

Tổng quan về lập trình

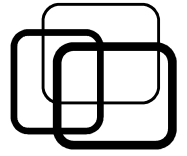
GV. Nguyễn Minh Huy



- Giới thiệu môn học.
- Khái niệm cơ bản về lập trình.
- Các ngôn ngữ lập trình.
- Môi trường lập trình.



- Giới thiệu môn học.
- **Khái niệm cơ bản về lập trình.**
- Các ngôn ngữ lập trình.
- Môi trường lập trình.



■ Khái niệm lập trình:

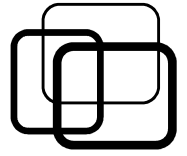
■ Bài toán:

- Dạy cách tính tổng số lớn nhất và số nhỏ nhất giữa 7, 1, 9.
- Ràng buộc:
 - Người học biết: phép cộng và phép so sánh giữa 2 số.
 - Liệt kê từng bước.

■ Các bước dạy:

- B1: so sánh 7 và 1 \Rightarrow 7 lớn hơn.
- B2: so sánh 7 và 9 \Rightarrow 9 lớn hơn \Rightarrow 9 lớn nhất.
- B3: so sánh 7 và 1 \Rightarrow 1 nhỏ hơn.
- B4: so sánh 1 và 9 \Rightarrow 1 nhỏ hơn \Rightarrow 1 nhỏ nhất.
- B5: cộng 9 và 1.

Khái niệm cơ bản về lập trình



■ Lập trình là gì?

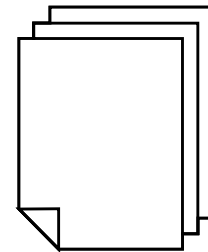
- Máy tính là một “đứa trẻ ngốc”.
- Cần phải dạy máy tính làm việc.
- Lập trình ~ giải toán:
 - Từng bước cụ thể.
 - Từ các lệnh có sẵn.
- Bảng mô tả lời giải → chương trình máy tính.



Máy tính

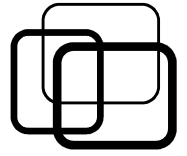


Lập trình



Chương trình

Khái niệm cơ bản về lập trình

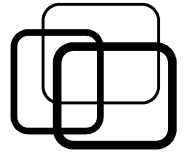


■ Chương trình máy tính:

- Máy tính hiểu sẵn một số lệnh cơ bản.
- Dạy máy tính từ những lệnh này.
- Chương trình:
 - Một tập hợp các lệnh cơ bản.
 - Sắp xếp theo trình tự.
 - Giải quyết một vấn đề.



Khái niệm cơ bản về lập trình



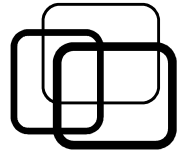
■ Lập trình viên:

- Người tạo ra chương trình máy tính.
- Programmer vs. coder vs. developer.
- Ai phù hợp làm lập trình viên?
 - Người có đầu óc logic.
 - Người thích loay hoay tìm tòi.
 - Người tỉ mỉ và cẩn trọng.
 - Phụ nữ!



Ada Lovelace

Khái niệm cơ bản về lập trình



■ Các loại mã lập trình:

■ Mã máy (machine code):

- Các dãy số '0' và '1'.
- Máy hiểu, người khó học!

■ Mã giả (pseudo code):

- Ngôn ngữ tự nhiên.
- Người hiểu, máy không hiểu!

■ Mã nguồn (source code):

- Ngôn ngữ lập trình.
 - Các lệnh ngắn gọn, cú pháp rõ ràng.
 - C, Java, Python, ...
- Trình biên dịch (compiler):
 - Chương trình dịch mã nguồn → mã máy.
 - Ai viết trình biên dịch?

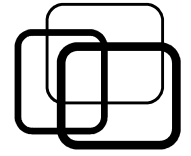


B1: cộng a và b.
B2: nhân a và c.
B3: so sánh c và d.
...

```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```


Khái niệm cơ bản về lập trình



■ Quy trình lập trình:

■ **Viết** chương trình (write):

- Soạn thảo mã nguồn.
- Dùng trình soạn thảo.

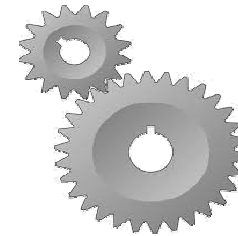


```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```

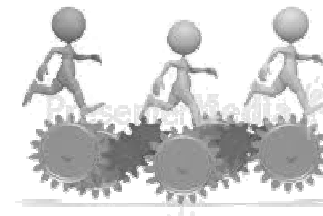
■ **Dịch** chương trình (compile):

- Mã nguồn → mã máy.
- Dùng trình biên dịch.



