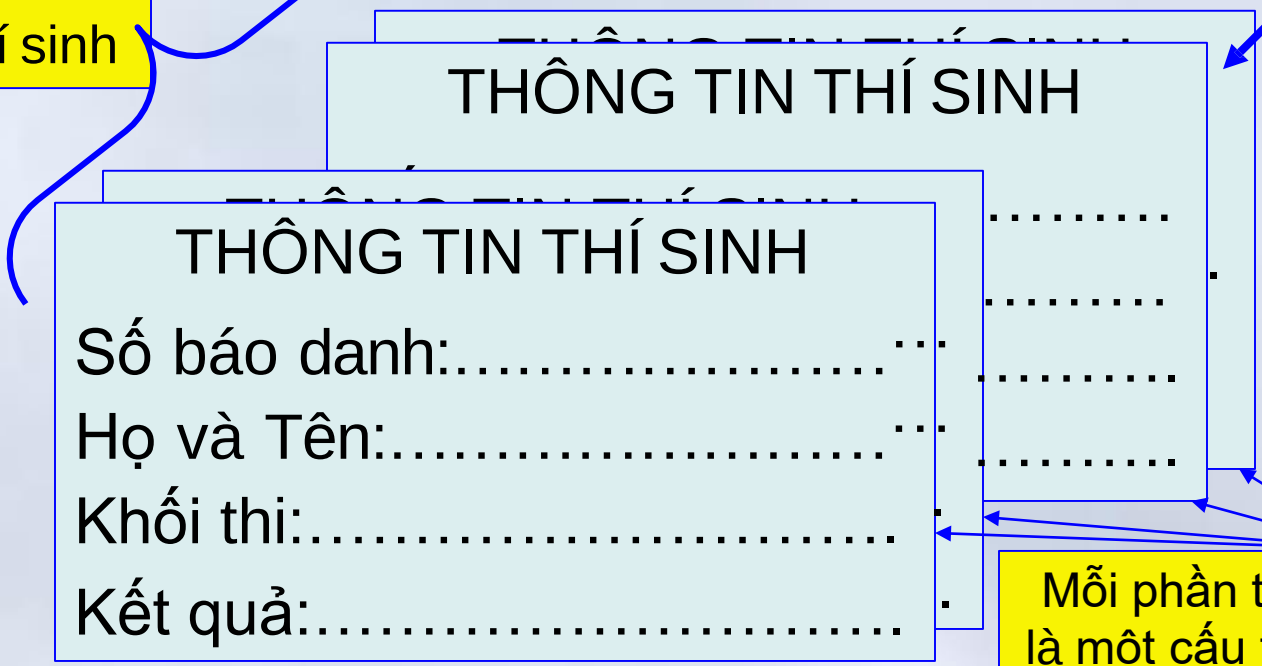
Ví dụ → Vấn đề & giải pháp

Dùng nhiều mảng

- Khó quản lý, dễ nhầm lẫn
- Không thể hiện cấu trúc thông tin dành cho từng thí sinh

Mảng các cấu trúc thông tin dành cho thí sinh

Cấu trúc thông tin dành cho một thí sinh



Mỗi phần tử của mảng là một cấu trúc thông tin

Khái niệm

- Cấu trúc là kiểu dữ liệu phức hợp, do người dụng tự định nghĩa
  - Kiểu cấu trúc bao gồm nhiều thành phần có thể thuộc các kiểu dữ liệu khác nhau
  - Các thành phần: gọi là trường dữ liệu (*field*)
  - Các thành phần, không được truy nhập theo chỉ số (*như mảng*) mà theo tên của trường.

Có thể coi một biến cấu trúc là một tập hợp của một hay nhiều biến rời rạc, thường có kiểu khác nhau thành một biến có một tên duy nhất để dễ dàng quản lý và sử dụng

Khái niệm → Ví dụ

- Kết quả học tập của sinh viên
  - TenSV: Chuỗi ký tự
  - MaSV: Chuỗi số/ số nguyên
  - Điem: Số thực
- Điểm trong mặt phẳng
  - Tên điểm: Ký tự (A, B, C..)
  - Hoành độ: Số thực
  - Tung độ: Số thực

Nội dung chính

1. Khái niệm cấu trúc

- Khái niệm

2. Khai báo cấu trúc

- Khái báo kiểu cấu trúc
- Khai báo biến cấu trúc
- Định nghĩa kiểu dữ liệu với **typedef**

3. Xử lý dữ liệu cấu trúc

- Truy nhập các trường dữ liệu
- Phép gán giữa các biến cấu trúc

4. Một số ví dụ

Khai báo kiểu cấu trúc

```
struct Tên_kiểu_cấu_ trúc {
    <Khai báo các trường dữ liệu>
};
```

- **struct:** từ khóa, cho phép người dùng khai báo kiểu dữ liệu mới: kiểu cấu trúc
- **Tên_kiểu_cấu_truc:** Tên của kiểu cấu trúc do người dùng tự định nghĩa
  - Tuân theo nguyên tắc đặt tên đối tượng trong C
- **Khai báo các trường dữ liệu:** Danh sách các khai báo thành phần (*trường:field*) của cấu trúc
  - Giống khai báo biến
  - Các trường có thể có kiểu bất kỳ

Khai báo kiểu cấu trúc → Ví dụ

Thẻ sinh viên

Số hiệu:…(*Chuỗi ký tự*)..

