Ngôn ngữ Lập trình C++

Chương I - Giới thiệu ngôn ngữ C++

Nội dung chính

- Mã máy, Hợp ngữ, và ngôn ngữ bậc cao
- Một số ngôn ngữ lập trình bậc cao
- Lịch sử C và C++
- Hệ thống và môi trường lập trình C++
- Giới thiệu về C++
  - ví dụ về chương trình C++ đơn giản
  - khái niệm biến
  - vào ra dữ liệu
  - các phép toán số học
  - ra quyết định - các phép toán quan hệ

1.1 Mã máy, Hợp ngữ, và Ngôn ngữ bậc cao

1. Mã máy (machine language)

- Là ngôn ngữ duy nhất máy tính trực tiếp hiểu được, là "ngôn ngữ tự nhiên" của máy tính
- Được định nghĩa bởi thiết kế phần cứng, phụ thuộc phần cứng
- Gồm các chuỗi số, => chuỗi các số 0 và 1
- Dùng để lệnh cho máy tính thực hiện các thao tác cơ bản, mỗi lần một thao tác
- Nặng nề, khó đọc đối với con người
- Ví dụ:

```
+1300042774
+1400593419
+1200274027
```

1.1 Mã máy, Hợp ngữ, và Ngôn ngữ bậc cao

2. Hợp ngữ (assembly)

- Những từ viết tắt kiểu tiếng Anh, đại diện cho các thao tác cơ bản của máy tính
- Dễ hiểu hơn đối với con người
- Máy tính không hiểu
  - Cần đến các chương trình dịch hợp ngữ (assembler) để chuyển từ hợp ngữ sang mã máy
- Ví dụ:

```
LOAD   BASEPAY
ADD    OVERPAY
STORE  GROSSPAY
```

1.1 Mã máy, Hợp ngữ, và Ngôn ngữ bậc cao

3. Các ngôn ngữ bậc cao (high-level languages)
   - Tương tự với tiếng Anh, sử dụng các ký hiện toán học thông dụng
   - Một lệnh thực hiện được một công việc mà hợp ngữ cần nhiều lệnh để thực hiện được.
   - Ví dụ:

```
grossPay = basePay + overTimePay
```

   - Các chương trình dịch (compiler) để chuyển sang mã máy
   - Các chương trình thông dịch (interpreter program) trực tiếp chạy các chương trình viết bằng ngôn ngữ bậc cao.
     - Chậm hơn
     - Thuận tiện khi đang phát triển chương trình

1.2 Một số ngôn ngữ lập trình bậc cao

- **FORTRAN**
  - FORmula TRANslator (1954-1957: IBM)
  - Tính toán toán học phức tạp, thường dùng trong các ứng dụng khoa học và kỹ thuật
- **COBOL**
  - COmmon Business Oriented Language (1959)
  - Thao tác chính xác và hiệu quả đối với các khối lượng dữ liệu lớn,
    - Các ứng dụng thương mại
- **Pascal**
  - Tác giả: Niklaus Wirth
  - Dùng trong trường học.
- **Java**
  - Tác giả: Sun Microsystems (1991)
  - Ngôn ngữ điều khiển theo sự kiện (event-driven), hoàn toàn hướng đối tượng, tính khả chuyển (portable) rất cao.
  - Các trang Web với nội dung tương tác động
  - Phát triển các ứng dụng quy mô lớn

1.2 Một số ngôn ngữ lập trình bậc cao

- BASIC
  - Beginner's All-Purpose Symbolic Instruction Code
  - Từ giữa những năm1960
- Visual Basic
  - GUI, xử lý sự kiện (event handling), sử dụng Win32 API, lập trình hướng đối tượng (object-oriented programming), bắt lỗi (error handling)
- Visual C++
  - C++ của Microsoft và mở rộng
    - Thư viện của Microsoft (Microsoft Foundation Classes -MFC)
    - Thư viện chung
      - GUI, đồ họa, lập trình mạng, đa luồng (multithreading), …
      - Dùng chung giữa Visual Basic, Visual C++, C#
- C#
  - Bắt nguồn từ C, C++ và Java
  - Ngôn ngữ điều khiển theo sự kiện (event-driven), hoàn toàn hướng đối tượng, ngôn ngữ lập trình trực quan (visual programming language)

