

Bài 1: Các Vấn Đề Cơ Bản Trong Ngôn Ngữ Lập Trình C++

Giảng viên:

MỤC TIÊU

- Một số khái niệm
- Môi trường làm việc
- Các bước phát triển phần mềm
- Cấu trúc một chương trình C++

Nội dung

- Một số khái niệm
- Môi trường làm việc
- Các bước phát triển phần mềm
- Cấu trúc một chương trình C++
- Bảng ký tự của C++
- Các từ khóa
- Tên gọi trong C++
- Một số quy tắc viết mã lệnh
- Lệnh xuất-nhập
- Kiểu dữ liệu cơ bản, biến, hằng

Một số khái niệm

Chương trình máy tính

Một loạt các câu lệnh mà nó chỉ thị cho máy tính giải quyết vấn đề như thế nào.

Dữ liệu

Những thông tin có thể được lưu trữ và xử lý bằng máy tính.

Lập trình

Thực hiện thiết kế, viết và bảo trì chương trình nhằm điều khiển máy tính làm việc.

Người lập trình

Người viết ra chương trình.

Một số khái niệm

Ngôn ngữ lập trình

Hệ thống các câu lệnh dùng để tạo thành chương trình.

Chương trình dịch

Chương trình dùng để chuyển đổi chương trình nguồn thành chương trình ngôn ngữ máy thực thi được.

Thời gian dịch

Thời gian thực hiện dịch chương trình nguồn.

Thời gian thực thi

Thời gian chạy chương trình ngôn ngữ máy.

Môi trường làm việc của C++

Để viết và chạy các chương trình C++, bạn cần phải có một **trình soạn thảo văn bản** và một **trình biên dịch C++** đã được cài trên máy tính.

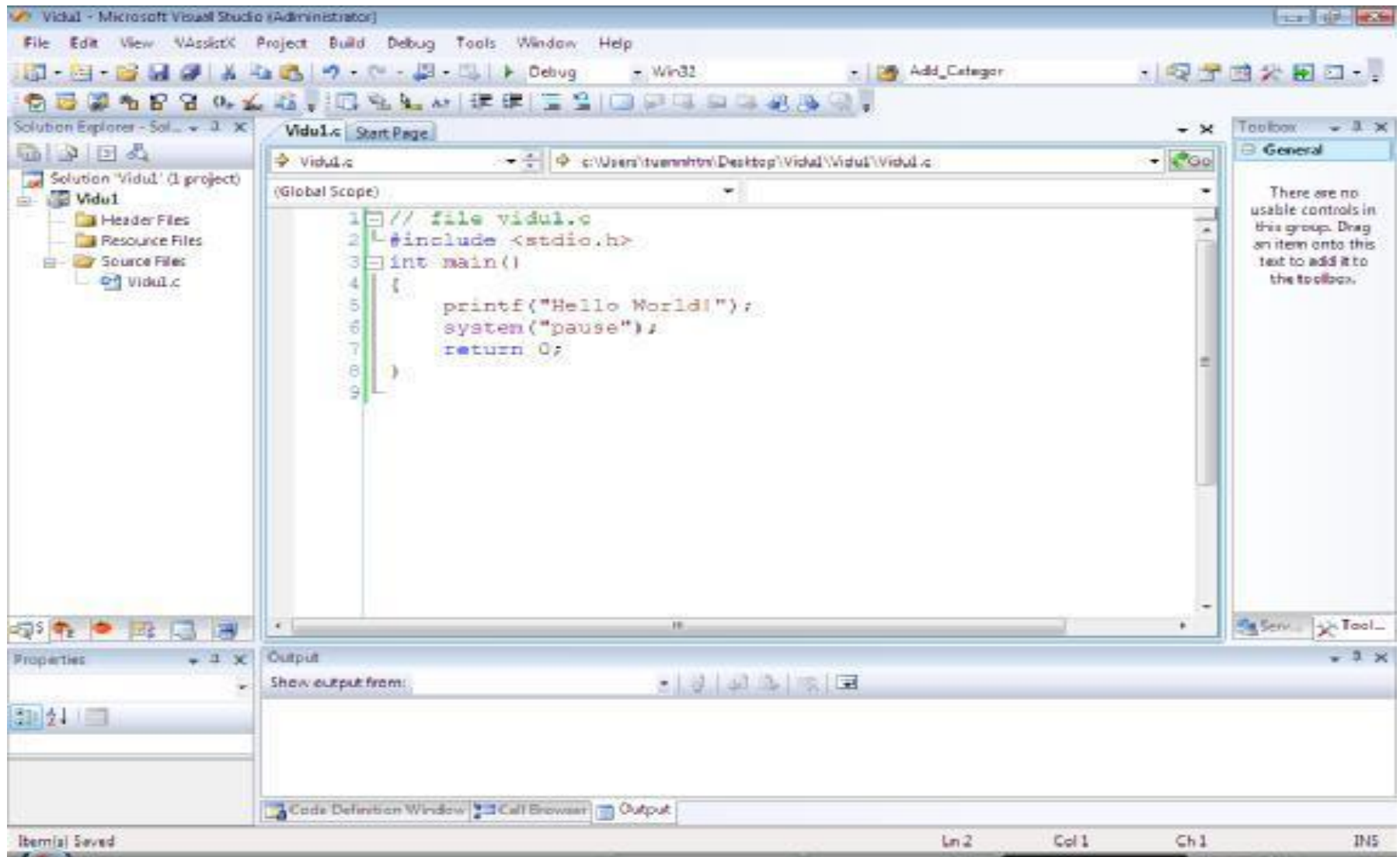
Một trình soạn thảo văn bản : là một hệ thống phần mềm cho phép tạo và soạn thảo các file văn bản trên máy tính.