Tên sinh viên: (*Chuỗi ký tự*)

Năm sinh:…(*Số nguyên*)…

Khóa:……(*Số nguyên*)…….

Lớp:…… :.(*Chuỗi ký tự*). …

```
struct SinhVien{
    char SHSV[10];
    char Ten[30];
    int NS;
    int Khoa;
    char Lop [10];
};
```

Point2D

Hoành độ (x)…(*Số thực*)..

Tung độ (y)…..(*Số thực*)..

```
struct Point{
    float x, y;
};
```

Khai báo biến cấu trúc

- Khai báo **kiểu** cấu trúc nhằm tạo định nghĩa toàn thể cho các cấu trúc sẽ được dùng sau này
  - Không cung cấp không gian nhớ cho kiểu
- Khai báo **biến** cấu trúc nhằm yêu cầu chương trình tạo vùng nhớ để lưu trữ các dữ liệu cho biến cấu trúc
  - Chứa dữ liệu của các trường của cấu trúc

Khai báo biến cấu trúc → Cú pháp

Tồn tại định nghĩa kiểu cấu trúc

```
struct Kiểu_cấu_ trúc Tên_biến;
```

Khai báo trực tiếp

```
struct {
   <Khai báo các trường dữ liệu>
}Tên_biến;
```

Kết hợp với khai báo kiểu

```
struct Kiểu_cấu_ trúc {
   <Khai báo các trường dữ liệu>
}Tên_biến;
```

Khai báo biến cấu trúc → Ví dụ

Tồn tại định nghĩa kiểu cấu trúc

```
struct SinhVien SV1, SV2, Thu khoa;
```

Khai báo trực tiếp

```
struct {
   float x, y; //Tọa độ trên mặt phẳng
}A, B;       //Khai báo 2 điểm A, B
```

Kết hợp với khai báo kiểu

```
struct Point_3D{
   float x, y, z;// Tọa độ không gian
}A, B;
```

Khai báo biến cấu trúc →Chú ý

Các cấu trúc có thể được khai báo lồng nhau

```
struct diem_thi {
        float dToan, dLy, dHoa;
}
struct thi_sinh{
        char SBD[10];
        char ho_va_ten[30];
        struct  diem_thi   ket_qua;
} thi_sinh_1, thi_sinh_2;
```

Khai báo biến cấu trúc →Chú ý

Có thể khai báo trực tiếp các trường dữ liệu của một cấu trúc bên trong cấu trúc khác

```
struct thi_sinh{
        char SBD[10];
        char ho_va_ten[30];
        struct{
                float dToan, dLy, dHoa;
        } ket_qua;
} thi_sinh_1, thi_sinh_2;
```

Khai báo biến cấu trúc →Chú ý

Có thể gán giá trị khởi đầu cho một biến cấu trúc, theo nguyên tắc như kiểu mảng

Ví dụ:

```
struct Date{
   int day;
   int month;
   int year;
};
struct SinhVien{
   char Ten[30];
   struct Date NS;
} SV = {“Tran Anh", 20, 12, 1990 };
```

```
struct SinhVien{
  char Ten[20];
  struct Date{
        int day;
        int month;
        int year;
        } NS;
} SV = {“Tran Anh", 20,12,1990 };
```

Định nghĩa kiểu dữ liệu với **typedef**

typedef <tên_cũ> <tên_mới>;

Mục đích

- Đặt tên mới đồng nghĩa với tên của một kiểu dữ liệu đã được định nghĩa
  - Thường được sử dụng cho kiểu cấu trúc
    - Giúp cho khai báo trở nên quen thuộc và ít bị sai hơn

Ví dụ

```
typedef char Str80[80] ;
typedef long mask;
Str80 str="Bonjour tout le monde !";
mask a, b;
```

Định nghĩa kiểu dữ liệu với **typedef**

Thường được kết hợp với kiểu cấu trúc để khai báo một bí danh cho một cấu trúc

– Giúp khai báo trở nên quen thuộc và ít bị sai hơn

```
typedef struct {  //Định nghĩa một cấu trúc
    char SHSV[10];
    char Ten[30];
    int NS;
    int Khoa;
    char Lop [10];
} SinhVien;          //Đặt tên cho cấu trúc là SinhVien

SinhVien SV;         //Tạo một biến cấu trúc
```

