

Chương 1: Tổng quan về kỹ thuật lập trình

Thông tin giảng viên

- PGS.TS. Lê Đức Hậu
- Khoa KHMT, 603 B1
- Email: hauld@soict.hust.edu.vn
- ĐT: 0912324564

- Teams code:

- zsllspd



Mục tiêu môn học?

- Học phần *Kỹ thuật lập trình* trang bị cho sinh viên những kỹ thuật cơ bản nhất mà một lập trình viên chuyên nghiệp cần phải nắm vững để viết mã nguồn hiệu quả.
 - Các kiến thức giảng dạy góp phần quan trọng giúp sinh viên phát triển được các ứng dụng phần mềm chất lượng cao trong thực tế.
- Học phần này trang bị cho sinh viên các kỹ thuật lập trình quan trọng như quản lý bộ nhớ, hàm, kỹ thuật đệ quy, kỹ thuật sử dụng các cấu trúc dữ liệu để giải quyết vấn đề, kỹ thuật viết mã nguồn hiệu quả, kỹ thuật lập trình phòng ngừa, kỹ thuật gỡ rối, tinh chỉnh mã nguồn, phong cách lập trình.
 - Học phần có các buổi thực hành nhằm rèn luyện và nâng cao kỹ năng lập trình của sinh viên.

Tài liệu học tập

- [1] Bài giảng trên lớp
- [2] Trần Đan Thư (2014). Kỹ thuật lập trình. NXB Khoa học và kỹ thuật
- [3] McConnell, Steve (2004). Code Complete: A Practical Handbook of Software Construction, 2d Ed. Redmond, Wa.: Microsoft Press.
- [4] Kernighan & Plauger (1978). The elements of programming style. McGraw-Hill; 2nd edition
- [5] Brian W. Kernighan and Rob Pike (1999). The Practice of Programming. Addison-Wesley; 1st Edition
- [6] Nicolai M. Josuttis. The C++ Standard Library: A Tutorial and Reference (2nd Edition), 2012.

Đánh giá học phần

Điểm thành phần	Phương pháp đánh giá cụ thể	Mô tả	Tỷ trọng
A1. Điểm quá trình	Đánh giá quá trình		40%
	Bài thực hành		20%
	Thi giữa kỳ	Tự luận và/ hoặc trắc nghiệm	20%
A2. Điểm cuối kỳ	Thi cuối kỳ	Tự luận và/ hoặc trắc nghiệm	60%

Tổng quan về lập trình

Hoạt động của chương trình máy tính và ngôn ngữ lập trình

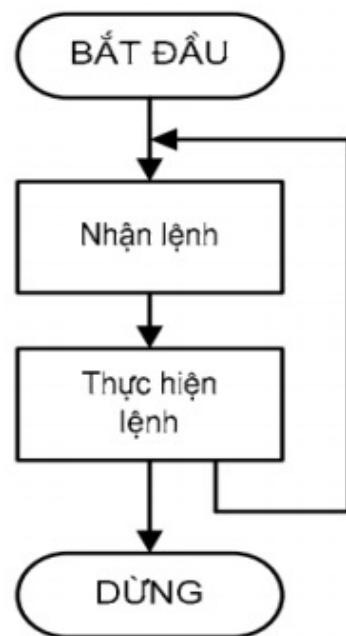
Chương trình máy tính và ngôn ngữ lập trình

- Chương trình máy tính: Tập hợp các lệnh chỉ dẫn cho máy tính thực hiện nhiệm vụ
- Ngôn ngữ lập trình: Dùng để viết các lệnh, chỉ thị



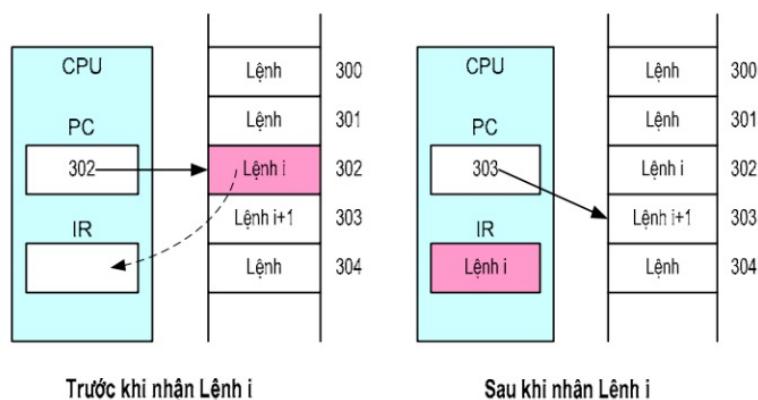
Hoạt động của chương trình máy tính

- Chương trình máy tính được nạp vào bộ nhớ chính (primary memory) như là một tập các lệnh viết bằng ngôn ngữ mà máy tính hiểu được, tức là một dãy tuần tự các số nhị phân (binary digits).
- Tại bất cứ một thời điểm nào, máy tính sẽ ở một trạng thái (state) nào đó. Đặc điểm cơ bản của trạng thái là con trỏ lệnh (instruction pointer) trỏ tới lệnh tiếp theo để thực hiện.
- Thứ tự thực hiện các nhóm lệnh được gọi là luồng điều khiển (flow of control).



