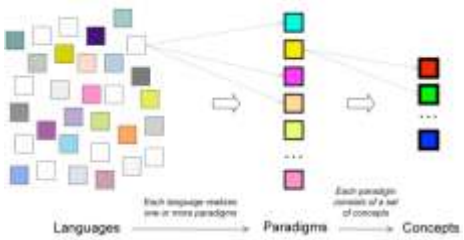
CHƯƠNG I.
NHỮNG KHÁI NIỆM CƠ BẢN TRONG KỸ THUẬT LẬP TRÌNH

I. TỔNG QUAN VỀ LẬP TRÌNH

I. Tổng quan về lập trình

- Với mỗi bài toán, làm thế nào để:
  - Thiết kế giải thuật nhằm giải quyết bài toán đó
  - Cài đặt giải thuật bằng một chương trình máy tính



I. Tổng quan về lập trình

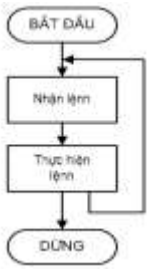
- Chương trình máy tính (computer program): Tập hợp các lệnh chỉ dẫn cho máy tính thực hiện nhiệm vụ
- Ngôn ngữ lập trình (programming language): Dùng để viết các lệnh, chỉ thị

I. TỔNG QUAN VỀ LẬP TRÌNH

1. Hoạt động của chương trình máy tính
2. Ngôn ngữ lập trình

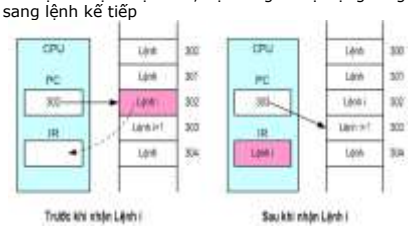
1. Hoạt động của chương trình máy tính

- 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).



1. 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



2. Ngôn ngữ lập trình (NNLT)

- Một NNLT là 1 hệ thống các ký hiệu dùng để liên lạc, trao đổi với máy tính nhằm thực thi một nhiệm vụ tính toán.
- Các thành phần căn bản của 1 NNLT:
  - Cú pháp (syntax): luật dùng để ghép các ký hiệu thành câu lệnh, thành chương trình hợp lệ về mặt cấu trúc
  - Ngữ nghĩa (semantic): luật dùng để ghép các ký hiệu thành câu lệnh, thành chương trình có ý nghĩa
- Có rất nhiều NNLT, khoảng 1000 ngôn ngữ (60's đã có hơn 700) – phần lớn là các ngôn ngữ hàn lâm, có mục đích riêng hay phạm vi ứng dụng hạn chế
  - Ngôn ngữ máy
  - Ngôn ngữ assembly
  - Các ngôn ngữ khác

2.1. Ngôn ngữ máy

- Máy tính chỉ nhận các tín hiệu điện tử - có, không có - tương ứng với các dòng bits.
- Một chương trình ở dạng đó gọi là mã máy (machine code).
- Ban đầu chúng ta phải dùng machine code để viết chương trình:

→ Quá phức tạp, giải quyết các bài toán lớn là không tưởng

```
23fc 0000 0001 0000 0040
0cb9 0000 000a 0000 0040
6e0c
06b9 0000 0001 0000 0040
60e8
```

2.2. Ngôn ngữ ASSEMBLY

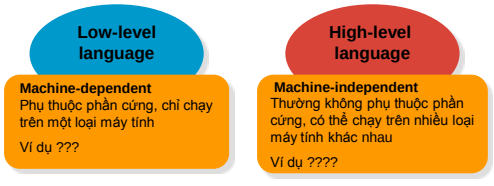
- Là bước đầu tiên của việc xây dựng cơ chế viết chương trình tiện lợi hơn – thông qua các ký hiệu, từ khóa và cả mã máy.
- Tất nhiên, để chạy được các chương trình này thì phải chuyển thành thành machine code.
- Vẫn còn phức tạp, cải thiện không đáng kể

```
 movl    #0x1,n
compare:
 cmpl    #oxa,n
 cgt     end_of_loop
 acddl   #0x1,n
 bra     compare
end_of_loop:
```

