Tin học cơ sở 4

Buổi 2. Biến, kiểu dữ liệu, phép toán

Bộ môn Khoa học máy tính - 2017

Nội dung buổi học

1. Biến và khai báo biến, hằng số
2. Các kiểu dữ liệu cơ bản
3. Các phép toán, biểu thức

Bộ nhớ máy tính

- Bộ nhớ: chuỗi các phần tử 0 – 1 (bit)
  - 4GB RAM = 2^{32} byte = 2^{35} bit (~ 32 tỷ bit)

BITS & BYTES

1 BIT - stands for Binary digIT
1 Byte = 8 bits
½ byte = 1 nibble = 4 bits
2x byte = 1 word = 16 bits
2x word = 1 long word = 32 bits

1KByte = 1,024 bytes = 2^{10}
1MByte = 1,048,576 bytes = 2^{20}
1GByte = 2^{30} bytes
1TByte = 2^{40} bytes

Bộ nhớ máy tính

- Ngôn ngữ lập trình bậc cao (C++)
  - Đặt tên và kiểu cho các vùng nhớ (biến)
    - Ví dụ: vùng nhớ chứa thời gian máy bay tới đích đến là **một số thực** được đặt tên là **timeToArrival**.
    - Ví dụ: vùng nhớ chứa họ và tên của sinh viên là một **chuỗi ký tự** được đặt tên là **studentName**.
  - Đem lại ý nghĩa cho các vùng nhớ
    - Thay vì các con số 0 – 1

CT1. Nhập và in dữ liệu

Mục tiêu: Nhập tên và in câu chào

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

int main()
{
    cout << "Please enter your name: ";
    string firstName;
    cin >> firstName;
    cout << "Hello, " << firstName << endl;
    return 0;
}
```

CT1. Mô tả

- In câu thông báo
- Khai báo biến firstName có kiểu string
  - Vùng nhớ là một chuỗi ký tự
- Nhập chuỗi ký tự: phép toán >>
  - Vào ***biến*** (vùng nhớ) có tên firstName
- In ra câu chào

Kiểu dữ liệu của biến

- Kiểu dữ liệu của biến quyết định các phép toán trên biến
  - cin >> firstName; đọc ký tự đến khi gặp một khoảng trắng (đọc 1 từ)
  - Khoảng trắng: dấu cách, dấu tab, dấu xuống dòng

Đối tượng cin

- cin: đại diện cho ***luồng nhập chuẩn***
  - Nếu không có gì đặc biệt: bàn phím
  - cout: luồng xuất chuẩn (màn hình)
- cin sử dụng phép toán >> (right-shift)
  - Ngược lại với cout (<<)

CT2. Phép toán trên chuỗi

Mục tiêu: Nhập tên, họ và in câu chào

```
int main() {
    cout << "Enter your first and second names\n";
    string first, second;
    cin >> first >> second;
    string name = first + ' ' + second;
    cout << "Hello, " << name << endl;
    return 0;
}
```

CT2. Mô tả

- In câu thông báo
- Khai báo 2 biến first, second kiểu string
- Nhập chuỗi ký tự vào hai biến
  - Biến first rồi đến biến second
- Nối hai chuỗi với dấu cách ở giữa
  - Phép toán cộng chuỗi (+)
  - Phép toán gán kết quả vào biến name (=)

Phép gán

- Dấu bằng =: Gán giá trị của biểu thức về phải vào biến (vùng nhớ) ở về trái
  - Vùng nhớ ở về trái có kiểu tương thích với giá trị của biểu thức về phải
- Ví dụ:
  - name = first + ' ' + second;
  - distance = velocity * timePassed;
  - circleArea = 3.14 * circleRadius * circleRadius;

CT2. Một số lưu ý

- Các giá trị chuỗi có thể viết liên tiếp
  - C++ tự động nối các giá trị chuỗi
  - Ví dụ: "Hello, " "World"
- Ký tự `'\n'`: dấu xuống dòng
  - `'\t': dấu tab`
- Phép cộng + nối 2 xâu
  - Phép cộng với số ?

CT3. Nhập và in số nguyên