Hoạt động của chương trình máy tính

- Bắt đầu mỗi chu trình lệnh, CPU nhận lệnh từ bộ nhớ chính.
 - PC (Program Counter): thanh ghi giữ địa chỉ của lệnh sẽ được nhận
 - Lệnh được nạp vào thanh ghi lệnh IR (Instruction Register)
- Sau khi lệnh được nhận vào, nội dung PC tự động tăng để trỏ sang lệnh kế tiếp



Ngôn ngữ lập trình

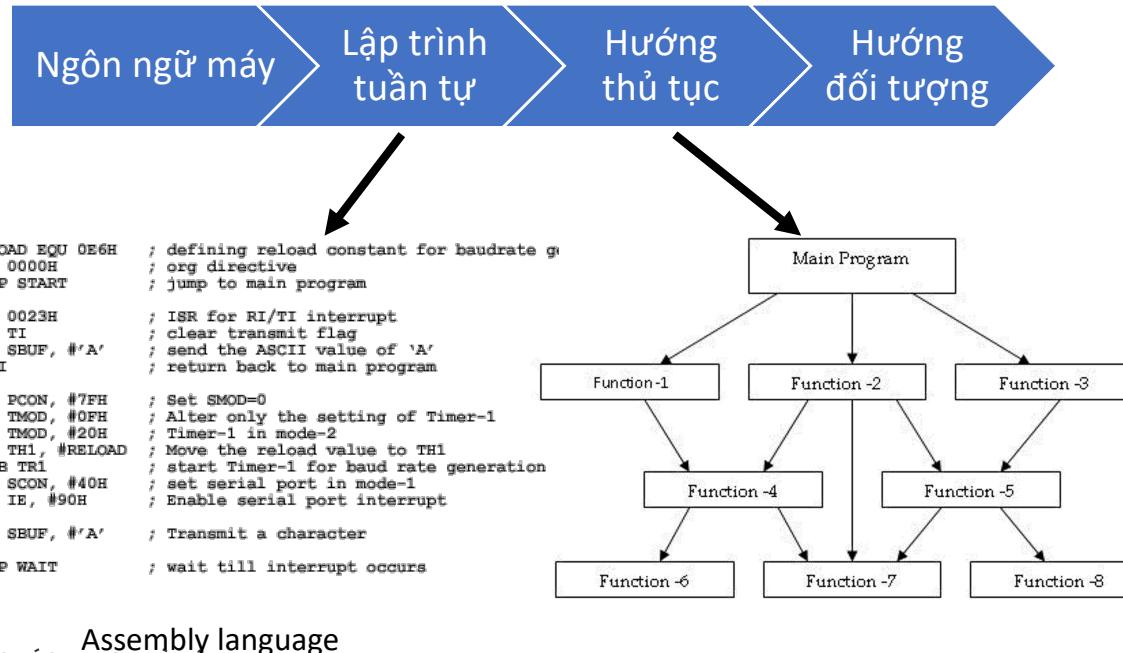
- Ngôn ngữ lập trình là một hệ thống các ký hiệu dùng để liên lạc, trao đổi với máy tính nhằm thực thi một nhiệm vụ tính toán.
- Có rất nhiều ngôn ngữ lập trình (khoảng hơn 1000), phần lớn là các ngôn ngữ hàn lâm, có mục đích riêng hay phạm vi.

Ngôn ngữ lập trình

Có 3 thành phần căn bản của bất cứ 1 NNLT nào:

- *Mô thức lập trình* là những nguyên tắc chung cơ bản, dùng bởi LTV để xây dựng chương trình.
- *Cú pháp* của ngôn ngữ là cách để xác định cái gì là hợp lệ trong cấu trúc các câu của ngôn ngữ; Nắm được cú pháp là cách để đọc và tạo ra các câu trong các ngôn ngữ tự nhiên, như tiếng Việt, tiếng Anh. Tuy nhiên điều đó không có nghĩa là nó giúp chúng ta hiểu hết ý nghĩa của câu văn.
- *Ngữ nghĩa* của 1 program trong ngôn ngữ ấy. Rõ ràng, nếu không có semantics, 1 NNLT sẽ chỉ là 1 mớ các câu văn vô nghĩa; như vậy semantics là 1 thành phần không thể thiếu của 1 ngôn ngữ.

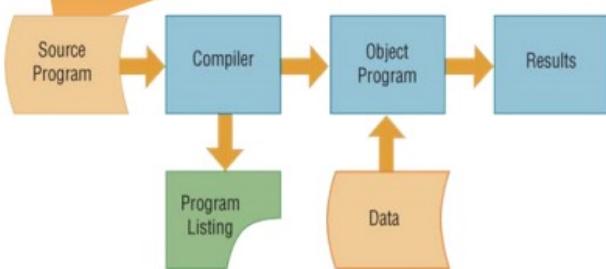
Sự phát triển của ngôn ngữ lập trình



Trình dịch - compiler

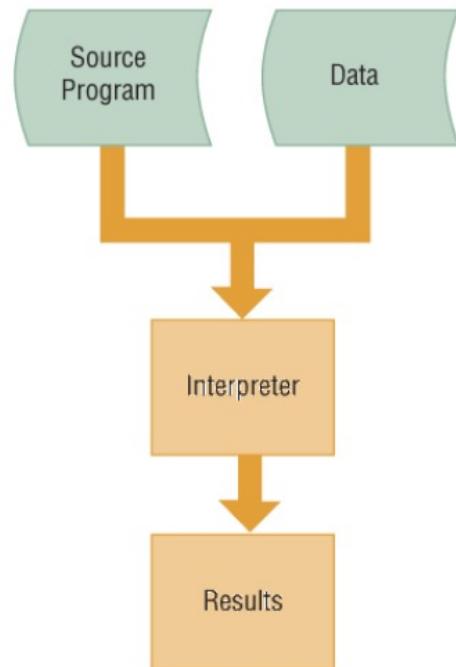
- Chương trình thực hiện biên dịch toàn bộ chương trình nguồn thành mã máy trước khi thực hiện

* COMPUTE REGULAR TIME PAY
MULTIPLY REGULAR-TIME-HOURS BY HOURLY-PAY-RATE
GIVING REGULAR-TIME-PAY.
* COMPUTE OVERTIME PAY
IF OVERTIME-HOURS > 0
COMPUTE OVERTIME-PAY = OVERTIME-HOURS * 1.5 * HOURLY-PAY-RATE
ELSE
MOVE 0 TO OVERTIME-PAY.
* COMPUTE GROSS PAY
ADD REGULAR-TIME-PAY TO OVERTIME-PAY
GIVING GROSS-PAY.
* PRINT GROSS PAY
MOVE GROSS-PAY TO GROSS-PAY-OUT.
WRITE REPORT-LINE-OUT FROM DETAILED-LINE
AFTER ADVANCING 2 LINES.