■ **Chạy** chương trình (run):

- Thực hiện mã máy đã dịch.
- Dùng run của hệ điều hành.



■ **Sửa lỗi** chương trình (debug):

- Tìm lỗi sai trong mã nguồn.
- Dùng trình gỡ rối (debugger).

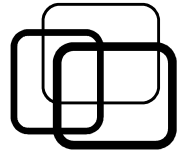


```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```



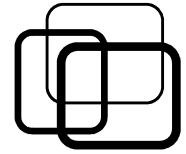
- Giới thiệu môn học.
- Khái niệm cơ bản về lập trình.
- **Các ngôn ngữ lập trình.**
- Môi trường lập trình.



■ Ngôn ngữ lập trình:

- Ngôn ngữ trung gian giữa người và máy.
- Ngắn gọn, chính xác và rõ ràng.
- Học ngôn ngữ lập trình là chuyện nhỏ!!
 - ➔ Học tư duy lập trình mới khó!!

Các ngôn ngữ lập trình



■ Thời kỳ đầu:

- 1950s: UNIVAC, IBM 701.

■ Ngôn ngữ cấp thấp:

- Ngôn ngữ thể hệ 1 → mã máy.
- Ngôn ngữ thể hệ 2 → hợp ngữ.

■ Đặc điểm:

- Tập lệnh đơn giản.
- Gần ngôn ngữ máy → khó học.
- Lập trình tốn công sức.
- Chương trình nhỏ gọn, nhanh.
- Can thiệp sâu hệ thống.

```
fib:
    mov edx, [esp+8]
    cmp edx, 0
    ja @f
    mov eax, 0
    ret

@@:
    cmp edx, 2
    ja @f
    mov eax, 1
    ret

@@:
    push ebx
    mov ebx, 1
    mov ecx, 1

@@:
    lea eax, [ebx+ecx]
    cmp edx, 3
    jbe @f
    mov ebx, ecx
    mov ecx, eax
    dec edx
    jmp @b

@@:
    pop ebx
    ret
```

Các ngôn ngữ lập trình



■ Thời kỳ phát triển:

- Những năm 1960s – 1970s.
- E. Dijkstra → lập trình cấu trúc.
- Ngôn ngữ cấp cao:

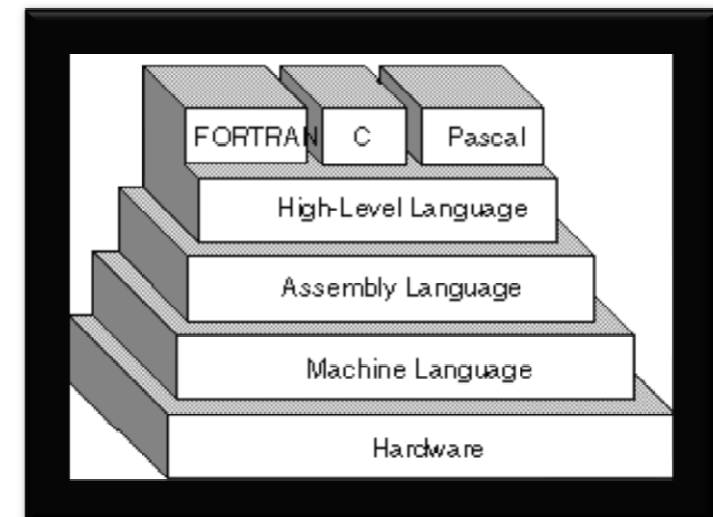
➢ Ngôn ngữ thế hệ 3: FORTRAN, ALGOL, C, Pascal, ...

■ Đặc điểm:

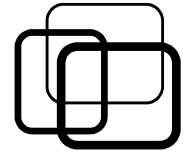
- Tập lệnh mở rộng, chặt chẽ.
- Gần ngôn ngữ tự nhiên → dễ học.
- Lập trình đỡ tốn công sức.
- Chương trình lớn hơn, chậm hơn.
- Ít can thiệp sâu hệ thống.