Ví dụ: WordPad hoặc Notepad.

Một trình biên dịch: là một hệ thống phần mềm giúp dịch các chương trình sang ngôn ngữ máy (được gọi là *mã nhị phân*) mà hệ điều hành máy tính có thể chạy được.

Ví dụ: Borland C++ Builder, Metrowerks CodeWarrior, Microsoft Visual C++ (nằm trong Visual Studio).


Môi trường làm việc của Microsoft Visual C++





DEMO

Môi trường Visual
Microsoft C++

A white hand cursor icon, resembling a computer mouse pointer, is pointing at the letter 'O' in the word 'DEMO'. The hand is stylized with a white palm and fingers.

Các bước trong phát triển phần mềm

Thiết kế

Phân tích, đặc tả giải thuật để giải quyết vấn đề.

Viết mã lệnh

Viết đặc tả giải thuật bằng cú pháp của ngôn ngữ lập trình.

Kiểm tra, thực thi, phát hiện lỗi

Tìm tất cả những lỗi phát sinh và chỉnh sửa lại chương trình.

Bảo trì

Cập nhật, sửa đổi theo yêu cầu sử dụng.

Vấn đề

- Một em học sinh đi học từ nhà đến trường bằng xe đạp. Biết khoảng cách từ nhà đến trường là 5km. Tốc độ em học sinh đi là 2km/h. Hỏi thời gian em đi từ nhà đến trường mất bao nhiêu giờ?



Thực hiện

- Nhập quãng đường em học sinh đi: 5
- Nhập tốc độ em đi: 2
- Kết quả:
 - Thời gian em đi từ nhà đến trường là: 2.5 h

Giải thuật

- Xuất thông báo nhắc người sử dụng nhập quãng đường.
- Đọc giá trị thực từ bàn phím vào biến **quangduong**.
- Xuất thông báo nhắc người sử dụng nhập tốc độ.
- Đọc giá trị thực từ bàn phím vào biến **tocdo**.
- Tính thời gian **thoigian** = **quangduong/tocdo**.
- Xuất thời gian học sinh đã đi **thoigian**.

Viết mã, thực thi và kiểm tra

- Tạo khung chương trình
 - Tạo tập tin chương trình nguồn.
 - Thêm vào các chỉ dẫn biên dịch.
 - Thêm vào hàm `main()`.
- Chuyển đổi từng bước giải thuật thành mã lệnh
 - Thêm các khai báo cho những đối tượng chưa được khai báo.
 - Khai báo bao gồm tên và kiểu dữ liệu.
- Xem mã nguồn và kết quả

Viết mã

/*Chương trình tính thời gian đi học

nhập: quãng đường và tốc độ

xuất: thời gian đi đến trường*/

include <iostream> //cin, cout,<<,>>

//Thư viện iostream có chứa các câu lệnh vào ra như cin, cout,...

using namespace std;

//Theo chuẩn ANSI C++, tất cả định nghĩa của các lớp, đối tượng và hàm của thư viện

//chuẩn đều được định nghĩa trong namespace std

int main() //khởi tạo hàm main()

{

return 0;

}

Viết mã

/*Chương trình tính thời gian đi học

nhập: quãng đường và tốc độ

xuất: thời gian đi đến trường*/

```
# include <iostream> //cin,cout,<<,>>
```

```
//Thư viện iostream có chứa các câu lệnh vào ra như cin, cout,...
```

```
using namespace std;
```

```
int main( ) {
```

```
    //khai báo biến quãng đường, tốc độ
```

```
    double quangduong,tocdo;
```

```
    cout << "Moi ban nhap quang duong: "; cin >> quangduong;
```

```
    cout << "Moi ban nhap toc do: "; cin >> tocdo;
```

```
    cout << "Thoi gian hoc sinh di tu nha den truong: "<< quangduong/tocdo << " h";
```

```
    cout << endl;
```

```
    system ("pause");
```

```
    return 0;
```

```
}
```

Kiểm tra, thực thi, phát hiện lỗi

Moi ban nhap quang duong : 5

Moi ban nhap toc do : 2

Thoi gian hoc sinh di tu nha den truong: 2.5 (h)

Kiểm tra, thực thi, phát hiện lỗi

Các lỗi thường gặp:

- Lỗi vi phạm các qui tắc văn phạm của ngôn ngữ cấp cao (lỗi cú pháp).
- Lỗi xảy ra lúc thực thi chương trình (lỗi thời gian thực thi).
- Lỗi do thiết kế giải thuật (lỗi logic).

