KỸ THUẬT LẬP TRÌNH

Chương 1. Tổng quan về C/C++

Liên hệ gv: thuong.bui@ut.edu.vn

1.1 Thuật toán

Giải thuật (hay còn gọi là thuật toán - tiếng Anh là Algorithms) là một chuỗi hữu hạn các thao tác để giải một bài toán nào đó. Giải thuật là độc lập với các ngôn ngữ lập trình, tức là một giải thuật có thể được triển khai trong nhiều ngôn ngữ lập trình khác nhau.

Ví dụ: Thuật toán để giải phương trình P(x): ax + b = 0.
Input: a, b
Output: Nghiệm x
Thực hiện:

- Nếu a = 0:
  - b = 0 thì P(x) có nghiệm bất kì
  - b ≠ 0 thì P(x) vô nghiệm
- Nếu a ≠ 0 thì P(x) có duy nhất một nghiệm x = (- b)/a

1.1 Thuật toán

Các tính chất quan trọng của giải thuật bao gồm:

- **Tính hữu hạn**: Giải thuật phải luôn kết thúc sau một số hữu hạn bước.
- **Tính xác định**: Mỗi bước của giải thuật phải được xác định rõ ràng, phải được thực hiện chính xác, và phải mang một mục đích nhất định.
- **Tính đúng**: Giải thuật phải đảm bảo tính đúng và chính xác.
- **Tính phổ biến**: Giải thuật phải giải quyết được lớp các vấn đề tương tự.
- **Tính hiệu quả**: Các thao tác trong giải thuật phải có khả năng giải quyết hiệu quả vấn đề trong điều kiện thời gian và tài nguyên cho phép.
- **Dữ liệu đầu vào xác định (input)**: Giải thuật phải xử lý một số lượng dữ liệu đầu vào xác định.
- **Kết quả đầu ra (output)**: Giải thuật sẽ giải quyết các bài toán bằng cách xử lý các loại dữ liệu đầu vào để cho kết quả dữ liệu đầu ra đã xác định.

1.2 Cách biểu diễn thuật toán

Dưới đây là một số cách thường được sử dụng để biểu diễn giải thuật:

- **Sử dụng ngôn ngữ tự nhiên**: Liệt kê tuần tự các bước để giải quyết bài toán. Cách này đơn giản và không cần kiến thức về biểu diễn giải thuật, tuy nhiên nó thường dài dòng và đôi khi khó hiểu.
- **Sử dụng lưu đồ (sơ đồ khối):** Sử dụng các hình khối khác nhau để biểu diễn giải thuật. Cách này trực quan và dễ hiểu, tuy nhiên hơi cồng kềnh.
- **Sử dụng mã giả**: Sử dụng ngôn ngữ tựa ngôn ngữ lập trình để biểu diễn giải thuật. Cách làm này đỡ cồng kềnh hơn sơ đồ khối tuy nhiên không trực quan.
- **Sử dụng ngôn ngữ lập trình**: Sử dụng các ngôn ngữ máy tính như C, Pascal, ... để biểu diễn giải thuật. Cách này đòi hỏi phải có kiến thức và kỹ năng về ngôn ngữ lập trình được sử dụng.

1.2 Cách biểu diễn thuật toán

Quy tắc biểu diễn thuật toán bằng sơ đồ khối:


Bắt đầu hoặc kết thúc

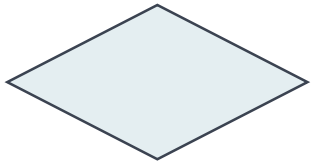

Khối chương trình con

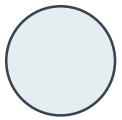

Phần dữ liệu nhập, xuất

Hướng đi của thuật toán

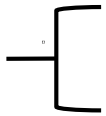

Quá trình xử lý, tính toán


Bước chuẩn bị


Điều kiện rẽ nhánh

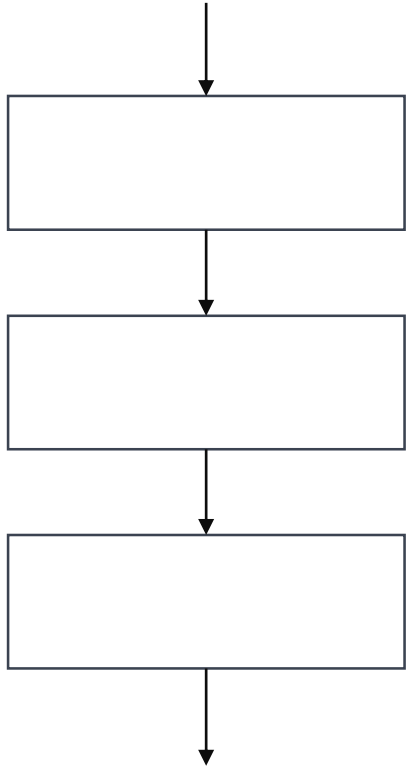

Điểm nối


Tệp tin dữ liệu

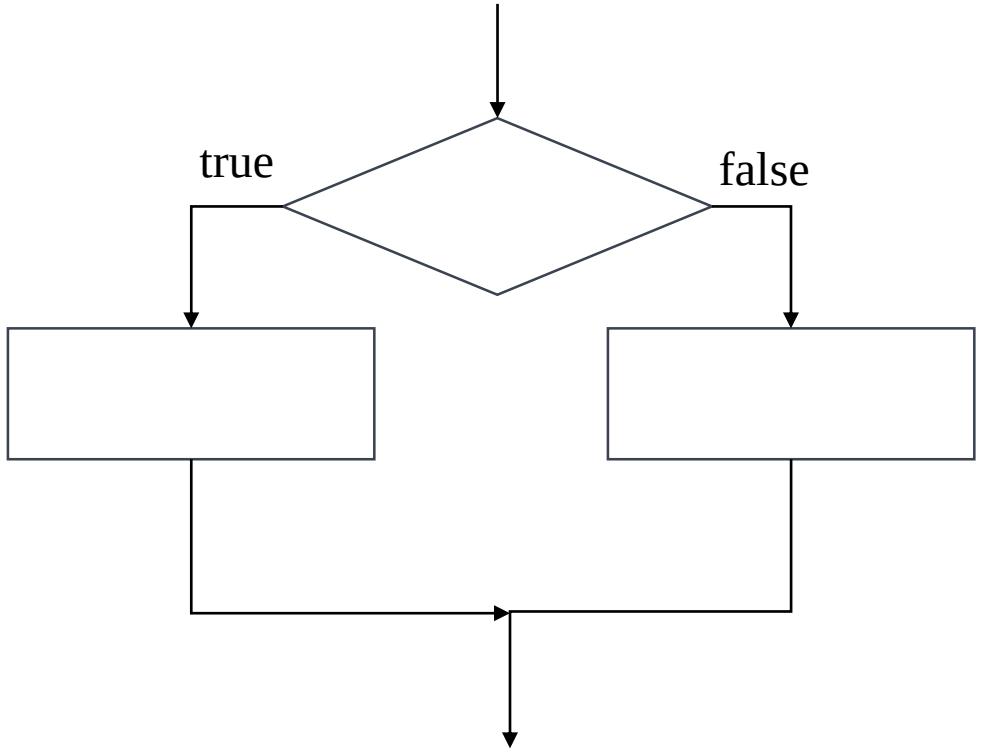

Chú thích

1.2 Cách biểu diễn thuật toán

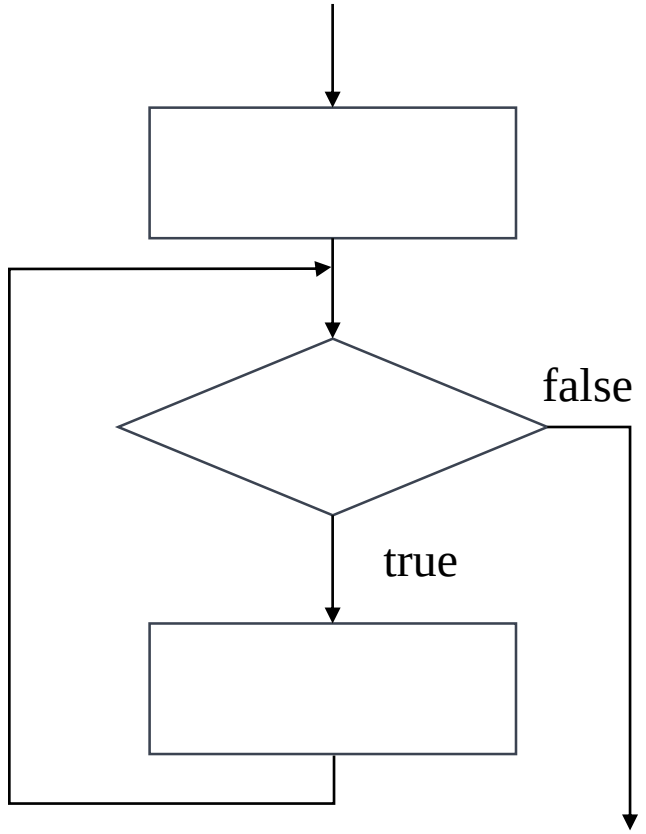
Các dạng thuật toán phổ biến:



Cấu trúc tuần tự



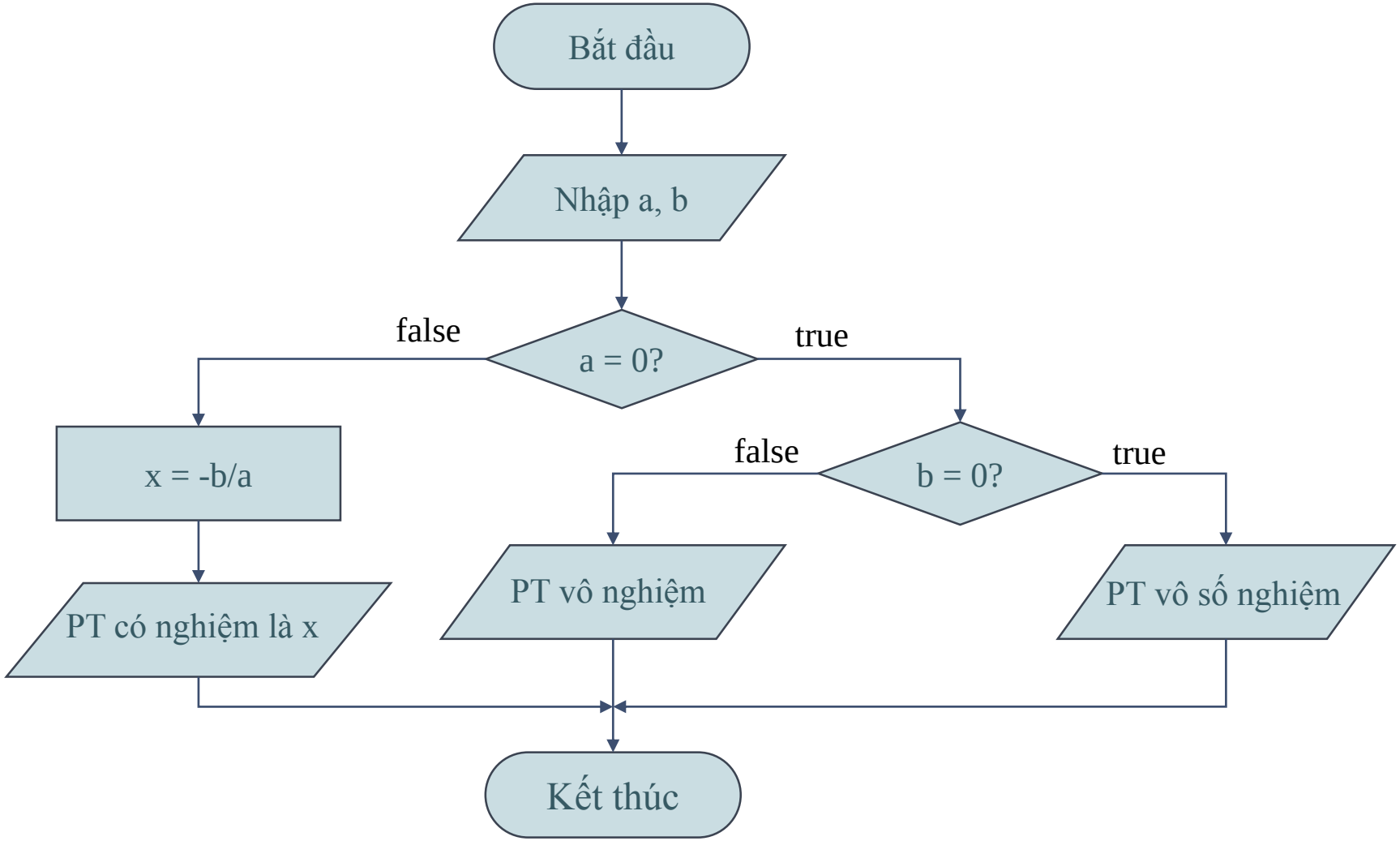
Cấu trúc rẽ nhánh



Cấu trúc lặp

1.2 Cách biểu diễn thuật toán

Ví dụ: Lưu đồ thuật toán giải phương trình: ax + b = 0



1.3 Lập trình và ngôn ngữ lập trình

Lập trình (Programming) - là kỹ thuật cài đặt một hoặc nhiều thuật toán trừu tượng có liên quan với nhau bằng một hoặc nhiều ngôn ngữ lập trình (NNLT) để tạo ra một chương trình máy tính.