2.3. Phân loại ngôn ngữ lập trình - Theo thời gian

- 1940s: Ngôn ngữ máy tính hiểu được
  - Machine code
- 1950s: Khai thác sức mạnh của máy tính
  - Assembler code, Fortran v.1
- 1960s: Tăng khả năng tính toán
  - Cobol, Lisp, Algol 60, Basic, PL/1
- 1970s: Giảm sự phụ thuộc vào máy, tăng tính đúng đắn của CT
  - Structured Programming, Modular Programming: Pascal, Algol 68 and C.
- 1980s: Giảm sự phức tạp
  - Object-oriented, functional programming: Java
- 1990s: Khai thác triệt để các tài nguyên
  - Parallel, distributed computing: occam
- 2000s: Phát triển các mô hình tính toán mới
  - genetic programming languages, DNA computing, bio-computing, service-based computing
- ….

2.3. Phân loại ngôn ngữ lập trình - Theo mức độ trừu tượng



Machine và assembly languages là ngôn ngữ bậc thấp

High(er) level languages gần với ngôn ngữ con người hơn: Algol, Fortran, Pascal, Basic, Ada, C, …

2.3. Phân loại ngôn ngữ lập trình - Theo mức độ trừu tượng

Level	Instructions	Memory handling
Low level languages	Dạng bits – giống các lệnh machine	Truy cập và cấp phát trực tiếp bộ nhớ
High level languages	Dùng các biểu thức và các dòng điều khiển xác định	Truy cập và cấp phát bộ nhớ qua các lệnh, toán tử - operators
Very high level languages	Hoàn toàn trừu tượng, độc lập phần cứng	Che dấu hoàn toàn việc truy cập và tự động cấp phát bộ nhớ

2.3. Phân loại ngôn ngữ lập trình - Theo mục đích sử dụng

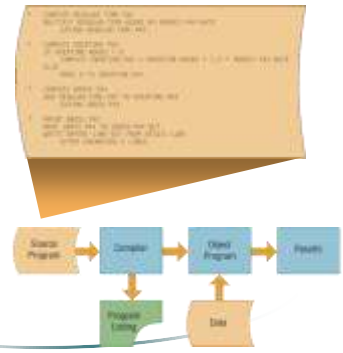
- Các ngôn ngữ lập trình cấp cao hơn ngôn ngữ assembly và mã máy có thể được phân thành 2 nhóm:
  - Declarative languages: ngôn ngữ lập trình dạng tường thuật
    - Trả lời câu hỏi: Cần làm gì / Cần lưu trữ cái gì
    - Còn gọi là functional languages, logic languages
  - Non-declarative langues: ngôn ngữ lập trình dạng phi tường thuật
    - Trả lời câu hỏi: Làm như thế nào / Lưu trữ như thế nào
    - Còn gọi là imperative languages, procedural languages

2.4. Ngôn ngữ lập trình dạng mệnh lệnh

Lập trình viên viết các chỉ thị hướng dẫn cho máy tính cai gì cần làm và làm như thế nào	**Sử dụng hàng loạt các từ giống tiếng anh để viết các chỉ thị - instructions**
Còn gọi là third-generation language (3GL)	**Các ngôn ngữ thông dụng là BASIC, COBOL, PASCAL, C,C++ và JAVA**

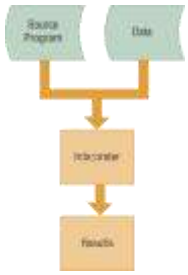
2.4. Ngôn ngữ lập trình dạng mệnh lệnh

- Trình dịch (Compiler): chương trình thực hiện biên dịch toàn bộ chương trình nguồn thành mã máy trước khi thực hiện



2.4. Ngôn ngữ lập trình dạng mệnh lệnh

- Trình thông dịch (Interpreter):
  - là 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