Bảo trì

- Các chương trình trong thực tế được sử dụng nhiều năm bởi nó là tài nguyên đã được đầu tư để xây dựng theo yêu cầu của khách hàng.
- Những yêu cầu mới phát sinh trong quá trình người dùng sử dụng chương trình.
- Cập nhật lại được gọi là "bảo trì".
- Ví dụ:

Khách hàng thay đổi yêu cầu tính thời gian đi của học sinh theo phút, thay vì tính theo giờ".

Lỗi nào dưới đây thuộc lỗi biên dịch?

Lỗi cú pháp

Lỗi chia cho 0

Lỗi logic

Tràn số



1.4. Cấu trúc chung của một chương trình C++

1. `#include <...>` → Chỉ dẫn biên dịch
 2. `using namespace std;` → Sử dụng không gian tên chuẩn std
 3. `void main ()`
 4. `{`
 5. `//chú thích`
 6. `//các câu lệnh ghi ở đây`
 7. `...`
 8. `}`
- Phần chính của chương trình chứa các lệnh C++

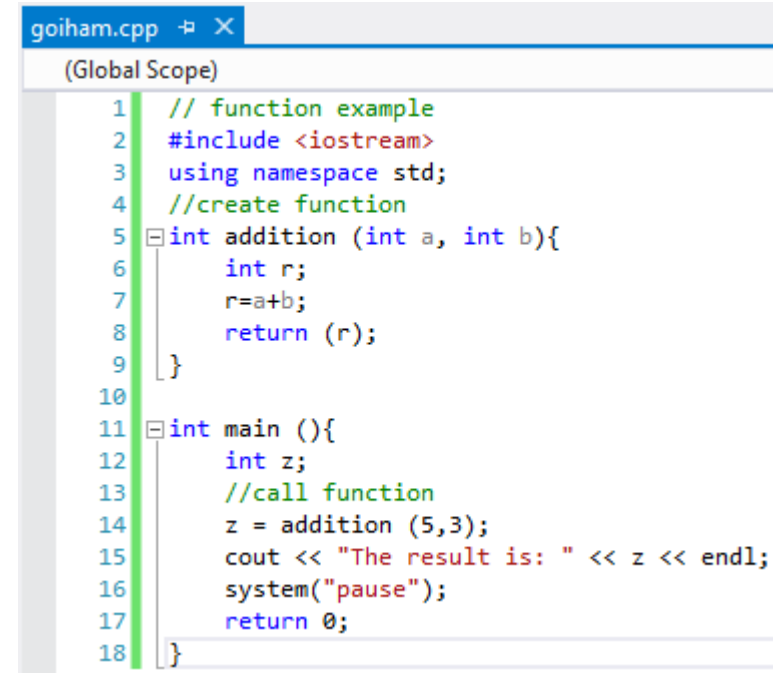
1.4. Cấu trúc khai báo các hàm

Kiểu_dữ_liệu Tên_hàm (Các tham số)

```
{  
    //các câu lệnh của hàm  
    .....  
    return (...); //trả về giá trị cho hàm  
}
```

Biên dịch chương trình: F6;

Chạy chương trình: F5;



The screenshot shows a code editor window titled 'goiham.cpp'. The code is written in C++ and demonstrates a function 'addition' and its use in a 'main' function. The code is as follows:

```
1 // function example  
2 #include <iostream>  
3 using namespace std;  
4 //create function  
5 int addition (int a, int b){  
6     int r;  
7     r=a+b;  
8     return (r);  
9 }  
10  
11 int main (){  
12     int z;  
13     //call function  
14     z = addition (5,3);  
15     cout << "The result is: " << z << endl;  
16     system("pause");  
17     return 0;  
18 }
```

CÁC THÀNH PHẦN CƠ BẢN TRONG C++

- Bảng ký tự của C++
- Các từ khóa
- Tên gọi trong C++
- Một số quy tắc viết mã lệnh
- Lệnh xuất-nhập

Bảng ký tự

Bộ ký tự dùng trong ngôn ngữ lập trình C++:

- Các chữ cái hoa: A, B,..., Z.
- Các chữ cái thường: a, b,..., z.
- Các chữ số: 0,...,9.
- Các ký hiệu đặc biệt khác: , ; : [], { }, #, dấu cách, ...
- Các ký hiệu toán học: +, -, *, /, %, &, ||, !, >, <, = ...
- Dấu gạch dưới _.

Các từ khoá

Một *từ khoá* trong một ngôn ngữ lập trình là một từ đã được định nghĩa trước và được dành riêng cho một mục đích sử dụng duy nhất trong các chương trình viết bằng ngôn ngữ đó.

