NGUYÊN LÝ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Bài 1: Giới thiệu ngôn ngữ lập trình C++

Giảng viên: TS. Lý Anh Tuấn

Email: tuanla@tlu.edu.vn

Nội dung

1. Giới thiệu C++
2. Biến, Biểu thức, Lệnh gán
3. Xuất nhập dữ liệu
4. Phong cách lập trình
5. Thư viện và Không gian tên

Giới thiệu C++

- Nguồn gốc C++
  - Ngôn ngữ bậc thấp: *Mã máy*, *Assembly*
  - Ngôn ngữ bậc cao: *C, C++, FORTRAN, COBOL*
  - Lập trình hướng đối tượng trong C++
- Thuật ngữ C++
  - Chương trình (*Program*), Hàm (*Function*), Thư viện (*Library*)
  - Xuất nhập cơ bản (IO) với *cin/cout*

Sự khác nhau giữa C và C++

- C là ngôn ngữ lập trình hướng thủ tục, trong khi C++ là ngôn ngữ lập trình hướng đối tượng
- C chỉ hỗ trợ con trỏ, trong khi C++ hỗ trợ cả con trỏ và tham chiếu
- C không có nạp chồng hàm (function overloading), trong khi C++ hỗ trợ tính năng này
- C sử dụng nhập (scanf), xuất (printf) trong khi C++ sử dụng cin và cout dễ dùng hơn
- C không có kiểu (string, bool) trong khi C++ có 2 kiểu dữ liệu này
- Đuôi mở rộng của C là .c, còn đuôi mở rộng của C++ là .cpp

Chương trình C++ mẫu (1/2)

