

KỸ THUẬT LẬP TRÌNH 1

GIẢNG VIÊN: NGUYỄN VĂN GIANG

CHƯƠNG 1

CÁC KHÁI NIỆM CƠ BẢN CỦA NGÔN NGỮ LẬP TRÌNH C/C++

- 1.1. Khái quát về ngôn ngữ lập trình C/C++
- 1.2. Các kiểu dữ liệu cơ sở
- 1.3. Biểu thức toán tử câu lệnh

1.1.KHÁI QUÁT VỀ NGÔN NGỮ LẬP TRÌNH C/C++

1.1.1. GIỚI THIỆU

1.1.1.1. Các khái niệm cơ bản

Chương trình (program): Tập hợp mã nguồn nhằm giải quyết một vấn đề cụ thể.

File chương trình viết bằng ngôn ngữ C: *.cpp.
Sau khi được biên dịch: *.exe

Lập trình (programming): Viết chương trình dựa vào một giải thuật (algorithm) và được cài đặt bằng một ngôn ngữ lập trình (programming language).

Phần mềm (software): Hệ thống các chương trình liên kết với nhau để giải quyết một công việc nào đó

1.1.1.2. Giới thiệu ngôn ngữ lập trình C/C++

C là một ngôn ngữ lập trình cấp cao được phát triển bởi Brian Kernighan và Dennis Ritchie làm việc tại AT&T Bell Labs của Mỹ năm 1972.

1990: C được tổ chức tiêu chuẩn quốc tế ISO chính thức công nhận với tên gọi là Standard C. C++ là một sự mở rộng của C chuẩn với mô hình lập trình hướng đối tượng (object-oriented programming).

1.1.1.3. Môi trường lập trình C/C++

DevCpp (4.9.9.2) là phần mềm mã nguồn mở của hãng Bloodshed Software.

Code::Blocks 17.12 là phần mềm mã nguồn mở của Code Blocks

Microsoft Visual Studio.Net 2010/2015/2017/2019 của Microsoft

1.1.2. CÁC THÀNH PHẦN CƠ BẢN CỦA NGÔN NGỮ LẬP TRÌNH C/C++

1.1.2.1. Bộ ký tự của C

Bộ chữ cái Latin: A, B, C, ..., Z, ..., a, b, c, ..., z

Bộ chữ số thập phân: 0, 1, 2, ..., 9

Các ký hiệu toán tử: +, -, *, /, %, =, !, ...

Các ký hiệu khác: , ; : _ ! # [] () , % , ? & . . .

Dấu cách (space) dùng để cách các từ.

1.1.2.2. Các từ khóa của C

Là các từ dành riêng của C, không được dùng nó vào các việc khác. Thông thường các từ hóa của C đều dùng chữ thường

Ví dụ :

if, else, switch, case, do, while, for, return, break, continuous, include, typedef, struct, int, char, float,...

1.1.2.3. Tên và cách đặt tên

Tên của C được bắt đầu bằng một chữ cái hoặc dấu gạch dưới , tiếp theo có thể là chữ cái, chữ số, dấu gạch nối

- Không được có khoảng trống trong tên
- C phân biệt chữ hoa chữ thường đối với tên
- Khi viết chương trình ta nên đặt tên sao cho chúng nói lên được ý nghĩa của đối tượng mà chúng biểu diễn.

1.1.2.4. Cách ghi chú thích trong chương trình

Khi viết chương trình nên đưa vào các câu chú thích (comment) để làm cho chương trình rõ ràng, dễ hiểu.

Cách 1:

```
//Tac gia chuong trinh: Nguyen Thanh Nam  
//Lop 19T4-LTM1  
int main ()  
... .
```

