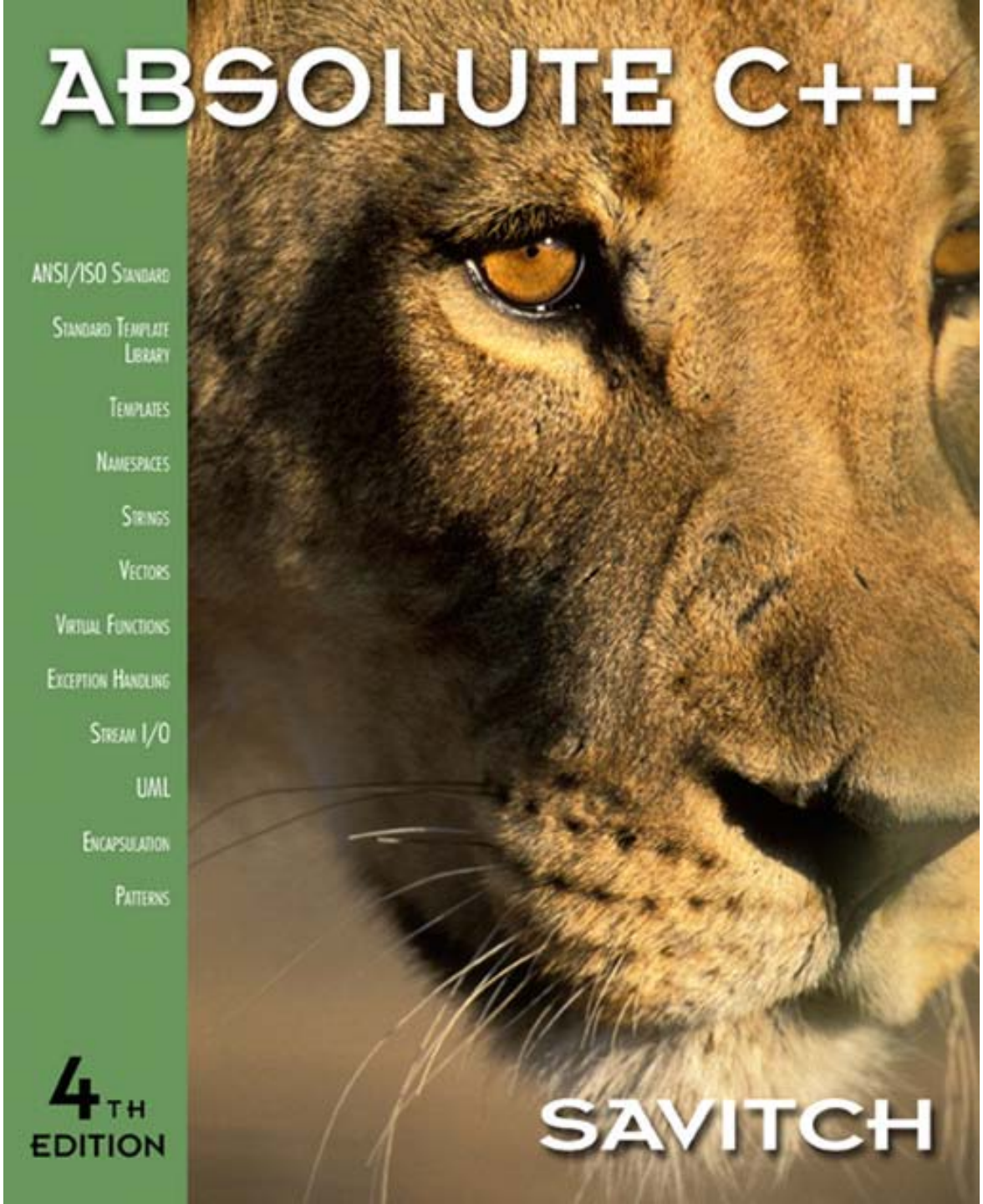
Bài 1: Căn bản về C++

Giảng viên: Hoàng Thị Điệp
Khoa Công nghệ Thông tin – ĐH Công Nghệ



Chapter 1

C++ Basics

Copyright © 2010 Pearson Addison-Wesley.
All rights reserved

PEARSON Addison Wesley

Mục tiêu bài học

- Giới thiệu C++
  - Nguồn gốc, Lập trình hướng đối tượng, Thuật ngữ
- Biến, Biểu thức và Câu lệnh gán
- Đọc ghi trên thiết bị vào/ra chuẩn
- Phong cách lập trình
- Thư viện và Không gian tên (namespace)