Trong C++ chuẩn hiện nay có 74 từ khóa. Dưới đây sẽ là một số từ khóa thường gặp:

int, char, float, long, double, if, else, switch, case, while, do, for, return, break, struct, unsigned, void,...

Tên gọi

Để phân biệt các đối tượng với nhau chúng cần có một tên gọi. Tên có thể cho: chương trình, biến, hàm, hằng,... Dưới đây là một số quy tắc đặt tên.

Tên là các kí tự đứng liền nhau, gồm:

- Các chữ cái: A, ... , Z, a, ... , z.
- Các chữ số: 0, ... , 9.
- Dấu gạch dưới: _
- Không bắt đầu bằng số.
- Không chứa dấu cách trắng, dấu chấm câu.
- Không trùng với từ khoá.
- Không dùng dấu tiếng Việt.

Lưu ý: Trong C++ phân biệt chữ thường và chữ hoa.

Một số quy tắc viết mã lệnh

Câu lệnh và dấu chấm câu

Mỗi lệnh trong C++ được kết thúc bằng dấu chấm phẩy (;).

Ví dụ: `int x = 5;`

Trong chương trình có nhiều chỗ không dùng đến dấu ; vì đó không phải câu lệnh.

Ví dụ: `#include <conio.h>`

Khối lệnh

Một **khối lệnh** là một chuỗi lệnh được bao trong các dấu ngoặc nhọn { }.

Ví dụ: `{ int temp = x ; x = y; y = temp; }`

Một số quy tắc viết mã lệnh

Cách ghi lời giải thích

- Lời giải thích không có tác dụng tạo ra mã chương trình. Nó chỉ đơn giản là lời thuyết minh cho dễ hiểu
- Phần lời giải thích đặt trên 1 dòng:
 - Dùng cặp kí hiệu `//`
 - Mỗi cặp `//` chỉ có tác dụng từ sau `//` đến hết dòng
- Nếu muốn chú thích trên nhiều dòng thì dùng cặp ký tự: `/* */`

Ví dụ:

```
// Bắt đầu chương trình chính
void main ()
{
    ...
}
```

Tên biến nào sai qui định của ngôn ngữ C++?

Bien3

3bien

bien_3

a_3_bien

Lệnh Xuất-Nhập

Xuất dữ liệu ra màn hình

Cú pháp: `cout << < Tham số > ;`

Trong đó: Tham số có thể là hằng, biến, biểu thức, hàm, phần tử mảng, ... mà giá trị của nó cần hiển thị lên màn hình.

Ví dụ: `cout << "Gia tri cua x la" << x ;`

Công dụng: Dùng để in dữ liệu ra màn hình.

Lệnh Xuất-Nhập

Lệnh nhập dữ liệu

Cú pháp: `cin >> < Tham số > ;`

Tham số: chỉ có thể là biến.

Ví dụ:

```
cout << "\nNhap x= ";  
cin >> x;
```

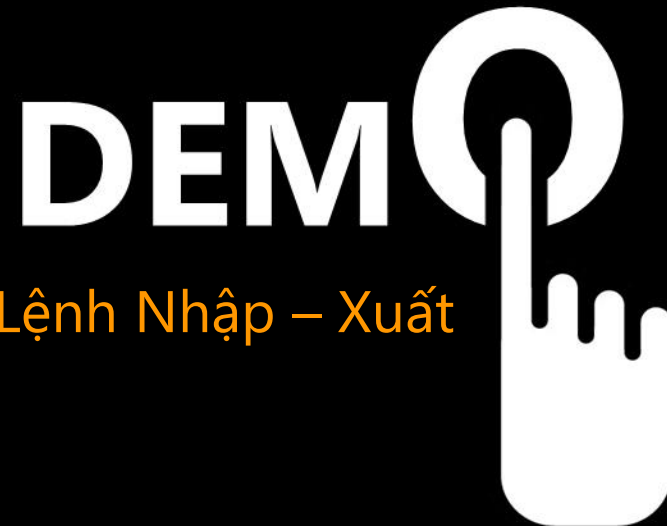
Công dụng: Dùng để đọc các kí tự, số, chuỗi ký tự từ bàn phím, chuyển dịch và lưu nó vào một ô nhớ xác định.

Chú ý:

Khi dùng hàm `cout` và `cin` phải khai báo.