Thông dịch - interpreter

- Chương trình dịch và thực hiện từng dòng lệnh của chương trình cùng lúc
- Dịch từ ngôn ngữ này sang ngôn ngữ khác, không tạo ra chương trình dạng mã máy hay assembly

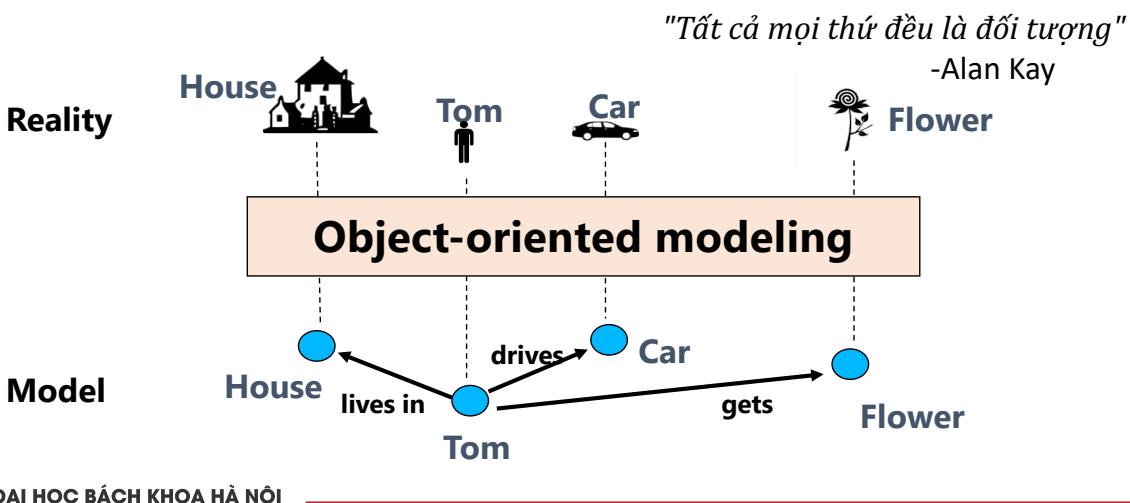


Ngôn ngữ lập trình hướng đối tượng

Object-oriented programming language

Lập trình hướng đối tượng

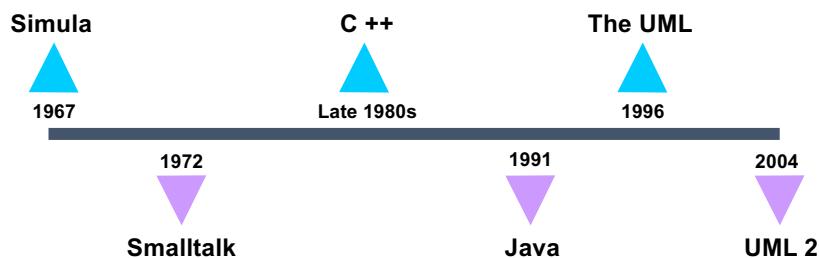
- Thể hiện các thành phần của bài toán là các “đối tượng” (object).
- Hướng đối tượng là một kỹ thuật để mô hình hóa hệ thống thành nhiều đối tượng tương tác với nhau



Công nghệ đối tượng (OOT)

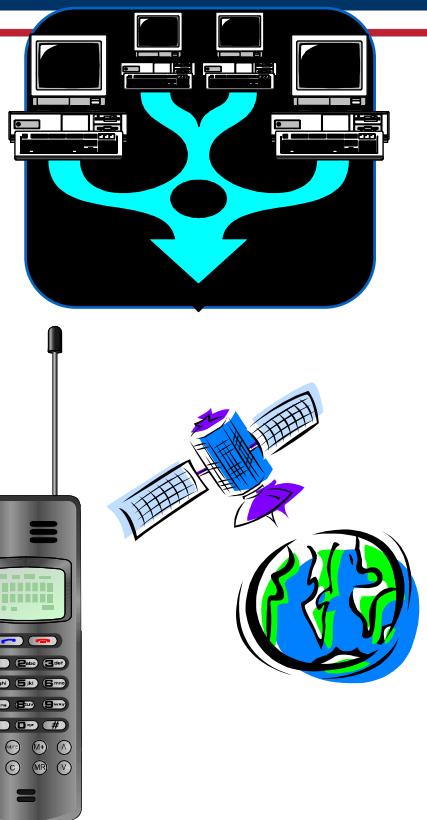
- Công nghệ đối tượng là một tập các quy tắc (trừu tượng hóa, đóng gói, đa hình), các hướng dẫn để xây dựng phần mềm, cùng với ngôn ngữ, cơ sở dữ liệu và các công cụ khác hỗ trợ các quy tắc này.
- Các mốc chính của công nghệ đối tượng

(Object Technology - A Manager's Guide, Taylor, 1997)



OOT được sử dụng ở đâu?

- Các hệ thống Client/Server và phát triển Web
- Hệ nhúng (embedded system)
- Hệ thống thời gian thực (real-time)
- Hệ thống phần mềm nói chung...



Đối tượng là gì?

- Đối tượng trong thế giới thực, là một thực thể cụ thể mà thông thường chúng ta có thể **sờ, nhìn thấy** hay **cảm nhận** được.
- Tất cả có trạng thái (state) và hành vi (behaviour)

	Trạng thái	Hành động	
Con chó	Tên Màu Giống Vui sướng	Sửa Vẩy tai Chạy Ăn	
Xe đạp	Bánh răng Bàn đạp Dây xích Bánh xe	Tăng tốc Giảm tốc Chuyển bánh răng ...	



