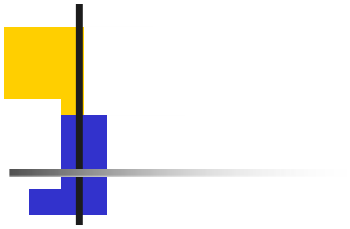
TIN HỌC ĐẠI CƯƠNG

Bài 2: Một số khái niệm cơ sở

Trần Thị Ngân

Bộ môn Công nghệ phần mềm, Khoa CNTT

Trường đại học Thủy Lợi



Nội dung chính

1. Cấu trúc một chương trình C++
2. Các thành phần cơ bản của C++
3. Bài tập

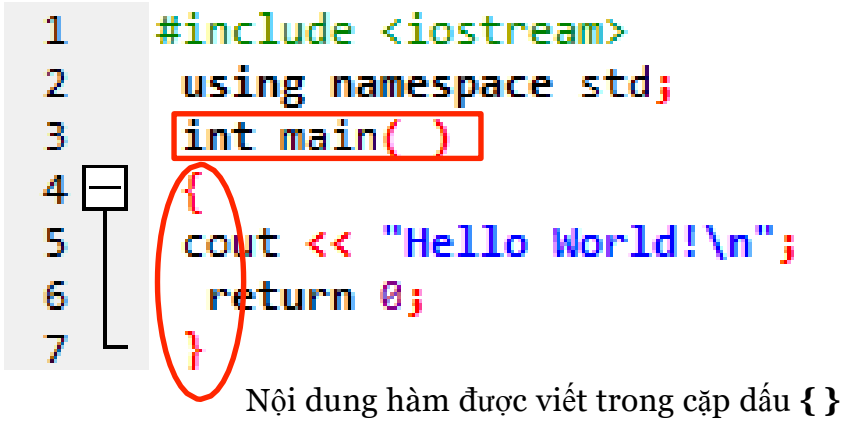
Thiết lập môi trường C++

- Tải phần mềm miễn phí Dev-C++ tại:

  https://sourceforge.net/projects/orwelldevcpp/
- Tiến hành cài đặt phần mềm trên máy tính
- Tạo file C++ mới: Vào File->New->Source File hoặc ấn Ctrl+N
- Lưu file dưới dạng file nguồn C++ hoặc với đuôi .cpp

Cấu trúc một chương trình C++

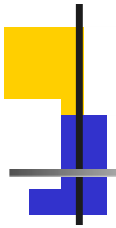
- Soạn thảo ví dụ đơn giản sau trong Dev-C++



Hàm chính: Chương trình sẽ bắt đầu từ hàm này

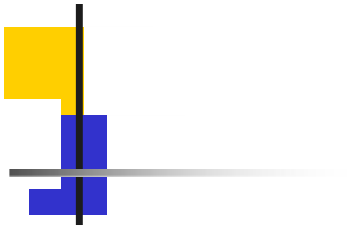
Nội dung hàm được viết trong cặp dấu **{ }**

- Lưu và đặt tên cho ví dụ
- Ấn **F9** để biên dịch, ấn **F10** để chạy



Các thành phần cơ bản củaC++

- Tập kí tự của C++
- Từ khóa
- Tên (định danh)
- Cấu trúc một chương trình C++
- Kiểu dữ liệu
- Biến
- Hằng
- Các toán tử
- Biểu thức
- Câu lệnh
- Một số hàm toán học



Tập ký tự của C++

- Các chữ cái la tinh: a .. z và A .. Z
- Dấu gạch dưới: _
- Các chữ số thập phân: 0, 1, . ., 9
- Các ký hiệu toán học: +, -, *, /, % , &, ||, !, >, <, = ...
- Các ký hiệu đặc biệt khác: . , ; : [] {} # $, dấu cách, ...

Từ khóa

- Từ khoá là từ được qui định trước trong NNLT, mỗi từ có một ý nghĩa nhất định
- Thường dùng để chỉ các loại dữ liệu hoặc kết hợp thành câu lệnh
- Một số từ khóa thường gặp: auto, break, case, char, continue, default, do, double, else, externe, float, for, goto, if, int, long, register, return, short, sizeof, static, struct, switch, typedef, union, unsigned, while
- **Lưu ý:** trong các chương trình C++, các từ khóa được in đậm

Tên (định danh)

- Tên là một dãy liên tiếp các chữ cái, chữ số và dấu gạch dưới (không chứa dấu cách)
- Phải bắt đầu bằng chữ cái hoặc dấu gạch dưới
- Không được trùng với từ khóa
- Chiều dài của tên không bị giới hạn
- Phân biệt chữ hoa và chữ thường

