



# TOÁN TỬ

NHẬP MÔN LẬP TRÌNH

GVHD: Trương Toàn Thịnh

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

# GIỚI THIỆU

- Ví dụ

Dòng	C++	C
1	//Tập tin Hello.cpp	//Tập tin Hello.c
2	#include <iostream>	#include <stdio.h>
3	using namespace std;	
4	void main()	void main()
5	{	{
6	cout<<"Hello World!!!"<<endl;	printf("Hello World!!!\n");
7	}	}

# GIỚI THIỆU

- Dòng 2: thư viện `iostream` trong C++ hỗ trợ input/output (tương tự `scanf` & `printf`)
- Dòng 3: Sử dụng `'namespace std'` để gọi `cout` (có thể dùng `std::cout` nếu không có `'namespace std'`)
- Dòng 6: Dùng `cout` để in “Hello world” với toán tử `“endl”` tương tự như `“\n”`

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

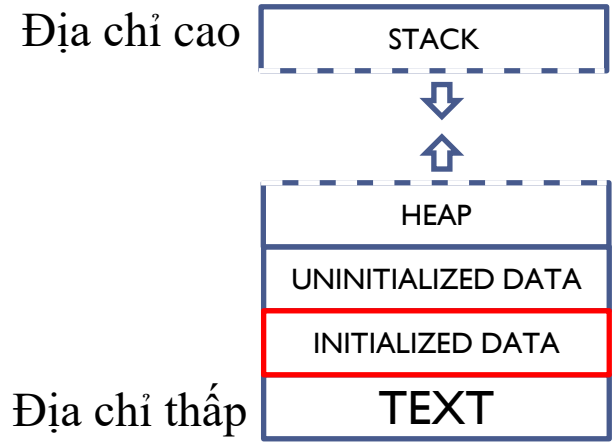
# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Kiểu dữ liệu
  - C hỗ trợ kiểu
    - Ký tự (**char**): ví dụ 'a', 'b'
    - Số nguyên (**int**, **long**): ví dụ 23, 24L
    - Số thực độ chính xác đơn (**float**): ví dụ 1.2F, 2.2F
    - Số thực độ chính xác kép (**double**): ví dụ 1.2, 2.2
  - Mỗi kiểu đều có giá trị min & max
  - Kiểu ký tự & số nguyên sử dụng như nhau
    - Ví dụ: **char** a = 65
  - Kiểu số nguyên có thể có dấu hay không dấu

# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Hằng: là các đại lượng mà giá trị của nó không thay đổi

- Hằng kí tự: ‘a’, ‘b’
- Hằng số nguyên: 22, 22L
- Hằng số thực: 1.2, 1.2F
- Hằng chuỗi: “hello”



- Định nghĩa tên hằng: ví dụ muốn đặt tên cho đại lượng 3.14 là PI
  - `#define` PI 3.14 (Ngôn ngữ C)
  - `const int` PI 3.14; (Ngôn ngữ C++)

# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Biến: là các đại lượng mà giá trị của nó có thể thay đổi

- Ví dụ: `int` a, `char` c, `float` f...

- Qui ước đặt tên biến:

- Gọi nhớ: `int` dienTich, `char` kt...
  - Luôn bắt đầu với ký tự (A → Z), dấu gạch chân(\_)
  - Cũng có thể tuân thủ theo hợp đồng riêng

- Kích thước kiểu dữ liệu có thể lấy bằng toán tử `sizeof` với đầu vào là một tên biến hoặc kiểu dữ liệu (Đơn vị là byte)

- Ví dụ: `sizeof(int)` (= 4), `sizeof(c)` (= 1)

Địa chỉ cao

STACK



HEAP

UNINITIALIZED DATA

INITIALIZED DATA

TEXT

Địa chỉ thấp



# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Ví dụ

Dòng	Mô tả
1	//Tap tin Circle.c
2	#include <stdio.h>
3	void main()
4	{
5	#define PI 3.14159
6	float R = 1.25;
7	float DienTich;
8	DienTich = PI*R*R;
9	printf("Hinh tron, ban kinh = %f\n", R);
10	printf("Dien tich = %f", DienTich);
11	}