Trạng thái và hành vi

- Trạng thái của một đối tượng là một trong các điều kiện tại đó mà đối tượng tồn tại
- Trạng thái của một đối tượng có thể thay đổi theo thời gian
- Hành vi quyết định đối tượng đó hành động và đáp ứng như thế nào đối với bên ngoài



Name: J Clark
Employee ID: 567138
Date Hired: July 25, 1991
Status: Tenured
Discipline: Finance
Maximum Course Load: 3 classes



Professor Clark's behavior
Submit Final Grades
Accept Course Offering
Take Sabbatical

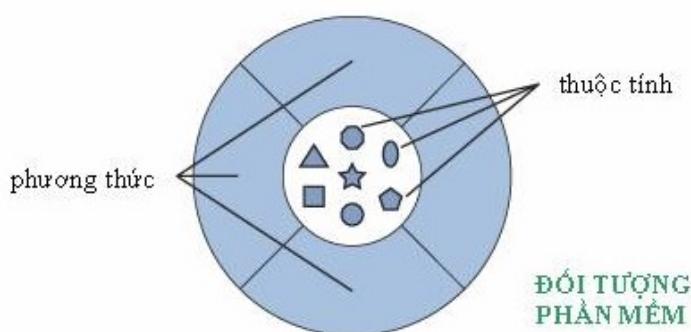


ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

21

Đối tượng phần mềm

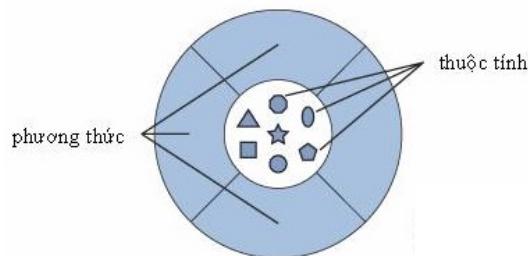
- Các **đối tượng phần mềm** được dùng để *biểu diễn* các đối tượng thế giới thực.
- Cũng có trạng thái và hành vi
 - Trạng thái: **thuộc tính** (attribute; property)
 - Hành vi: **phương thức** (method)



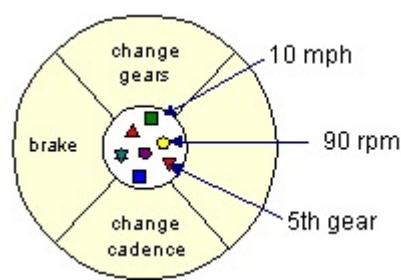
ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

22

Ví dụ đối tượng phần mềm



Đối tượng phần mềm



Đối tượng phần mềm Xe Đạp

Đối tượng (object) là một thực thể phần mềm bao bọc các **thuộc tính** và các **phương thức** liên quan.

Thuộc tính được xác định bởi giá trị cụ thể gọi là **thuộc tính thể hiện**. Một đối tượng cụ thể được gọi là một **thể hiện**.

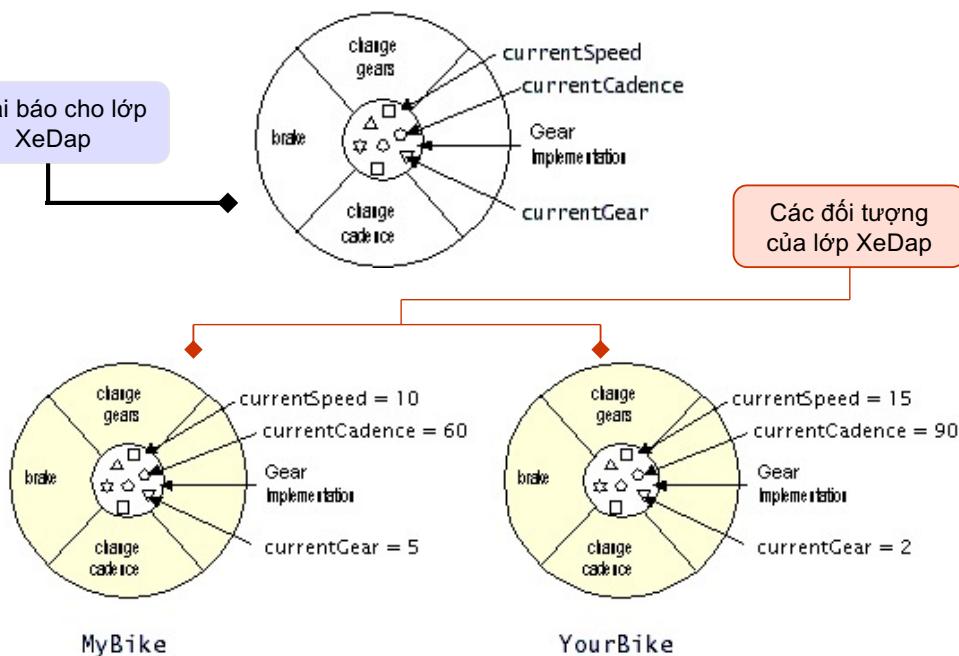


Lớp và đối tượng

- Một **lớp** là một thiết kế (blueprint) hay mẫu (prototype) cho các đối tượng cùng kiểu
 - Ví dụ: lớp XeDap là một thiết kế chung cho nhiều đối tượng xe đạp được tạo ra
- Lớp định nghĩa các thuộc tính và các phương thức chung cho tất cả các đối tượng của cùng một loại nào đó
- Một **đối tượng** là một thể hiện (instance) cụ thể của một lớp.
 - Ví dụ: mỗi đối tượng xe đạp là một thể hiện của lớp XeDap
- Mỗi thể hiện có thể có những thuộc tính thể hiện khác nhau
 - Ví dụ: một xe đạp có thể đang ở số (gear) thứ 5 trong khi một xe khác có thể là đang ở số (gear) thứ 3.



Ví dụ Lớp Bike



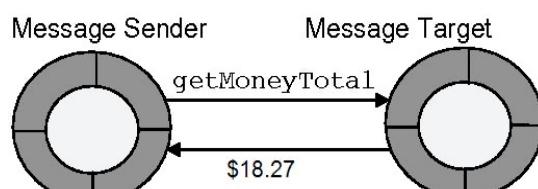
Tương tác giữa các đối tượng

- Sự giao tiếp giữa các đối tượng trong thế giới thực:



- Các đối tượng và sự tương tác giữa chúng trong lập trình

- Các đối tượng giao tiếp với nhau bằng cách gửi thông điệp (message)



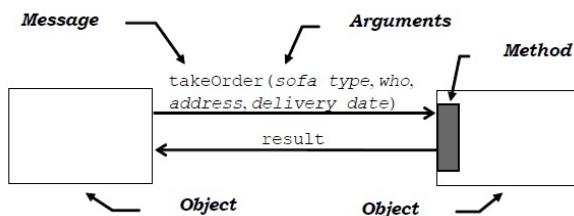
Thông điệp vs. Phương thức

- **Thông điệp**

- Được gửi từ đối tượng này đến đối tượng kia, không bao gồm đoạn mã thực sự sẽ được thực thi.

