

NGÔN NGỮ LẬP TRÌNH C/C++

Nguyễn Hải Châu
Khoa Công nghệ thông tin
Trường Đại học Công nghệ
(Bài giảng tuần 2)

1

Nội dung

- Kiểu dữ liệu
- Biểu thức
- Câu lệnh

2

Kiểu dữ liệu đơn giản

3

Khái niệm

- Các ngôn ngữ lập trình (NNLT) đều có một số kiểu dữ liệu cơ bản
- Các yếu tố gắn với kiểu dữ liệu:
 - Tên kiểu
 - Số byte trong bộ nhớ để lưu trữ một đơn vị dữ liệu thuộc kiểu này
 - Miền giá trị của kiểu

4

Một số kiểu dữ liệu đơn giản trong C++

Loại dữ liệu	Tên kiểu	Số ô nhớ	Miền giá trị
Kí tự	char	1 byte	-128 .. 127
	unsigned char	1 byte	0 .. 255
Số nguyên	int	4 byte	$-2^{31} .. 2^{31}-1$
	unsigned int	4 byte	$0 .. 2^{32}-1$
	short	2 byte	- 32768 .. 32767
	long	4 byte	$-2^{31} .. 2^{31}-1$
Số thực	float	4 byte	$\pm 10^{-37} .. \pm 10^{+38}$
	double	8 byte	$\pm 10^{-307} .. \pm 10^{+308}$

5

Kiểu ký tự

char c, d; // c, d được phép gán giá trị từ -128 đến 127
unsigned char e, f; // e được phép gán giá trị từ 0 đến 255
c = 65; d = 179; // d có giá trị ngoài miền cho phép
e = 179; f = 330; // f có giá trị ngoài miền cho phép
cout << c << int(c); // in ra chữ cái 'A' và giá trị số 65
cout << d << int(d); // in ra là kí tự 'I' và giá trị số -77
cout << e << int(e); // in ra là kí tự 'I' và giá trị số 179
cout << f << int(f); // in ra là kí tự 'J' và giá trị số 74

6

Ví dụ: Tính diện tích và chu vi hình tròn

```
#include <iostream.h>
#include <iomanip.h>
void main()
{
    float r = 2; // r là tên biến dùng để chứa bán kính
    cout << "Diện tích = " << setiosflags(ios::showpoint);
    cout << setprecision(3) << r * r * 3.1416;
    getch();
}
```

7

Hằng: Khai báo và sử dụng



8

Hằng là gì?

- Là các giá trị cố định, được đặt tên gọi trong chương trình C/C++
- Giá trị của hằng không thay đổi trong khi chương trình thực hiện

9

Hằng nguyên

- Cách viết hằng nguyên (hệ 10):
 - Kiểu short, int: 3, -7
 - Kiểu unsigned: 3, 12345
 - Kiểu long, long int: 3L, -7L, 12345L
- Hằng nguyên có thể viết ở hệ 16 hoặc 8:
 - Hệ 16: 0xA1 (11 ở hệ 10)
 - Hệ 8: 013 (11 ở hệ 10)

10

Hằng thực

- Hằng thực có thể viết theo 2 cách
- Dạng dấu phẩy tĩnh: 3.2, -7.1, 3.1416
- Dạng dấu phẩy động:
 - Tổng quát: mEn hoặc mEn , trong đó m là phần định trị, n là phần bậc (phần mũ)
 - Ví dụ: 3.2 → 3.2e1, 3.2E1; 0.32 → 3.2e-1, 3.2E-1

11

Hằng ký tự

- Có hai cách viết hằng ký tự:
 - Với các ký tự có mặt chữ: 'A'
 - Các ký tự không có mặt chữ: Dùng chữ số hệ 8 hoặc 16 để biểu diễn mã của ký tự đó: '\33', '\x1B'
 - Một số hằng ký tự đặc biệt có cách viết riêng để tiện lợi và dễ nhớ
- Hằng ký tự không có khái niệm rỗng