`#include <iostream>`

`using namespace std;`



Lệnh Nhập – Xuất

KIỂU DỮ LIỆU, BIẾN, HẲNG

- Tìm hiểu các kiểu dữ liệu cơ bản của C++
- Tìm hiểu về biến và hằng

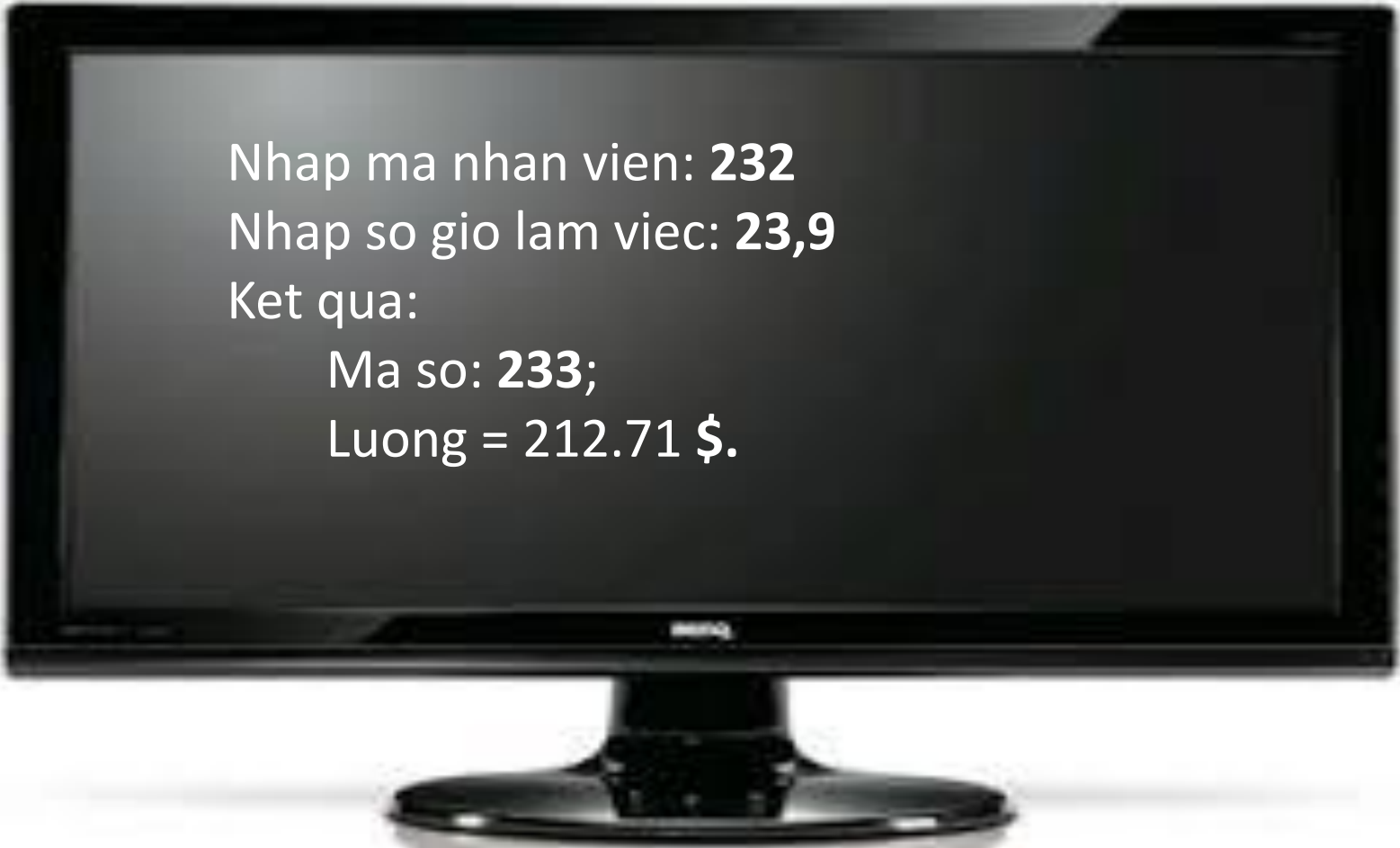
Vấn đề

Viết chương trình để thực hiện các tác vụ sau:

- Tính lương cho mỗi nhân viên.
- Tất cả nhân viên đều được trả 8.9 \$ mỗi giờ.



Thực hiện

A black computer monitor with a silver stand, displaying white text on its screen. The text is centered and reads: 'Nhap ma nhan vien: 232', 'Nhap so gio lam viec: 23,9', 'Ket qua:', 'Ma so: 233;', and 'Luong = 212.71 \$.'.

Nhap ma nhan vien: **232**
Nhap so gio lam viec: **23,9**
Ket qua:
Ma so: **233;**
Luong = 212.71 \$.

Giải thuật

- Khai báo hằng **luonggio** = 8.9.
- Xuất thông báo nhắc người sử dụng nhập mã nhân viên.
- Đọc giá trị nguyên từ bàn phím vào biến **ma_nv**.
- Xuất thông báo nhắc người sử dụng nhập số giờ làm việc.
- Đọc giá trị thực từ bàn phím vào biến **giolamviec**.
- Tính lương **pay** = **giolamviec** * **luonggio**.
- Xuất mã nhân viên **ma_nv** và lương **pay**.