Định nghĩa kiểu dữ liệu với **typedef** →Chú ý

Cho phép đặt tên mới trùng với tên cũ

Ví dụ

```
struct point_3D{
        float x, y, z;
  }
struct point_3D  M;
typedef struct point_3D point_3D;
point_3D  N;
```

```
typedef struct {
        float x, y, z;
 }point_3D;
point_3D M;
point_3D N;
```

Định nghĩa kiểu dữ liệu với **typedef** →Chú ý

```
typedef struct point_2D    {
        float x, y;
}point_2D, diem_2_chieu, ten_bat_ki;
point_2D   X;
diem_2_chieu   Y;
ten_bat_ki    Z;
```

Chú ý:

point_2D, **diem_2_chieu**, **ten_bat_ki** là các tên cấu trúc, không phải tên biến

Nội dung chính

1. Khái niệm cấu trúc

- Khái niệm

2. Khai báo cấu trúc

- Khái báo kiểu cấu trúc
- Khai báo biến cấu trúc
- Định nghĩa kiểu dữ liệu với **typedef**

3. Xử lý dữ liệu cấu trúc

- Truy nhập các trường dữ liệu
- Phép gán giữa các biến cấu trúc

4. Một số ví dụ

Truy cập các trường dữ liệu

- Cú pháp

  tên_biến_cấu_trúc.tên_trường

- Lưu ý
  - Dấu "." là toán tử truy cập vào trường dữ liệu trong cấu trúc
  - Nếu trường dữ liệu là một cấu trúc => sử dụng tiếp dấu "." để truy cập vào thành phần mức sâu hơn

Ví dụ

```
#include <stdio.h>
void main(){
    struct{
      char Ten[20];
      struct Date{
                  int day;
                        int month;
                        int year;
                  } NS;
     } SV = {"Tran Anh", 20,12, 1990 };
    printf(" Sinh vien %s (%d/%d/%d)",
    SV.Ten,SV.NS.day,SV.NS.month,SV.NS.year);
 }
```

```
Sinh vien Tran Anh (20/12/1990)
```

Ví dụ

Bài toán: Xây dựng một cấu trúc biểu diễn điểm trong không gian 2 chiều.

- Nhập giá trị cho một biến kiểu cấu trúc này
- Hiển thị giá trị các trường dữ liệu của biến này ra màn hình.

Thực hiện:

- Cấu trúc gồm: tên điểm, tọa độ x, tọa độ y
- Nhập, hiển thị từng trường của biến cấu trúc như các biến dữ liệu khác

Ví dụ

```
#include<stdio.h>
#include<conio.h>
typedef struct{
   char ten[5];
   int x,y;
}toado;
void main(){
  toado t;
  printf("Nhap thong tin toa do\n");
  printf("Ten diem: ");gets(t.ten);
    printf("Toa do x: ");scanf("%d",&t.x);
    printf("Toa do y: ");scanf("%d",&t.y);
    printf("Gia tri cac truong\n");
    printf("%-5s%3d%3d\n",t.ten,t.x,t.y);
    getch();
}
```

```
Nhap thong tin toa do
Ten diem: A
Toa do x: 12
Toa do y: 24
Gia tri cac truong
A     12 24
```

Phép gán giữa các biến cấu trúc

- C cho phép gán hai biến cấu trúc cùng kiểu:

  Biến_cấu_trúc_1 = biến_cấu_trúc_2

- Ví dụ
  - Xây dựng cấu trúc gồm họ tên và điểm TĐC của sinh viên
  - Khai báo 3 biến cấu trúc: **a**, **b**, **c**
  - Nhập giá trị cho biến **a**.
  - Gán biến **a** cho biến **b**
  - gán từng trường của **a** cho **c**.
  - So sánh **a**, **b** và **c** ?

Ví dụ

```
#include<stdio.h>
typedef struct{
   char hoten[20];
   int diem;
}sinhvien;
void main(){
   sinhvien a,b,c;
   printf("Nhap thong tin sinh vien\n");
   printf("Ho ten: ");gets(a.hoten);
   printf("Diem:");scanf("%d",&a.diem);
```

Ví dụ

```
    b=a;                                  //Gán biến cấu trúc
   strcpy(c.hoten,a.hoten);     //Gán từng trường
   c.diem=a.diem;
   printf(“Bien a: ");
    printf("%-20s%3d\n",a.hoten,a.diem);
    printf(“Bien b: ");
    printf("%-20s%3d\n",b.hoten,b.diem);
    printf(“Bien c: ");
    printf("%-20s%3d\n",c.hoten,c.diem);
}
```