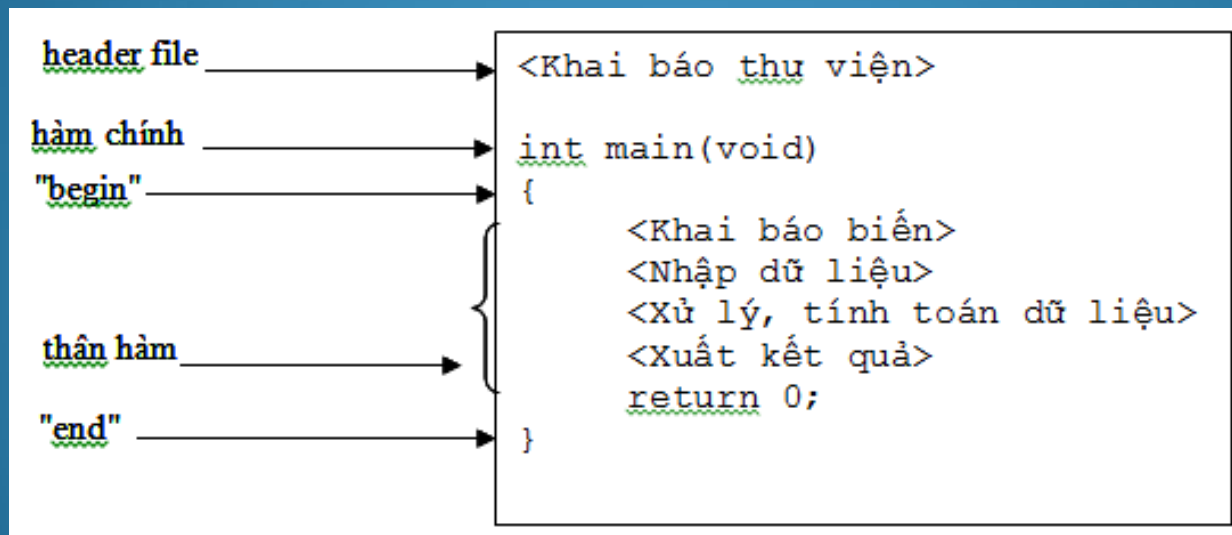
Cách 2:

```
/*Tac gia chuong trinh: Nguyen Thanh Nam  
Lop 19T4-LTM1*/  
int main ()  
... .
```

1.1.2.5. Câu lệnh và dấu chấm câu

Mỗi câu lệnh trong chương trình phải được kết thúc bằng dấu chấm phẩy (;) tuy nhiên có một số lệnh đặc biệt thì không có dấu chấm phẩy (;) ở cuối câu.

1.1.12.6. Cấu trúc của chương trình



1.1.2.7. Các bước lập trình cơ bản

1. Tìm hiểu và phân tích và mô hình hóa bài toán
2. Xây dựng thuật toán phù hợp để giải quyết bài toán
3. Cài đặt thuật toán bằng ngôn ngữ lập trình
4. Biên dịch chương trình và sửa lỗi
5. Chạy thử nghiệm và hoàn thiện chương trình
6. Tối ưu chương trình

1.1.3. CÁCH TẠO FILE MÃ NGUỒN



1.2. CÁC KIỂU DỮ LIỆU CƠ SỞ

1.2.1. Các kiểu dữ liệu cơ sở

1.2.1.1. Kiểu ký tự (character)

Mỗi ký tự là một **ký hiệu** xác định được định nghĩa trong bảng mã chuẩn ASCII (xem phụ lục).

Mỗi ký tự có độ dài **1 byte** và tương ứng với một giá trị số là mã ASCII.

Mỗi ký tự được bao bằng dấu nháy đơn, ví dụ 'a', 'A', '1', '2', ..và có định dạng là **%c**.

Ngoài ra, cũng có thể dùng kiểu ký tự để biểu diễn số nguyên

Kiểu	Số byte	Minimum	Maximum	Định dạng
char	1	-128	127	%d, %i
unsigned char	1	0	255	%d, %i

1.2.1.2. Kiểu số nguyên (integer number)

Kiểu	Số byte	Minimum	Maximum	Định dạng
short	2	-32.768	32.767	%hi, %hd
unsigned short	2	0	65.535	%hu
int	4	-2.147.483.648	2.147.483.647	%i, %d
unsigned int	4	0	4.294.967.296	%u
long	4	-2.147.483.648	2.147.483.647	%li, %ld
unsigned long	4	0	4.294.967.245	%lu
long long	8	-9.223.372.036.854.775.808	9223372036854775807	%lld
unsigned long long	8	0	18446744073709551616	%llu

1.2.1.2. Kiểu số thực (real number)

Kiểu	Số byte	Minimum	Maximum	Định dạng
float	4	-3.4×10^{38}	3.4×10^{38}	%f, %e, %g
double	8	-1.7×10^{308}	1.7×10^{308}	%lf, %le, %lg
long double	10	-3.4×10^{4932}	3.4×10^{4932}	%Lf, %Le, %Lg

1.2.2. KHÁI NIỆM VỀ HẰNG VÀ BIẾN

1.2.2.1. Hằng (Constant)

Một hằng có thể là một ký tự, số nguyên, số thực
Giá trị của hằng không thay đổi trong phạm vi mà hằng được định nghĩa.