- **Phương thức**

- Thủ tục/hàm trong ngôn ngữ lập trình cấu trúc.
- Là sự thực thi dịch vụ được yêu cầu bởi thông điệp.
- Là đoạn mã sẽ được thực thi để đáp ứng thông điệp được gửi đến cho đối tượng.



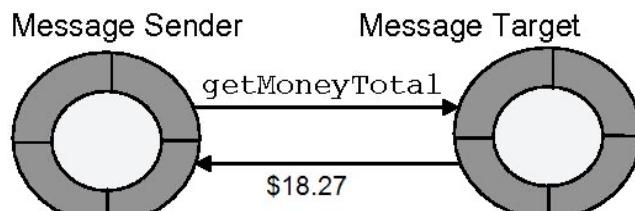
Đặc tính của LT HDT

1. Tất cả đều là **đối tượng**
2. Chương trình phần mềm có thể coi là một tập hợp các đối tượng **tương tác** với nhau
3. Mỗi đối tượng trong chương trình có các **dữ liệu độc lập** của mình và **chiếm bộ nhớ riêng** của mình.
4. Mỗi đối tượng đều có dạng **đặc trưng** của lớp các đối tượng đó
5. Tất cả các đối tượng thuộc về cùng một lớp đều có các **hành vi** giống nhau



Hướng cấu trúc vs. Hướng ĐT

- Hướng cấu trúc:
 - data structures + algorithms = Program
 - (cấu trúc dữ liệu + giải thuật = Chương trình)
- Hướng đối tượng:
 - objects + messages = Program
 - (đối tượng + thông điệp = Chương trình)



Giới thiệu về ngôn ngữ C++

Lịch sử ngôn ngữ C

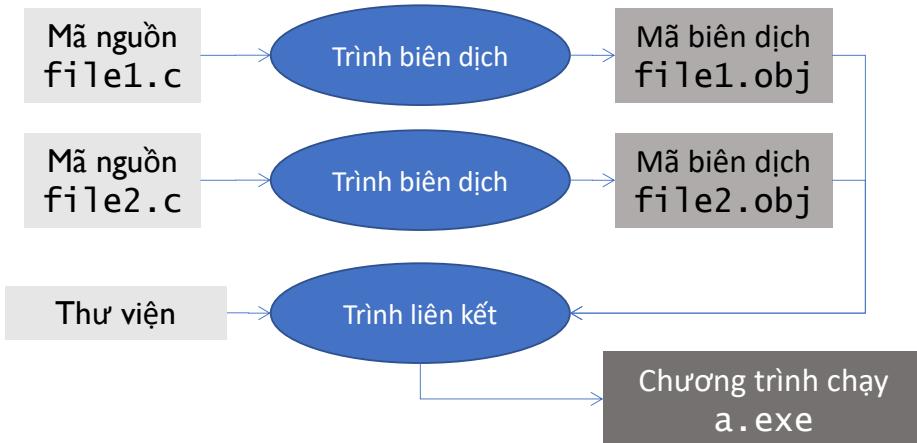
- Ra đời trong những năm 1970, gắn liền với sự phát triển của HĐH Unix. Tác giả: Dennis Ritchie
- Mục tiêu:
 - Đè cao tính hiệu quả
 - Có khả năng truy xuất phần cứng ở cấp thấp
 - Ngôn ngữ có cấu trúc (thay cho lập trình bằng hợp ngữ)
- C là ngôn ngữ trung gian giữa cấp thấp...
 - Có khả năng truy xuất bộ nhớ trực tiếp
 - Cú pháp ngắn gọn, ít từ khoá
- ... và cấp cao
 - Không phụ thuộc phần cứng
 - Cấu trúc, hàm, khả năng đóng gói
 - Kiểm tra kiểu dữ liệu

Lịch sử ngôn ngữ C++

- Ra đời năm 1979 bằng việc mở rộng ngôn ngữ C. Tác giả: Bjarne Stroustrup
- Mục tiêu:
 - Thêm các tính năng mới
 - Khắc phục một số nhược điểm của C
- Bổ sung những tính năng mới so với C:
 - Lập trình hướng đối tượng (OOP)
 - Lập trình tổng quát (template)
 - Nhiều tính năng nhỏ giúp lập trình linh hoạt hơn nữa (thêm kiểu bool, khai báo biến bất kỳ ở đâu, kiểu mạnh, định nghĩa chồng hàm, namespace, xử lý ngoại lệ,...)

Biên dịch chương trình C/C++

- Là quá trình chuyển đổi từ mã nguồn (do người viết) thành chương trình ở dạng mã máy để có thể thực thi được



Biên dịch chương trình C/C++

- Cho phép dịch từng file riêng rẽ giúp:
 - Dễ phân chia và quản lý từng phần của chương trình
 - Khi cần thay đổi, chỉ cần sửa đổi file liên quan
→ giảm thời gian bảo trì, sửa đổi
 - Chỉ cần dịch lại những file có thay đổi khi cần thiết
→ giảm thời gian dịch
- Các trình biên dịch hiện đại còn cho phép tối ưu hóa dữ liệu và mã lệnh
- Một số trình biên dịch thông dụng: MS Visual C++, gcc, Intel C++ Compiler, Watcom C/C++,...

Vào ra trong C++

Header file cho I/O trong C++

Header File	Miêu tả
<iostream>	File này định nghĩa các đối tượng cin, cout, cerr và clog, tương ứng với Standard Input Stream (Luồng đầu vào chuẩn), Standard Output Stream (Luồng đầu ra chuẩn), Unbuffered Standard Error Stream (Luồng lỗi chuẩn không được đệm) và Buffered Standard Error Stream (Luồng lỗi chuẩn được đệm).
<iomanip>	File này cung cấp các trình thao tác (manipulator) để định dạng đầu ra khi sử dụng các luồng I/O như std::cout, std::cin, std::cerr. Giúp kiểm soát cách các dữ liệu được hiển thị trên màn hình hoặc trong các file, cho phép bạn điều chỉnh độ rộng, độ chính xác, căn lề, và nhiều yếu tố khác của đầu ra.
<fstream>	File này khai báo các dịch vụ xử lý file được kiểm soát bởi người dùng.