Viết mã, thực thi và kiểm tra

- Tạo khung chương trình
Tạo tập tin chương trình nguồn.
Thêm vào các chỉ dẫn biên dịch.
Thêm vào hàm `main()`.
- Chuyển đổi từng bước giải thuật thành mã lệnh
Thêm các khai báo cho những đối tượng chưa được khai báo.
Khai báo bao gồm tên và kiểu dữ liệu.
- Xem mã nguồn và kết quả

```
#include <iostream>
using namespace std;
//chuong trinh tinh luong theo gio
int main()
{
    //khai bao hang luong gio lam viec

    double const gioluong=8.9;
    double pay;
    double giolamviec;
    int ma_nv;
    cout << "Moi ban nhap ma nhan vien: " ;
    cin >> ma_nv ;
    cout << "Moi ban nhap gio lam viec: " ;
    cin >> giolamviec;
    cout << "Ma nhan vien: " << ma_nv << endl;
    cout << "Luong : " << gioluong*giolamviec << endl;
    system ("pause");
    return 0;
}
```

C:\Users\hieunh3\Documents\Visual Stu...

```
Moi ban nhap ma nhan vien: 232
Moi ban nhap gio lam viec: 23.9
Ma nhan vien: 232
Luong : 212.71
Press any key to continue . . .
```

Các kiểu dữ liệu cơ bản

Kiểu dữ liệu cơ bản: Là những kiểu dữ liệu đơn giản, không có cấu trúc thường được các ngôn ngữ lập trình cấp cao xây dựng sẵn như một thành phần của ngôn ngữ để giảm nhẹ công việc cho người lập trình. Các kiểu dữ liệu cơ bản gồm:

- Kiểu số nguyên (số dương, âm): **int**
- Các biến thể của kiểu số nguyên:
short, long, unsigned.
- Kiểu số thực (phân số):
float, double, long double.
- Kiểu ký tự (chữ cái, chữ số, ký hiệu, ...): **char.**
- Kiểu nhị phân (các giá trị logic đúng hay sai):
 - **int/unsigned** hay **bool** (trong Visual C++)

Kiểu Số Nguyên

Từ khoá	Số byte	Miền giá trị
unsigned char	1 byte (8 bits)	Từ 0 đến 255
char	1 byte (8 bits)	Từ -128 đến 127
enum	2 bytes (16 bits)	Từ -32,768 đến 32,767
unsigned int	2 bytes (16 bits)	Từ 0 đến 65,535
short int	2 bytes (16 bits)	Từ -32,768 đến 32,767
int	2 bytes (16 bits)	Từ -32,768 đến 32,767
unsigned long	4 byte (32 bit)	Từ 0 đến 4,294,967,295
long	4 byte (32 bit)	Từ -2,147,483,648 đến 2,147,483,647

Kiểu số thực

Từ khoá	Số byte	Miền giá trị
float	32 bit (4 byte)	Từ $3.4 * 10^{-38}$ đến $3.4 * 10^{38}$
double	64 bit (8 byte)	Từ $1.7 * 10^{-308}$ đến $1.7 * 10^{308}$
long double	80 bit (10 byte)	Từ $3.4 * 10^{-4932}$ đến $1.1 * 10^{4932}$

Kiểu dữ liệu int (kiểu số nguyên) có thể xử lý số nguyên nằm trong khoảng nào?

0...255

-128...127

-32768...32767

0...65535

Kiểu nhị phân

Kiểu **nhị phân** là một kiểu nguyên mà các biến thuộc kiểu này chỉ có thể có hai giá trị là: **false** và **true**. Các giá trị này được lưu trữ dưới dạng số nguyên **0** và **1**. Kiểu nguyên trong C++ chuẩn được đặt tên là **bool**.

Ví dụ: Các biến nhị phân

```
int main()  
{ // in ra giá trị của một biến nhị phân  
    bool flag = false;  
    cout << "flag = " << flag << endl;  
    flag = true;  
    cout << "flag = " << flag << endl;  
}  
flag = 0  
flag = 1
```

Kiểu liệt kê

Kiểu liệt kê là một kiểu nguyên được định nghĩa bởi người sử dụng với cú pháp:

```
enum typename { enumerator-list };
```

Ở đây **enum** là từ khóa trong C++, *typename* là định danh dùng để đặt tên kiểu được định nghĩa, và *enumerator-list* là danh sách các tên của các hằng số kiểu nguyên.

Ví dụ:

Cú pháp sau định nghĩa kiểu liệt kê tên là **Semester**, xác định ba giá trị mà một biến thuộc kiểu này có thể có:

```
enum Semester {FALL, SPRING, SUMMER};
```

Kiểu kí tự

- Kiểu **char**.
- Được thể hiện bằng các ký tự riêng biệt
 - Xem bảng mã ASCII
- Các ký tự được thể hiện trong bộ nhớ bằng giá trị nguyên một byte.
- Hằng ký tự:
 - Các ký tự đặt trong cặp dấu nháy đơn (' ').
 - Ví dụ: 'X', '7', '>', 'e'

Các ký tự điều khiển

Dấu \ được kết hợp với ký tự khác thành chuỗi ký tự điều khiển có ý nghĩa đặc biệt:

Ký tự	Chuỗi ký tự điều khiển
Xuống dòng	\n
Tab ngang	\t
Phím backspace	\b
Trở về đầu dòng	\r
Tiếng beep	\a

Chuỗi ký tự

Có quan hệ với ký tự:

- Là một dãy các ký tự.
- Được đặt trong cặp dấu nháy kép.