Ví dụ →Kết quả

```
Nhap thong tin sinh vien
Ho ten: Nguyen Van Anh
Diem:9
Bien a: Nguyen Van Anh          9
Bien b: Nguyen Van Anh          9
Bien c: Nguyen Van Anh          9
```

Nội dung chính

1. Khái niệm cấu trúc

- Khái niệm

2. Khai báo cấu trúc

- Khái báo kiểu cấu trúc
- Khai báo biến cấu trúc
- Định nghĩa kiểu dữ liệu với **typedef**

3. Xử lý dữ liệu cấu trúc

- Truy nhập các trường dữ liệu
- Phép gán giữa các biến cấu trúc

4. Một số ví dụ

Một số ví dụ

1. Nhập vào 2 số phức và đưa ra tổng và tích của chúng
2. Nhập vào một danh sách (<100) sinh viên gồm họ tên, năm sinh. Kết thúc nhập khi gặp SV có tên là rỗng
   - Đưa danh sách vừa nhập ra màn hình.
   - Đưa ra màn hình sinh viên lớn tuổi nhất
3. Nhập danh sách có N (N < 100, nhập từ bàn phím) thí sinh gồm họ tên, số báo danh, khoa dự thi và điểm thi
   - Đưa ra DSSV đã sắp xếp theo kết quả thi
   - Đưa ra danh sách sinh viên dự thi khoa CNTT có điểm thi từ 22.5 trở lên
   - Nhập vào một số báo danh và in ra họ tên, điểm thi và khoa đăng ký của thí sinh nếu tìm thấy. Nếu không tìm thấy thí sinh thì đưa ra thông báo « không tìm thấy »

Ví dụ 1

```
#include <stdio.h>
typedef struct {float re, im;} Complex;
void main(){
    Complex R, R1, R2;
    printf("Phan thuc & phan ao cho so thu nhat :");
    scanf("%f%f",&R1.re,&R1.im);
    printf("Phan thuc & phan ao cho so thu hai  :");
    scanf("%f%f",&R2.re,&R2.im);
    R.re = R1.re+R2.re; R.im = R1.im+R2.im;          //phép cộng số ảo
    printf("(%.1f+%.1fi)+(%.1f+%.1fi)=(%.1f+%.1fi)\n",
                        R1.re,R1.im,R2.re,R2.im,R.re,R.im);
    R.re = R1.re*R2.re - R1.im*R2.im;                //nhân số ảo
    R.im = R1.re*R2.im + R1.im*R2.re;
    printf("(%.1f+%.1fi)*(%.1f+%.1fi)=(%.1f+%.1fi)\n",
                        R1.re,R1.im,R2.re,R2.im,R.re,R.im);
}
```

```
Phan thuc & phan ao cho so thu nhat : 2 5
Phan thuc & phan ao cho so thu hai  : 3 4
(2.0+5.0i)+(3.0+4.0i)=(5.0+9.0i)
(2.0+5.0i)*(3.0+4.0i)=(-14.0+23.0i)
```

Ví dụ 2

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  typedef struct{
4.      char Ten[30];
5.      int NS;
6.  }SinhVien;
7.  void main(){
8.      SinhVien DS[100], SV;
9.      int n=0,i;
10.     do{
11.         fflush(stdin);
12.         printf("Nhap du lieu cho sinh vien %d: \n", n+1);
13.         printf("Ho ten : "); gets(SV.Ten);
14.         if (strlen(SV.Ten) >0) {
15.             printf("Nam sinh :");scanf("%d", &SV.NS);
16.             DS[n] = SV; n = n + 1;
17.         }
18.     } while (strlen(SV.Ten) > 0);
```

```
for(ni =00;i<n1<0100;+n++){+){
    printf("Nhap du lieu cho sinh vien %d: \n", in++11));
    printf("Ho ten : "); fflush(stdin); gets(SV.[Tne].nT)e;n);
    if (strlen(SV.[Tne].nT)e=n=) 0=)=b0)ebarke;ak;
    printf("Nam sinh :");scanf("%d", &SV.[Nn]S.N);S);
}   DS[n] = SV; n = n + 1;
}
```

Ví dụ 2

```
19.  //In danh sach sinh vien
20.  printf("\n\n");
21.printf("              HO & TEN              NAM SINH\n");
22.  for(i=0; i < n;i++)
23.     printf("%-3d%-30s %6d\n",i+1,DS[i].Ten,DS[i].NS);
24.  printf("\n\n");
25.  //Tim sinh vien lon tuoi nhat
26.  SV = DS[0];
27.  for(i=0; i < n;i++)
28.     if(DS[i].NS < SV.NS)
29.        SV = DS[i];
30.  printf("Sinh vien: %s sinh nam: %d lon tuoi nhat",SV.Ten,SV.NS);
31. }//main
```

Ví dụ 2→Kết quả thực hiện