Standard Output Stream (cout) trong C++

- Đối tượng cout đã được định nghĩa trước cout là một thể hiện của lớp ostream. Đối tượng cout được xem như "được kết nối tới" thiết bị đầu ra chuẩn, thường là màn hình.
- Đối tượng cout được sử dụng kết hợp với toán tử chèn luồng (insertion operator), được viết là <<, như ví dụ dưới đây:

```
#include <iostream>
using namespace std;
int main( ) {
    char str[] = "Xin chao C++";
    cout << "Gia tri cua str la: " << str <<
    endl;
}
```

- Toán tử chèn luồng << có thể được sử dụng nhiều hơn một lần trong một lệnh và endl được sử dụng để thêm một dòng mới tại cuối dòng đó.

Standard Input Stream (cin) trong C++

- Đối tượng tiền định nghĩa cin là một minh họa của lớp isrtream. Đối tượng cin được xem như đính kèm với thiết bị đầu vào chuẩn, mà thường là bàn phím. Đối tượng cin được sử dụng kết hợp với toán tử trích luồng (extraction operator), viết là >>, như trong ví dụ sau:

```
#include <iostream>
using namespace std;
int main( ) {
    char ten[50];
    cout << "Nhap ten cua ban (viet lien): ";
    cin >> ten;
    cout << "Ten ban la: " << ten << endl;
}
```

Standard Input Stream (cin) trong C++

- Bộ biên dịch C++ cũng quyết định kiểu dữ liệu của giá trị đã nhập và chọn toán tử trích luồng thích hợp để trích giá trị và lưu giữ nó trong các biến đã cung cấp.
- Toán tử trích luồng >> có thể được sử dụng nhiều hơn một lần trong một lệnh. Để yêu cầu nhiều hơn một dữ liệu chuẩn, bạn có thể sử dụng:

```
cin >> ten >> tuoi;
```

Nó tương đương với hai lệnh sau:

```
cin >> ten; cin >> tuoi;
```

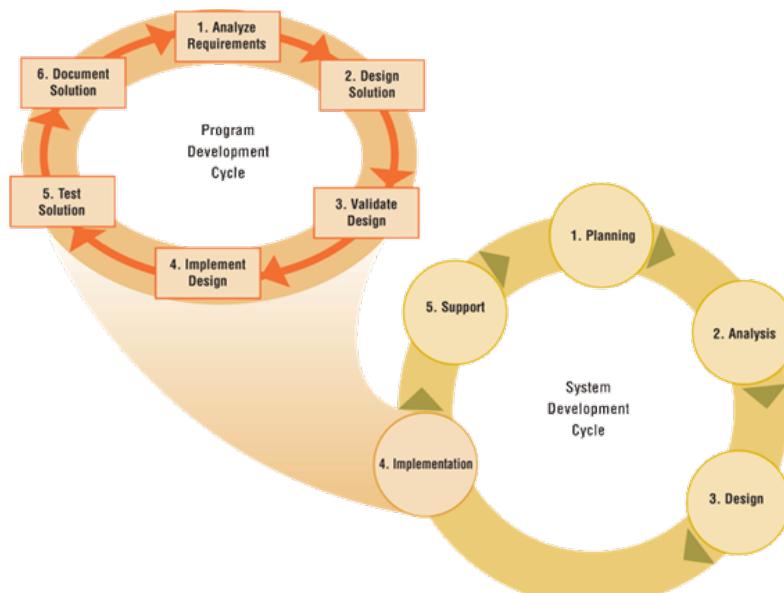
IDE lập trình

- Visual Studio Code (VSCode)
 - <https://code.visualstudio.com/>
 - Cài đặt thêm extension Microsoft C/C++

Chu trình phát triển phần mềm

Chu trình phát triển phần mềm

- Program development cycle?
- Là các bước mà lập trình viên dùng để xây dựng chương trình



Step 1 — Analyze Requirements

- Các việc cần làm khi phân tích yêu cầu?

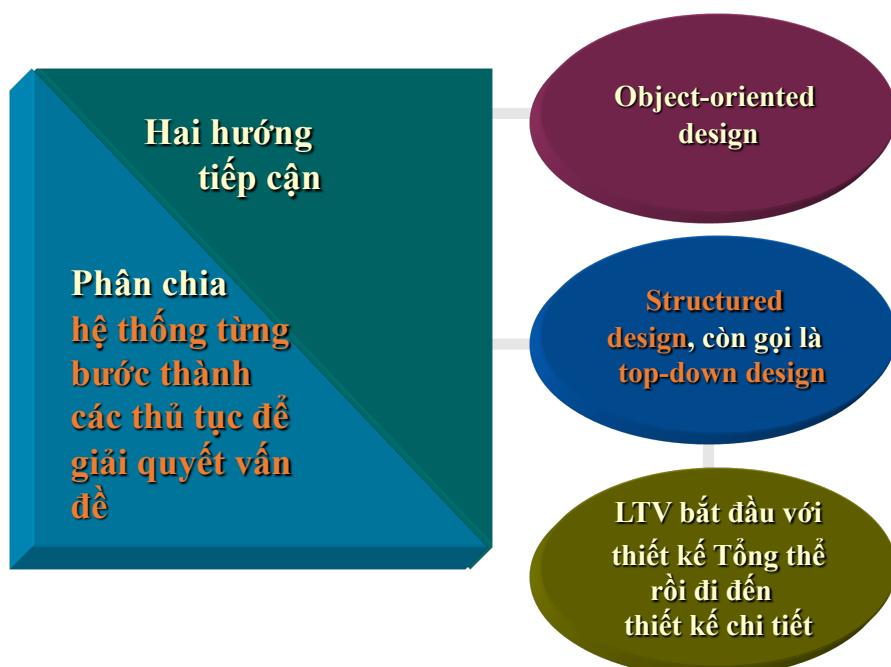
1. Thiết lập các requirements
2. Gặp các nhà phân tích hệ thống và users
3. Xác định input, output, processing, và các thành phần dữ liệu

- IPO chart—Xác định đầu vào, đầu ra và các bước xử lý

IPO CHART		
Input	Processing	Output
Regular Time Hours Worked	Read regular time hours worked, overtime hours worked, hourly pay rate.	Gross Pay
Overtime Hours Worked	Calculate regular time pay.	
Hourly Pay Rate	If employee worked overtime, calculate overtime pay. Calculate gross pay. Print gross pay.	