1.3 Lịch sử ngôn ngữ C và C++

- C
  - Dennis Ritchie (Bell Laboratories)
  - Là ngôn ngữ phát triển của hệ điều hành UNIX
  - Độc lập phần cứng => có thể viết các chương trình khả chuyển
  - Chuẩn hóa năm 1990 – ANSI C
  - Kernighan & Ritchie "The C Programming Language", 2nd, 1988
- C++
  - Là mở rộng của C
  - Đầu những năm 1980: Bjarne Stroustrup (phòng thí nghiệm Bell)
  - Cung cấp khả năng lập trình hướng đối tượng.
  - Ngôn ngữ lai
    - Lập trình cấu trúc kiểu C
    - Lập trình hướng đối tượng
    - Cả hai
- Có cần biết C trước khi học C++?

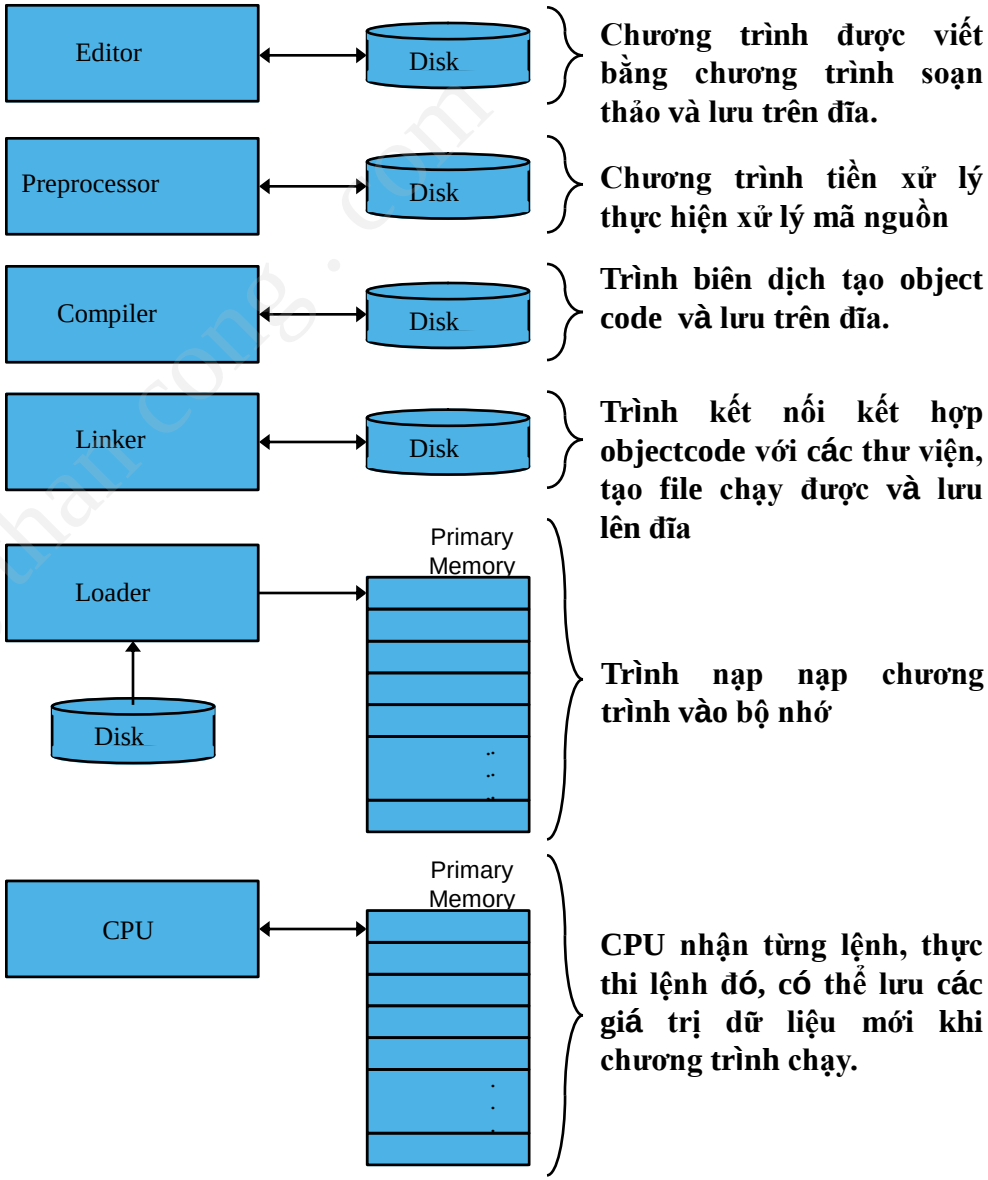
1.4 Hệ thống C++

- Môi trường phát triển chương trình (Program-development environment)
- Ngôn ngữ
- Thư viện chuẩn (C++ Standard Library)

1.4 Môi trường cơ bản cho lập trình C++

Các giai đoạn của chương trình C++:

1. Soạn thảo - Edit
2. Tiền xử lý - Preprocess
3. Biên dịch - Compile
4. Liên kết - Link
5. Nạp - Load
6. Chạy - Execute



1.4 Môi trường cơ bản cho lập trình C++

- Soạn thảo
  - File có kiểu mở rộng *.cpp, *cxx, *.cc, *.C
  - Unix/Linux: vi, emacs
  - MS.Windows: các môi trường soạn thảo tích hợp: Dev-cpp, Microssoft Visual C++, Borland C++ Builder, …
    - Chú ý mức độ hỗ trợ C++ chuẩn – ANSI/ISO C++

Ví dụ 1: Hello World!

```
1   /* A first program in C++.
2        Print a line of text to standard output */
3   #include <iostream>
4
5   // function main begins program execution
6   int main()
7   {