# KIỂU DỮ LIỆU, HẰNG & BIẾN




- Dòng 5: Đặt tên cho hằng số 3.14 là PI
- Dòng 6: Khai báo & định nghĩa giá trị cho biến số thực R giá trị là 1.25
- Dòng 7: Khai báo biến số thực DienTich
- Dòng 8: Tính diện tích theo công thức, sau đó gán giá trị cho biến DienTich
- Dòng 9 & 10: Xuất giá trị ra màn hình

# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Ví dụ

Dòng	Mô tả
1	//Tap tin varSize.c
2	#include <stdio.h>
3	void main()
4	{
5	short Delta = 9;
6	printf("Kích thước delta = %d\n", sizeof(delta));
7	printf("Kích thước kiểu int = %d\n", sizeof(int));
8	printf("Kích thước kiểu long = %d\n", sizeof(long));
9	printf("Kích thước kiểu float = %d\n", sizeof(float));
10	printf("Kích thước kiểu char = %d\n", sizeof(char));
11	}

# KIỂU DỮ LIỆU, HẰNG & BIẾN

- Có thể khái quát quá trình xử lý vấn đề bao gồm ba bước
  - Nhập dữ liệu
  - Xử lý dữ liệu
  - Xuất kết quả
- Các kí hiệu trong lưu đồ thuật toán
  -  : Bắt đầu/kết thúc (Start/End)
  -  : Nhập/xuất (Input/output)
  -  : Xử lý kết quả (Process)

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Kiểu số nguyên
  - **char**: ký tự 8-bit có dấu có giá trị  $\in [-128, 127]$
  - **unsigned char**: ký tự không dấu 8-bit có giá trị từ  $[0, 255]$
  - **int**: số nguyên có dấu 32-bit có giá trị từ  $[-2^{31}, 2^{31} - 1]$
  - **unsigned int**: số nguyên không dấu 32-bit có giá trị từ  $[0, 2^{32} - 1]$
  - **short**: số nguyên có dấu 16-bit có giá trị từ  $[-32768, 32767]$
  - **unsigned short**: số nguyên không dấu 16-bit có giá trị từ  $[0, 65535]$
- Hỗ trợ các thao tác:  $+$ ,  $-$ ,  $\times$ ,  $\div$ ,  $\%$
- Cách biểu diễn kiểu số nguyên
  - Dạng thập phân: ví dụ  $15_{10}$
  - Dạng nhị phân: ví dụ  $1111_2$
  - Dạng thập lục: ví dụ  $0xF$
  - Dạng bát phân: ví dụ  $17_8$

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Ví dụ số nguyên có dấu kích thước 16-bit
  - Xét số âm (bit đầu là 1)

1xxxxxxxxxxxxxxxxx

Có  $2^{15}$  số âm { 0000000000000000  
0000000000000001  
0000000000000010  
...  
1111111111111111

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Ví dụ số nguyên có dấu kích thước 16-bit
  - Xét số dương (bit đầu là 0)

0xxxxxxxxxxxxxxxxx

0000000000000000

0000000000000001

0000000000000010

...

1111111111111111

Có  $2^{15} - 1$  số dương



# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Các thao tác trên kiểu số nguyên
  - Xét  $A = 10_{10} (1010_2)$  &  $B = 15_{10} (1111_2)$
  - $A \& B = (1010_2) \& (1111_2) = 1010_2 (10_{10})$
  - $A | B = (1010_2) | (1111_2) = 1111_2 (15_{10})$
  - $A \wedge B = (1010_2) \wedge (1111_2) = 0101_2 (5_{10})$
  - $\sim A = \sim(1010_2) = \sim(00001010_2) = (11110101_2) = -11$  (Nhớ công thức  $\sim(n) = -n-1$ )
  - $A \ll 1 = (00001010_2) \ll 1 = (00010100_2) = 20_{10}$
  - $A \gg 1 = (00001010_2) \gg 1 = (00000101_2) = 5_{10}$
- Xét các trường hợp:
  - `unsigned char`  $c = -1$ ; //  $c = 255 = -1 + 256$
  - `char`  $b = -129$ ; //  $b = 127 = -129 + 256$
  - `char`  $a = 128$ ; //  $a = -128 = 128 - 256$