```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```



Các ngôn ngữ lập trình



■ Hiện nay:

■ Lập trình hệ thống:

- C, C++, Assembly, ...

■ Ứng dụng desktop:

- C, C++, Java, C#, Python, ...

■ Ứng dụng web:
























- JavaScript, Java, C#, Python, ...

■ Hướng đối tượng:

- C++, Java, C#, Python, ...

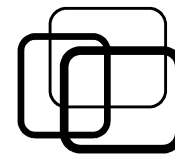
■ Đa nền tảng:

- Java, C#, ...

Language Rank	Types
1. Python	 
2. C	  
3. Java	  
4. C++	  
5. C#	  
6. R	
7. JavaScript	 
8. Go	 
9. Swift	 
10. Ruby	 



- Giới thiệu môn học.
- Khái niệm cơ bản về lập trình.
- Các ngôn ngữ lập trình.
- **Môi trường lập trình.**



■ Môi trường lập trình tích hợp:

■ Viết chương trình:

- Chương trình soạn thảo (editor).

■ Dịch chương trình:

- Trình biên dịch (compiler).

■ Chạy chương trình:

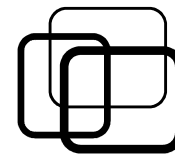
- Run program của hệ điều hành.

■ Sửa lỗi chương trình:

- Trình gỡ rối (debugger).

➔ Giải pháp tích hợp: IDE
(Integrated **D**evelopment **E**nvironment)

Môi trường lập trình



■ Môi trường lập trình C/C++:

Công cụ	Windows	MacOS	Linux
Editor	VS Code Atom Notepad++	VS Code Atom Vim, Emacs	VS Code Atom Vim, Emacs
Compiler	MSVC (*) GCC (**) Clang (**)	Clang GCC	GCC Clang
Debugger	MSVC (*)	LLVM/LLDB, GDB	GDB, LLVM/LLDB
IDE	Visual Studio (*) Code::Blocks CodeLite	Xcode (*) Code::Blocks CodeLite	Code::Blocks CodeLite

(*): Phần mềm độc quyền, mã nguồn đóng.

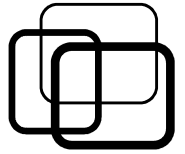
(**): Phải chạy trên nền tảng giả lập MinGW, Cygwin, hoặc WSL.



■ Môi trường lập trình C/C++:

■ Demo:

- Soạn thảo chương trình.
- Dịch chương trình.
- Chạy chương trình.
- Sửa lỗi chương trình.



■ Lập trình:

- Mô tả cụ thể các bước giải quyết bài toán.

■ Chương trình:

- Bảng mô tả các bước giải.
- Ở 3 dạng: mã máy, mã giả, mã nguồn.

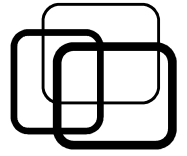
■ Ngôn ngữ lập trình:

- Ngôn ngữ trung gian giữa người và máy.
- Trình biên dịch: dịch ra cho máy hiểu.

■ Môi trường lập trình:

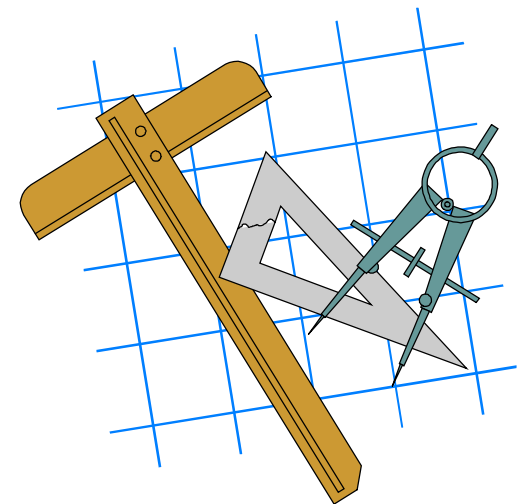
- Editor + Compiler + Runner + Debugger.

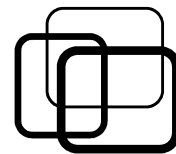




■ Bài tập 1.1:

Lựa chọn và cài đặt môi trường lập trình C/C++ trên máy tính của bạn.