8      std::cout << "Hello World!\n";
9
10     return 0;   // indicate that program ended successfully
11
12  } // end function
```

```
Hello World!
```

Chú thích

hàm **main** trả về một giá trị kiểu số nguyên.

Định hướng tiền xử lý (preprocessor directive) để khai báo sử dụng thư viện ra/vào chuẩn **<iostream>**.

Viết một dòng ra output chuẩn (màn hình)

hàm **main** xuất hiện đúng một lần trong mỗi chương trình C++.

Ngoặc trái **{** bắt đầu thân hàm.

Các lệnh kết thúc bằng dấu chấm phảy **;**

Tương ứng, ngoặc phải **}** kết thúc thân hàm.

Từ khóa **return** là một cách thoát khỏi hàm; giá trị **0** được trả về có nghĩa chương trình kết thúc thành công.

1.5 Các thành phần cơ bản
Chú thích và định hướng tiền xử lý

- Chú thích - comment

```
// A first program in C++.
```

  - Làm tài liệu cho các chương trình
  - Làm chương trình dễ đọc dễ hiểu hơn
  - được trình biên dịch (compiler) bỏ qua
  - 1 dòng chú thích bắt đầu với **//**

- Các định hướng tiền xử lý - directive

```
#include <iostream>
```

  - Được xử lý ngay trước khi biên dịch
  - Bắt đầu bằng **#**

Ví dụ 1 - mở rộng 1

fig01_04.cpp

```
1  // Fig. 1.4: fig01_04.cpp
2  // Printing a line with multiple statements.
3  #include <iostream>
4
5  // function main begins program execution
6  int main()
7  {
8     std::cout << "Welcome ";
9     std::cout << "to C++!\n";
10
11    return 0;   // indicate that program ended successfully
12
13 } // end function main
```