2.4. Ngôn ngữ lập trình dạng mệnh lệnh

- BASIC
  - Được thiết kế để cho những người mới học, giúp họ tiếp cận một cách đơn giản NNLT
  - Beginner's All-purpose Symbolic Instruction Code

2.4. Ngôn ngữ lập trình dạng mệnh lệnh

- COBOL
  - Dùng cho các ứng dụng kinh doanh, thương mại
  - Các lệnh giống tiếng Anh làm cho code dễ đọc, viết và chỉnh sửa
  - Common Business-Oriented Language

2.4. Ngôn ngữ lập trình dạng mệnh lệnh

- C
  - Là NNLT rất mạnh, ban đầu được thiết kế để lập trình phần mềm hệ thống
  - Yêu cầu những kỹ năng lập trình chuyên nghiệp

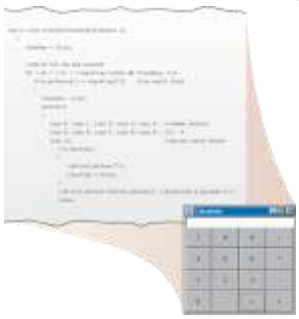
2.5. Ngôn ngữ lập trình hướng đối tượng

2.5. Ngôn ngữ lập trình hướng đối tượng

- C++
  - Chứa đựng các thành phần của C, loại bỏ những nhược điểm và thêm vào những tính năng mới để làm việc với object-oriented concepts
  - Được dùng để phát triển các Database và các ứng dụng Web

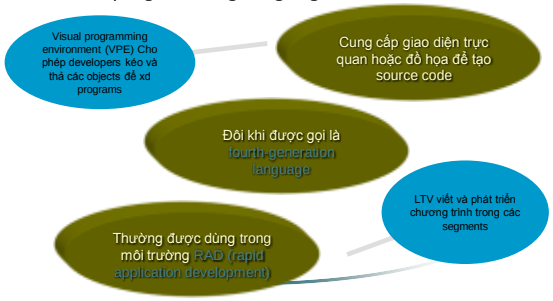
2.5. Ngôn ngữ lập trình hướng đối tượng

- Java
  - Phát triển bởi Sun Microsystems
  - Giống C++ nhưng dùng trình dịch just-in-time (JIT) để chuyển source code thành machine code



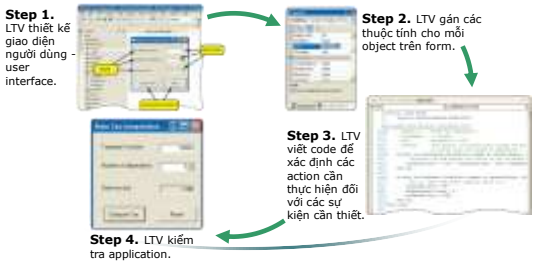
2.5. Ngôn ngữ lập trình hướng đối tượng

- Visual programming language



2.5. Ngôn ngữ lập trình hướng đối tượng

- Visual Studio .NET 2003, 2005
  - Bước phát triển của visual programming languages và RAD tools
  - .NET là tập hợp các công nghệ cho phép program chạy trên Internet
  - Visual Basic .NET 2003-5 dùng để xây dựng các chương trình hướng đối tượng phức tạp



2.5. Ngôn ngữ lập trình hướng đối tượng

- Delphi
  - Là 1 công cụ lập trình trực quan mạnh
  - Hợp với những ứng dụng chuyên nghiệp và Web lớn

2.5. Ngôn ngữ lập trình hướng đối tượng

- PowerBuilder
  - Một công cụ lập trình trực quan mạnh khác
  - Phù hợp với các ứng dụng Web-based hay các ứng dụng lớn HĐT - object-oriented applications