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5    int numberOfLanguages;
6    cout << "Xin chao.\n" << "Chao mung den voi C++.\n";
7    cout << "Ban biet bao nhieu ngon ngu lap trinh? ";
8    cin >> numberOfLanguages;
9    if (numberOfLanguages < 1)
10     cout << "Ban nen hoc lop lap trinh co ban hon.\n";
11   else
12     cout << "Chuc ban hoc tot.\n";
13   return 0;
14 }
```

Chương trình C++ mẫu (2/2)

Tình huống kết quả 1:

Xin chao.
Chao mung den voi C++.
Ban biet bao nhieu ngon ngu lap trinh? 0 ← Người dùng nhập vào 0
Ban nen hoc lop lap trinh co ban hon.

Tình huống kết quả 2:

Xin chao.
Chao mung den voi C++.
Ban biet bao nhieu ngon ngu lap trinh? 1 ← Người dùng nhập vào 1
Chuc ban hoc tot.

Biến trong C++

- Định danh trong C++
  - Không trùng với từ khóa hoặc từ dành riêng
  - Phân biệt chữ hoa, chữ thường
  - Nên là tên có nghĩa
- Biến trong C++
  - Là một vùng bộ nhớ để lưu trữ dữ liệu cho một chương trình
  - Phải được khai báo trước khi sử dụng trong chương trình

Các kiểu dữ liệu (1/2)

Loại dữ liệu	Kiểu dữ liệu	Số ô nhớ	Miền giá trị
Boolean	bool	1 byte	0 hoặc 1. Trong đó 0 => FALSE và 1 => TRUE
Ký tự	char	1 byte	-128 tới 127
	unsigned char	1 byte	0 tới 255
Số nguyên	int	4 byte	-2147483648 tới 2147483647
	unsigned int	4 byte	0 tới 4294967295
	short int	2 byte	-32768 tới 32767
	long int	4 byte	-2147483648 tới 2147483647

Các kiểu dữ liệu (2/2)

Số thực	float	4 byte	+/- 3.4e +/- 38 (~7 chữ số)
	double	8 byte	+/- 1.7e +/- 308 (~15 chữ số)
	long double	8 byte	+/- 1.7e +/- 308 (~15 chữ số)

Lưu ý: Kích thước của biến có thể khác với kích thước được chỉ ra trong bảng, tùy thuộc vào trình biên dịch và máy tính bạn mà sử dụng

Gán dữ liệu cho biến

- Khởi tạo dữ liệu trong câu lệnh khai báo
  - Nếu chưa được khởi tạo, biến sẽ có giá trị "undefined"!
    - *int myVar = 0;*
- Gán dữ liệu trong khi chạy chương trình
  - Cú pháp: Vế trái = Vế phải
    - Vế trái phải là biến
    - Vế phải có thể là bất kỳ biểu thức nào
  - Ví dụ: *distance = rate * time;*
    - Vế trái là *distance*
    - Vế phải là *rate * time*

Gán dữ liệu: Ký hiệu viết tắt

Ví dụ	Tương đương với
count += 2;	count = count + 2;
total -= discount;	total = total - discount;
bonus *= 2;	bonus = bonus * 2;
time /= rushFactor;	time = time/rushFactor;
change %= 100;	change = change % 100;
amount *= cnt1 + cnt2;	amount = amount * (cnt1 + cnt2);

Những quy tắc gán dữ liệu

- Tính tương thích của phép gán
  - Khác kiểu dữ liệu:
    - Quy tắc: không gán giá trị thuộc một kiểu dữ liệu cho biến thuộc một kiểu dữ liệu khác
  - intVar = 2.99; // 2 được gán cho intVar!
    - Chỉ phần nguyên là hợp kiểu, vì thế chỉ phần này được gán cho biến intVar
    - Được gọi là *ép kiểu ngầm định* hoặc *chuyển đổi dữ liệu tự động*
  - 2, 5.75, "Z", "Hello World" là các *hằng literal*, không thể thay đổi giá trị khi chạy chương trình

Chuỗi escape (1/2)

- Cú pháp: \<ký_tự>
- Báo với trình biên dịch đây là một ký tự đặc biệt

Chuỗi escape	Ý nghĩa
\n	Xuống dòng mới
\r	Về đầu dòng
\t	Dấu Tab ngang
\a	Chuông (tiếng bíp)
\\	Dấu ngạch chéo ngược

Chuỗi escape (2/2)

\'	Dấu nháy đơn
\"	Dấu nháy kép
\v	Dấu Tab dọc
\b	Dấu Backspace (Kí tự xóa)
\f	Form Feed
\?	Dấu chấm hỏi

Ví dụ về chuỗi escape

```
1  #include <iostream>
2  using namespace std;
3  int main() {
4      cout << "Ban dang\nHoc lap trinh\n\'C++\'\n"
5      << "\"Chuc ban hoc vui ve!\"";
6      return 0;
7  }
```

Kết quả:

Ban dang
Hoc lap trinh
'C++'
"Chuc ban hoc vui ve!"

Hằng

- Đặt tên cho hằng
  - Hằng literal ít mang ngữ nghĩa
    - VD: bản thân số 24 không cho biết nó diễn đạt thông tin gì
- Việc đặt tên cung cấp cho hằng ngữ nghĩa nó muốn diễn đạt
  - VD: *const int NUMBER_OF_STUDENTS = 24*;
    - Gọi là hằng được khai báo hay hằng được đặt tên
    - Có thể sử dụng tên hằng ở bất cứ chỗ nào trong chương trình

Hằng được đặt tên

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      const double RATE = 6.9;
6      double deposit;
7      cout << "Nhap vao khoan tien ban gui (nghin dong):";
8      cin >> deposit;
9      double newBalance;
10     newBalance = deposit + deposit*(RATE/100);
11     cout << "Sau mot nam, khoan tien gui tang len thanh "
12     << newBalance << " nghin dong.";
13     return 0;
14 }
```

Tình huống kết quả:

Nhap vao khoan tien ban gui (nghin dong):500
Sau mot nam, khoan tien gui tang len thanh 534.5 nghin dong.