Nhiều dòng lệnh tạo output trên một dòng.

```
Welcome to C++!
```

Ví dụ 1 - mở rộng 2

```
1   // Fig. 1.5: fig01_05.cpp
2   // Printing multiple lines with a single statement
3   #include <iostream>
4
5   // function main begins program execution
6   int main()
7   {
8      std::cout << "Welcome\nto\n\nC++!\n";
9
10     return 0;   // indicate that program ended successfully
11
12  } // end function main
```

Dùng ký tự dòng mới **\n** để in trên nhiều dòng.

```
Welcome
to

C++!
```

Ví dụ 2:
Chương trình tính tổng hai số nguyên

```
1  // Fig. 1.6: fig01_06.cpp
2  // Addition program.
3  #include <iostream>
4
5  // function main begins program execution
6  int main()
7  {
8     int integer1;  // first number to be input by user
9     int integer2;  // second number
10    int sum;       // variable in wh
11
12    std::cout << "Enter first integer\n";  // prompt
13    std::cin >> integer1;                  // read an int
14
15    std::cout << "Enter second integer\n"; // prompt
16    std::cin >> integer2;
17
18    sum = integer1 + integer2;
19
20    std::cout << "Sum is " << sum << endl; // print sum
21
22    return 0;   // indicate that program ended successfully
23
24 } // end function main
```

```
Enter first integer
45
Enter second integer
72
Sum is 117
```

Khai báo các biến nguyên.

Nhập một số nguyên từ input chuẩn, ghi vào biến integer1

endl cho kết quả là một dòng trống.

Tính toán có thể được thực hiện trong lệnh output: Thay cho các dòng 18 và 20:

```
cout << "Sum is " << integer1 + integer2 << endl;
```

1.5 Các thành phần cơ bản
Biến chương trình

- Biến - variable: Một nơi trong bộ nhớ, có thể lưu các giá trị thuộc một kiểu nào đó.
- Các kiểu dữ liệu cơ bản
  - **`int`** - số nguyên
  - **`char`** – ký tự
  - **`double`** - số chấm động
  - **`bool`** – các giá trị logic true hoặc false
- Các biến phải được khai báo tên và kiểu trước khi sử dụng

```
int integer1;
int integer2;
int sum;
```

- Có thể khai báo nhiều biến thuộc cùng một kiểu dữ liệu trong một dòng khai báo biến.

```
int integer1, integer2, sum;
```

1.5 Biến chương trình

- Quy tắc đặt tên biến
  - Chuỗi ký tự (chữ cái a..z, A..Z, chữ số 0..9, dấu gạch dưới _)
  - Không được bắt đầu bằng chữ số
  - Phân biệt chữ hoa chữ thường.

  Ví dụ:

  Tên biến hợp lệ: h678h_m2, _adh2, taxPayment…

  Không hợp lệ: áadàn, so chia, 2n, …

1.5 Biến chương trình

- Các khái niệm về bộ nhớ (memory)
  - Mỗi biến tương ứng với một khu trong bộ nhớ máy tính
  - Mỗi biến có tên, kiểu, kích thước, và giá trị
  - Khi biến được gán một giá trị mới, giá trị cũ bị ghi đè
  - Đọc giá trị của các biến trong bộ nhớ không làm thay đổi các biến trong bộ nhớ.

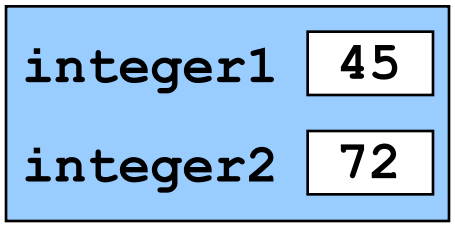
1.5 Biến chương trình

```
std::cin >> integer1;
```