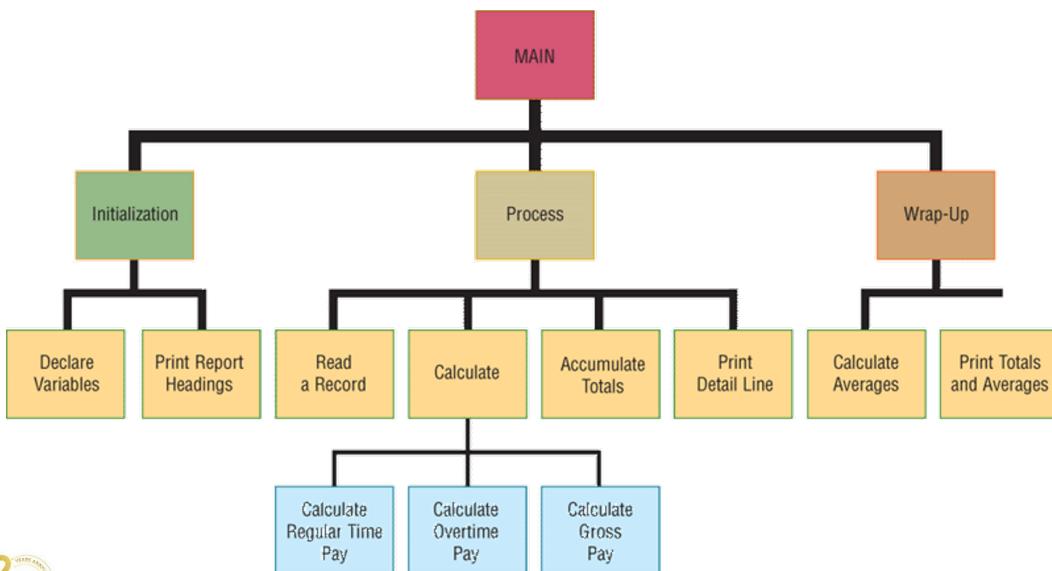
Step 2 — Design Solution

- Những việc cần làm trong bước thiết kế giải pháp?



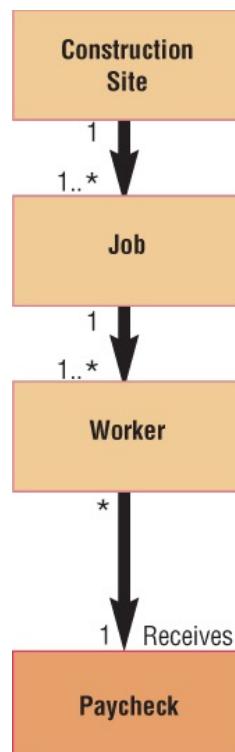
Step 2 — Design Solution

- Sơ đồ phân cấp chức năng- hierarchy chart?
- **Trực quan hóa các modules chương trình**
- **Còn gọi là sơ đồ cấu trúc**



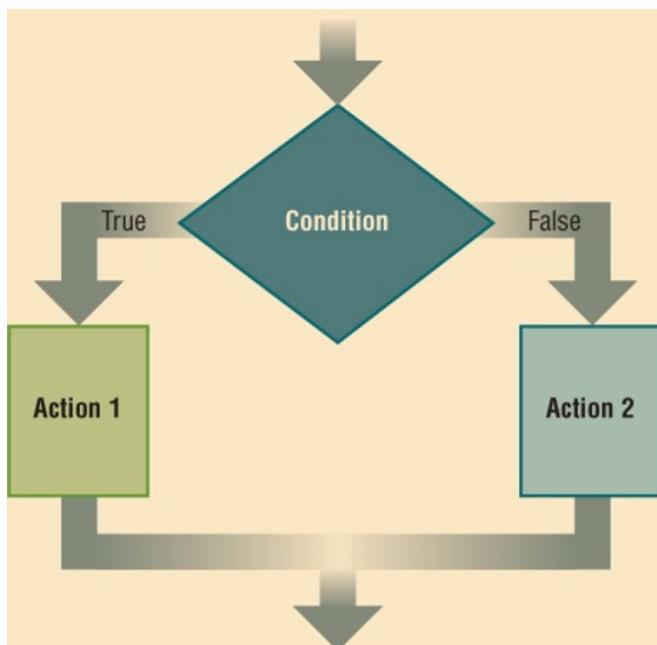
Step 2 — Design Solution

- Object-oriented (OO) design là gì?
- **LTV đóng gói dữ liệu và các thủ tục xử lý dữ liệu trong 1 object**
 - Các objects được nhóm lại thành các classes
 - Biểu đồ lớp thể hiện trực quan các quan hệ phân cấp quan hệ của các classes