Giới thiệu C++

- Nguồn gốc
  - Ngôn ngữ bậc thấp
    - Ngôn ngữ máy, hợp ngữ
  - Ngôn ngữ bậc cao
    - C, C++, ADA, COBOL, FORTRAN
  - Lập trình hướng đối tượng trong C++
- Thuật ngữ C++
  - *Program* và *function*
  - Đọc/ghi cơ bản với cin và cout

Display 1.1
Một chương trình C++ mẫu (1/2)

Display 1.1 A Sample C++ Program

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

3   int main( )
4   {
5       int numberOfLanguages;

6       cout << "Hello reader.\n"
7            << "Welcome to C++.\n";

8       cout << "How many programming languages have you used? ";
9       cin >> numberOfLanguages;

10      if (numberOfLanguages < 1)
11          cout << "Read the preface. You may prefer\n"
12               << "a more elementary book by the same author.\n";
13      else
14          cout << "Enjoy the book.\n";

15      return 0;
16  }
```

Display 1.1
Một chương trình C++ mẫu (2/2)

SAMPLE DIALOGUE 1

```
Hello reader.
Welcome to C++.
How many programming languages have you used? 0
Read the preface. You may prefer
a more elementary book by the same author.
```

User types in `0` on the keyboard.

SAMPLE DIALOGUE 2

```
Hello reader.
Welcome to C++.
How many programming languages have you used? 1
Enjoy the book
```

User types in `1` on the keyboard.

Biến

- Định danh trong C++
  - Phân biệt khái niệm từ khóa và định danh
  - Định danh phân biệt viết hoa viết thường và có quy tắc
  - Hãy đặt tên có nghĩa!
- Biến
  - Là nơi trong bộ nhớ để lưu dữ liệu cho chương trình
  - Tất cả dữ liệu cần được khai báo trước khi sử dụng trong chương trình

Các kiểu dữ liệu:
Display 1.2 Các kiểu dữ liệu đơn giản (1/2)

Display 1.2 **Simple Types**

TYPE NAME	MEMORY USED	SIZE RANGE	PRECISION
`short` (also called `short int`)	2 bytes	−32,768 to 32,767	Not applicable
`int`	4 bytes	−2,147,483,648 to 2,147,483,647	Not applicable
`long` (also called `long int`)	4 bytes	−2,147,483,648 to 2,147,483,647	Not applicable
`float`	4 bytes	approximately 10^{-38} to 10^{38}	7 digits
`double`	8 bytes	approximately 10^{-308} to 10^{308}	15 digits

Các kiểu dữ liệu:
Display 1.2 Các kiểu dữ liệu đơn giản (2/2)

`long double`	10 bytes	approximately 10^{-4932} to 10^{4932}	19 digits
`char`	1 byte	All ASCII characters (Can also be used as an integer type, although we do not recommend doing so.)	Not applicable
`bool`	1 byte	`true`, `false`	Not applicable

The values listed here are only sample values to give you a general idea of how the types differ. The values for any of these entries may be different on your system. *Precision* refers to the number of meaningful digits, including digits in front of the decimal point. The ranges for the types `float`, `double`, and `long double` are the ranges for positive numbers. Negative numbers have a similar range, but with a negative sign in front of each number.