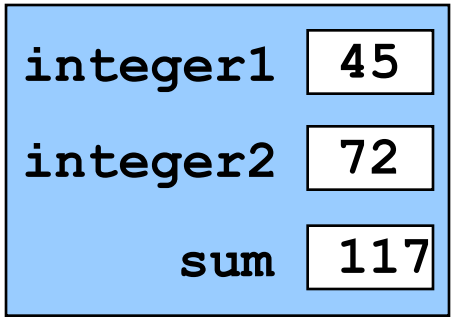
– giả sử người dùng nhập 45


```
std::cin >> integer2;
```

– giả sử người dùng nhập 72




```
sum = integer1 + integer2;
```



1.6 Vào ra dữ liệu

Các đối tượng vào/ra cơ bản

- **`cin`**
  - dòng dữ liệu vào chuẩn - Standard input stream
  - thường là từ bàn phím
- **`cout`**
  - dòng dữ liệu ra chuẩn - Standard output stream
  - thường là màn hình máy tính
- **`cerr`**
  - dòng báo lỗi chuẩn - Standard error stream
  - hiện các thông báo lỗi

1.6 Vào ra dữ liệu

In dòng văn bản ra màn hình

```
std::cout << "Enter first integer\n";   // prompt
```

- Đối tượng ra chuẩn - Standard output stream object
  - **`std::cout`**
  - "nối" với màn hình
  - **`<<`**
    - toán tử chèn vào dòng dữ liệu ra – stream insert operator
    - giá trị bên phải (right operand) được chèn vào dòng dữ liệu ra
- Không gian tên - Namespace
  - **`std::`** có nghĩa là sử dụng tên thuộc "namespace" **`std`**
  - **`std::`** được bỏ qua nếu dùng các khai báo **`using`**
- Escape characters **`\`**
  - đánh dấu các ký tự đặc biệt
    - ví dụ \\, \', \n, \t

1.6 Vào ra dữ liệu

Các chuỗi escape

Chuôi Escape	Mô tả
\n	Dòng mới. Đ àn đ òng
\t	Tab. Di chuyển con trỏ đ
\r	Về đ òng đ òng
\a	Chuông. Bật chuông hệ thống.
****	Chéo ngư
\"	Nháy kép. Dùng đ

1.6 Vào ra dữ liệu

Nhập dữ liệu từ thiết bị vào chuẩn

```
std::cin >> integer1;  // read an integer
```

- Đối tượng dòng dữ liệu vào - Input stream object
  - **>>** (toán tử đọc từ dòng dữ liệu vào)
    - được sử dụng với **`std::cin`**
    - đợi người dùng nhập giá trị, rồi gõ phím *Enter* (Return)
    - lưu giá trị vào biến ở bên phải toán tử
      - đổi giá trị được nhập sang kiểu dữ liệu của biến
- **=** (toán tử gán)
  - gán giá trị cho biến
  - toán tử hai ngôi - Binary operator
  - Ví dụ:

```
sum = variable1 + variable2;
```

1.7 Tính toán số học

- Các phép toán số học
  - `*` Phép nhân
  - `/` Phép chia
    - Phép chia với số nguyên lấy thương là số nguyên và bỏ phần dư
      - `7 / 5` cho kết quả `1`
    - Phép chia với số thực cho kết quả là số thực
      - `7.0 / 5.0` cho kết quả `1.4`
  - `%` Phép lấy số dư
    - `7 % 5` cho kết quả `2`

1.7 Tính toán số học

- Các quy tắc ưu tiên - Rules of operator precedence
  - Các phép toán trong ngoặc được tính trước
    - ngoặc lồng nhau
      - các phép toán ở bên trong nhất được tính trước nhất
  - tiếp theo là các phép nhân, chia, và phép lấy số dư
    - các phép toán được tính từ trái sang phải
  - cộng và trừ được tính cuối cùng
    - các phép toán được tính từ trái sang phải

```
1  // Fig. 1.14: fig01_14.cpp
2  // Using if statements, relational
3  // operators, and equality operators.
4  #include <iostream>
5
6  using std::cout;  // program uses cout
7  using std::cin;   // program uses cin
8  using std::endl;  // program uses endl
9
10 // function main begins program execution
11 int main()
12 {
13    int num1;  // first number to
14    int num2;  // second number t
15
16    cout << "Enter two integers, and I will tell you\n"
17         << "the relationships they satisfy: ";
18    cin >> num1 >> num2;  // read two integers
19
20    if ( num1 == num2 )
21       cout << num1 << " is equal to " << num2 << endl;
22
23    if ( num1 != num2 )
24       cout << num1 << " is not equal to " << num2 << endl;
25
```