```
Ho ten : Hoang Mai Huynh
Nam sinh :1990
Nhap du lieu cho sinh vien 4:
Ho ten : Le Danh Tra
Nam sinh :1989
Nhap du lieu cho sinh vien 5:
Ho ten : Tran Hoai Thanh
Nam sinh :1990
Nhap du lieu cho sinh vien 6:
Ho ten : Nguyen Duc Cuong
Nam sinh :1990
Nhap du lieu cho sinh vien 7:
Ho ten :


            HO & TEN                    NAM SINH
1   Nguyen Thanh Nga                      1990
2   Le Anh Nam                            1991
3   Hoang Mai Huynh                       1990
4   Le Danh Tra                           1989
5   Tran Hoai Thanh                       1990
6   Nguyen Duc Cuong                      1990


Sinh vien: Le Danh Tra sinh nam: 1989 lon tuoi nhat
```

Ví dụ 3 (1/5): Khai báo cần thiết

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  typedef struct{
4.          int     SBD;
5.          char Ten[30];
6.          char Khoa[10];
7.          float Diem;
8.  }ThiSinh;
9.  void main(){
10.     ThiSinh DS[100],  SV;
11.     int N, i, j, d=0, SBD;
12.     char Ch; //Sử dụng trong thực hiện tìm kiếm nhiều lần
```

Ví dụ 3 (2/5): Nhập dữ liệu từ bàn phím

```
13.     printf("Nhap so thi sinh : "); scanf("%d",&N);
14.     for ( i=0; i < N; i++ ){
15.        printf("Nhap du lieu cho thi sinh  %d: \n", i+1);
16.        printf("So bao danh : "); scanf("%d",&DS[i].SBD);
17.        printf("Ho ten : ");
18.        fflush(stdin); gets(DS[i].Ten);
19.        printf("Khoa dang ky : ");
20.        fflush(stdin); gets(DS[i].Khoa);
21.        printf("Ket qua thi : ");scanf("%f", &DS[i].Diem);
22.        printf("\n");
23.     }//for
24.     printf("\n\nNhan phim bat ky de xem ket qua thi...");
25.     fflush(stdin); getch();
```

C scanf: floating point format not linked

Ví dụ 3 (3/5): Danh sách sắp xếp theo điểm thi

```
26.     for(i=0; i < N - 1; i ++) //Sắp xếp DS theo điểm thi
27.             for(j=i+1; j < N; j ++)
28.                 if(DS[i].Diem > DS[j].Diem) {
29.                                 SV=DS[i];
30.                                 DS[i]=DS[j];
31.                                 DS[j]=SV;
32.                         }
33.     printf("\n\n                    KET QUA THI \n\n");
34.     for(i=0; i < N; i ++)
35.         printf("%-3d BKA- %-6d %-24s %-6s %-6.1f\n",
                ++d, DS[i].SBD, DS[i].Ten, DS[i].Khoa, DS[i].Diem);
36.     printf("\n\nNhan phim bat ky de xem DSSV khoa CNTT");
37.     fflush(stdin); getch();
```

Ví dụ 3 (4/5): Danh sách trúng tuyển khoa CNTT

```
//Dua ra danh sach du thi khoa CNTT
38.    d = 0;
39.    printf("\n\nDanh sach thi sinh trung tuyen khoa CNTT\n\n");
40.    for(i=0; i < N; i ++)
41.       if(strcmp(DS[i].Khoa,"CNTT")==0 && DS[i].Diem >=22.5)
42.      printf("%-3d BKA%-6d %-24s %-f\n",
                                ++d,DS[i].SBD,DS[i].Ten,DS[i].Diem);
43.    printf("\n\nNhan phim bat ky de bat dau tim kiem theo so bao danh");
44.    fflush(stdin); getch();
```

Ví dụ 3 (5/5): Tìm kiếm sinh viên

```
45.   do{
46.printf("\nNhap so bao danh can tim : "); scanf("%d",&SBD);  47.
          for(i=0; i < N; i++)
48.           if(DS[i].SBD == SBD){
49.printf("So bao danh : %d \n",SBD);  50.
51.                 printf("Ho ten              : %s \n",DS[i].Ten);
52.                 printf("Khoa du thi : %s \n", DS[i].Khoa);
53.                 printf("So bao danh : %.1f \n",DS[i].Diem);
54.           }     break;
55.         if (i == N)
56.                printf("So bao danh %d khong ton tai\n",SBD);
57.         printf("\nCo tiep tuc tim kiem nua khong (C/K) :");
58.     }while(toupper(getche())!='K');


59.  }//main
```