Khai báo hằng:

#define <Tên hằng> <giá trị>

(Không có “;” ở cuối câu lệnh)

Ví dụ: Khai báo hằng hằng PI có giá trị là 3.14159

#define PI 3.14159

2.2.2.2. Biến (Variable)

Biến là một tên gọi đại diện cho một địa chỉ xác định trong bộ nhớ.

Biến được khai báo phải thuộc một kiểu dữ liệu xác định

Khai báo biến:

<Kiểu dữ liệu> <Tên biến>;

Ví dụ:

int n;

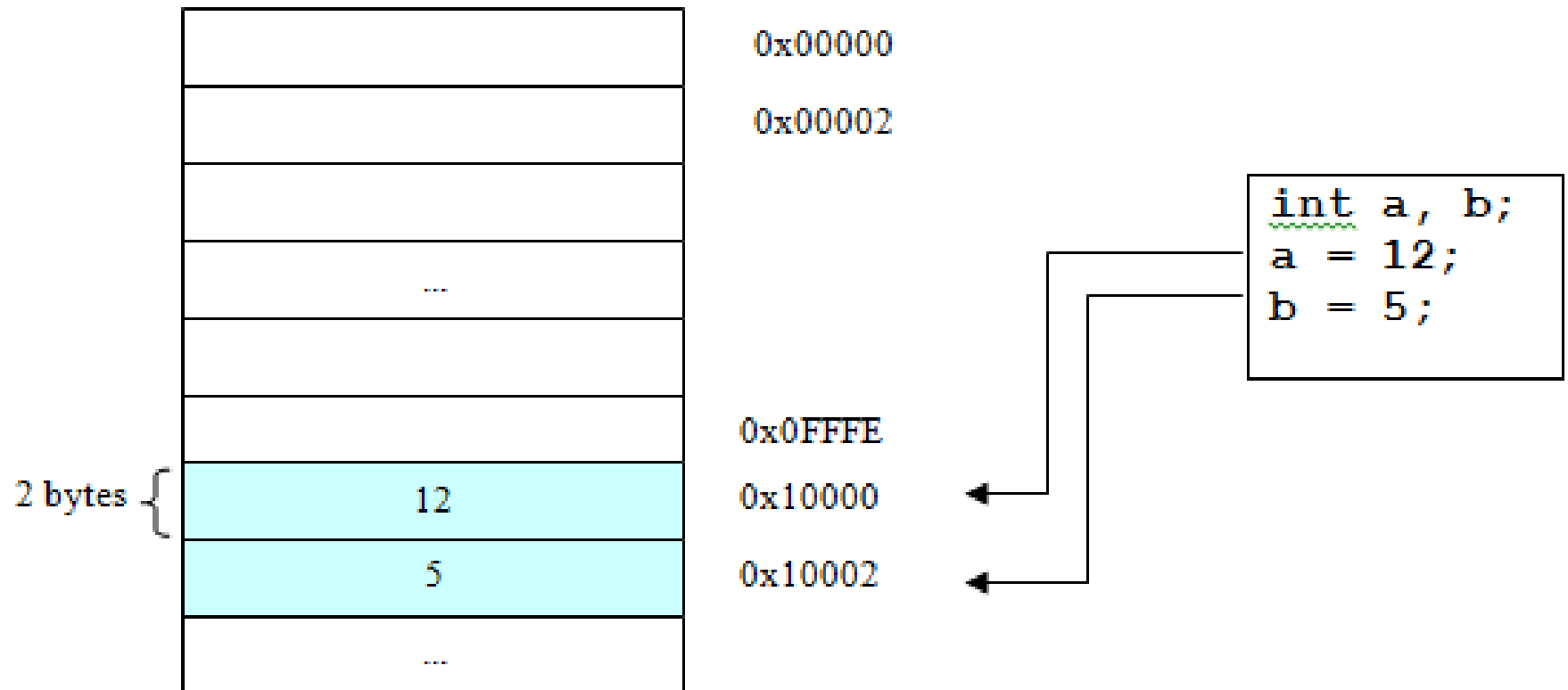
<Kiểu dữ liệu> <Danh sách tên biến>;