Gán giá trị cho biến

- Dùng câu lệnh khai báo để khởi tạo giá trị cho biến
  - Nếu không khởi tạo, kết quả sẽ là “không xác định”!
    - int myValue = 0;
- Gán giá trị cho biến khi đang thực thi
  - Lvalue (vế trái) & Rvalue (vế phải)
    - Lvalue phải là biến
    - Rvalue có thể là biểu thức bất kì
    - Ví dụ:
      distance = rate * time;
      Lvalue: "distance"
      Rvalue: "rate * time"

Phép gán: Kí hiệu tắt

EXAMPLE	EQUIVALENT TO
`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);`

Các quy tắc gán

- Dữ liệu gán phải tương thích
  - Lệch kiểu
    - Quy tắc chung: Không thể gán giá trị kiểu này cho biến kiểu khác
  - intVar = 2.99; // 2 sẽ được gán cho intVar!
    - Chỉ có phần nguyên là “vừa” nên ta chỉ lấy được phần này cho biến
    - Đây là “chuyển kiểu tự động” hay “không tường minh"
  - Giá trị hằng
    - 2, 5.75, "Z", "Hello World"
    - Coi là hằng số vì chúng không thay đổi trong suốt chương trình

Dữ liệu hằng

- Giá trị hằng
  - Ví dụ:
    - 2 // Hằng int
    - 5.75 // Hằng double
    - "Z" // Hằng char
    - "Hello World" // Hằng string
- Không thể thay đổi các giá trị này trong suốt quá trình thực hiện chương trình
- Called "literals" because you "literally typed" them in your program!

Các xâu escape

- “Mở rộng” tập kí tự
- Gồm dấu xược ngược (\) đứng trước một kí tự
  - Báo cho trình biên dịch chuẩn bị làm việc với một kí tự escape đặc biệt
  - Display 1.3 trong slide sau liệt kê các xâu escape

Display 1.3
Một số xâu escape (1/2)

Display 1.3 Some Escape Sequences

SEQUENCE	MEANING
\n	New line
\r	Carriage return (Positions the cursor at the start of the current line. You are not likely to use this very much.)
\t	(Horizontal) Tab (Advances the cursor to the next tab stop.)
\a	Alert (Sounds the alert noise, typically a bell.)
\\	Backslash (Allows you to place a backslash in a quoted expression.)

Display 1.3
Một số xâu escape (2/2)

\'	Single quote (Mostly used to place a single quote inside single quotes.)
\"	Double quote (Mostly used to place a double quote inside a quoted string.)
The following are not as commonly used, but we include them for completeness:	
\v	Vertical tab
\b	Backspace
\f	Form feed
\?	Question mark

Hằng

- Hãy đặt tên hằng số của bạn
  - Dùng giá trị hằng cũng tạm được nhưng đặt tên hằng sẽ cung cấp một chút ý nghĩa
    - ví dụ: khi bạn thấy số 24 trong 1 chương trình bạn sẽ không hiểu được nó biểu diễn gì
- Hãy sử dụng các hằng đặt tên
  - Đặt tên ý nghĩa để biểu diễn dữ liệu
    const int NUMBER_OF_STUDENTS = 24;
    - Gọi là một "hằng đã khai báo" hoặc "hằng có tên"
    - Sau đó hãy dùng tên hằng ở bất cứ chỗ nào bạn cần tới
    - Giá trị gia tăng: chỉ cần sửa đổi giá trị của hằng ở 1 chỗ

Các phép toán số học:
Display 1.4 Hằng có tên (1/2)

- Các phép toán số học chuẩn
  - Luật ưu tiên – luật chuẩn

Display 1.4 Named Constant

