**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH**

**--------------**

**Ths. Nguyễn Thị Thanh Bình**

**BÀI GIẢNG LÝ THUYẾT**

**NHẬP MÔN LẬP TRÌNH**

**Trình độ: ĐẠI HỌC**

**Ngành: KHOA HỌC MÁY TÍNH**

**Môn: NHẬP MÔN LẬP TRÌNH**

**Thời lượng: 15 tiết**

TP. HỒ CHÍ MINH - 2016

LƯU HÀNH NỘI BỘ

**MỤC LỤC**

[CHƯƠNG 1 Tổng Quan Về Lập Trình Vá Các Bước Giải Quyết Trên Máy Tính 4](#_Toc127294333)

[1.1. Các khái niệm cơ bản 4](#_Toc127294334)

[1.1.1. Lập trình máy tính 4](#_Toc127294335)

[1.1.2. Thuật toán 4](#_Toc127294336)

[1.1.3. Ví dụ 4](#_Toc127294337)

[1.2. Các tính chất của thuật toán 4](#_Toc127294338)

[1.3. Các bước xây dựng chương trình 4](#_Toc127294339)

[1.4. Xây dựng thuật toán/ thuật giải: 4](#_Toc127294340)

[1.4.1. Ngôn ngữ tự nhiên 4](#_Toc127294341)

[1.4.2. Lưu đồ - Sơ đồ khối 5](#_Toc127294342)

[1.4.3. Mã giả 6](#_Toc127294343)

[1.5. Bài tập lý thuyết 6](#_Toc127294344)

[CHƯƠNG 2 Các Thành Phần Cơ Bản Của Ngôn Ngữ C 6](#_Toc127294345)

[2.1. Giới thiệu 6](#_Toc127294346)

[2.1.1. Ưu điểm của C 6](#_Toc127294347)

[2.1.2. Môi trường phát triển tích hợp IDE (Integrated Development Environment) 6](#_Toc127294348)

[2.1.3. Môi trường lập trình 6](#_Toc127294349)

[2.2. Bộ từ vựng của C 7](#_Toc127294350)

[2.2.1. Các ký tự được sử dụng 7](#_Toc127294351)

[2.2.2. Từ khóa (keyword) 7](#_Toc127294352)

[2.2.3. Tên/Định danh (Identifier) 7](#_Toc127294353)

[2.2.4. Dấu chấm phẩy ; 7](#_Toc127294354)

[2.2.5. Câu chú thích 7](#_Toc127294355)

[2.2.6. Hằng ký tự và hằng chuỗi 7](#_Toc127294356)

[2.3. Cấu trúc chương trình C 8](#_Toc127294357)

[2.4. Bài tập lý thuyết 8](#_Toc127294358)

[CHƯƠNG 3 Nhập Xuất Dữ Liệu Và Khai Thác Các Thư Viện Của C 8](#_Toc127294359)

[3.1. Các kiểu dữ liệu cơ sở 8](#_Toc127294360)

[3.1.1. Kiểu số nguyên 8](#_Toc127294361)

[3.1.2. Kiểu số thực 9](#_Toc127294362)

[3.1.3. Kiểu luận lý 9](#_Toc127294363)

[3.1.4. Kiểu ký tự 9](#_Toc127294364)

[3.2. Biến 9](#_Toc127294365)

[3.3. Hằng 10](#_Toc127294366)

[3.4. Biểu thức 10](#_Toc127294367)

[3.5. Toán tử gán 10](#_Toc127294368)

[3.6. Các toán tử toán học 10](#_Toc127294369)

[3.7. Các toán tử trên bit 11](#_Toc127294370)

[3.8. Các toán tử quan hệ 12](#_Toc127294371)

[3.9. Các toán tử luận lý 12](#_Toc127294372)

[3.10. Toán tử điều kiện 12](#_Toc127294373)

[3.11. Toán tử phẩy 12](#_Toc127294374)

[3.12. Độ ưu tiên của các toán tử 13](#_Toc127294375)