Ngôn ngữ lập trình (Programming language) - là một dạng ngôn ngữ được thiết kế và chuẩn hóa để truyền các chỉ thị cho máy tính. NNLT có thể được dùng để tạo ra các chương trình nhằm mục đích điều khiển máy tính hoặc mô tả các thuật toán để người khác đọc hiểu.

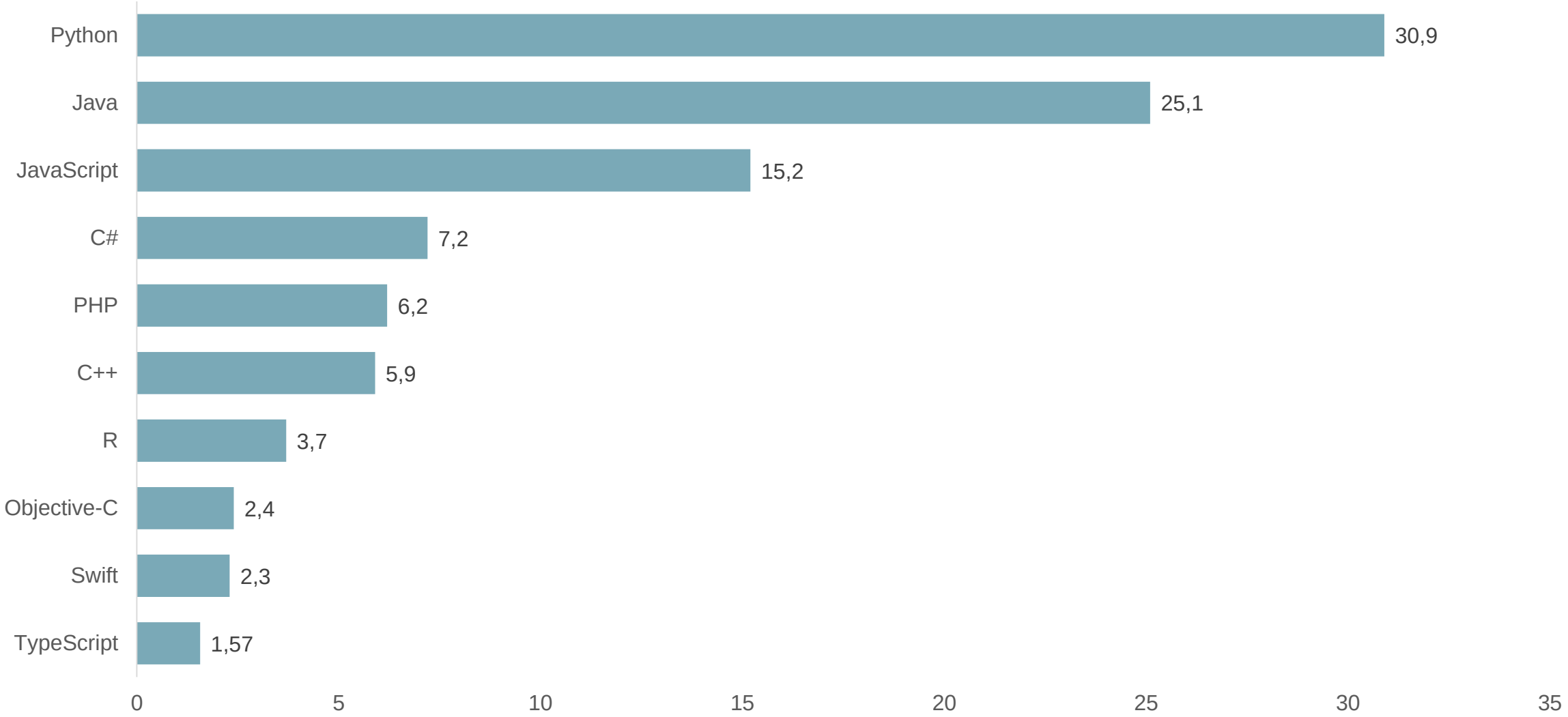
1.3 Lập trình và ngôn ngữ lập trình

Phân loại ngôn ngữ lập trình:

- **Ngôn ngữ máy (mã máy):** Là ngôn ngữ nền tảng của bộ vi xử lý. Các chương trình được viết trong tất cả các loại ngôn ngữ khác cuối cùng đều được chuyển thành ngôn ngữ máy trước khi chương trình đó được thi hành.
- **Hợp ngữ:** Hợp ngữ tương tự như ngôn ngữ máy nhưng lại sử dụng các ký hiệu gợi nhớ (hay mã lệnh hình thức) để biểu diễn cho các mã lệnh của máy. Các chương trình hợp ngữ được chuyển sang mã máy thông qua một chương trình đặc biệt gọi là trình hợp dịch (assembler).
- **Ngôn ngữ cấp cao:** Bao gồm các danh từ, động từ, ký hiệu toán học, liên hệ và các thao tác luận lý. Các chương trình viết bằng ngôn ngữ cấp cao có thể chạy trên các loại máy tính khác nhau (sử dụng các bộ vi xử lý khác nhau).

1.3 Lập trình và ngôn ngữ lập trình

Các ngôn ngữ lập trình thông dụng (tỷ lệ % thống kê 2024):



1.4 Quy trình viết và thực thi chương trình

1.5 Bộ ký tự và từ khóa trong C/C++

Bộ kí tự (Character set):

Có phân biệt hoa thường

- 26 chữ cái Latinh lớn: A, B, C..., Z
- 26 chữ cái Latinh nhỏ: a, b, c ..., z
- 10 chữ số thập phân: 0, 1, 2...9
- Các ký hiệu toán học: +, -, *, /, =, <, >
- Các ký hiệu đặc biệt: . , ; : " ' _ % # ! ^ [ ] { } ( ) ...
- Dấu cách hay khoảng trống, xuống hàng (\n) và tab (\t)

1.5 Bộ ký tự và từ khóa trong C/C++

Bộ từ khóa (Keywords):

asm	auto	bool	break
case	catch	char	class
const	const_cast	continue	default
delete	else	extern	do
enum	false	double	explicit
float	dynamic_cast	export	for
friend	goto	if	inline
int	long	mutable	namespace
new	operator	private	protected
public	register	reinterpret_cast	return
short	signed	sizeof	static
static_cast	struct	switch	template
this	throw	true	try
typedef	typeid	typename	union
unsigned	using	virtual	void
volatile	wchar_t	while	

1.6 Định danh (identifier)

Định danh (đặt tên) là dùng một dãy kí tự để gọi tên các đối tượng trong chương trình như biến, hằng, hàm, mảng, …

Một số qui tắc cần tuân theo khi định danh:

- Không được bắt đầu bằng chữ số, không được trùng với từ khóa.
- Chỉ được sử dụng các ký tự gồm chữ cái (A..Z,a..z), chữ số (0..9) và dấu gạch dưới ‘_’.

1.7 Biến và hằng

Biến

Biến là một đại lượng được người lập trình định nghĩa và được đặt tên thông qua việc khai báo biến:

Cú pháp khai báo chung:

```
<kiểu_dữ _liệu> <tên_biến>;
```

Khai báo nhiều biến có cùng một kiểu dữ liệu:

```
<kiểu_dữ _liệu> <tên_biến1>, <tên_biến2>, …;
```

Khai báo và khởi tạo giá trị cho biến:

```
<kiểu_dữ _liệu> <tên_biến> = <giá trị_khởi_tạo>;
```

1.7 Biến và hằng

Các kiểu dữ liệu cơ bản

Kiểu dữ liệu	Từ khóa	Kích thước	Miền giá trị
Kí tự	char	1 byte	-128 → +127
	unsigned char	1 byte	0 → 255
Số nguyên	int	4 byte	-32768 → 32767 (- 2^{15} →2^{15}-1)
	unsigned int	4 byte	0 → 65535 (0→2^{16} - 1)
	long	4 byte	-2147483648 → 2147483647 (-2^{31}→2^{31}-1)
	unsigned long	4 byte	0 → 4294967295 (0→ 2^{32} -1)
Số thực	float	4 byte	$3.4*10^{-38}$ → $3.4*10^{38}$
	double	8 byte	$1.7*10^{-308}$ → $1.7*10^{308}$
	long double	12 byte	$3.4*10^{4932}$ → $1.1*10^{4932}$
Logic	bool	1 byte	true/false

1.7 Biến và hằng

Hằng (constant)

Là đại lượng không đổi trong suốt quá trình thực thi của chương trình.