Ví dụ:

- Các tên đúng: i, i1, j, delta, PT_Bac_2
- Các tên sai: Bai tap, 3abc, case
- Các tên sau đây là khác nhau: ha_noi, Ha_noi, HA_NOI

Cấu trúc một chương trình C++

```
//chuong trinh C++ dau tien
#include <iostream>
using namespace std;
int main( )
{
  cout << "Hello World";
  return 0;
}
```

`//chuong trinh C++ dau tien` tất cả các dòng bắt đầu bằng // được xem là các dòng chú thích và không ảnh hưởng đến việc thực hiện của chương trình

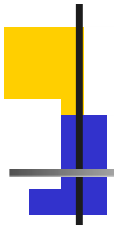
`#include <iostream>` đảm bảo rằng chương trình có thể sử dụng các định nghĩa trong thư viện vào ra chuẩn

Cấu trúc một chương trình C++

- `using namespace std;` khai báo sử dụng không gian tên std, định danh cout được định nghĩa trong không gian tên này
- `int main()` điểm bắt đầu quá trình thực hiện của các chương trình C++, tất cả các chương trình C++ đều có một hàm main
- `cout << "Hello World";` đây là một câu lệnh C++, làm nhiệm vụ in ra dòng chữ **`Hello World`**
- `return 0;` Kết thúc hàm main, trả về giá trị 0 cho hệ điều hành
- Các câu lệnh trong C++ phải kết thúc bằng dấu chấm phẩy

Kiểu dữ liệu

Loại dữ liệu	Tên	Kích thước	Miền giá trị
Kí tự	char	1 byte	signed: -128 to 127 unsigned: 0 tới 255
Số nguyên	short int (short)	2 byte	signed: -32768 to 32767 unsigned: 0 tới 65535
	int	4 byte	signed: -2147483648 tới 2147483647 unsigned: 0 tới 4294967295
	long int (long)	4 byte	signed: -2147483648 tới 2147483647 unsigned: 0 tới 4294967295
Kiểu logic	bool	1 byte	true hoặc false
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)



Biến

- Biến là một phần của bộ nhớ được dành để lưu trữ một giá trị xác định
- Giá trị của biến có thể thay đổi trong quá trình làm việc
- Cách khai báo biến:

  kiểu_dữ_liệu tên_biến;

- Ví dụ

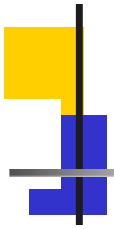
```
int a;
double mynumber;
```

Làm việc với biến

```
// Làm việc với biến

#include <iostream>
using namespace std;

int main ()
{
  int a, b, result; // Khai báo biến

  a = 5; b = 2;
  a = a + 3;
  result = a - b;
  cout << result;

  return 0;
}
```

Khởi tạo giá trị cho biến

```
#include<iostream>
using namespace std;
int main ()
{
        int a = 5; //Gia tri cua a la 5
        int b(2);  //Gia tri cua b la 2
        int result;  //Gia tri cua result la chua xac dinh
        a = a+3;
        result = a - b;
        cout<<result;
        return 0;
}
```



Hằng

- Hằng là một giá trị cố định nào đó
- Hằng thông thường được sử dụng để gán trị cho biến hoặc để biểu diễn thông điệp chúng ta muốn in ra
  Ví dụ:
  - Hằng nguyên: 1776, 707, -273
  - Hằng thực: 3.14159, 6.02e23, 1.6e-19
  - Hằng kí tự và xâu kí tự: 'z', 'p',"Xin chao"
  - Hằng logic: true, false

Một số hằng kí tự đặc biệt

\n	Kí tự xuống dòng
\r	Quay ngược lại đầu dòng
\t	Kí tự tab
\b	Kí tự khoảng trống
\f	Kéo trang
\a	Kí tự chuông (bíp)
\'	Dấu nháy đơn (')
\"	Dấu nháy kép (")
\?	Dấu hỏi(?)
\\	Dấu gạch chéo ngược (\)

Khai báo hằng

- Đôi khi sẽ thuận lợi hơn nếu ta đặt tên cho một hằng được sử dụng nhiều lần trong chương trình
- Cách khai báo hằng:

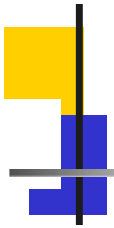
  #define tên_hằng giá_trị_hằng hoặc:

  const kiểu_dữ_liệu tên_hằng = giá_trị_hằng ;
- Ví dụ:

```
#define PI 3.14159265
#define NEWLINE '\n`
const int sosv = 50 ;
```