Ví dụ: "Nice to meet you!"

Có thể chứa các ký tự điều khiển: "\nThe answer is "

Lưu ý:

- "A" là một hằng chuỗi.
- 'A' là một hằng ký tự.



DEMO

Các kiểu dữ liệu



Biến

Khái niệm

Là một đại lượng có thể thay đổi giá trị khi thực hiện chương trình.

Khai báo

Cú pháp: <KieuDuLieu> <TenBien> ;

- Trong đó **TenBien** được đặt tên theo **qui tắc đặt tên**.
- Muốn khai báo từ 2 biến trở lên có cùng kiểu dữ liệu thì các tên biến đặt cách nhau bởi dấu phẩy (,).

Ví dụ:

float y;

float a, b;

Biến

Vị trí khai báo biến: Các biến phải được khai báo trước khi sử dụng.

Khai báo biến ngoài: Biến được khai báo sau khi khai báo thư viện.

Do đó biến có thể được sử dụng bất kỳ đâu trong chương trình (**biến công cộng** hay **toàn cục**).

Ví dụ:

```
#include<iostream.h>
int x;
void main( )
{
    ....
}
```

Biến

Khai báo biến cục bộ: Các biến được đặt trong hàm hoặc trong khối lệnh. Biến chỉ có tác dụng trong hàm hoặc trong khối lệnh

Ví dụ:

```
void InSo()  
{  
    int i = 100;  
    cout << " i = " << i;  
}
```

Trong ví dụ trên biến **i** chỉ có tác dụng trong hàm **InSo**.

Khai báo đối số hàm

Vị trí của biến đặt trong phần định nghĩa tham số hàm.

Ví dụ:

```
int tong ( int x, int y)  
{  
    return (x + y);  
}
```

Biến

Gán và lấy giá trị

Mỗi biến đều có kiểu dữ liệu. Nếu không gán giá trị ban đầu cho biến thì biến sẽ có giá trị mặc định trong miền kiểu dữ liệu (KDL) được khai báo.

Khai báo và gán: **<KDL> <TenBien> = <GiaTri>;**

Ví dụ 1:

```
int    x;  
cout << "x = " << x;
```

Ví dụ 2:

```
long   a = 150;  
cout << "a = " << a;
```

Hằng

Khái niệm và khai báo

Hằng là những đại lượng mà giá trị của nó không thay đổi trong quá trình tính toán.

Khai báo với từ khoá **const**

```
const <KDL> <TenHang> = <GiaTri>;
```

Khai báo với **#define**: **#define TenHang GiaTri**

Ví dụ:

```
const float pi = 3.14;  
#define hundred 100
```

Cách sử dụng

Sau khi khai báo hằng, trong chương trình khi sử dụng **TenHang** sẽ được thay bằng **GiaTri**

Ví dụ:

```
const float hundred = 100;  
...  
cout << "Gia = " << 5*hundred;
```

Để định nghĩa một hằng nguyên $ABC = 10$ câu lệnh nào dưới đây là đúng?

`const ABC=10;`

`int ABC=10;`

`#define ABC 10`

`#define ABC =10`



DEMO

Biến & hằng



Biểu thức và các phép toán

Biểu thức

Khái niệm

Biểu thức là công thức tính toán để có một giá trị theo một qui tắc toán học nào đó.

Ví dụ:

$$8 * 3 + 4;$$

$$x + \sin(y);$$

...

Các kiểu biểu thức

Biểu thức số học: là biểu thức tính ra giá trị bằng số.

Biểu thức quan hệ: Là biểu thức chứa toán tử quan hệ
($<$, $>$, $<=$, $>=$, $=$, $!=$).

Biểu thức nhị phân (logic): Là biểu thức về nguyên tắc có giá trị đúng (**True**) hoặc sai (**False**).

Biểu thức và các phép toán

Phép gán: **biến** = **biểu thức**;

Ví dụ:

```
long x, y;  
x = 100;  
y = x + 200;
```

Các phép toán gồm:

Các phép toán số học

$+$, $-$, $*$, $/$, $\%$

Các phép toán quan hệ

$<$, $>$, $<=$, $>=$, $==$, $!=$

Các phép toán nhị phân (logic)

$\&\&$, $\|\|$, $!$

Các toán tử gán kết hợp

$+=$, $-=$, $*=$, $/=$, và $\%=$.