```
int main() {
    cout << "please enter your first name and age\n";
    string firstName;
    int age;
    cin >> firstName >> age;
    cout << "Hello, " << firstName
         << " age " << age << endl;
    return 0;
}
```

CT3. Mô tả

- Khai báo biến firstName kiểu string và biến age kiểu int (số nguyên)
- Nhập chuỗi ký tự vào firstName
- Nhập tuổi (số nguyên) vào biến age
- In ra câu chào
  - Tên trước rồi đến tuổi

Số và chuỗi ký tự

String

- cin >>: đọc chuỗi
- cout <<: in chuỗi
- phép +: nối chuỗi
- phép += s: nối chuỗi s vào cuối
- phép ++: không có (sinh lỗi)
- phép -: không có (sinh lỗi)

int, float, double

- cin >>: đọc số
- cout <<: in số
- phép +: cộng số học
- phép += n: tăng lên n
- phép ++: tăng lên 1
- phép -: trừ số học

Tên

- Bắt đầu bằng chữ cái hoặc dấu gạch dưới _
- Có thể có chữ cái [a-zA-Z], chữ số [0-9], dấu gạch dưới _
  - ✓ x, timeToArrival, _myBeauty1
  - ~~X main land, 1234x, timeToArrival~~
- Không dùng các từ khóa của C++
  - int, if, while, double …

Bảng từ khóa C++11

alignas	bitor	concept	dynamic_cast
alignof	bool	const	else
and	break	constexpr	enum
and_eq	case	const_cast	explicit
asm	catch	continue	export
atomic_cancel	char	decltype	extern
atomic_commit	char16_t	default	false
atomic_noexcept	char32_t	delete	float
auto	class	do	for
bitand	compl	double	friend

Bảng từ khóa C++11

goto	not	requires	template	using
if	not_eq	return	this	virtual
import	nullptr	short	thread_local	void
inline	operator	signed	throw	volatile
int	or	sizeof	true	wchar_t
long	or_eq	static	try	while
module	private	static_assert	typedef	xor
mutable	protected	static_cast	typeid	xor_eq
namespace	public	struct	typename	
new	register	switch	union	
noexcept	reinterpret_cast	synchronized	unsigned	

Cách đặt tên

- Chọn tên có ý nghĩa
  - Lập trình như kể chuyện
  - Tên viết tắt thường khó đọc
    - Cho người khác
    - Cho chính người lập trình sau 1 ngày, 1 tuần
  - Dùng tên viết tắt chỉ khi chắc chắn ai cũng hiểu
    - Tọa độ x, y
    - Biến đếm i

Cách đặt tên

- Đặt tên không quá dài
  - Tốt: arrivalTime
  - Không nên: the_time_that_John_arrive
- Phong cách
  - Dùng dấu gạch dưới
    - partial_sum, people_count
  - Dùng kiểu “lạc đà” (CamelCase)
    - timeToArrival, ageOfJohn

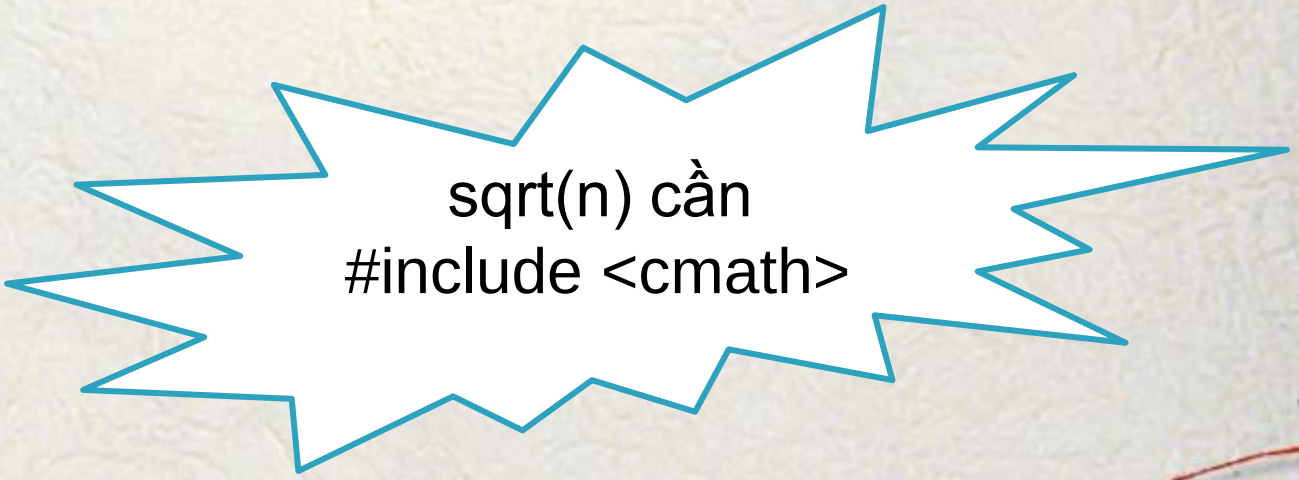
CT4. Phép toán số học