Ví dụ:

float x,y;

Memory/RAM

Address



1.3. BIỂU THỨC – TÓÁN TỬ – CÂU LỆNH

1.3.1. BIỂU THỨC VÀ TÓÁN TỬ

1.3.1.1. Khái niệm về biểu thức

Biểu thức là một công thức tính toán bao gồm các phép toán, hằng, biến, hàm, và các dấu ngoặc.

Thứ tự ưu tiên:

1. Biểu thức trong ngoặc, phép gọi hàm,
2. !
3. / , * , \ , % , &&
4. + , - , ||
5. < , > , == , <= , >= , !=

Ví dụ :

Nếu $a=3$, $b=5$, $c= 2$ thì:

Biểu thức : $a + b * c$

có giá trị là 13

Biểu thức: $(b * b - 4 * a * c) / (2 * a)$

có giá trị là 1/6

Biểu thức: $(a \% 2) != 0$

Có giá trị là 1

1.3.1.2. Toán tử

Toán tử số học:

+, -, *, /, % (chia lấy dư)

Ví dụ:

```
int a = 5, b = 2;
```

```
float x = 11;
```

```
char c = 'm';
```

a+b

có giá trị là **7**

(c+1)

có giá trị là **'n'**

a-b

có giá trị là **3**

(c - 2)

có giá trị là **'k'**

a*b

có giá trị là **10**

char (c*2)

có giá trị là **'f'**

a / b

có giá trị là **2**

x/b

có giá trị là **5.5**

int (c/2)

có giá trị là **54**

a%b

có giá trị là **1**

c%b

có giá trị là **1**

x%b

→ **lỗi**

Toán tử quan hệ (so sánh):

`==`, `<`, `<=`, `>`, `>=`, `!=` (không bằng)

Kết quả của phép so sánh là một giá trị logic true (=1) hoặc false (=0)

Ví dụ:

```
int a = 5, b = 2;
```

`a==b`

có giá trị là **false**

`a>b`

có giá trị là **true**

`a>=b`

có giá trị là **true**

`a!=b`

có giá trị là **true**

Toán tử logic: ! (NOT), && (AND), || (OR)

Kết quả thực hiện phép toán logic là một giá trị logic true hoặc false.)

Ví dụ:

a = true, b = false;

&&	False	True
False	False	False
True	False	True

a&&b

có giá trị là **false**

a||b

có giá trị là **true**

!b

có giá trị là **true**

	False	True
False	False	True
True	True	True

Toán tử tăng giảm: ++, --

Các toán tử này thực hiện việc tăng (++) hay giảm (--) giá trị của biến hay biểu thức 1 đơn vị.

Ví dụ:

```
int a = 5, b=9, c=7;
```

```
a++ ; //a=6
```

```
b--; // b=8
```

```
cout<<a++ ;
```

xuất ra màn hình số 5 rồi mới tăng a lên 6

```
cout<<++b;
```

tăng b lên 10 rồi xuất ra màn hình số b (số 10)

```
cout<<c++<<--c;
```

in ra màn hình số 77

1.3.2. Lệnh và khối lệnh

Lệnh đơn là một khai báo hay phát biểu được ghi trên 1 dòng và kết thúc bằng dấu chấm phẩy (;).

Khối lệnh là tập hợp các lệnh đơn và được bao trong cặp ngoặc nhọn ({}).

```

    int count=5; // Lệnh đơn

    printf("Cac so tu 5 den 0: \n"); // Lệnh đơn

    while(count>=0)
    {
        printf("%d, ", count); // Lệnh đơn
        count--; // Lệnh đơn
    }
    
```

Khối lệnh {

1.3.3. Chuyển đổi kiểu dữ liệu

(*<Kiểu dữ liệu đích>*) *<Biểu thức>*

Ví dụ:

```
int a = 5, b=10;
```

```
a/b ;
```

có giá trị là 0

```
float a)/b;
```

có giá trị là 0,5