[3.13. Câu lệnh 13](#_Toc127294376)

[3.14. Câu lệnh xuất 14](#_Toc127294377)

[3.14.1. Định dạng xuất 15](#_Toc127294378)

[3.14.2. Chuỗi định dạng 15](#_Toc127294379)

[3.15. Câu lệnh nhập 15](#_Toc127294380)

[3.16. Một số hàm hữu ích khác 16](#_Toc127294381)

[3.17. Bài tập lý thuyết 16](#_Toc127294382)

# Tổng Quan Về Lập Trình Vá Các Bước Giải Quyết Trên Máy Tính

## Các khái niệm cơ bản

### Lập trình máy tính

* + Gọi tắt là lập trình (programming).
  + Nghệ thuật cài đặt một hoặc nhiều thuật toán trừu tượng có liên quan với nhau bằng một ngôn ngữ lập trình để tạo ra một chương trình máy tính.

### Thuật toán

* + Là tập hợp (dãy) hữu hạn các chỉ thị (hành động) được định nghĩa rõ ràng nhằm giải quyết một bài toán cụ thể nào đó.

### Ví dụ

* + Thuật toán giải PT bậc nhất: ax + b = 0 (a, b là các số thực).
    - **Đầu vào:** a, b thuộc R
    - **Đầu ra:** nghiệm phương trình ax + b = 0
      * Nếu a = 0
      * b = 0 thì phương trình có nghiệm bất kì.
      * b ≠ 0 thì phương trình vô nghiệm.
      * Nếu a ≠ 0
      * Phương trình có nghiệm duy nhất x = -b/a

## Các tính chất của thuật toán

* Bao gồm 5 tính chất sau:
  + Tính chính xác: quá trình tính toán hay các thao tác máy tính thực hiện là chính xác.
  + Tính rõ ràng: các câu lệnh minh bạch được sắp xếp theo thứ tự nhất định.
  + Tính khách quan: được viết bởi nhiều người trên máy tính nhưng kết quả phải như nhau.
  + Tính phổ dụng: có thể áp dụng cho một lớp các bài toán có đầu vào tương tự nhau.
  + Tính kết thúc: hữu hạn các bước tính toán

## Các bước xây dựng chương trình

* + Xác định vấn đề - bài toán
  + Lựa chọn phương pháp giải
  + Xây dựng thuật toán/ thuật giải: Biểu diễn bằng:
    - Ngôn ngữ tự nhiên
    - Lưu đồ - Sơ đồ khối
    - Mã giả
  + Cài đặt chương trình
  + Hiệu chỉnh chương trình
    - Lỗi cú pháp
    - Lỗi ngữ nghĩa
  + Thực hiện chương trình

## Xây dựng thuật toán/ thuật giải:

### Ngôn ngữ tự nhiên

Đầu vào: a, b thuộc R

Đầu ra: nghiệm phương trình ax + b = 0

1. Nhập 2 số thực a và b.

2. Nếu a = 0 thì

2.1. Nếu b = 0 thì

2.1.1. Phương trình vô số nghiệm

2.1.2. Kết thúc thuật toán.

2.2. Ngược lại

2.2.1. Phương trình vô nghiệm.

2.2.2. Kết thúc thuật toán.

3. Ngược lại

3.1. Phương trình có nghiệm.

3.2. Giá trị của nghiệm đó là x = -b/a

3.3. Kết thúc thuật toán.

* Ưu điểm: dễ viết, dùng trong các chương trình lớn và có lời giải phức tạp nhiều điều kiện.