Ví dụ 3→Kết quả thực hiện

```
Nhap so thi sinh : 10
Nhap du lieu cho thi sinh  1:
So bao danh : 1200
Ho ten : Le Tu Anh
Khoa dang ky : CNTT
Ket qua thi : 21.5

Nhap du lieu cho thi sinh  2:
So bao danh : 1256
Ho ten : Tran Huu Cuong
Khoa dang ky : DTVT
Ket qua thi : 27

Nhap du lieu cho thi sinh  3:
So bao danh : 1288
Ho ten : Nguyen Anh Dung
Khoa dang ky : CNTT
Ket qua thi : 24.5

Nhap du lieu cho thi sinh  4:
So bao danh : 1297
Ho ten : Le Thu Ha
Khoa dang ky : KTQL
Ket qua thi : 22.5

Nhap du lieu cho thi sinh  5:
So bao danh : 1322
Ho ten : Tran Hoai Nam
Khoa dang ky : CNTT
Ket qua thi : 22.5

Nhap du lieu cho thi sinh  6:
So bao danh : 1345
Ho ten : Bui Trong Son
Khoa dang ky : CNTT
Ket qua thi : 24.5

Nhap du lieu cho thi sinh  7:
So bao danh : 1350
Ho ten : Le Thanh Son
Khoa dang ky : KTQL
Ket qua thi : 19.5

Nhap du lieu cho thi sinh  8:
So bao danh : 1355
Ho ten : Tran Thanh Son
Khoa dang ky : CNTT
Ket qua thi : 23

Nhap du lieu cho thi sinh  9:
So bao danh : 1410
Ho ten : Nguyen Anh Tuan
Khoa dang ky : DTVT
Ket qua thi : 21

Nhap du lieu cho thi sinh  10:
So bao danh : 1485
Ho ten : Tran Trong Viet
Khoa dang ky : CNTT
Ket qua thi : 21



Nhan phim bat ky de xem ket qua thi...
```

Ví dụ 3→Kết quả thực hiện

```
                    KET QUA THI

1    BKA-1350    Le Thanh Son                KTQL    19.5
2    BKA-1410    Nguyen Anh Tuan             DTVT    21.0
3    BKA-1485    Tran Trong Viet             CNTT    21.0
4    BKA-1200    Le Tu Anh                   CNTT    21.5
5    BKA-1322    Tran Hoai Nam               CNTT    22.5
6    BKA-1297    Le Thu Ha                   KTQL    22.5
7    BKA-1355    Tran Thanh Son              CNTT    23.0
8    BKA-1288    Nguyen Anh Dung             CNTT    24.5
9    BKA-1345    Bui Trong Son               CNTT    24.5
10   BKA-1256    Tran Huu Cuong              DTVT    27.0


Nhan phim bat ky de xem DSSV khoa CNTT

Danh sach thi sinh trung tuyen khoa CNTT

1    BKA1322    Tran Hoai Nam               22.5
2    BKA1355    Tran Thanh Son              23.0
3    BKA1288    Nguyen Anh Dung             24.5
4    BKA1345    Bui Trong Son               24.5


Nhan phim bat ky de bat dau tim kiem theo so bao danh_
```

Ví dụ 3→Kết quả thực hiện

```
Nhan phim bat ky de bat dau tim kiem theo so bao danh
Nhap so bao danh can tim : 1297
So bao danh : 1297
Ho ten      : Le Thu Ha
Khoa du thi : KTQL
So bao danh : 22.5

Co tiep tuc tim kiem nua khong (C/K) :c
Nhap so bao danh can tim : 1350
So bao danh 1350 khong ton tai

Co tiep tuc tim kiem nua khong (C/K) :c
Nhap so bao danh can tim : 1355
So bao danh : 1355
Ho ten      : Tran Thanh Son
Khoa du thi : CNTT
So bao danh : 23.0

Co tiep tuc tim kiem nua khong (C/K) :k
```

Bài tập

1. Lập trình đọc vào một danh sách không quá 100 sinh viên gồm: Họ tên, năm sinh
   1. Đưa ra DS những sinh viên sinh năm 1990
   2. Nhập tên sinh viên, cho biết năm sinh nếu tìm thấy
   3. Đưa ra DSSV đã sắp xếp theo thứ tự ABC của họ và tên
2. Lập trình đọc vào DS thí sinh gồm Họ tên, điểm thi 3 môn Toán, Lý,Hóa, kết thúc nhập khi gặp sinh viên có tên rỗng
   1. Đọc tiếp vào một điểm chuẩn; đưa ra danh sách thí sinh trúng tuyển (*không có điểm liệt - 0*)
   2. Đưa ra thí sinh có kết quả thi cao nhất
   3. Tìm điểm chuẩn, nếu chỉ lấy K SV, K nhập vào. Nếu có nhiều người bằng điểm nhau; loại cả