Hằng có thể là một chuỗi ký tự, ký tự, con số xác định.

Khai báo:

```
#define <Tên_hằng> <Giá_trị>
const <Kiểu_dữ _liệu> <Tên_hằng> = <Giá_trị>;
```

Ví dụ:

```
#define  PI  3.14
const int MAX = 100;
```

1.8 Chú thích

Trong ngôn ngữ lập trình C/C++, nội dung chú thích có thể được viết bằng hai cách:

- Cách 1: */*chú_thích*/*

Cách này có thể viết chú thích trên một hoặc nhiều dòng.

- Cách 2: *//Chú thích*

Cách này chỉ viết chú thích trên một dòng (tức là chú thích kết thúc khi ta ấn phím enter).

Chú ý: *Chú thích có thể được viết ở bất kì vị trị nào trong chương trình và nó không ảnh hưởng gì đến kết quả chạy chương trình.*

1.9 Vào/ra trong C++

Xuất dữ liệu ra màn hình với cout và toán tử <<

```
cout << biểu thức 1 ;
cout << biểu thức 2;
cout << biểu thức 3 ;
…
```

Hoặc:

```
cout << biểu thức 1 << biểu thức 2  << …<< biểu thức n;
```

Biểu thức là chuỗi ký tự thì đặt nội dung trong cặp dấu nháy đôi “ ”

Để xuất ra 1 ký tự đặc biệt ta sử dụng trước ký tự đó dấu \ (Ví dụ \’ sẽ xuất ra dấu ’)

1.9 Vào/ra trong C++

Định dạng xuất ra màn hình:

- Xuống dòng: endl hoặc ‘\n’
- Tab ngang: ‘\t’

Cần khai báo **#include<iomanip>**

- **setw(n):** Qui định độ rộng dành để in ra các giá trị là n cột màn hình.
- **setprecision(n):** Chỉ định số chữ số sẽ in ra là n.
- **setprecision(n)** kết hợp với **fixed**: Chỉ định số chữ số của phần thập phân in ra là n. Số sẽ được làm tròn trước khi in ra.

1.9 Vào/ra trong C++

Nhập dữ liệu từ bàn phím với cin và toán tử ≫

Để nhập dữ liệu vào cho các biến có tên biến_1, biến_2, biến_3 chúng ta sử dụng câu lệnh:

```
cin >> biến_1 ;
cin >> biến_2 ;
cin >> biến_3 ;
```

Hoặc:

```
cin >> biến_1 >> biến_2 >> biến_3 ;
```

1.9 Vào/ra trong C++

Nhập dữ liệu từ bàn phím với cin và toán tử ≫

Toán tử nhập >> chủ yếu làm việc với dữ liệu kiểu số. Để nhập kí tự hoặc xâu kí tự, C++ cung cấp các phương thức (hàm) sau:

- **cin.get(c):** cho phép nhập một kí tự vào biến kí tự c;
- **cin.getline(s, n):** cho phép nhập tối đa n-1 kí tự vào xâu s.

Trước khi sử dụng các phương thức cin.get(c) hoặc cin.getline(s, n) nên sử dụng phương thức cin.ignore(1) để lấy ra kí tự xuống dòng còn sót lại trong bộ đệm nếu trước đó ta nhập dữ liệu bằng toán tử nhập >>.

1.10 Biểu thức và toán tử

Biểu thức:

Biểu thức là sự kết hợp các toán tử và các toán hạng. Biểu thức có thể là một hằng, một biến, một hàm, …

- Toán tử có thể là dấu các phép toán: +, - * , /,… hay >, >=, …
- Toán hạng có thể là hằng, biến, hàm.

1.10 Biểu thức và toán tử

Toán tử số học:

Phép toán	Ý nghĩa	Ví dụ
+	Phép cộng	a+b
-	Phép trừ	a-b
*	Phép nhân	a*b
/	Phép chia	a/b - Chia 2 số nguyên sẽ bỏ phần thập phân
%	Phép lấy phần dư	a%b - Dành cho số nguyên.

1.10 Biểu thức và toán tử

Toán tử gán:

=, +=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=

Ví dụ:

a += 5; tương đương với a = a + 5;

a -= 5; tương đương với a = a - 5;

a /= b; tương đương với a = a / b;

a *= b + 1; tương đương với a = a * (b + 1);

1.10 Biểu thức và toán tử

Toán tử tăng giảm:

Các toán tử: ++ và --

a++; a+=1; a = a+1; // Các câu lệnh là tương đương.