```
#include <cmath>
```

```
int main()
{
    cout << "please enter a floating-point number: ";
    double n;
    cin >> n;
    cout << "n == " << n
         << "\nn+1 == " << n+1
         << "\nthree times n == " << 3*n
         << "\ntwice n == " << n+n
         << "\nn squared == " << n*n
         << "\nhalf of n == " << n/2
         << "\nsquare root of n == " << sqrt(n)
         << '\n';
}
```

CT4. Mô tả

- Nhập một số thực vào biến n
  - Viết tắt của number
- In giá trị các biểu thức tính trên n
  - n+1
  - n*3
  - n+n
  - n*n
  - n/2



CT5. Đổi đơn vị inch ra cm

```
int main()
{
    const double cmPerInch = 2.54;
    int lengthInInch = 1;
    cout << "Please enter a length "
         << "in inches: ";
    cin >> lengthInInch;
    cout << lengthInInch << "in.  = "
         << cmPerInch*lengthInInch
         << "cm.\n";
}
```

CT5. Mô tả

- Biến cmPerInch có giá trị 2.54
  - Không thay đổi trong chương trình
  - Thêm khai báo const
    - Đặt tên cho giá trị 2.54 (để kể chuyện !!!)
    - Tránh lỗi thay đổi giá trị
  - Gọi là biến hằng hoặc hằng (constant)

Kiểu và giá trị

Kiểu cơ bản	Mô tả	Giá trị
bool	logic, đúng sai	true, false
char	ký tự	`'a', '1', '\n'`
int, short, long	số nguyên	1, 2, -1, -1024
float, double	số thực	1.35, 1.2E2, 0.54E-3, 1.5f

Kiểu và giá trị

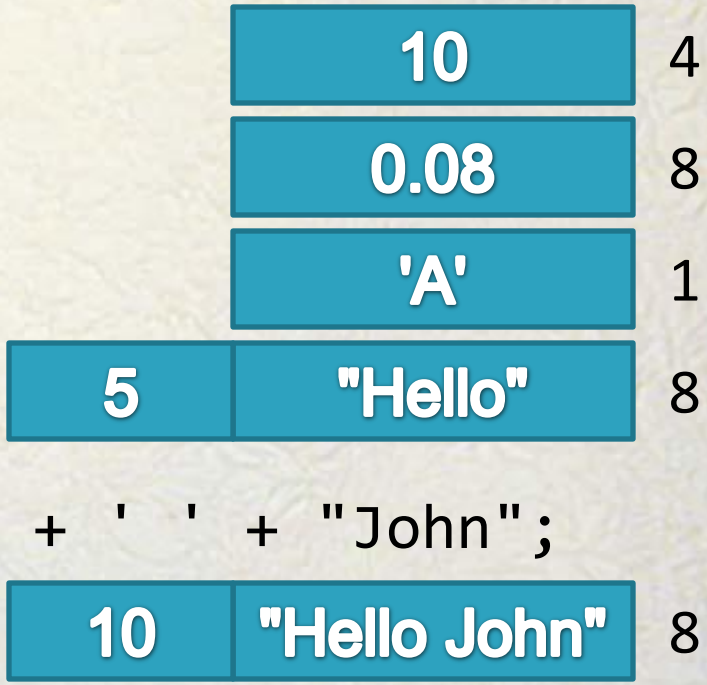
Kiểu của thư viện chuẩn	Mô tả	Giá trị
string	chuỗi ký tự	“Hello, World”
complex<T>	số phức	`complex<float>(1.0f, 2.0f)` `complex<double>(2.34, -1.26)`
và nhiều kiểu khác		

Kiểu và giá trị

- Kiểu có sẵn trong ngôn ngữ C++
  - int, bool, double, char …
- Kiểu định nghĩa trong thư viện chuẩn
  - string, complex, … (#include <string>)
- Kiểu do người dùng tự định nghĩa

Khai báo và khởi tạo biến