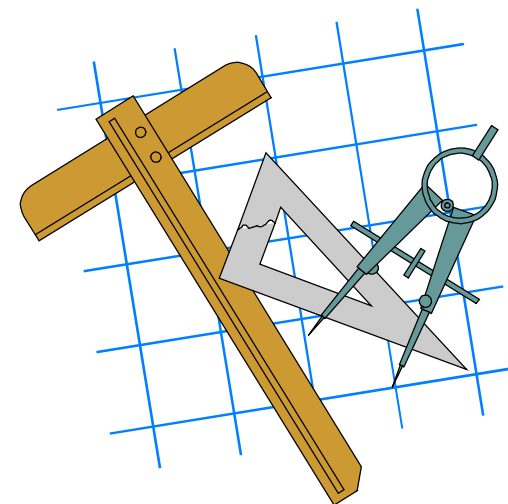
■ Bài tập 1.2:

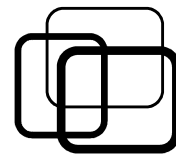
Viết và chạy thử chương trình sau:

- a) Chương trình in gì ra màn hình?
- b) Thay chữ “Hello World” trong chương trình bằng chữ “Chao mung”, chạy lại chương trình, màn hình có gì thay đổi?

```
#include <stdio.h>
```

```
void main()  
{  
    printf(“Hello World\n”);  
}
```





■ Bài tập 1.3:

Viết và chạy thử chương trình sau:

Chương trình yêu cầu nhập gì và xuất kết quả gì ra màn hình?

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    int  a, b, c, max;
```

```
    printf("Nhập a, b, c = ");
```

```
    scanf("%d %d %d", &a, &b, &c);
```

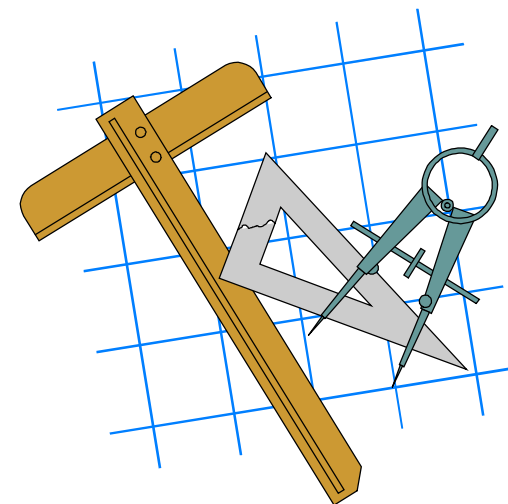
```
    max = a;
```

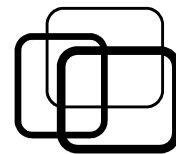
```
    if (b > max) max = b;
```

```
    if (c > max) max = c;
```

```
    printf("max = %d", max);
```

```
}
```





■ Bài tập 1.4:

Viết và chạy thử chương trình sau:

Chương trình yêu cầu nhập gì và xuất kết quả gì ra màn hình?

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    int    n;
```

```
    long   s;
```

```
    printf("Nhập n = ");
```

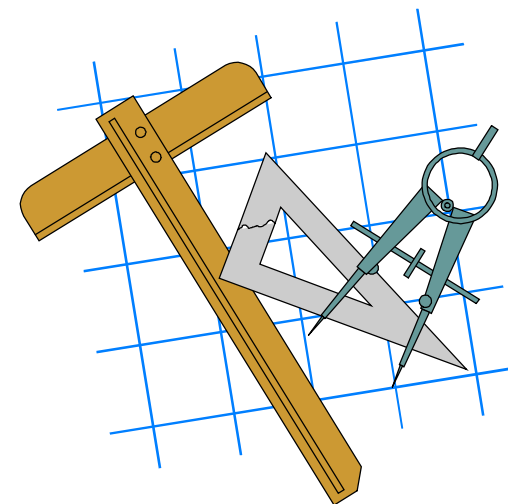
```
    scanf("%d", &n);
```

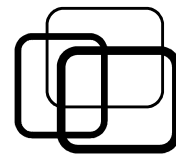
```
    for (s = 1; n > 0; n--)
```

```
        s = s * n;
```

```
    printf("s = %d ", s);
```

```
}
```





■ Bài tập 1.5:

Debug chương trình ở bài 1.3 và 1.4.

- a) Ở bài 1.3, nhập $a = 1$, $b = 5$, $c = 3$, hãy cho biết max lần lượt nhận những giá trị nào khi chương trình thực hiện.
- b) Ở bài 1.4, nhập $n = 10$, hãy cho biết giá trị của s khi $n = 3$.