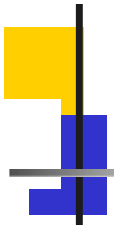
Khai báo và sử dụng hằng

```
#include<iostream>
using namespace std;
#define PI 3.14159 //Dinh nghia hang so PI
#define NEWLINE '\n'//Dinh nghia lenh tao 1 dong moi
int main ()
{
        double r = 1.5;
        double circle;
        circle = 2*PI*r;
        cout<<circle;
        cout<<NEWLINE;
//      cout<<circle;
        return 0;
}
```



Các toán tử

- Phép gán
- Toán tử số học
- Toán tử tăng/giảm
- Toán tử quan hệ
- Toán tử logic
- Toán tử điều kiện



Phép gán

- Gán một giá trị cho một biến
- Khi biến được gán giá trị mới, giá trị cũ sẽ được tự động xoá
- Cú pháp của phép gán:

  tên_biến = biểu_thức;

- Ví dụ:

  a = 5;

  a = b;

Phép gán

```
// phép gán

#include <iostream>
using namespace std;

int main ()
{
  int a, b;
  a = 10;
  b = 4;
  a = b;
  b = 7;

  cout << "a:";
  cout << a;
  cout << " b:";
  cout << b;

  return 0;
}
```

Viết gọn phép gán

- C++ cho phép viết gọn phép gán theo cách
  - sau:

x+=y	↔	x=x+y
x-=y	↔	x=x-y
x*=y	↔	x=x*y
x/=y	↔	x=x/y
x%=y	↔	x=x%y

Viết gọn phép gán

```
// phép gán được viết gọn

#include <iostream>
using namespace std;

int main ()
{
   int a, b=3;
   a = b;
   a+=2;                    // tương đương với a=a+2
   cout << a;
   return 0;
}
```

Toán tử số học

Toán tử số học	Ý nghĩa	Ví dụ
+	Cộng	12 + 4.9 // cho 16.9
-	Trừ	3.98 - 4 // cho -0.02
*	Nhân	2 * 3.4 // cho 6.8
/	Chia	9 / 2.0 // cho 4.5
%	Lấy phần dư	13 % 3 // cho 1

Ngoại trừ toán tử lấy phần dư (%) thì tất cả các toán tử số học cho phép pha trộn các toán hạng số nguyên và số thực

Toán tử tăng/giảm

- Các toán tử *tăng một* (++) và *giảm một* (--) giúp tiện lợi trong việc tăng thêm 1 hoặc giảm đi 1 đối với biến số.

Toán tử	Ý nghĩa	Ví dụ (với i = 3, j = 10)
++	Tăng một (tiền tố)	i = ++j ; // được i = 11, j = 11
++	Tăng một (hậu tố)	i = j++ ; // được i = 10, j = 11
--	Giảm một (tiền tố)	i = --j ; // được i = 9, j = 9
--	Giảm một (hậu tố)	i = j-- ; // được i = 10, j = 9

Toán tử quan hệ

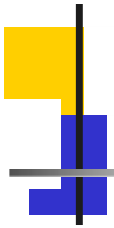
- Được sử dụng để so sánh giá trị của hai biểu thức
- Giá trị trả về thuộc kiểu logic: true (đúng) hoặc false (sai)

Toán tử quan hệ	Ý nghĩa	Ví dụ
<	Nhỏ hơn	5 < 5.5 // được true
>	Lớn hơn	5 > 5.5 // được false
<=	Nhỏ hơn hoặc bằng	5 <= 5 // được true
>=	Lớn hơn hoặc bằng	6.3 >= 5 // được true
==	Bằng	5 == 5 // được true
!=	Khác	5 != 5 // được false

Toán tử logic

Toán tử logic	Ý nghĩa	Ví dụ
!	Phủ định	!(5 == 5) // được false
&&	Và	5 < 6 && 6 < 6 // được false
||	Hoặc	5 < 6 || 6 < 5 // được true

- Các toán hạng của toán tử logic phải thuộc kiểu logic tức là có giá trị true (đúng) hoặc false (sai)
- Giá trị trả về cùng thuộc kiểu logic
- Phép toán "phủ định" đúng khi và chỉ khi toán hạng của nó sai
- Phép toán "và" đúng khi và chỉ khi hai toán hạng cùng đúng
- Phép toán "hoặc" sai khi và chỉ khi hai toán hạng cùng sai