3.4. Biểu thức và các phép toán

Các phép tăng, giảm một giá trị

Phép toán tăng:

$i = i + 1$ được viết: $i++$ hoặc $++i$

Phép toán giảm:

$i = i - 1$ được viết: $i--$ hoặc $--i$

Sự khác nhau giữa $++i$ và $i++$

$++i$: i được tăng trước khi gán.

$i++$: i được gán trước khi tăng.

Ví dụ: cho $x = 3$

$a = x++$; $a = ++x$;

$a = x--$; $a = --x$;

Cho $b = 5$ và $c = 8$. Hãy cho biết giá trị của a sau khi thi hành dòng lệnh sau ? $a = b++ + c++;$

13

14

15

16

Chuyển kiểu

Chuyển kiểu

Là cách mà một số có thể được chuyển tự động thành một kiểu khác.

Cú pháp: (kiểu) (biểu thức)

Ví dụ:

```
cout << 15/ 4;
```

```
cout << 15/ (float) (4);
```

Chuyển kiểu

Ép kiểu

Ví dụ :

```
int main()  
{ // ép kiểu double thành kiểu int:  
  double v = 1234.56789;  
  int n = int(v);  
  cout << "v = " << v << ", n = " << n << endl;  
}  
//kết quả:  
v = 1234.57, n = 1234
```

Giá trị **double** 1234.56789 được chuyển thành giá trị **int** 1234.

Chương trình trên đã minh họa việc ép giá trị kiểu **double** thành giá trị kiểu **int**

Chuyển kiểu

Nâng kiểu

Khi một kiểu dữ liệu được chuyển lên kiểu **cao hơn**, thì không cần dùng đến toán tử ép kiểu. Đây gọi là **nâng kiểu**.

Ví dụ:

```
int main ()
{ // in ra giá trị nâng kiểu của 65 từ kiểu char lên double:
  char c='A'; cout << " char c = " << c << endl;
  short k=c;   cout << " short k = " << k << endl;
  int m=k;     cout << " int m = " << m << endl;
  long n=m;    cout << " long n = " << n << endl;
  float x=m;   cout << " float x = " << x << endl;
  double y=x;  cout << "double y = " << y << endl;
}
//kết quả
char c = A; short k = 65; int m = 65;
long n = 65; float x = 65; double y = 65
```

Chương trình trên đã minh họa việc nâng một biến **char** lên các kiểu **short, int, float, double**



DEMO

Biểu thức,
Phép toán,
Chuyển kiểu



Tràn số

Ví dụ

```
int main()  
{  
    // in ra n cho tới khi nó bị tràn:  
    int n = 1000;  
    cout << "n = " << n << endl;  
    n *= 1000; // nhân n với 1000  
    cout << "n = " << n << endl;  
    n *= 1000; // nhân n với 1000  
    cout << "n = " << n << endl;  
    n *= 1000; // nhân n với 1000  
    cout << "n = " << n << endl;  
}
```

//kết quả

n = 1000; n = 1000000;

n = 1000000000; n = -727379968

Đoạn mã in đậm cho thấy máy tính chạy chương trình này không thể thực hiện đúng phép nhân **1,000,000,000** với **1000**.

Chương trình trên đã minh họa việc lặp phép nhân **n** với **1000** cho đến khi xảy ra tràn số.

Lỗi làm tròn số

Ví dụ:

```
int main()  
{ // mô phỏng lỗi làm tròn  
    double x = 1000/3.0; cout << "x = " << x << endl; // x = 1000/3  
    double y = x-333.0; cout << "y = " << y << endl; // y = 1/3  
    double z = 3*y - 1.0; cout << "z = " << z << endl; // z = 3(1/3) - 1  
    if (z == 0) cout << "z == 0.\n";  
    else cout << "z does not equal 0.\n"; // z != 0  
}
```

//kết quả

x = 333.333; y = 0.333333;

z = -5.68434e-14 ; z does not equal 0.

Trong lý thuyết số học, các biến sẽ có giá trị $x = 333 \frac{1}{3}$, $y = \frac{1}{3}$, và $z = 0$.

Nhưng $\frac{1}{3}$ không thể được biểu diễn một cách chính xác bằng số dấu phẩy động. Sự sai lệch được phản ánh trong giá trị dư của **z**.

Chương trình trên đã minh họa một số phép tính đơn giản mô phỏng lỗi làm tròn số.

Kết luận

- Một số khái niệm
- Môi trường làm việc
- Các bước phát triển phần mềm
- Cấu trúc một chương trình C++
- Bảng ký tự của C++
- Các từ khóa
- Tên gọi trong C++
- Một số quy tắc viết mã lệnh
- Lệnh xuất-nhập
- Kiểu dữ liệu cơ bản, biến, hằng

Chuẩn bị bài sau

Sinh viên đọc sách và slide trước bài học kế tiếp về cấu trúc điều khiển gồm:

- Lệnh lựa chọn
- Lệnh lặp



FPT POLYTECHNIC

THANK YOU!

www.poly.edu.vn