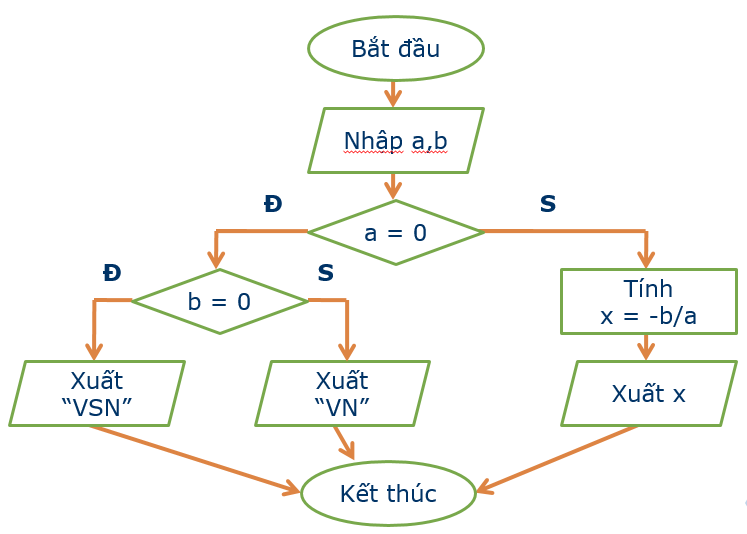
### Lưu đồ - Sơ đồ khối

Thực hiện câu lệnh

Câu lệnh nhập xuất

Biểu thức điều kiện

Đường chỉ dẫn thao tác tiếp theo



* Ưu điểm: rõ ràng dễ hiểu tuy nhiên sơ đồ khối dùng cho những bài toán có thuật giải nhỏ.

### Mã giả

* Vay mượn ngôn ngữ nào đó (ví dụ Pascal) để biểu diễn thuật toán.

Đầu vào: a, b thuộc R

Đầu ra: nghiệm phương trình ax + b = 0

If a = 0 Then

Begin

If b = 0 Then

Xuất “Phương trình vô số nghiệm”

Else

Xuất “Phương trình vô nghiệm”

End

Else

Xuất “Phương trình có nghiệm x = -b/a”

* Ưu điểm: gần với ngôn ngữ cài đặt nhất giúp cho việc cài đặt dễ dàng hơn.

## Bài tập lý thuyết

1. Thuật toán là gì? Trình bày các tính chất quan trọng của một thuật toán?
2. Các bước xây dựng chương trình?
3. Các cách biểu diễn thuật toán? Ưu và khuyết điểm của từng phương pháp?

Cho ví dụ minh họa.

1. Nhập năm sinh của một người. Tính tuổi người đó.
2. Nhập 2 số a và b. Tính tổng, hiệu, tính và thương của hai số đó.
3. Nhập tên sản phẩm, số lượng và đơn giá. Tính tiền và thuế giá trị gia tăng phải trả, biết:
   1. tiền = số lượng \* đơn giá
   2. thuế giá trị gia tăng = 10% tiền

# Các Thành Phần Cơ Bản Của Ngôn Ngữ C

## Giới thiệu

### Ưu điểm của C

* + Rất mạnh và linh động, có khả năng thể hiện bất cứ ý tưởng nào.
  + Được sử dụng rộng rãi bởi các nhà lập trình chuyên nghiệp.
  + Có tính khả chuyển, ít thay đổi trên các hệ thống máy tính khác nhau.
  + Bộ lệnh rõ ràng, cô đọng, phù hợp với phương pháp lập trình có cấu trúc
  + Kiểu dữ liệu phong phú, cho phép định nghĩa kiểu dữ liệu mới.
  + Lập trình đơn thể, tái sử dụng thông qua hàm

### Môi trường phát triển tích hợp IDE (Integrated Development Environment)

* + Biên tập chương trình nguồn (Trình EDIT).
  + Biên dịch chương trình (Trình COMPILE).
  + Chạy chương trình nguồn (Trình RUNTIME).
  + Sửa lỗi chương trình nguồn (Trình DEBUG).