```
int x = 10;
double rate = 0.08;
char c = 'A';
string hello = "Hello";
string hello_john = hello + ' ' + "John";
```




```
cout << sizeof(string);
```

Đối tượng

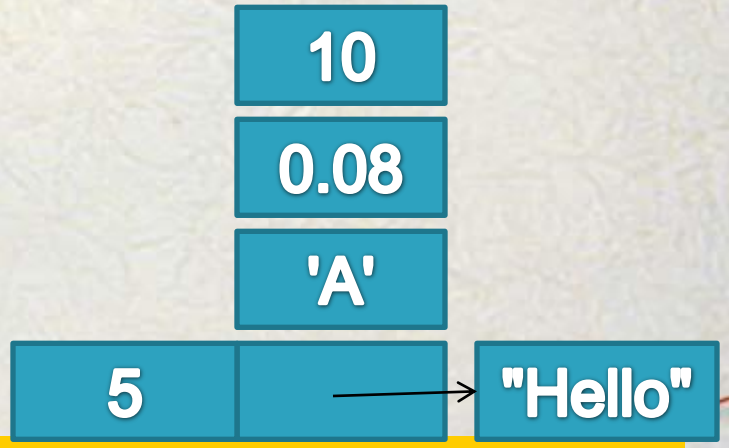
- Đối tượng: bộ nhớ chứa giá trị có kiểu nhất định
- Biến: đối tượng có tên
- Khai báo: đặt tên cho đối tượng

```
int x = 10;

double rate = 0.08;

char c = 'A';

string hello = "Hello";
```



An toàn kiểu (type safety)

- Kiểu giới hạn cách dùng đối tượng
  - Phép toán phụ thuộc vào kiểu
    - Có nghĩa / không có nghĩa
    - Kết quả phép toán
  - Biến chỉ có nghĩa sau khi khai báo
    - Mọi phép toán phù hợp với kiểu của biến trả về một giá trị xác định

An toàn kiểu (type safety)

- Cố gắng dùng các phép toán phù hợp với kiểu
- Tuy nhiên, C++ cho phép "vi phạm"
  - Mã lệnh dễ viết, sáng tạo hơn

Vi phạm an toàn kiểu

```
int main()
{
    int a = 20000;
    char c = a;
    int b = c;
    if (a != b)      //  != means "not equal"
        cout << "oops!: " << a << "!=" << b << '\n';
    else
        cout << "Wow! We have large characters\n";
}
```

C++ cho phép chuyển giá trị từ kiểu "lớn hơn" sang biến có kiểu "nhỏ hơn"

Vi phạm an toàn kiểu