2.6. Ngôn ngữ lập trình dạng tường thuật

Nonprocedural Language

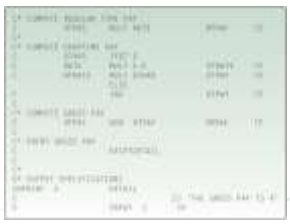
LTV viết các lệnh giống tiếng anh hoac tương tác với môi trường trực quan để nhận được các dữ liệu từ files hay database

Program Development Tools

Các chương trình thân thiện với người sử dụng được thiết kế để trợ giúp cả LTV lẫn người sử dụng trong việc tạo chương trình

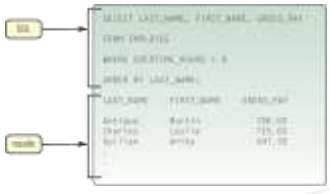
2.6. Ngôn ngữ lập trình dạng tường thuật

- RPG (Report Program Generator)
  - Các ngôn ngữ LT phi thủ tục được dùng để tạo các báo cáo, thiết lập các thao tác tính toán và cập nhật files



2.6. Ngôn ngữ lập trình dạng tường thuật

- NN thế hệ IV fourth-generation language (4GL)
  - Là các ngôn ngữ phi thủ tục cho phép truy cập dữ liệu trong csdl
  - NNLT 4GL thông dụng là SQL,Access, là các ngôn ngữ truy vấn . Cho phép users quản trị dữ liệu trong csdl quan hệ relational DBMS



2.6. Ngôn ngữ lập trình dạng tường thuật

- Application generator
  - Là chương trình tạo source code hoặc machine code từ các specification
  - Bao gồm các chương trình tạo Report và tạo Form nhập dữ liệu

2.6. Ngôn ngữ lập trình dạng tường thuật

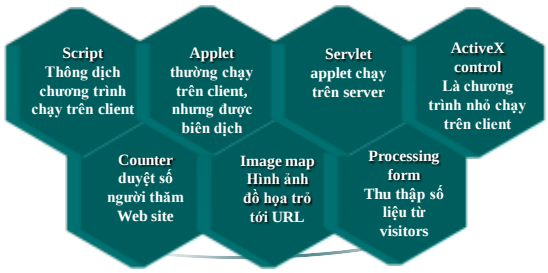
- Visual Basic for Applications (VBA)
  - Macro programming language
    - Macro—Dãy các lệnh dùng để tự động hóa các công việc

2.6. Ngôn ngữ lập trình dạng tường thuật

  - HTML (Hypertext Markup Language)
  - Dùng để tạo các trang Web

2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

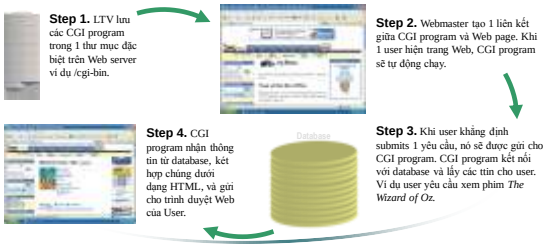
- Các hiệu ứng đặc biệt và các phần tử tương tác được thêm vào trang Web như thế nào ?



2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

- Common gateway interface (CGI): Chuẩn giao tiếp xác định cách thức Web server giao tiếp với các nguồn tài nguyên bên ngoài

CGI script /program – chương trình quản trị việc gửi và nhận dữ liệu qua CGI



2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

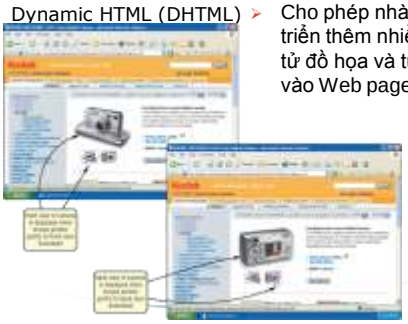
- Scripting language?