12

Một số hằng ký tự đặc biệt

'\n'	: biểu thị kí tự xuống dòng (cũng tương đương với endl)
'\t'	: kí tự tab
'\a'	: kí tự chuông (tức thay vì in kí tự, loa sẽ phát ra một tiếng 'bíp')
'\r'	: xuống dòng
'\f'	: kéo trang
'\'	: dấu \
'\?'	: dấu chấm hỏi ?
'\"'	: dấu nháy đơn '
'\"'	: dấu nháy kép "
'\kkk'	: kí tự có mã là kkk trong hệ 8
'\xkk'	: kí tự có mã là kk trong hệ 16

13

Hằng xâu ký tự

- Là dãy ký tự bất kỳ đặt giữa dấu nháy kép
- Ví dụ:
 - "Dien tu Vien thong"
 - "Cong nghe thong tin"
- Chú ý:
 - 'A' là một hằng ký tự, khác với
 - "A" là một hằng xâu ký tự
 - Xâu ký tự có thể rỗng: ""

14

Tại sao cần có hằng trong chương trình?

- Chương trình dễ đọc hơn vì các con số được thay bởi các tên gọi có ý nghĩa, ví dụ: **3.1415** được thay bởi **Pi**
- Chương trình dễ sửa chữa hơn

15

Cách khai báo hằng

```
#define <tên hằng> <giá trị hằng>
hoặc
const <tên hằng>=<giá trị hằng>;
Ví dụ:
#define sosinhvien 50
#define MAX 100
const sosinhvien = 50;
```

16

Biến: Khai báo và sử dụng

Khai báo biến

- Biến là các tên gọi để lưu giá trị khi chương trình thực hiện
- Biến khác hằng ở chỗ giá trị của nó có thể thay đổi trong khi chương trình thực hiện
- Có hai cách khai báo biến:
 - Khai báo không khởi tạo
 - Khai báo có khởi tạo

17

18

Khai báo không khởi tạo

```
<tên kiểu 1> <tên biến 1>;  
<tên kiểu 2> <tên biến 2>;  
<tên kiểu 3> <tên biến 3>, <tên biến 4>;
```

Chú ý: Các biến có cùng kiểu có thể khai báo theo cách 3

19

Ví dụ về khai báo biến không khởi tạo

```
void main()  
{  
    int i, j; // khai báo 2 biến i, j có kiểu nguyên  
    float x; // khai báo biến thực x  
    char c, d[100]; // biến kí tự c, chuỗi d  
                    // chứa tối đa 100 kí tự  
    unsigned int u; // biến nguyên không dấu u  
    ...  
}
```

20

Khai báo có khởi tạo

```
<tên kiểu 1> <tên biến 1>=<giá trị 1>;  
<tên kiểu 2> <tên biến 2>=<giá trị 2>;  
<tên kiểu 3> <tên biến 3>=<giá trị 3>, <tên  
    biến 4>=<giá trị 4>;
```

Các giá trị khởi tạo có thể là hằng, biến hoặc biểu thức

21

Ví dụ về khai báo biến có khởi tạo

```
const int n = 10 ;  
void main()  
{  
    int i = 2, j, k = n + 5; // khai báo i và khởi tạo  
                            // bằng 2, k bằng 15  
    float eps = 1.0e-6 ; // khai báo biến thực  
                          // epsilon khởi tạo bằng 10-6  
    char c = 'Z'; // khai báo biến kí tự c  
                 // và khởi tạo bằng 'A'  
    char d[100] = "Tin hoc"; // khai báo chuỗi kí tự d  
                             // chứa dòng chữ "Tin hoc"  
    ...  
}
```

22

Ví dụ về tên gọi trong C++

- Tên gọi đúng: `i, i1, j, tinhoc, tin_hoc, luu_luong`
- Tên gọi sai: `1i, tin hoc, luu-luong-nuoc`
- Các tên sau đây là khác nhau: `ha_noi, Ha_noi, HA_Noi, HA_NOI, ...`