### Môi trường lập trình

* + Borland C++ 3.1 for DOS.
  + Visual C++ 6.0, Win32 Console Application.
  + CFree

## Bộ từ vựng của C

### Các ký tự được sử dụng

* + Bộ chữ cái 26 ký tự Latinh A, B, C, …, Z, a, b, c, …, z
  + Bộ chữ số thập phân : 0, 1, 2, …, 9
  + Các ký hiệu toán học : + – \* / = < > ( )
  + Các ký tự đặc biệt : . , : ; [ ] % \ # $ ‘
  + Ký tự gạch nối \_ và khoảng trắng ‘ ’

### Từ khóa (keyword)

* + Các từ dành riêng trong ngôn ngữ.
  + Không thể sử dụng từ khóa để đặt tên cho biến, hàm, tên chương trình con.
  + Một số từ khóa thông dụng:
    - const, enum, signed, struct, typedef, unsigned…
    - char, double, float, int, long, short, void
    - case, default, else, if, switch
    - do, for, while
    - break, continue, goto, return

### Tên/Định danh (Identifier)

* + Một dãy ký tự dùng để chỉ tên một hằng số, hằng ký tự, tên một biến, một kiểu dữ liệu, một hàm hay một thủ tục.
  + Không được trùng với các từ khóa và được tạo thành từ các chữ cái và các chữ số nhưng bắt buộc chữ đầu phải là chữ cái hoặc \_.
  + Số ký tự tối đa trong một tên là 255 ký tự và được dùng ký tự \_ chen trong tên nhưng không cho phép chen giữa các khoảng trắng.
* Ví dụ Tên/Định danh (Identifier)
  + Các tên hợp lệ: GiaiPhuongTrinh, Bai\_Tap1
  + Các tên không hợp lệ: 1A, Giai Phuong Trinh
  + Phân biệt chữ hoa chữ thường, do đó các tên sau đây khác nhau:
    - A, a
    - BaiTap, baitap, BAITAP, bAItaP, …

### Dấu chấm phẩy ;

* + Dùng để phân cách các câu lệnh.
  + Ví dụ: printf(“Hello World!”); printf(“\n”);

### Câu chú thích

* + Đặt giữa cặp dấu /\* \*/ hoặc // (C++)
  + Ví dụ: /\*Ho & Ten: NVA\*/, // MSSV: 0712078

### Hằng ký tự và hằng chuỗi

* + Hằng ký tự: ‘A’, ‘a’, …
  + Hằng chuỗi: “Hello World!”, “Nguyen Van A”
  + Chú ý: ‘A’ khác “A”

## Cấu trúc chương trình C

#include “…”; // Khai báo file tiêu đề

int x; // Khai báo biến hàm

void Nhap(); // Khai báo hàm

void main() // Hàm chính

{

// Các lệnh và thủ tục

}

* Ví dụ

#include <stdio.h>

#include <conio.h>

void main()

{

int x, y, tong;

printf(“Nhap hai so nguyen: ”);

scanf(“%d%d”, &x, &y);

tong = x + y;

printf(“Tong hai so la %d”, tong);

getch();

}

## Bài tập lý thuyết

1. Tên (định danh) nào sau đây đặt không hợp lệ, tại sao?
   * Tin hoc co SO A, 1BaiTapKHO
   * THucHaNH, NhapMon\_L@pTrinH
2. Câu ghi chú dùng để làm gì? Cách sử dụng ra sao? Cho ví dụ minh họa.
3. Trình bày cấu trúc của một chương trình C. Giải thích ý nghĩa của từng phần trong cấu trúc.

# Nhập Xuất Dữ Liệu Và Khai Thác Các Thư Viện Của C

## Các kiểu dữ liệu cơ sở

* Turbo C có 4 kiểu cơ sở như sau:
  + Kiểu số nguyên: giá trị của nó là các số nguyên như 2912, -1706, …
  + Kiểu số thực: giá trị của nó là các số thực như 3.1415, 29.12, -17.06, …
  + Kiểu luận lý: giá trị đúng hoặc sai.
  + Kiểu ký tự: 256 ký tự trong bảng mã ASCII.