➢ Rất dễ học và dễ sử dụng

- JavaScript—thêm các nội dung động và các phần tử tương tác vào Web page
- VBScript (Visual Basic, Scripting Edition)—Thêm tính thông minh và tương tác vào Web page
- Perl (Practical Extraction and Report Language)—Có khả năng xử lý văn bản rất mạnh

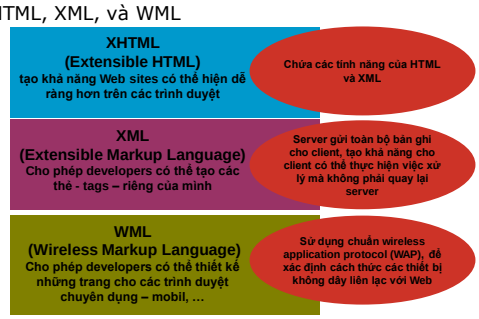
2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

- Dynamic HTML (DHTML)
  - Cho phép nhà phát triển thêm nhiều phần tử đồ họa và tương tác vào Web page



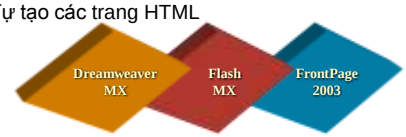
2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

- XHTML, XML, và WML



2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các trang web

- Web page authoring software?
  - Tạo các trang Web hoàn hảo mà không cần dùng HTML
  - Tự tạo các trang HTML

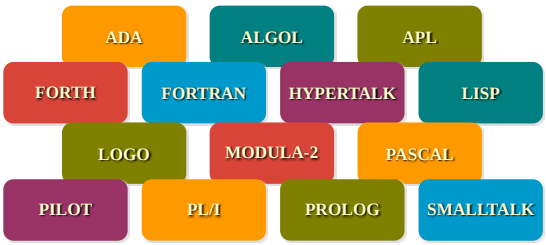


2.6. Ngôn ngữ lập trình dạng tường thuật - Tạo các ứng dụng đa phương tiện

- Multimedia authoring software?
  - Kết hợp văn bản, đồ họa, hoạt hình, âm thanh và video trong 1 bài trình diễn có tương tác
  - Sử dụng cho computer-based training (CBT) và Web-based training (WBT)

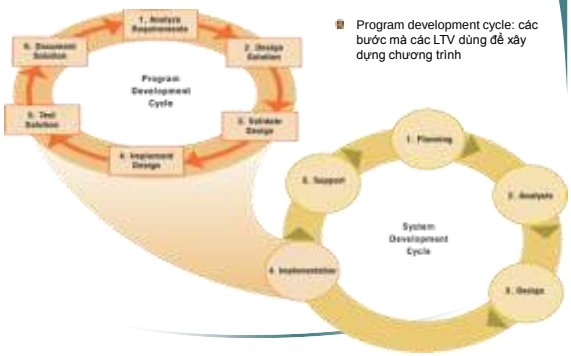
  - Software includes Toolbook, Authorware, và Director

2.7. Các ngôn ngữ lập trình khác



II. CHU TRÌNH PHÁT TRIỂN CHƯƠNG TRÌNH

Mô tả



- Program development cycle: các bước mà các LTV dùng để xây dựng chương trình

Bước 1: phân tích yêu cầu (*analyze requirements*)

- Phân tích hệ thống
  - Dựa trên các hệ thống có thực (do con người vận hành hoặc hệ thống tự động)
  - Do các nhà phân tích hệ thống tiến hành, sẽ hiệu quả hơn nếu phỏng vấn người dùng
  - Mục tiêu:
    - Xác định xem hệ thống hiện tại đã làm được những gì, làm như thế nào, còn tồn tại các vấn đề gì

  → Quyết định xem có nên thực hiện bước tiếp theo hay không (Return-on-Investment – ROI estimation)

Bước 1: phân tích yêu cầu (*analyze requirements*)