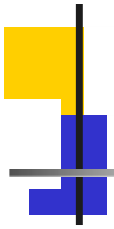


Toán tử điều kiện

- Toán tử điều kiện tính giá trị của một biểu thức và trả về một giá trị nếu biểu thức đúng; trả về một giá trị khác nếu biểu thức sai
- Cú pháp: điều_kiện ? kết_quả1: kết_quả;
- Nếu điều kiện đúng kết_quả1 được trả về, ngược lại kết_quả2 sẽ được trả về
- Ví dụ:
  - 7==5 ? 4 : 3 // trả về 3, vì 7 không bằng 5.
  - 7==5+2 ? 4 : 3 // trả về 4, vì 7 bằng 5+2.
  - 5>3 ? a : b // trả về giá trị của a, vì 5 lớn hơn 3.
  - a>b ? a : b // trả về số lớn hơn trong hai số a, b.

Toán tử điều kiện

```
// toán tử điều kiện

#include <iostream>
using namespace std;

int main ()
{
  int a,b,c;

  a=2;
  b=7;
  c = (a>b) ? a : b;

  cout << c;

  return 0;
}
```



Biểu thức

- Biểu thức là dãy kí hiệu kết hợp giữa các toán hạng,
  - toán tử và cặp dấu () theo một qui tắc nhất định
- Các toán hạng là hằng, biến, hàm
- Biểu thức cung cấp cách thức tính giá trị mới dựa trên các toán hạng và toán tử trong biểu thức.
- Ví dụ:
  - (x + y) * 2 - 4 ;
  - 3 - x + sqrt(y) ;
  - (-b + sqrt(delta)) / (2*a) ;

Thứ tự ưu tiên của các toán tử

- C++ qui định trật tự tính toán theo các mức
  độ ưu tiên như sau:
  1. Các biểu thức trong cặp dấu ngoặc ()
  2. Các toán tử 1 ngôi (phủ định, tăng, giảm, ...)
  3. Các toán tử số học
  4. Các toán tử quan hệ
  5. Các toán tử logic
  6. Các phép gán

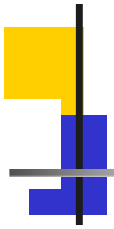
Thứ tự ưu tiên của các toán tử

Toán tử
( )
- ++ -- ! *(toán tử một ngôi)*
* / %
+ -
< <= > >=
== !=
&&
||
?:
= += -= *= /= %=

Ví dụ: 7+3*5 // = 22

(65 > 21) && ('B' < 'A') // = false

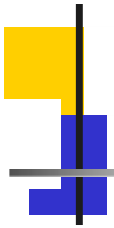
! (16.25 + 2 < 17) || (2 > 4 / 2) // = true



Một số hàm toán học

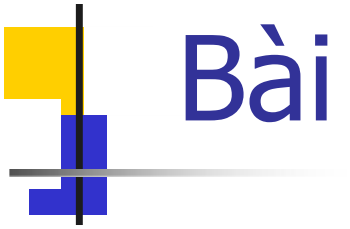
Một số hàm toán học trong thư viện **cmath**:

- sin(x), cos(x), tan(x), asin(x), acos(x), atan(x): các hàm lượng giác
- exp(x): hàm mũ, trả lại giá trị e mũ x (e^x).
- log(x), log10(x): trả lại lôgarit cơ số e và lôgarit thập phân của x (lnx, logx) .
- pow(x, y): hàm mũ, trả lại giá trị x lũy thừa y (x^y).
- sqrt(x): trả lại căn bậc 2 của x.
- abs(x), labs(x), fabs(x): trả lại giá trị tuyệt đối của x.
- ceil(x): trả lại giá trị làm tròn lên của x
- floor(x): trả lại giá trị làm tròn xuống của x



Câu lệnh

- Câu lệnh trong C++ được thiết lập từ các từ khoá và các biểu thức …
- Câu lệnh luôn luôn được kết thúc bằng dấu chấm phẩy
- Các câu lệnh được phép viết trên cùng một hoặc nhiều dòng
- Câu lệnh gồm nhiều lệnh được bao bởi cặp dấu ngoặc {} và được gọi là *khối lệnh*.
- Các biến được khai báo trong khối lệnh nào thì chỉ có tác dụng trong khối lệnh đó



Bài tập

Cho x là số nguyên không âm có 2 chữ số. Viết chương trình tính tổng 2 chữ số của x.

Ví dụ : nếu x là 98 thì kết quả cho ra là 9 + 8 = 17.