### Kiểu số nguyên

* Các kiểu số nguyên (có dấu)
  + n bit có dấu: –2n – 1 … +2n – 1 – 1

|  |  |  |
| --- | --- | --- |
| **Kiểu (Type)** | **Độ lớn (Byte)** | **Miền giá trị (Range)** |
| char | 1 | –128 … +127 |
| int | 2 | –32.768 … +32.767 |
| short | 2 | –32.768 … +32.767 |
| long | 4 | –2.147.483.648 … +2.147.483.647 |

* Các kiểu số nguyên (không dấu)
  + n bit không dấu: 0 … 2n – 1

|  |  |  |
| --- | --- | --- |
| **Kiểu (Type)** | **Độ lớn (Byte)** | **Miền giá trị (Range)** |
| unsigned char | 1 | 0 … 255 |
| unsigned int | 2 | 0 … 65.535 |
| unsigned short | 2 | 0 … 65.535 |
| unsigned long | 4 | 0 … 4.294.967.295 |

### Kiểu số thực

* Các kiểu số thực (floating-point)

|  |  |  |
| --- | --- | --- |
| **Kiểu (Type)** | **Độ lớn (Byte)** | **Miền giá trị (Range)** |
| float (\*) | 4 | 3.4\*10–38 … 3.4\*1038 |
| double (\*\*) | 8 | 1.7\*10–308 … 1.7\*10308 |

* + Ví dụ
    - 17.06 = 1.706\*10 = 1.706\*101
    - (\*) Độ chính xác đơn (Single-precision) chính xác đến 7 số lẻ.
    - (\*\*) Độ chính xác kép (Double-precision) chính xác đến 19 số lẻ.

### Kiểu luận lý

* Đặc điểm
  + C ngầm định một cách không tường minh:
    - false (sai): giá trị 0.
    - true (đúng): giá trị khác 0, thường là 1.
  + C++: bool
* Ví dụ
  + 0 (false), 1 (true), 2 (true), 2.5 (true)
  + 1 > 2 (0, false), 1 < 2 (1, true)

### Kiểu ký tự

* Đặc điểm
  + Tên kiểu: char
  + Miền giá trị: 256 ký tự trong bảng mã ASCII.
  + Chính là kiểu số nguyên do:
    - Lưu tất cả dữ liệu ở dạng số.
    - Không lưu trực tiếp ký tự mà chỉ lưu mã ASCII của ký tự đó.
* Ví dụ
  + Lưu số 65 tương đương với ký tự ‘A’…
  + Lưu số 97 tương đương với ký tự ‘a’.

## Biến

* Cú pháp

<kiểu> <tên biến>;

<kiểu> <tên biến 1>, <tên biến 2>;

* Ví dụ

int i;

int j, k;

unsigned char dem;

float ketqua, delta;

## Hằng

* Cú pháp

#define <tênhằng> <giá trị>

hoặc sử dụng từ khóa const.

* Ví dụ

#define MAX 100 // Không có ;

#define PI 3.14 // Không có ;

const int MAX = 100;

const float PI = 3.14;

## Biểu thức

* Khái niệm
  + Tạo thành từ các toán tử (Operator) và các toán hạng (Operand).
  + Toán tử tác động lên các giá trị của toán hạng và cho giá trị có kiểu nhất định.
  + Toán tử: +, –, \*, /, %….
  + Toán hạng: hằng, biến, lời gọi hàm...
* Ví dụ
  + 2 + 3, a / 5, (a + b) \* 5, …

## Toán tử gán

* Khái niệm
  + Thường được sử dụng trong lập trình.
  + Gán giá trị cho biến.
* Cú pháp
  + <biến> = <giá trị>;
  + <biến> = <biến>;
  + <biến> = <biểu thức>;
  + Có thể thực hiện liên tiếp phép gán.
* Ví dụ

void main()

{

int a, b, c, d, e, thuong;

a = 10;

b = a;

thuong = a / b;

a = b = c = d = e = 156;

e = 156;

d = e;

c = d;

b = c;

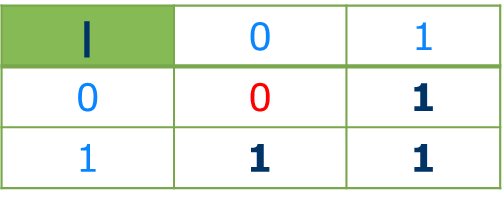
a = b;