Bài tập 1

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  typedef struct{
4.      char Ten[30];
5.      int NS;
6.  }SinhVien;
7.  void main(){
8.      SinhVien DS[100], SV;
9.      int N, i, j, d=0;
10.     char ten[30];
11.printf("Nhap so sinh vien : "); scanf("%d",&N);
12.     for ( i=0; i < N; i++ ){
13.         printf("Nhap du lieu cho sinh vien %d: \n", i+1);
14.         printf("Ho ten : ");  fflush(stdin);gets(DS[i].Ten);
15.         printf("Nam sinh :  ");scanf("%d", &DS[i].NS);
16.     }
```

Bài tập 1 (*tiếp*)

```
17.    printf("\n\nSINH VIEN SINH NAM 1990\n\n");
18.    for(i = 0; i < N; i ++)
19.        if(DS[i].NS ==1990)
20.            printf("%s\n",DS[i].Ten);

21.     printf("\n\nTim SV : "); fflush(stdin); gets(Ten);
22.      for(int i=0;i <N; i++)
23.         if(strcmp(Ten,DS[i].Ten)==0){
24.printf("Sinh vien: %s\nSinh nam %d\n", DS[i].Ten,DS[i].NS);  25.
26.        }    d = d + 1;
27.      if(d==0)
28.
            printf("Khong co sinh vien: %s trong danh sach\n",Ten);
```

Tìm kiếm chỉ theo tên ?

strcmp(Ten,strrchr(DS[i].Ten,32)+1))

Bài tập 1 (*tiếp*)

```
29.       for(i = 0; i < N - 1; i ++) //sắp xếp theo pp lựa chọn
30.         for(j = i+1; j < N; j ++)
31.               if(strcmp(DS[i].Ten,DS[j].Ten) > 0){
32.                         SV= DS[i];
33.                         DS[i]=DS[j];
34.                         DS[j] = SV;
35.               }
36.
37.      printf("\n\n DANH SACH SAP XEP\n\n");
38.      for(i = 0; i < N; i ++)
39.            printf("%d %-20s %d \n",i+1, DS[i].Ten, DS[i].NS);
40.   }//main
```

Bài tập 1 → Kết quả thực hiện

```
Nhap so sinh vien : 8
Nhap du lieu cho sinh vien 1:
Ho ten : Nguyen Tuan Anh
Nam sinh : 1992
Nhap du lieu cho sinh vien 2:
Ho ten : Do Trong Khang
Nam sinh : 1990
Nhap du lieu cho sinh vien 3:
Ho ten : Bui Hai Thanh
Nam sinh : 1991
Nhap du lieu cho sinh vien 4:
Ho ten : Hoang Tuan Anh
Nam sinh : 1992
Nhap du lieu cho sinh vien 5:
Ho ten : Le Thu Ha
Nam sinh : 1990
Nhap du lieu cho sinh vien 6:
Ho ten : Bui Thanh Huong
Nam sinh : 1991
Nhap du lieu cho sinh vien 7:
Ho ten : Tran Thanh Son
Nam sinh : 1990
Nhap du lieu cho sinh vien 8:
Ho ten : Nguyen Anh Dung
Nam sinh : 1992
```

```
SINH VIEN SINH NAM 1990

Do Trong Khang
Le Thu Ha
Tran Thanh Son
```

```
Tim SV : Hoang Tuan Anh
Sinh vien: Hoang Tuan Anh
Sinh nam 1992
```

```
 DANH SACH SAP XEP

1 Bui Hai Thanh          1991
2 Bui Thanh Huong        1991
3 Do Trong Khang         1990
4 Hoang Tuan Anh         1992
5 Le Thu Ha              1990
6 Nguyen Anh Dung        1992
7 Nguyen Tuan Anh        1992
8 Tran Thanh Son         1990
```

Bài tập 2 (*1/5*)

```
#include <stdio.h>
#include <string.h>

typedef struct{
	char Ten[30];
	struct{
		int T, L, H, S; //S = T+L+H
	} DT;
}SinhVien;

void main(){
	SinhVien DS[100], TK, SV;
	int N,i,j,K;
	float C;
```

Bài tập 2 (2/5)

```
N = 0; //N chứa số sinh viên đã nhập
do{
    printf("\nNhap DL cho sv thu %d\n",N+1);
    printf("Ten SV : ");
    fflush(stdin); gets(DS[N].Ten);
    if(strlen(DS[N].Ten)==0) //Độ dài bằng 0 ⇒ xâu rỗng
        break;
    else{
       printf(« Nhập điểm thi T L H cua SV %s : ",DS[N].Ten);
       scanf("%d%d%d",&DS[N].DT.T,&DS[N].DT.L,&DS[N].DT.H);
       DS[N].DT.S = DS[N].DT.T + DS[N].DT.L + DS[N].DT.H;
       N++;
    }
}while(1);
```

Bài tập 2 (*3/5*)