23

Phạm vi của biến

- Phạm vi của biến là nơi mà biến có tác dụng hay tại đó giá trị của biến có thể sử dụng được
- Chi tiết: sẽ nói trong các bài học sau

24

Gán giá trị cho biến

- Sử dụng phép gán để gán giá trị cho biến:
<tên biến> = <biểu thức>;
Ví dụ:
int n, i = 3; // khởi tạo i bằng 3
n = 10; // gán cho n giá trị 10
cout << n << ", " << i << endl; // in ra: 10, 3
i = n / 2; // gán lại giá trị của i bằng n/2 = 5
cout << n << ", " << i << endl; // in ra: 10, 5

25

Một số lưu ý về phép gán

- Phép gán là một phép toán và nó trả lại giá trị của <biểu thức>
- Do đó có thể thực hiện nhiều phép gán:
<biến 1>=<biến 2>=...=<biểu thức>
- Tuy nhiên không nên lạm dụng nhiều phép gán như trên dẫn đến chương trình khó đọc

26

Phép toán, biểu thức và câu lệnh

27

Phép toán

- C++ có nhiều phép toán chia thành các loại 1 ngôi, 2 ngôi và thậm chí 3 ngôi
- Các thành phần tên gọi tham gia trong phép toán gọi là hạng thức hoặc toán hạng, các kí hiệu phép toán gọi là toán tử
- Ví dụ: $a+b$: a, b là toán hạng, + là toán tử
- Số ngôi của phép toán chính là số toán hạng

28

Các phép toán số học

- Cộng (+), trừ (-), nhân (*)
- Chia (/):
 - Chia lấy phần nguyên: $5/2 = 2$
 - Chia thực: $5.0/2.0 = 2.5$
- Lấy phần dư (%)
 - $5 \% 2 = 1$
 - $4 \% 2 = 0$
- Đây là các phép toán 2 ngôi
- Phép trừ còn là 1 ngôi (khi đảo dấu)

29

Các phép toán tự tăng giảm

- $i++$, $++i$: Tăng i (biến nguyên) lên 1 đơn vị
- $i--$, $--i$: Giảm i (biến nguyên) đi 1 đơn vị
- Đây là các phép toán 1 ngôi

Phép toán	Tương đương	Kết quả
$i = ++j$; // tăng trước	$j = j + 1$; $i = j$;	$i = 16$, $j = 16$
$i = j++$; // tăng sau	$i = j$; $j = j + 1$;	$i = 15$, $j = 16$
$j = ++i + 5$;	$i = i + 1$; $j = i + 5$;	$i = 4$, $j = 9$
$j = i++ + 5$;	$j = i + 5$; $i = i + 1$;	$i = 4$, $j = 8$

30

Các phép toán so sánh và logic

- Các phép toán so sánh: Bằng nhau (==), khác nhau (!=), lớn hơn (>), lớn hơn hoặc bằng (>=), nhỏ hơn (<), nhỏ hơn hoặc bằng (<=)
- Đây là các phép toán 2 ngôi
- Các phép toán logic: Và (&&), hoặc (||), phủ định (!)

31

Các phép gán

- Gán thông thường <biến> = <biểu thức>
- Gán có điều kiện:
<biến> = <điều kiện>?<biểu thức 1>:<biểu thức 2>
<điều kiện> là một biểu thức logic, <biểu thức 1> và <biểu thức 2> là các biểu thức cùng kiểu với kiểu của <biến>
Nếu <điều kiện> đúng thì <biến> nhận giá trị của <biểu thức 1> ngược lại nhận giá trị của <biểu thức 2>

32

Ví dụ phép gán có điều kiện

$x = (3 + 4 < 7) ? 10 : 20$ // $x = 20$ vì $3+4 < 7$ là sai
 $x = (3 + 4) ? 10 : 20$ // $x = 10$ vì $3+4$ khác 0, tức điều kiện đúng
 $x = (a > b) ? a : b$ // $x =$ số lớn nhất trong 2 số a, b .