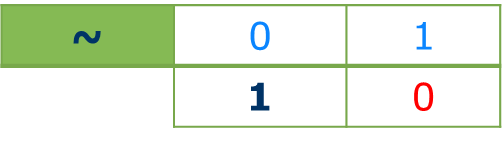
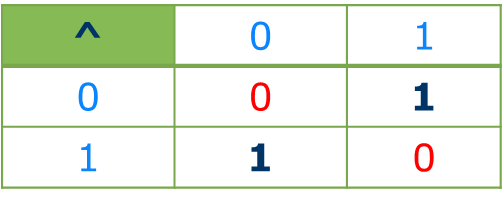
}

## Các toán tử toán học

* Toán tử 1 ngôi
  + Chỉ có một toán hạng trong biểu thức.
  + ++ (tăng 1 đơn vị), -- (giảm 1 đơn vị)
  + Đặt trước toán hạng
    - Ví dụ ++x hay --x: thực hiện tăng/giảm trước.
  + Đặt sau toán hạng
    - Ví dụ x++ hay x--: thực hiện tăng/giảm sau.
* Ví dụ
  + x = 10; y = x++; // y = 10 và x = 11
  + x = 10; y = ++x; // x = 11 và y = 11
* Toán tử 2 ngôi
  + Có hai toán hạng trong biểu thức.
  + +, –, \*, /, % (chia lấy phần dư)
  + x = x + y ⬄ x += y;
* Ví dụ
  + a = 1 + 2; b = 1 – 2; c = 1 \* 2; d = 1 / 2;
  + e = 1\*1.0 / 2; f = float(1) / 2; g = float(1 / 2);
  + h = 1 % 2;
  + x = x \* (2 + 3\*5); ⬄ x \*= 2 + 3\*5;

## Các toán tử trên bit

* Các toán tử trên bit
  + Tác động lên các bit của toán hạng (nguyên).
  + & (and), | (or), ^ (xor), ~ (not hay lấy số bù 1)
  + >> (shift right), << (shift left)
  + Toán tử gộp: &=, |=, ^=, ~=, >>=, <<=



* Ví dụ

void main()

{

int a = 5; // 0000 0000 0000 0101

int b = 6; // 0000 0000 0000 0110

int z1, z2, z3, z4, z5, z6;

z1 = a & b; // 0000 0000 0000 0100

z2 = a | b; // 0000 0000 0000 0111

z3 = a ^ b; // 0000 0000 0000 0011

z4 = ~a; // 1111 1111 1111 1010

z5 = a >> 2;// 0000 0000 0000 0001

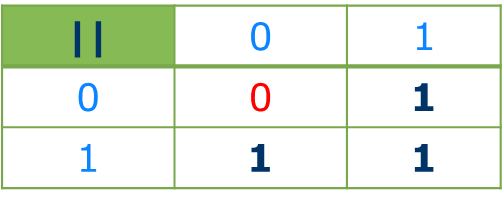
z6 = a << 2;// 0000 0000 0001 0100

}

## Các toán tử quan hệ

* Các toán tử quan hệ
  + So sánh 2 biểu thức với nhau
  + Cho ra kết quả 0 (hay false nếu sai) hoặc 1 (hay true nếu đúng)
  + ==, >, <, >=, <, <=, !=
* Ví dụ
  + s1 = (1 == 2); s2 = (1 != 2);
  + s3 = (1 > 2); s4 = (1 >= 2);
  + s5 = (1 < 2); s6 = (1 <= 2);

## Các toán tử luận lý

* Các toán tử luận lý
  + Tổ hợp nhiều biểu thức quan hệ với nhau.
  + && (and), || (or), ! (not)
  + Ví dụ
    - s1 = (1 > 2) && (3 > 4);
    - s2 = (1 > 2) || (3 > 4);
    - s3 = !(1 > 2);