Lưu ý: Các toán tử ++/-- đặt trước (tiền tố) hay sau (hậu tố) trong biểu thức.

Ví dụ:

B = 3; A = ++B; // A bằng 4, B bằng 4

Ví dụ:

B = 3; A = B++; // A bằng 3, B bằng 4

1.10 Biểu thức và toán tử

Toán tử so sánh:

Toán tử	Ý nghĩa
>	Lớn hơn
>=	Lớn hơn hoặc bằng
<	Nhỏ hơn
<=	Nhỏ hơn hơn hoặc bằng
==	Bằng
!=	Khác
Kết quả cho true (đúng) hoặc false (sai).	

1.10 Biểu thức và toán tử

Toán tử logic:

Toán tử	Ý nghĩa
&&	AND: Cho kết quả true nếu cả 2 điều kiện có kết quả true, còn lại cho false
||	OR: Cho kết quả true nếu chỉ cần 1 trong 2 điều kiện có kết quả true, còn lại cho false khi cả 2 điều kiện cùng false
!	NOT: Tác động lên các giá trị riêng lẻ, chuyển true thành false hoặc ngược lại.

1.10 Biểu thức và toán tử

Toán tử xử lý bit:

Toán tử	Mô tả
(x & y)	Mỗi vị trí của bit trả về kết quả là 1 nếu bit của hai toán hạng là 1, còn lại là 0
(x | y)	Mỗi vị trí của bit trả về kết quả là 1 nếu bit của một trong hai toán hạng là 1.
(~ x)	Đảo ngược giá trị của toán hạng (1 thành 0 và ngược lại).
(x ^ y)	Giống nhau cho 0, khác nhau cho 1 (0^0 = 0, 1^1 = 0, 0^1 = 1, 1^0 = 1)
(x <<n)	Dịch sang trái n bit, tương đương x* 2^n
(x >>n)	Dịch sang phải n bit, tương đương x/ 2^n

1.10 Biểu thức và toán tử

Toán tử điều kiện 3 ngôi:

Điều kiện ? X : Y

Nếu điều kiện là true ? thì nó trả về giá trị X : nếu không thì trả về Y

1.10 Biểu thức và toán tử

Một số toán tử khác:

Toán tử	Ý nghĩa
sizeof	Toán tử sizeof trả về kích cỡ của một biến. Ví dụ: sizeof(a), với a là int, sẽ trả về 4 bytes
, (phẩy)	Giá trị của toàn biểu thức phẩy là giá trị của biểu thức cuối cùng trong danh sách được phân biệt bởi dấu phẩy (biểu _thức_1, biểu_thức_2,…, biểu_thức_n) m = (t = 2, t*t + 3); ⇔ m = 7
. (dot) và -> (arrow)	Toán tử thành viên được sử dụng để tham chiếu các phần tử đơn của các lớp, các cấu trúc và union
cast	Toán tử ép kiểu biến đổi một kiểu dữ liệu thành kiểu khác. Ví dụ: int(2.2000) sẽ trả về 2. Cú pháp: type_cast <new_type> (expression); type: const, dynamic, reinterpret, static Ví dụ: reinterpret_cast<char*>(&x);
&	Toán tử con trỏ & (toán tử 1 ngôi) trả về địa chỉ của một biến.
*	Toán tử con trỏ * là trỏ tới một biến.

1.10 Biểu thức và toán tử

Chuyển đổi kiểu ngầm định:

Trong cùng 1 biểu thức, nếu các toán hạng không cùng kiểu với nhau thì trước khi tính toán giá trị của biểu thức, chương trình dịch sẽ thực hiện việc chuyển đổi kiểu ngầm định (nếu được) theo nguyên tắc "Kiểu có phạm vi giá trị biểu diễn nhỏ hơn sẽ được chuyển sang kiểu có phạm vị giá trị biểu diễn lớn hơn".