```
//In ra danh sách vừa nhập
printf("\n\n DANH SACH SINH VIEN\n\n");
printf("      Ten SV          Toan   Ly    Hoa Tong \n");
for(i = 0; i < N; i ++)
   printf("%-20s%5d%5d%5d%6d\n",DS[i].Ten,
              DS[i].DT.T,DS[i].DT.L,DS[i].DT.H,DS[i].DT.S);
//In danh sách trúng tuyển theo điểm chuẩn
printf("\n\nDiem Chuan : ");scanf("%f",&C);
printf("\n\n DANH SACH SINH VIEN TRUNG TUYEN \n\n");
for(i = 0; i < N; i ++)
   if( (DS[i].DT.S >= C)&&(DS[i].DT.T*DS[i].DT.L*DS[i].DT.H>0))
         printf("%s\n",DS[i].Ten);
```

Bài tập 2 (*4/5*)

```
//Tìm điểm của thủ khoa, bỏ qua tường hợp điểm liệt
TK = DS[0];
for(i = 1; i < N; i ++)
      if(DS[i].DT.S > TK.DT.S)
            TK = DS[i];
//Đưa ra danh sách thí sinh có điểm bằng điểm cao nhât
for(i = 0; i < N; i ++)
      if(DS[i].DT.S == TK.DT.S)
          printf("\n\n THU KHOA: %s \n\n",TK.Ten);
```

Bài tập 2 (5/5)

```
    printf("\nSo nguoi trung tuyen:"); scanf("%d",&K);
    for(i = 0; i < N - 1; i ++) //Sắp xếp theo tổng điểm thi
          for(j = i+1; j < N; j ++)
                if(DS[i].DT.S < DS[j].DT.S ){
                      SV= DS[i];
                      DS[i]=DS[j];
                      DS[j] = SV;
                }
  while((K>0)&&(DS[K-1].DT.S==DS[K].DT.S))K--;
  if(K>0){
    printf("Diem Chuan La : %4d",DS[K-1].DT.S);
    printf("\n\n Danh Sach sinh vien trung tuyen \n");
    for(i=0; i < K; i++)
        printf("%s\n",DS[i].Ten);
   }
}//main
```

Bài tập 1

Lập trình thực hiện các công việc sau

- Đọc vào từ bàn phím một danh sách thuốc gồm
  - Tên thuốc (chuỗi không quá 20 ký tự)
  - Năm hết hạn
  - Số lượng còn
  - Đơn giá

  Kết thúc nhập khi gặp thuốc có tên »*** »
- Đưa danh sách thuốc ra màn hình
- Đưa ra danh sách các thuốc đã hết hạn
- Xóa khỏi danh sách những thuốc đã hết hạn. Đưa danh sách mới ra màn hình
- Tính tổng giá trị các thuốc đã hết hạn
- Đưa ra DS thuốc được sắp xếp theo năm hết hạn

Bài tập 2

Cho một danh sách thành tích thi đấu bóng đá của 32 đội tuyển bao gồm: Tên đội bóng, số bàn thắng, số bàn thua, số thẻ đỏ, số thẻ vàng

Viết chương trình thực hiện

- Nhập dữ liệu vào từ bàn phím
- Nhập vào tên đội bóng,
  - Đưa ra thành thích của đội này
  - Nếu không tồn tại, thông báo: không tìm thấy
- Tính và đưa ra màn hình số điểm của các đội nếu
  - Mỗi bàn thắng được tính 10 điểm
  - Mỗi bàn thua bị phạt 5 điểm, mỗi thẻ vàng trừ 2 điểm, thẻ đỏ trừ 5 điểm

1. Kết quả đưa ra màn hình

```
#include<stdio.h>
 typedef struct {
    int SHSV;
    char Ten[25];
  }SV;
void main(){
   SV DS[] = { {12, "Mai"},
               {13, "Nam"},
               {14, "Minh"}};
    printf("%d ", DS[1].SHSV);
    printf("%s\n", (*(DS+2)).Ten);
}
```

a	12 Mai
b	12 Nam
c	13 Nam
d	13 Minh
e	14 Minh

2. Chỉ ra câu trả lời đúng

Chỉ ra khai báo hợp lệ cho biến **SV** có kiểu cấu trúc chỉ gồm 2 trường: **Tên** có kiểu xâu ký tự và **NS** có kiểu số nguyên

a	struct SinhVien{
	char Ten[20];
	unsigned NS;
	};
	SinhVien SV;
		b	struct {
			char Ten[20];
			unsigned NS;
			} SinhVien;
			struct SinhVien SV;

c	struct { unsigned NS; char Ten[20]; } SV;	d	typedef struct { unsigned NS; char Ten[20]; } SV;
e	Không có câu trả lời nào đúng		