```
1  #include <iostream>
2  using namespace std;
3
4  int main( )
5  {
6      const double RATE = 6.9;
7      double deposit;

8      cout << "Enter the amount of your deposit $";
9      cin >> deposit;
```

Các phép toán số học:
Display 1.4 Hằng có tên (2/2)

```
10        double newBalance;
11        newBalance = deposit + deposit*(RATE/100);
12        cout << "In one year, that deposit will grow to\n"
13             << "$" << newBalance << " an amount worth waiting for.\n";

14        return 0;
15    }
```

SAMPLE DIALOGUE

Enter the amount of your deposit $**100**
In one year, that deposit will grow to
$106.9 an amount worth waiting for.

Độ chính xác số học

- Độ chính xác của các phép tính
  - Rất quan trọng!
    - C++ có thể tính giá trị biểu thức khác với mong đợi của bạn!
  - “Toán hạng bậc cao nhất" xác định kiểu của “độ chính xác” số học sẽ được thực hiện
  - Lỗi hay gặp!

Ví dụ độ chính xác số học

- Ví dụ:
  - 17 / 5 có giá trị bằng 3 trong C++
    - Cả 2 toán hạng đều là số nguyên
    - Phép chia số nguyên được thực hiện
  - 17.0 / 5 có giá trị bằng 3.4 trong C++
    - Toán hạng bậc cao nhất có kiểu double
    - Phép chia với độ chính xác double được thực hiện
  - int intVar1 =1, intVar2=2;
    intVar1 / intVar2;
    - Thực hiện phép chia số nguyên
    - Kết quả: 0

Độ chính xác số học riêng lẻ

- Từng phép tính một được thực hiện
  - 1 / 2 / 3.0 / 4 thực hiện 3 phép chia riêng rẽ.
    - Đầu tiên → 1 / 2 bằng 0
    - Sau đó → 0 / 3.0 bằng 0.0
    - Sau đó → 0.0 / 4 bằng 0.0
- Do đó chỉ biến đổi một toán hạng trong biểu thức lớn là không đủ
  - Bạn cần nhớ là từng phép tính sẽ được thực hiện khi tính giá trị biểu thức.

Chuyển đổi kiểu

- Đổi kiểu cho biến
  - Với giá trị hằng, có thể bổ sung ".0“ để ép độ chính xác số học. Nhưng với biến thì sao?
    - Ta không thể viết "myInt.0"
  - static_cast<double>intVar
  - Chuyển kiểu tường minh cho intVar thành kiểu double
    - Sau đó kết quả của phép chuyển đổi sẽ được sử dụng
    - Biểu thức ví dụ:
      doubleVar = static_cast<double>intVar1 / intVar2;
      - Phép chuyển đổi kiểu ép thực hiện phép chia double cho 2 biến nguyên.

Chuyển đổi kiểu (2)

- Hai loại
  - Không tường minh – còn gọi là “tự động”
    - Chương trình tự động làm việc này cho bạn
      17 / 5.5
      Biểu thức này dẫn tới phép chuyển kiểu không tường minh, chuyển 17 → 17.0
  - Tường minh
    - Người lập trình xác định phép chuyển đổi sử dụng toán tử chuyển đổi

    (double)17 / 5.5

    Cũng giống biểu thức trên, dùng chuyển kiểu tường minh

    (double) myInt / myDouble

    Cách dùng phổ biến hơn, chuyển đổi trên biến

Các toán tử viết tắt

- Toán tử tự tăng và tự giảm
  - Chỉ là kí hiệu viết tắt
  - Toán tử tự tăng, ++
    intVar++; tương đương với
    intVar = intVar + 1;
  - Toán tử tự giảm, --
    intVar--; tương đương với
    intVar = intVar – 1;

Các toán tử viết tắt: Hai lựa chọn

- Tăng sau
  intVar++
  - Sử dụng giá trị hiện thời, SAU ĐÓ mới tăng nó
- Tăng trước
  ++intVar
  - Tăng giá trị của biến trước, SAU ĐÓ sử dụng giá trị mới
- “Sử dụng" có nghĩa là bất cứ ngữ cảnh hiện thời nào của biến
- Không khác nhau nếu câu lệnh chỉ có phép tự tăng:
  intVar++; và ++intVar; → kết quả giống hệt nhau

Ví dụ tăng sau

- Phép tăng sau trong biểu thức:

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

  - Đoạn mã này cho output:
    4
    3
  - vì nó dùng phép tăng sau

Ví dụ tăng trước

- Giờ ta sử dụng phép tăng trước:

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

  - Đoạn mã này cho output:
    6
    3
  - vì nó dùng phép tăng trước

Đọc/ghi chuẩn

- Các đối tượng đọc/ghi cin, cout, cerr
- Định nghĩa trong thư viện <iostream> của C++
- Bạn phải viết những dòng sau (gọi là chỉ thị tiền xử lý) ở gần đầu file:
  - #include <iostream>
    using namespace std;
  - Báo cho C++ dùng những thư viện thích hợp để ta có thể sử dụng các đối tượng cin, cout, cerr trong chương trình

Ghi chuẩn

- Ta có thể output những gì?
  - Có thể output bất cứ dữ liệu nào ra màn hình
    - Biến
    - Hằng
    - Giá trị hằng
    - Biểu thức (bao gồm tất cả)
  - cout << numberOfGames << " games played.";
    2 giá trị được output:
    giá trị của biến numberOfGames,
    giá trị hằng " games played."
- Cascading: nhiều giá trị trong một cout

Tách output thành nhiều dòng

- Kí hiệu xuống dòng trong output
  - Nhắc lại: "\n" là chuỗi escape cho kí tự xuống dòng
- Cách thứ 2: đối tượng endl
- Ví dụ:

```
cout << "Hello World\n";
```

  - Gửi xâu "Hello World" ra màn hình và xâu escape nhảy sang dòng tiếp theo