Sơ đồ chuyển đổi kiểu ngầm định:

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

Ép kiểu (type casting): Trong một số trường hợp, ta bắt buộc phải sử dụng ép kiểu để tạo ra một biểu thức hợp lệ như sau:

(<tên kiểu>) (<biểu thức>)

Ví dụ: Để tạo biểu thức số học hợp lệ sau 8.0 % 3, ta cần thực hiện ép kiểu như sau:

(int) 8.0 % 3

1.10 Biểu thức và toán tử

Độ ưu tiên của các toán tử:

Độ ưu tiên	Nhóm ưu tiên	Toán tử	Thứ tự thực hiện
1	Scope (Phạm vi)	::	Left-to-right
2	Postfix (unary) Hậu tố (một ngôi)	++ -- () [] . →	Left-to-right
3	Prefix (unary) Tiền tố (một ngôi)	++ -- ~ ! + - & * new delete sizeof (type)	Right-to-left

1.10 Biểu thức và toán tử

Độ ưu tiên của các toán tử:

Độ ưu tiên	Nhóm ưu tiên	Toán tử	Thứ tự thực hiện
4	Pointer-to-member (Trỏ đến thành viên)	.* ->*	Left-to-right
5	Arithmetic: scaling (Số học: tỷ lệ)	* / %	Left-to-right
6	Arithmetic: addition (Số học: tăng giảm)	+ -	Left-to-right
7	Bitwise shift (Dịch bit)	<< >>	Left-to-right
8	Relational (So sánh hơn)	< > <= >=	Left-to-right
9	Equality (So sánh bằng)	== !=	Left-to-right

1.10 Biểu thức và toán tử

Độ ưu tiên của các toán tử:

Độ ưu tiên	Nhóm ưu tiên	Toán tử	Thứ tự thực hiện
10	Bit AND	&	Left-to-right
11	Bit XOR	^	Left-to-right
12	Bit OR	|	Left-to-right
13	Logical AND		Left-to-right
14	Logical OR	||	Left-to-right
15	Assignment-level expressions (Gán)	= *= /= %= += -= >>= <<= &= ^= |= ?:	Right-to-left
16	Sequencing (Sắp xếp)	,	Left-to-right

1.11 Cấu trúc của một chương trình C/C++

Cấu trúc	Giải thích
# Tiền xử lý	Khai báo thư viện và macro
Khai báo biến, hàm,..;	Khai báo các biến toàn cục, khai báo nguyên mẫu hàm được sử dụng trong chương trình chính

int main()
{
Thân chương trình chính;
}
	Chương trình chính

Định nghĩa các hàm (thân hàm)	Định nghĩa thân hàm đã được khai báo

BÀI TẬP CHƯƠNG 1

Bài 1. Vẽ lưu đồ giải thuật giải phương trình:

$$ax^2 + bx + c = 0$$

với a, b, c bất kỳ.

Bài 2. Vẽ lưu đồ giải thuật giải hệ phương trình:

$$\begin{cases} ax + by = c \\ dx + ey = f \end{cases}$$

Với x, y là biến và a,b,c,d,e,f là tham số bất kỳ.

Bài 3. Vẽ lưu đồ giải thuật tính tổng của các số chẵn từ 10 số nguyên dương bất kỳ do người dùng nhập.

Bài 4. Vẽ lưu đồ thuật toán tìm số đảo ngược của một số nguyên dương (ghi chú: số đảo ngược của số 123 là 321)

BÀI TẬP CHƯƠNG 1

Bài 5. Viết chương trình nhập vào 2 số nguyên a và b. Xuất ra màn hình giá trị của a, b và kết quả của các phép tính: a+b, a-b, a/b, a*b, a%b.

Bài 6. Viết chương trình nhập vào 2 số thực a và b. Xuất ra màn hình giá trị của a, b và kết của quả các phép tính: a+b, a-b, a/b, a*b.

Bài 7. Nhập vào từ bàn phím họ tên và năm sinh cho 2 người. Xuất ra màn hình tên và tuổi của 2 người đó.

Bài 8. Viết chương trình tính giá trị trung bình của 3 số bất kỳ. Kết quả làm tròn đến 4 chữ số thập phân.

Bài 9. Viết chương trình tính khoảng cách từ điểm A(x, y, z) tới mặt phẳng:

$$ax + by + cz + d = 0$$

BÀI TẬP CHƯƠNG 1

Bài 10. Viết chương trình tính số tiền cước phải trả khi thuê dịch vụ taxi với cách tính như sau:

- Giá mở cửa: 10.000 đồng
- Giá cước từ km thứ 1 đến km thứ 10: 12.000 đồng/km
- Giá cước từ km thứ 11 trở đi: 10.000 đồng/km

Ví dụ nếu bạn đi 15km, thì số tiền bạn phải trả sẽ là:

- 10.000 đồng (giá mở cửa)
- 10km đầu: 10km * 12.000 đồng/km = 120.000 đồng
- 5km tiếp theo: 5km * 10.000 đồng/km = 50.000 đồng
- Tổng: 10.000 + 120.000 + 50.000 = 180.000 đồng