- Thiết lập các yêu cầu của hệ thống:
  - Dựa trên sự trao đổi giữa nhà phân tích hệ thống và nhà phân tích nghiệp vụ
  - Hình dung hệ thống mới: « look and feel »
  - Xác định
    - Cái gì cần thay đổi
    - Cần làm gì để có sự thay đổi đó (chưa quan tâm đến việc làm như thế nào)
  - Mô tả những việc cần làm: xác định
    - đầu vào (input): dữ liệu nào, từ đâu đến
    - đầu ra (output): dữ liệu nào, « mềm » (dữ liệu xuất ra màn hình) hay « cứng » (dữ liệu xuất ra các thiết bị khác)
    - xử lý (process): các hành động nào cần thực hiện để biến đầu vào thành đầu ra

    →Vẽ biểu đồ IPO

  → Quyết định xem có nên thực hiện bước tiếp theo hay không

Bước 1: phân tích yêu cầu
(*analyze requirements*)

- Biểu đồ IPO:
  - Input, Output: danh từ, phân biệt được các dữ liệu
  - Process: động từ, chỉ 1 hành động duy nhất
- Ví dụ: viết chương trình cho phép nhập vào 3 số, tính tổng của chúng và tính giá trị trung bình của chúng.

Input	Process	Output

Bước 1: phân tích yêu cầu
(*analyze requirements*)

Input	Process	Output
value1, value 2, value3	read the input values	
value1, value 2, value3	add the numbers together	Total
Total, value number	calculate average	Average
	display average	
	display total	

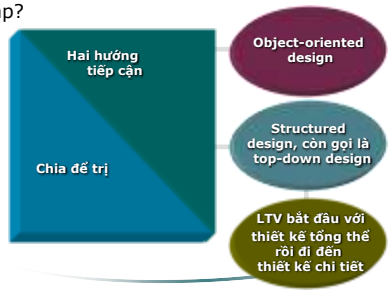
Input	Process	Output
3 numbers	read 3 numbers	(n1 + n2+ n3) / 3
	compute average and total	n1 + n2 + n3
	print average and total	

Bước 2 – thiết kế giải pháp
(design solution)

- Những việc cần làm trong bước thiết kế giải pháp?
  - Phân rã bài toán thành các bài toán nhỏ hơn
  - Tìm giải pháp cho từng bài toán nhỏ, phát triển lên thành giải thuật
  - Kết hợp các giải pháp cho bài toán nhỏ thành giải pháp tổng thể cho bài toán ban đầu

Bước 2 – thiết kế giải pháp
(design solution)

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



Bước 2 – thiết kế giải pháp
(design solution)

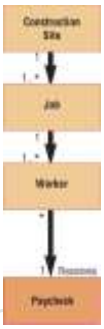
- Sơ đồ phân cấp chức năng (hierarchy chart) ?
  - Trực quan hóa các modules CT
  - Còn gọi là sơ đồ cấu trúc

Bước 2 – thiết kế giải pháp
(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



Bước 2 – thiết kế giải pháp
(design solution)

- Máy tính không thể tự nghĩ ra hay tự quyết định một sơ đồ hoạt động
- Máy tính chỉ có thể làm chính xác những gì được yêu cầu, theo cách được yêu cầu, chứ không phải làm những gì con người muốn máy tính làm
- Giải thuật là một tập các chỉ thị miêu tả cho máy tính nhiệm vụ cần làm và thứ tự thực hiện các nhiệm vụ đó.

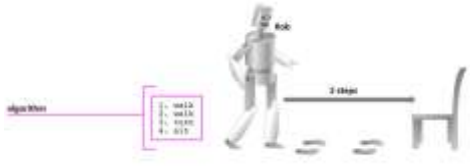
Bước 2 – thiết kế giải pháp
(design solution)

- Giải pháp cho mọi chương trình máy tính, dù đơn giản hay phức tạp, đều có thể được trình bày dựa trên 3 cấu trúc cơ bản sau:
  - Tuần tự
  - Chọn
  - Lặp
- Các cấu trúc này được gọi là các cấu trúc điều khiển hay các cấu trúc logic, vì nó điều khiển logic tính toán của chương trình máy tính

Bước 2 – thiết kế giải pháp
(design solution)