## Toán tử điều kiện

* Toán tử điều kiện
  + Đây là toán tử 3 ngôi (gồm có 3 toán hạng)
  + <biểu thức 1> ? <biểu thức 2> : <biểu thức 3>
    - <biểu thức 1> đúng thì giá trị là <biểu thức 2>.
    - <biểu thức 1> sai thì giá trị là <biểu thức 3>.
* Ví dụ
  + s1 = (1 > 2) ? 2912 : 1706;
  + int s2 = 0;
  + 1 < 2 ? s2 = 2912 : s2 = 1706;

## Toán tử phẩy

* Toán tử phẩy
  + Các biểu thức đặt cách nhau bằng dấu ,
  + Các biểu thức con lần lượt được tính từ trái sang phải.
  + Biểu thức mới nhận được là giá trị của biểu thức bên phải cùng.
* Ví dụ
  + x = (a++, b = b + 2);
  + ⬄ a++; b = b + 2; x = b;

## Độ ưu tiên của các toán tử

|  |  |
| --- | --- |
| **Toán tử** | **Độ ưu tiên** |
| () [] -> .  ! ++ -- - + \* (cast) & sizeof  \* / %  + -  << >>  < <= > >=  == !=  &  |  ^  &&  ||  ?:  = += -= \*= /= %= &= …  , | 🡪  🡨  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡨  🡨  🡨 |

* Quy tắc thực hiện
  + Thực hiện biểu thức trong ( ) sâu nhất trước.
  + Thực hiện theo thứ tự ưu tiên các toán tử.

* Ví dụ
  + n = 2 + 3 \* 5;

=> n = 2 + (3 \* 5);

* + a > 1 && b < 2

=> (a > 1) && (b < 2)

## Câu lệnh

* Khái niệm
  + Là một chỉ thị trực tiếp, hoàn chỉnh nhằm ra lệnh cho máy tính thực hiện một số tác vụ nhất định nào đó.
  + Trình biên dịch bỏ qua các khoảng trắng (hay tab hoặc xuống dòng) chen giữa lệnh.
* Ví dụ

a=2912;

a = 2912;

a

=

2912;

* Phân loại
  + Câu lệnh đơn: chỉ gồm một câu lệnh.
  + Câu lệnh phức (khối lệnh): gồm nhiều câu lệnh đơn được bao bởi { và }
* Ví dụ

a = 2912; // Câu lệnh đơn

{ // Câu lệnh phức/khối lệnh

a = 2912;

b = 1706;

}

## Câu lệnh xuất

* Thư viện
  + #include <stdio.h> (standard input/output)
* Cú pháp
  + printf(<chuỗi định dạng>[, <đs1>, <đs2>, …]);
  + <chuỗi định dạng> là cách trình bày thông tin xuất và được đặt trong cặp nháy kép “ ”.
    - Văn bản thường (literal text)
    - Ký tự điều khiển (escape sequence)
    - Đặc tả (conversion specifier)
* Chuỗi định dạng
* Văn bản thường (literal text)

Được xuất y hệt như lúc gõ trong chuỗi định dạng.

* Ví dụ

Xuất chuỗi Hello World

printf(“Hello ”); printf(“World”);

printf(“Hello World”);

Xuất chuỗi a + b

printf(“a + b”);

* Ký tự điều khiển (escape sequence)
  + Gồm dấu \ và một ký tự như trong bảng sau:

|  |  |
| --- | --- |
| **Ký tự điều khiển** | **Ý nghĩa** |
| \a  \b  \n  \t  \\  \?  \” | Tiếng chuông  Lùi lại một bước  Xuống dòng  Dấu tab  In dấu \  In dấu ?  In dấu “ |

* + Ví dụ

printf(“\t”); printf(“\n”);

printf(“\t\n”);

* Đặc tả (conversion specifier)
  + Gồm dấu % và một ký tự.
  + Xác định kiểu của biến/giá trị muốn xuất.
  + Các đối số chính là các biến/giá trị muốn xuất, được liệt kê theo thứ tự cách nhau dấu phẩy.