Step 2 — Design Solution

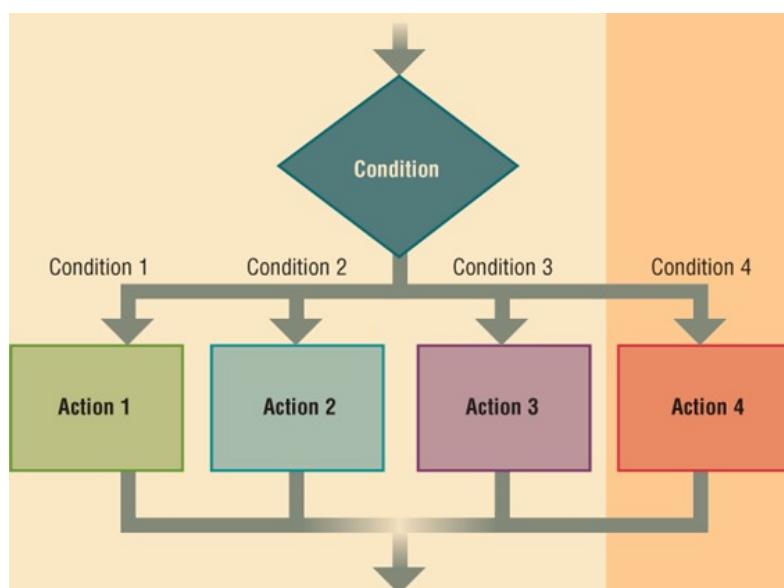
- Cấu trúc rẽ nhánh



- Chỉ ra action tương ứng điều kiện
- 2 kiểu
 - Case control structure
 - If-then-else control structure—dựa theo 2 khả năng: true or false

Step 2 — Design Solution

- Case control structure
- Dựa theo 3 hoặc nhiều hơn các khả năng

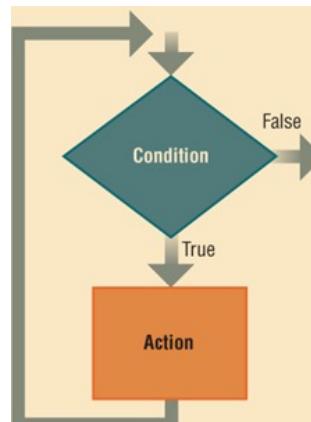


Step 2 — Design Solution

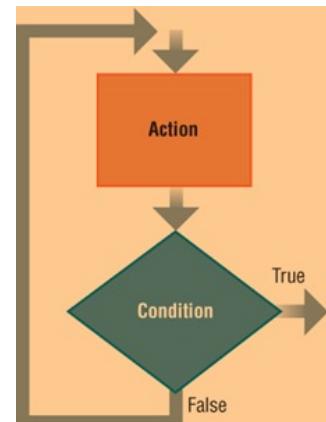
- Cấu trúc lặp

➤ Cho phép CT thực hiện 1 hay nhiều actions lặp đi lặp lại

- **Do-while control structure**—lặp khi điều kiện còn đúng
- **Do-until control structure**—Lặp cho đến khi điều kiện đúng



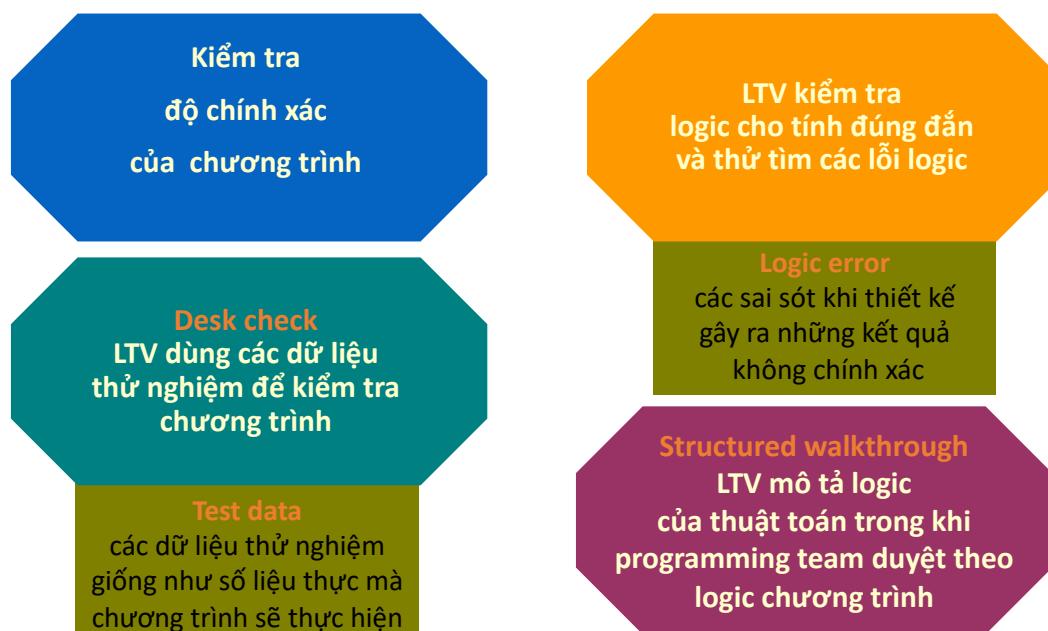
Do-While Control Structure



Do-Until Control Structure

Step 3 — Validate Design

- Những điều cần làm trong giai đoạn này?



Step 4 — Implement Design

- implementation?
- **Viết code : dịch từ thiết kế thành program**
 - **Syntax**—Quy tắc xác định cách viết các lệnh
 - **Comments**—program documentation
- **Extreme programming (XP)—coding và testing ngay sau khi các yêu cầu được xác định**

```
Chapter 2: Take-home pay calculator
Programmer: J. Quasney
Date: September 2, 2005
Purpose: This application calculates a worker's take-home pay per paycheck. The inputs are salary, percentage of salary contributed to a retirement account, and a selection of a health plan.

Private Sub CalculateTakeHomePay()
    Dim dblSocial, dblFed, dblState, dblMedicare, dblWeeklyPay As Double
    Dim dblRetirement, dblInsurance, dblTakeHomePay As Double
```

Step 5 — Test Solution

- Những việc cần làm ?

Đảm bảo CT chạy thông và cho kq chính xác

Debugging—Tìm và sửa các lỗi syntax và logic errors

Kiểm tra phiên bản **beta**,
giao cho Users dùng thử
và thu thập phản hồi

Step 6 — Document Solution

- Là bước không kém quan trọng
- 2 hoạt động

Rà soát lại program code—loại bỏ các **dead code**, tức các lệnh mà ct không bao giờ gọi đến

Rà soát, hoàn thiện documentation



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

53



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Xin cảm ơn!