```
cout << "Hello World" << endl;
```

  - Kết quả giống trên

Định dạng output

- Định dạng giá trị số cho output
  - Giá trị hiển thị có thể không như bạn mong đợi.
    cout << "The price is $" << price << endl;
    - Nếu biến price (với kiểu khai báo là double) có giá trị 78.5, bạn có thể nhận output:
      - The price is $78.500000 hay:
      - The price is $78.5
- Bạn cần báo tường minh cho C++ cách kết xuất các con số trong chương trình!

Định dạng các con số

- “Công thức màu nhiệm" để ép cỡ thập phân:
  cout.setf(ios::fixed);
  cout.setf(ios::showpoint);
  cout.precision(2);
- Những câu lệnh này ép tất cả các giá trị sẽ được cout phía sau:
  - phải có chính xác 2 chữ số sau dấu phẩy
  - Ví dụ:
    cout << "The price is $" << price << endl;
    - Giờ sẽ cho kết quả là:
      The price is $78.50
- Bạn cũng có thể chỉnh sửa độ chính xác mỗi khi cần

Ghi lỗi

- Bạn output lỗi với cerr
  - cerr làm việc giống như cout
  - Cung cấp cơ chế phân biệt ghi thông thường với ghi lỗi
- Chỉnh hướng luồng ghi
  - Hầu hết các hệ thống cho phép cout và cerr được “chỉnh hướng" sang thiết bị khác
    - ví dụ: máy in, ghi ra file, trình ghi lỗi, ...

Đọc dữ liệu bằng cin

- cin để đọc vào, cout để ghi ra
- Sự khác biệt:
  - ">>" (phép toán trích ra) hướng ngược lại
    - Hãy nghĩ nó là hướng dữ liệu sẽ đi
  - Đối tượng tên là "cin" được dùng thay cho "cout"
  - Giá trị hằng không được phép dùng với cin
    - Bạn phải đọc vào một biến
- cin >> num;
  - Màn hình đợi nhập vào bàn phím
  - Giá trị nhập từ bàn phím được gán cho num

Nhắc nhập dữ liệu: cin và cout

- Bạn nên “nhắc” người dùng nhập dữ liệu
  cout << "Enter number of dragons: ";
  cin >> numOfDragons;
  - Chú ý là không cần "\n" trong cout. Dấu nhắc sẽ “đợi” trên cùng dòng lời nhắc:

    Enter number of dragons: ____

    - Dấu gạch dưới báo hiệu nơi dữ liệu nhập từ bàn phím sẽ được ghi ra
- Mọi cin nên có một lời nhắc bằng cout
  - Cần tối đa tính thân thiện của input/output

Phong cách lập trình

- Yêu cầu tối thiểu: Chương trình dễ đọc và chỉnh sửa
- Có 2 phương pháp chú thích:
  - // Hai dấu xược báo hiệu cả dòng đó sẽ bị bỏ qua
  - /*Cặp xược sao báo hiệu mọi thứ nằm giữa chúng sẽ bị bỏ qua */
  - Cả 2 đều được dùng thường xuyên
- Đặt tên
  - VIẾT_HOA cho hằng
  - viếtThườngRồiViếtHoaChữCáiĐầuTiếng cho biến
  - Quan trọng nhất: TÊN PHẢI CÓ NGHĨA!

Thư viện

- Các thư viện chuẩn của C++
- #include <Tên_Thư_Viện>
  - Chỉ dẫn "thêm" nội dung của file thư viện vào chương trình của bạn
  - Gọi là "chỉ thị tiền xử lý"
    - Thực hiện trước trình biên dịch, chỉ đơn giản copy file thư viện vào file chương trình của bạn
- C++ có rất nhiều thư viện
  - Input/output, toán, xâu, ...

Không gian tên

- Các không gian tên đã định nghĩa:
  - Là tập các định nghĩa tên
- Tới giờ ta chỉ quan tâm tới không gian tên "std"
  - Có tất cả các định nghĩa trong thư viện chuẩn ta cần
- Ví dụ:

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

  - Bao gồm tất cả các định nghĩa tên trong thư viện chuẩn

```
#include <iostream>
using std::cin;
using std::cout;
```

  - Ta có thể chỉ đinh những đối tượng mà ta cần

Tóm tắt 1

- Định danh trong C++ phân biệt viết hoa viết thường
- Dùng các tên có nghĩa
  - cho biến và hằng
- Các biến phải được khai báo trước khi sử dụng
  - Nên khởi tạo biến
- Cẩn thận khi tính toán dữ liệu số
  - Độ chính xác, dấu ngoặc, thứ tự ưu tiên các phép toán
- #include các thư viện C++ khi cần

Tóm tắt 2

- Đối tượng cout
  - ghi ra thiết bị ra chuẩn
- Đối tượng cin
  - đọc từ thiết bị vào chuẩn
- Đối tượng cerr
  - ghi thông điệp báo lỗi
- Dùng chú thích để làm chương trình dễ hiểu hơn
  - không nên chú thích quá nhiều

Chuẩn bị bài tới

- Đọc chương 1, 2 giáo trình.