Độ chính xác phép toán

- Cách xác định giá trị biểu thức trong C++:
  - Có thể không như bạn mong đợi
  - Toán hạng bậc cao nhất sẽ quyết định kiểu độ chính xác phép toán được thực hiện
- VD1: 17 / 5 trả về giá trị 3
  - Cả hai toán hạng kiểu nguyên nên phép chia số nguyên được thực hiện!
- VD2: 17.0 / 5 trả về giá trị 3.4
  - Toán hạng bậc cao nhất kiểu thực nên phép chia độ chính xác số thực được thực hiện!
- VD3: int var1 = 1, var2 = 2;
  - var1/var2 = ?

Độ chính xác phép toán

- Các phép toán được thực hiện từng bước một
  - VD: 1 / 2 / 3.0 / 4 thực hiện 3 phép chia riêng rẽ
    - đầu tiên 1 / 2 = 0
    - tiếp đó 0 / 3.0 = 0.0
    - cuối cùng 0.0 / 4 = 0.0 !
- Chỉ thay đổi một toán hạng trong một biểu thức lớn là không đủ
  - Cần lưu ý đến tất cả các phép toán riêng rẽ sẽ được thực hiện

Ép kiểu

- Ép kiểu cho biến
  - Có thể thêm ".0" vào literal để ép độ chính xác phép toán, nhưng với biến?
    - Không thể sử dụng "myInt.0"!
  - static_cast<double> intVar:
  - Ép hoặc chuyển intVar sang kiểu double một cách tường minh
    - Kết quả chuyển đổi sau đó được sử dụng
    - VD: doubleVar=static_cast<double>intVar1/intVar2;

      Ép thực hiện phép chia số thực giữa hai biến nguyên

Ép kiểu

- Hai cách ép kiểu
  - Ép kiểu ngầm hoặc tự động:
    - *17/ 5.5* sẽ tự động chuyển 17 thành 17.0
  - Chuyển kiểu tường minh:
    - Người lập trình chỉ rõ việc chuyển kiểu bằng toán tử ép kiểu (double)17/ 5.5
    - Tương tự, sử dụng toán tử ép kiểu (double)myInt / myDouble để ép kiểu trên biến

Các toán tử viết tắt

- Các toán tử tăng giảm
  - Toán tử tăng, ++

    intVar++; tương đương với
    intVar = intVar + 1;
  - Toán tử giảm, --

    intVar--; tương đương với
    intVar = intVar – 1;
  - Tăng hậu tố: intVar++
    - Sử dụng giá trị hiện tại của biến, sau đó tăng biến
  - Tăng tiền tố: ++intVar
    - Trước hết tăng biến, sau đó sử dụng giá trị mới

Tăng hậu tố vs Tăng tiền tố

- VD 1: Giá trị của *valueProduced* và *n* ?

```
int  n = 2, valueProduced;
valueProduced = 2 * (n++);
cout << valueProduced << endl;
cout << n << endl;
```

- VD 2: Giá trị của *valueProduced* và *n* ?

```
int  n = 2, valueProduced;
valueProduced = 2 * (++n);
cout << valueProduced << endl;
cout << n << endl;
```

Xuất/nhập dữ liệu

- Các đối tượng I/O: *cin, cout, cerr*
- Được định nghĩa trong thư viện <iostream>
- Phải có các dòng sau đây ở đầu chương trình:
  - *#include <iostream>*
    using namespace std;
  - Báo cho C++ sử dụng thư viện phù hợp để ta có thể sử dụng các đối tượng I/O cin, cout, cerr

Xuất dữ liệu

- Tất cả các kiểu dữ liệu đều có thể hiển thị trên màn hình:
  - Biến, hằng, literal, biểu thức
- cout << numberOfGames << " games played."; hai giá trị được xuất ra:
  - giá trị của biến numberOfGames
  - chuỗi ký tự " games played."
- Cho phép xuất nhiều giá trị trong một lệnh cout