|  |  |  |
| --- | --- | --- |
| **Đặc tả** | **Ý nghĩa** | **Kiểu khai báo** |
| %c  %d, %ld  %f, %lf  %s  %u | Ký tự  Số nguyên có dấu  Số thực  Chuỗi ký tự  Số nguyên không dấu | char  char, int, short, long  float, double  char[], char\*  unsigned int/short/long |

* Ví dụ
  + int a = 10, b = 20;
  + printf(“%d”, a); 🡺 Xuất ra 10
  + printf(“%d”, b); 🡺 Xuất ra 20
  + printf(“%d %d”, a, b); 🡺 Xuất ra 10 20
  + float x = 15.06;
  + printf(“%f”, x); 🡺 Xuất ra 15.060000
  + printf(“%f”, 1.0/3); 🡺 Xuất ra 0.333333

### Định dạng xuất

* Cú pháp
  + Định dạng xuất số nguyên: %nd
  + Định dạng xuất số thực: %n.kd
* Ví dụ:

int a = 1706;

float x = 176.85;

printf(“%10d”, a);printf(“\n”);

printf(“%10.2f”, x);printf(“\n”);

printf(“%.2f”, x);printf(“\n”);

### Chuỗi định dạng

Phối hợp các thành phần

int a = 1, b = 2;

Xuất 1 cong 2 bang 3 và xuống dòng.

printf(“%d”, a); // Xuất giá trị của biến a

printf(“ cong ”); // Xuất chuỗi “ cong ”

printf(“%d”, b); // Xuất giá trị của biến b

printf(“ bang ”); // Xuất chuỗi “ bang ”

printf(“%d”, a + b); // Xuất giá trị của a + b

printf(“\n”); // Xuất điều khiển xuống dòng \n

🡺 printf(“%d cong %d bang %d\n”, a, b, a+b);

## Câu lệnh nhập

* Thư viện
  + #include <stdio.h> (standard input/output)
* Cú pháp
  + scanf(<chuỗi định dạng>[, <đs1>, <đs1>, …]);
  + <chuỗi định dạng> giống định dạng xuất nhưng chỉ có các đặc tả.
  + Các đối số là tên các biến sẽ chứa giá trị nhập và được đặt trước dấu &
* Ví dụ, cho a và b kiểu số nguyên
  + scanf(“%d”, &a); // Nhập giá trị cho biến a
  + scanf(“%d”, &b); // Nhập giá trị cho biến b
  + 🡺 scanf(“%d%d”, &a, &b);
  + Các câu lệnh sau đây sai
    - scanf(“%d”, a); // Thiếu dấu &
    - scanf(“%d”, &a, &b);// Thiếu %d cho biến b
    - scanf(“%f”, &a); // a là biến kiểu số nguyên
    - scanf(“%9d”, &a); // không được định dạng
    - scanf(“a = %d, b = %d”, &a, &b”);

## Một số hàm hữu ích khác

* Các hàm trong thư việc toán học
  + #include <math.h>
  + 1 đầu vào: double, Trả kết quả: double
    - acos, asin, atan, cos, sin, …
    - exp, log, log10
    - sqrt
    - ceil, floor
    - abs, fabs
  + 2 đầu vào: double, Trả kết quả: double
    - double pow(double x, double y)

## Bài tập lý thuyết

1. Trình bày các kiểu dữ liệu cơ sở trong C và cho ví dụ.
2. Trình bày khái niệm về biến và cách sử dụng lệnh gán.
3. Phân biệt hằng thường và hằng ký hiệu.  
   Cho ví dụ minh họa.
4. Trình bày khái niệm về biểu thức.

Tại sao nên sử dụng cặp ngoặc đơn.

1. Trình bày cách định dạng xuất.
2. Nhập năm sinh của một người và tính tuổi của người đó.
3. Nhập 2 số a và b. Tính tổng, hiệu, tính và thương của hai số đó.
4. Nhập tên sản phẩm, số lượng và đơn giá. Tính tiền và thuế giá trị gia tăng phải trả, biết:
   1. tiền = số lượng \* đơn giá
   2. thuế giá trị gia tăng = 10% tiền