```
int main()
{
    int x;
    char c;
    double d;

    double dd = d;

    cout << " x: " << x << " c: " << c
         << " d: " << d << '\n';
}
```

x, c, d chưa được khởi tạo, giá trị ngẫu nhiên

d, dd có thể không có nghĩa

Biểu diễn đối tượng trong bộ nhớ

- Đối tượng là một dãy bít (0-1)
  - 'A' = 65_{10} = 01000001_2
- Kiểu mang lại ý nghĩa cho dãy bít
  - Giống như: số 37 có nghĩa gì ?
    - Nhiệt độ ? Độ ẩm ? Chiều dài ? Thời gian ...

```
char c = 'A';
cout << c; // A
int i = c;
cout << i; // 65
```

Biểu diễn đối tượng trong bộ nhớ

Kiểu	Số byte (8-bit)	Số bit
int	4	32
char	1	8
double	8	64

```
cout << sizeof(<tên kiểu/tên biến>);
```

Biểu thức

- Tổ hợp của toán tử và toán hạng
  - Toán tử: các ký hiệu phép toán
  - Toán hạng: dữ liệu cho các phép toán
- Toán tử cần phù hợp với toán hạng
  - Phép toán hai ngôi: +,-, =, *, /, +=, -=
  - Phép toán một ngôi: ++, --

CT6. Tính quãng đường

```
int main()
{
    const double timeStep = 0.5;
    double position = 0;
    double velocity;

    cout << "Enter your velocity (m/s): ";
    cin >> velocity;

    cout << "Starting: position = " << position << " (m)" << endl;
    position = position + velocity * timeStep;
    cout << "Next step: position = " << position << " (m)"  << endl;
    position = position + velocity * timeStep;
    cout << "Next step: position = " << position << " (m)"  << endl;
}
```

CT6. Sử dụng hàm

```
double getPosition(double oldPosition, double velocity,
                   double timeStep)
{
    double distance = velocity * timeStep;
    double newPosition = oldPosition + distance;
    return newPosition;
}

...
    cout << "Starting: position = " << position << " (m)" << endl;
    position = getPosition(position, velocity, timeStep);
    cout << "Next step: position = " << position << " (m)"  << endl;
    position = getPosition(position, velocity, timeStep);
    cout << "Next step: position = " << position << " (m)"  << endl;
```

CT6. Sử dụng hàm

```
double getPosition(double oldPosition,
                   double velocity,
                   double timeStep)
{
    double distance = velocity * timeStep;
    double newPosition = oldPosition + distance;
    return newPosition;
}
```

CT6. Tiếp tục sử dụng hàm

```
double step(double oldPosition, double velocity, double timeStep)
{
    double newPosition = getPosition(oldPosition, velocity, timeStep);
    cout << "Next step: position = " << newPosition << " (m)"  << endl;
    return newPosition;
}

...
    cout << "Starting: position = " << position << " (m)\n";
    position = step(position, velocity, timeStep);
    position = step(position, velocity, timeStep);
```

CT6. Tiếp tục sử dụng hàm

```
double step(double oldPosition,
            double velocity,
            double timeStep)
{
    double newPosition
        = getPosition(oldPosition, velocity, timeStep);
    cout << "Next step: position = "
         << newPosition << " (m)"  << endl;
    return newPosition;
}
```

CT6. Tiếp tục sử dụng hàm

```
double step(double oldPosition,
            double velocity,
            double timeStep)
{
    double newPosition
        = getPosition(oldPosition, velocity, timeStep);
    // display current position
    display(newPosition);  // <-- vẽ hình ở đây
    return newPosition;
}
```

Câu hỏi

- Về nội dung buổi học ?
- Về nội dung khóa học ?
- Về cách tổ chức ?