fig01_14.cpp
(1 of 2)

khai báo **using** để sau đó không cần dùng tiền tố **std::**

Khai báo biến.

Có thể viết **cout** và **cin** mà không cần tiền tố **std::**

lệnh **if** kiểm tra xem các giá trị của **num1** và **num2** có bằng nhau không.

Nếu điều kiện là đúng (nghĩa là hai giá trị bằng nhau) thì thực hiện lệnh này.

lệnh **if** kiểm tra xem các giá trị của **num1** và **num2** có khác nhau không.

Nếu điều kiện là đúng (nghĩa là hai giá trị khác nhau) thì thực hiện lệnh này.

```
26     if ( num1 < num2 )
27        cout << num1 << " is less than " << num2 << endl;
28
29     if ( num1 > num2 )
30        cout << num1 << " is greater than " << num2 << end
31
32     if ( num1 <= num2 )
33        cout << num1 << " is less than or equal to "
34             << num2 << endl;
35
36     if ( num1 >= num2 )
37        cout << num1 << " is greater than or equal to "
38             << num2 << endl;
39
40     return 0;   // indicate that program ended successfully
41
42  } // end function main
```

Một lệnh có thể được tách thành nhiều dòng.

fig01_14.cpp
output (1 of 2)

```
Enter two integers, and I will tell you
the relationships they satisfy: 22 12
22 is not equal to 12
22 is greater than 12
22 is greater than or equal to 12
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 7 7
7 is equal to 7
7 is less than or equal to 7
7 is greater than or equal to 7
```

fig01_14.cpp
output (2 of 2)

1.8 Ra quyết định: Các phép toán quan hệ

Ký hiệu toán học	Toán tử của C++	Ví dụ điều kiện C++	Ý nghĩa của điều kiện
>	`>`	`x > y`	**x** lớn hơn **y**
<	`<`	`x < y`	**x** nhỏ hơn **y**
≥	`>=`	`x >= y`	**x** lớn hơn hoặc bằng **y**
≤	`<=`	`x <= y`	**x** nhỏ hơn hoặc bằng **y**
=	`==`	`x == y`	**x** bằng **y**
≠	`!=`	`x != y`	**x** khác **y**

1.8 Ra quyết định: Các phép toán quan hệ

- cấu trúc **`if`**
  - Đưa ra quyết định dựa vào kết quả đúng hoặc sai của điều kiện
    - Nếu điều kiện thỏa mãn thì thực hiện tập lệnh S
    - nếu không, tập lệnh S không được thực hiện

```
if ( num1 == num2 )
     cout << num1 << " is equal to " << num2 <<
  endl;
```

1.9 Khai báo using

- Khai báo sử dụng toàn bộ không gian tên
  - using namespace std;
  - Để không cần tiền tố std:: cho mọi tên trong std

```
1   // Fig. 1.4: fig01_04.cpp
2   // Printing a line with multiple statements.
3   #include <iostream>
4
5   using namespace std;
6   // function main begins program execution
7   int main()
8   {
9      cout << "Welcome ";
10     std::cout << "to C++!\n";
11
12     return 0;
13
14  } // end function main
```

1.9 Khai báo using

- Khai báo sử dụng từng tên

```
using std::cout;  // program uses cout
using std::cin;   // program uses cin
using std::endl;  // program uses endl
...
cout << "No need to write std::";
cin >> somevariable;
...
```