Xuống dòng khi xuất dữ liệu

- Xuống dòng mới:
  - Sử dụng chuỗi escape "\n"
    - VD: *cout << "Hello World\n";*
  - Sử dụng đối tượng *endl*
    - VD: *cout << "Hello World" << endl;*
  - Hai ví dụ cho kết quả giống nhau

Định dạng giá trị xuất ra

- Giá trị in ra có thể không như mong đợi
  cout << "The price is $" << price << endl;
  - Nếu price có giá trị 78.5, màn hình sẽ hiển thị:
    - The price is $78.500000 hoặc
    - The price is $78.5
- Quy định kích thước phần thập phân:
  - cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(2);
  - Kết quả bây giờ sẽ là:
    - The price is $78.50

Xuất ra lỗi

- Sử dụng đối tượng *cerr*
  - Làm việc tương tự như cout
  - Cung cấp cơ chế để phân biệt giữa xuất dữ liệu thông thường và xuất ra lỗi
- Chuyển hướng luồng xuất ra
  - Hầu hết các hệ thống cho phép cout và cerr được chuyển hướng sang các thiết bị khác. VD: máy in, file, …

Nhập dữ liệu bằng cin

- cin để nhập dữ liệu, cout để xuất dữ liệu
- Khác nhau:
  - Toán tử ">>" hướng theo chiều ngược lại
  - Tên là *cin* thay vì *cout*
  - Phải nhập dữ liệu cho biến
- cin >> num;
  - Dừng màn hình đợi người dùng nhập dữ liệu vào
  - Giá trị nhập vào được gán cho biến *num*

Nhắc nhập dữ liệu

- Luôn nhắc người dùng nhập dữ liệu

```
cout << "Enter number of dragons: ";
cin >> numOfDragons;
```

- Nếu không có "\n" con chạy đợi nhập dữ liệu nằm cùng dòng với lời nhắc

  Enter number of dragons: ____

- Tất cả *cin* nên có lời nhắc *cout*

Phong cách lập trình

- Mục tiêu: làm cho chương trình dễ đọc và sửa đổi
- Chú thích trong C++, 2 cách:
  - *// cau chu thich cho mot dong*
  - */* doan chu thich */*
- Quy ước đặt tên trong C++
  - Tên hằng: viết hoa hoàn toàn
    VD: NUMBER_OF_STUDENTS
  - Tên biến: viết theo cách lowerToUpper
    VD: numberStudent
  - Quan trọng là tên phải có ý nghĩa

Thư viện

- Các thư viện chuẩn của C++
- # include <tên_thư_viện>
  - Thêm nội dung của file thư viện vào chương trình của bạn
  - Được gọi là chỉ thị tiền xử lý
- C++ có nhiều thư viện:
  - Xuất/nhập dữ liệu, toán học, xâu ký tự, …

Không gian tên

- Không gian tên định nghĩa:
  - Một tập các định nghĩa tên
- Hiện tại sử dụng không gian tên *std*
  - Có tất cả các định nghĩa thư viện chuẩn cần thiết
- Ví dụ:

  ```
  #include <iostream>
  using namespace std;
  ```

  - Bao gồm toàn bộ thư viện chuẩn của các định nghĩa tên
-
  ```
  #include <iostream>
  using std::cin;
  using std::cout;
  ```

  - Có thể chỉ khai báo sử dụng các đối tượng chúng ta cần

Tóm tắt

- C++ phân biệt chữ hoa, chữ thường
- Sử dụng các tên có ý nghĩa
  - Cho biến và hằng
- Các biến phải được khai báo trước khi sử dụng
  - Và nên được khởi tạo
- Thận trọng trong các thao tác số học:
  - Độ chính xác, dấu ngoặc, thứ tự phép toán
- *#include* các thư viện chuẩn của C++ khi cần
- cin – nhập, cout – xuất, cerr – xuất thông điệp lỗi
- Sử dụng chú thích để giúp chương trình dễ hiểu