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Kiểu thực có hai loại
  - **float**: kích thước 4-byte, có giá trị từ  $[1.401298 \times 10^{-45}, 3.40282 \times 10^{38}]$
  - **double**: kích thước 8-byte, có giá trị từ  $4.94066 \times 10^{-324}, 1.79769 \times 10^{308}]$
- Hỗ trợ các thao tác  $+$ ,  $-$ ,  $\times$ ,  $\div$
- Ví dụ: xét số thực 209.8125F
  - $209_{10} = 11010001_2$
  - $0.8125 \times 2 \rightarrow 1.625 \times 2 \rightarrow 1.25 \times 2 \rightarrow 0.5 \times 2 \rightarrow 1.0$ : vậy  $0.8125 = 0.1101$
  - Vậy  $209.8125 = 11010001.1101_2$
  - **Chuẩn hóa** về '1.F': 1.**10100011101** (Đời dấu chấm qua trái **7** vị trí).
  - **Lấy** 7 + **127** = 134 = **10000110**<sub>2</sub> (Dùng số 'quá 127')
  - Cuối cùng ta có

31	30					23	22							0
0	1	0	0	0	0	1	1	0	1	0	1	0	0	0

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Kiểu luận lý: là kiểu nguyên đặc biệt trong C/C++
- Có hai giá trị {**true** ( $\neq 0$ ), **false** (0)}
- Hỗ trợ các thao tác:
  - **&&**: (A **&&** B) **true** khi hai biểu thức A và B đều **true**
  - **||**: (A **||** B) **false** khi hai biểu thức A và B đều **false**
  - **!**: (!B) là **true** khi B **false** và ngược lại

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Ví dụ kiểu luận lý

Dòng	Mô tả
1	<code>#include &lt;stdio.h&gt;</code>
2	<code>void main(){</code>
3	<code>bool bVal;</code>
4	<code>float x = 46.7F, y = 93F, z;</code>
5	<code>bVal = (x == y);</code>
6	<code>printf(“%d\n”, bVal);</code>
7	<code>bVal = (x &lt; y);</code>
8	<code>printf(“%d\n”, bVal);</code>
9	<code>z = (x &gt; y)*x + (x &lt;= y)*y;</code>
10	<code>printf(“%f\n”, z);</code>
11	<code>}</code>

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Kiểu ký tự có hai loại
  - Ký tự 8-bit: kiểu `char` hay `unsigned char`
  - Ký tự 16-bit: kiểu `wchar_t` (Lưu ý đây là kiểu riêng cho ký tự 16-bit nên không xét dấu)
- Ví dụ: `char c = 'a'; wchar_t d = L'a';`
- Ký tự 8-bit có giá trị  $[0, 255] \in \text{ASCII}$
- Ký tự 16-bit theo bảng mã Unicode
- Có thể dùng phép toán so sánh và  $+$ ,  $-$  với kiểu ký tự

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Ví dụ kiểu kí tự

Dòng	Mô tả
1	<code>#include &lt;stdio.h&gt;</code>
2	<code>void main(){</code>
3	<code>char ch = 65;</code>
4	<code>printf("%c\n", ch);</code>
5	<code>ch = 'A';</code>
6	<code>printf("%c\n", ch);</code>
7	<code>printf("Nhập ch: ");</code>
8	<code>scanf("%c", &amp;ch);</code>
9	<code>printf("ASCII code: %d\n", ch);</code>
10	<code>ch -= ('a' - 'A')*(ch &gt;= 'a' &amp;&amp; ch &lt;= 'z');</code>
11	<code>printf("Upper case: %c\n", ch);</code>
12	<code>}</code>

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Phép gán:

- $a = b$ : giá trị biến  $b$  giữ sẽ gán cho biến  $a$  (giá trị  $b$  và  $a$  lúc này bằng nhau)
- $a = b = c$ : ...
- $a = a + 1$ : giá trị biến  $a$  tăng lên 1, sau đó gán lại cho chính nó ( $a$  tăng 1 đơn vị)
- $a++$ :...
- $++a$ :...
- $a = b++$ : lấy giá trị  $b$  gán cho  $a$ , sau đó tăng  $b$  lên 1 đơn vị
- $a = ++b$ : tăng  $b$  lên 1 đơn vị, sau đó lấy giá trị mới tăng đó gán cho  $a$

# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Độ lớn & độ chính xác:
  - Kiểu dữ liệu trên máy tính có độ lớn giới hạn.
  - Ví dụ xét hai số nguyên kiểu `short`  
`short a = 400, b = 500, z;`  
`z = a*b; // z = 3392 = 200000 mod 216`
  - Khi số thực vượt quá độ lớn cho phép có thể xảy ra một vài hiện tượng lạ:
  - Ví dụ xét số `float`:
    - Số thực  $1.401298E-45F / 2 = 0$
    - Số thực:  $3.40282E+38F + 10 = 3.40282E+38F$



# CÁC KIỂU DỮ LIỆU CƠ SỞ

- Độ lớn & độ chính xác:
  - Nếu số thực không biểu diễn được ‘tròn vẹn’ sẽ có tình trạng xấp xỉ  $\rightarrow$  không chính xác
  - Xét ví dụ biểu diễn được:

[illegible]

**b = 1.250f**                      0 01111111 010000000000000000000000

**c = a + b = 2.375f**    0 10000000    001100000000000000000000

- Xét ví dụ không biểu diễn được

$$a = 1.123f$$
$$b = 1.456f$$
$$c = a + b \neq 2.579f$$

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

# THƯ VIỆN HÀM

- Tập các hàm được viết sẵn để phục vụ cho mục tiêu nào đó được gọi là thư viện hàm
  - Ví dụ thư viện hàm
    - <math.h>: tập các hàm toán học
    - <ctype.h>: tập các hàm xử lý kí tự
- Hàm là một đơn vị xử lý trong lập trình cần đầu vào hợp lệ và sẽ ra một kết quả dựa trên đầu vào.
  - Ví dụ hàm `sqrt(double)` có
    - Đầu vào là một số thực kiểu `double`
    - Đầu ra là kết quả của phép tính  $\sqrt{\phantom{x}}$

# THƯ VIỆN HÀM

- Một số hàm xử lý kiểu `char/wchar_t`

<code>int</code> <code>isupper(char)</code> <code>int</code> <code>iswupper(wchar_t)</code>	Kiểm tra xem ký tự đầu vào có là ký tự hoa hay không
<code>char</code> <code>toupper(char)</code> <code>wchar_t</code> <code>towupper(wchar_t)</code>	Trả về ký tự hoa tương ứng với ký tự thường đầu vào
<code>int</code> <code>islower(char)</code> <code>int</code> <code>iswlower(wchar_t)</code>	Kiểm tra xem ký tự đầu vào có là ký tự thường hay không
<code>char</code> <code>tolower(char)</code> <code>wchar_t</code> <code>towlower(wchar_t)</code>	Trả về ký tự thường tương ứng với ký tự hoa đầu vào

# THƯ VIỆN HÀM

- Ví dụ viết chương trình tính  $y = \log_2 8$
- Gợi ý:  $\log_a b = \frac{\log_c b}{\log_c a}$

Dòng	Mô tả
1	<code>#include &lt;stdio.h&gt;</code>
2	<code>#include &lt;math.h&gt;</code>
3	<code>void main(){</code>
4	<code>double a = 2, b = 8, c;</code>
5	<code>c = log(8)/log(2);</code>
6	<code>printf("%lf\n", c);</code>
7	<code>}</code>

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Ta có thể qui định cách thức hiển thị trên màn hình console
  - Kiểu số nguyên có dấu
    - Để in số nguyên ‘**int**’: dùng “%d”
    - Để in số nguyên dài ‘**long**’: dùng “%ld”
    - Để in số nguyên ngắn ‘**short**’: dùng “%hd”
  - Kiểu số nguyên không dấu: dùng “%u”
  - Kiểu số thực ‘**float**’: dùng “%f” hay “%e”
  - Kiểu số thực dài ‘**double**’: dùng “%lf” hay “%le”
  - Kiểu kí tự ‘**char**’: dùng “%c”
  - Kiểu chuỗi kí tự ‘**char\***’: dùng “%s”
  - Kiểu số nguyên theo cơ số:
    - Cơ số 16: “%x”
    - Cơ số 8: “%o”

# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Ví dụ

Dòng	Mô tả
1	<code>#include &lt;stdio.h&gt;</code>
2	<code>void main(){</code>
3	<code>int a = 28; long b = -9; short c = 8;</code>
4	<code>float d = 1.2E-8F;</code>
5	<code>printf("%d\n", a);</code>
6	<code>printf("%hd\n", c);</code>
7	<code>printf("%ld\n", b);</code>
8	<code>printf("%o\n", a);</code>
9	<code>printf("%x\n", a);</code>
10	<code>printf("%e\n", c);</code>
11	<code>}</code>



# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Ta có thể qui định độ rộng và độ chính xác cho kiểu nguyên và kiểu thực
- Dùng dạng ‘<bổ từ>width.precision’ để xác định kết quả xuất ra
- Ví dụ số thực:

```
float a = 2.335
```

```
printf(“%0.2f”, a); //2.33
```

- Ví dụ số nguyên với bổ từ ‘0’:

```
int a = 2;
```

```
printf(“%05d”, a); //000002
```

- Còn một số bổ từ như ‘-’ hay ‘\*’...

# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Định dạng nhập xuất với C++:
  - Được thực hiện với hai đối tượng cin và cout kết hợp toán tử “>>” và “<<”
  - Cần khai báo
    - `#include <iostream>`
    - `using namespace std;`
  - Khi muốn định dạng dữ liệu xuất cần khai báo thêm `#include <iomanip>`. Thư viện chứa một số toán tử và hàm tiện ích
    - `setw(n)`: thiết lập độ rộng (bên trái hay bên phải) để in ra đúng n ký tự ra màn hình (mặc định chèn khoảng trắng)
    - `setfill(ch)`: thường dùng kèm với `setw` để in ra n ký tự ch
    - `setprecision(n)`: xác định độ chính xác khi in ra số thực
    - Ngoài ra còn có các toán tử `hex`, `oct`, `endl`, `left`, `right`

# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Định dạng nhập xuất với C++:
  - Ví dụ
    - `#include <iostream>`
    - `#include <iomanip>`
    - `void main(){`
      - `int a = 970, h = 10, v = 9700;`
      - `cout << setw(8) << "Area" << setw(10) << a << endl;`
      - `cout << setw(8) << "H" << setw(10) << h << endl;`
      - `cout << setw(8) << "Volume" << setw(10) << v << endl;`
    - `}`

Kết quả khi chạy chương trình

Area      970

H        10

Volume    9700

8 ký tự

10 ký tự

# ĐỊNH DẠNG DỮ LIỆU NHẬP/XUẤT

- Định dạng nhập xuất với C++:
  - Ví dụ
    - `#include <iostream>`
    - `#include <iomanip>`
    - `void main(){`
      - `long n;`
      - `cout << "n (hexadecimal) = ";`
      - `cin >> hex >> n;`
      - `cout << "Octal representation: " << oct << n << endl;`
    - `}`

Kết quả khi chạy chương trình  
n (hexadecimal) = 8 ↵  
Octal representation: 10

# NỘI DUNG

- Giới thiệu
- Kiểu dữ liệu, hằng & biến
- Các kiểu dữ liệu cơ sở
- Thư viện hàm
- Định dạng dữ liệu nhập/xuất
- Bài tập

# BÀI TẬP

- Viết các chương trình thực hiện một số công việc sau đây
  - Cho người dùng nhập năm sinh, in ra tuổi
  - Cho người dùng nhập kí tự, in ra kí tự hoa
  - Cho người dùng nhập số tiền cần rút, in ra số lượng tiền xuất ra theo mệnh giá: 500,000 - 200,000 – 100,000 – 50,000 – 20,000 – 10,000
    - Ví dụ:  $2,600,000đ = 5 \times 500,000 + 0 \times 200,000 + 1 \times 100,000 + 0 \times 50,000 + 0 \times 20,000 + 0 \times 10,000$