33

Cách viết gọn của phép gán

- Phép gán dạng $x = x@a$, trong đó @ là một phép toán số học, xử lý bit.. có thể được viết gọn thành: $x @ = a$
- Ví dụ:
 - $x = x + 2 \rightarrow x += 2$
 - $y = y/2 \rightarrow y /= 2$

34

Biểu thức

- Biểu thức là dãy kí hiệu kết hợp giữa các toán hạng, phép toán và cặp dấu () theo một qui tắc nhất định
- Các toán hạng có thể là hằng, biến, hàm
- Ví dụ các biểu thức
 - $(x + y)^4 - 2$
 - $(-b + \sqrt{\Delta}) / (2 \cdot a)$
 - $3 - x + \sqrt{y}$

35

Thứ tự ưu tiên của các phép toán

- C++ qui định trật tự tính toán theo các mức độ ưu tiên từ cao đến thấp như sau:
 - Các biểu thức trong cặp dấu ngoặc ()
 - Nếu có nhiều cặp ngoặc lồng nhau thì cặp trong cùng (sâu nhất) được ưu tiên cao hơn
 - Các phép toán 1 ngôi (tự tăng-giảm, lấy địa chỉ, lấy nội dung con trỏ, phủ định ...)
 - Các phép toán số học.
 - Các phép toán quan hệ, logic.
 - Các phép gán.

36

Chú ý

- Để chương trình rõ ràng, sáng sủa: Với mỗi biểu thức, nên sử dụng dấu ngoặc để chỉ định một cách tường minh trật tự tính toán trong biểu thức đó

37

Phép toán chuyển đổi kiểu

- C++ hỗ trợ chuyển đổi kiểu tự động:
char ↔ int → long int → float → double
- Chuyển đổi kiểu không tự động:
(tên_kiểu)biểu_thức // cú pháp cũ trong C
hoặc
tên_kiểu(biểu_thức) // cú pháp mới trong C++

38

Câu lệnh

- Một **câu lệnh** trong C++ được thiết lập từ các từ khoá và các biểu thức ... và luôn luôn được kết thúc bằng dấu chấm phẩy
- Ví dụ:
cin >> x >> y ;
x = 3 + x ; y = (x = sqrt(x)) + 1 ;
cout << x ;
cout << y ;

39

Khối lệnh

- Một số câu lệnh được gọi là lệnh có cấu trúc, tức bên trong nó lại chứa dãy lệnh khác.
- Dãy lệnh này phải được bao giữa cặp dấu ngoặc {} và được gọi là **khối lệnh**.
- Ví dụ:
if (a==b) {
 x *= 2; y = y-2; // Font đỏ: Khối lệnh
}

40

Thư viện các hàm toán học

- #include <math.h>
- **abs(x)**, **labs(x)**, **fabs(x)**: trả lại giá trị tuyệt đối của một số nguyên, số nguyên dài và số thực.
- **pow(x, y)**: hàm mũ, trả lại giá trị x lũy thừa y (xy).
- **exp(x)**: hàm mũ, trả lại giá trị e mũ x (ex).
- **log(x)**, **log10(x)**: trả lại lôgarit cơ số e và lôgarit thập phân của x (lnx, logx) .
- **sqrt(x)**: trả lại căn bậc 2 của x.
- **atof(s_number)**: trả lại số thực ứng với số viết dưới dạng xâu kí tự s_number.
- Hàm lượng giác: **sin(x)**, **cos(x)**, **tan(x)**

41

Các vấn đề cần nhớ

- Các kiểu dữ liệu của C++
- Hằng và biến
- Phép toán (toán tử), biểu thức, toán hạng, độ ưu tiên của các phép toán
- Ngôi của phép toán
- Câu lệnh và khối lệnh

42

Bài tập

- Làm tất cả các bài tập từ số 1 đến số 20 trong giáo trình (trang 38, 39, 40)
- Giờ thực hành: Yêu cầu sinh viên chạy các chương trình trong tuần 1 và tuần 2 trên máy tính

43