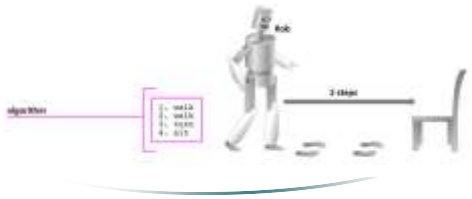
Cấu trúc tuần tự

- Cấu trúc tuần tự trong một chương trình máy tính chỉ thị cho máy tính xử lý lần lượt các lệnh (statement) của chương trình theo thứ tự được chỉ ra trong chương trình



Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc tuần tự

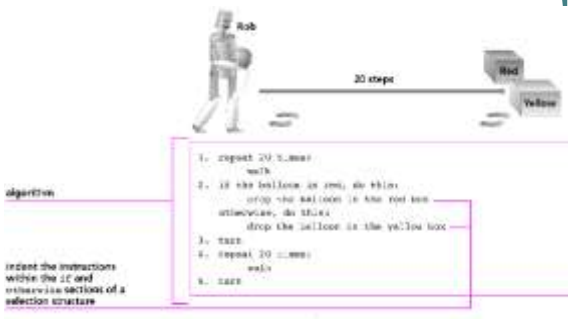
- Lệnh có thể là:
  - Lệnh gán
  - Lệnh vào / ra
  - Lệnh ghép



Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc chọn

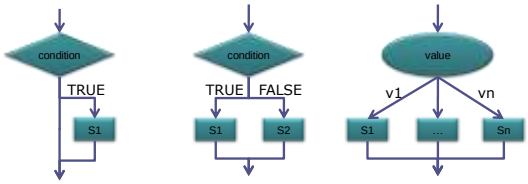
- Dùng để ra quyết định, và sau đó thì thực hiện một hành động dựa trên quyết định đó
- Phải chỉ ra được các hành động có khả năng được thực hiện sau khi có quyết định
- Quyết định phụ thuộc vào các điều kiện

Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc chọn

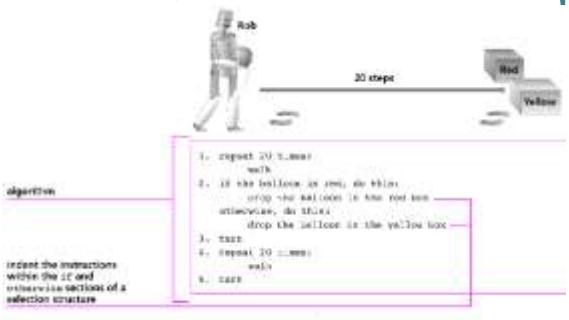


Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc chọn

- Single input – single output
- Single input – double output
- Single input – multiple output



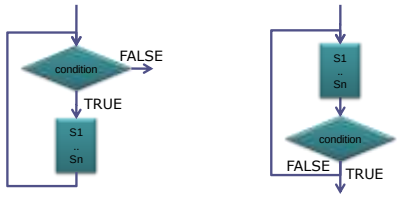
Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc lặp



Bước 2 – thiết kế giải pháp
(design solution)
Cấu trúc lặp

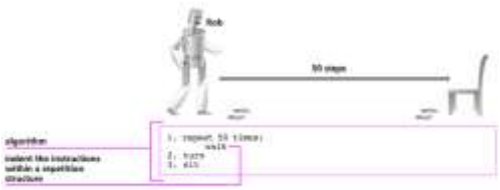
- Cho phép lập trình viên đặc tả một hành động cần thực hiện lặp đi lặp lại và có điều kiện
- Khi được sử dụng trong một chương trình, cấu trúc lặp chỉ thị cho máy tính thực hiện lặp đi lặp lại một hoặc nhiều lệnh, cho đến khi thỏa mãn điều kiện. Vào thời điểm đó, máy tính có thể kết thúc vòng lặp

Bước 2 – thiết kế giải pháp *(design solution)*
Cấu trúc lặp



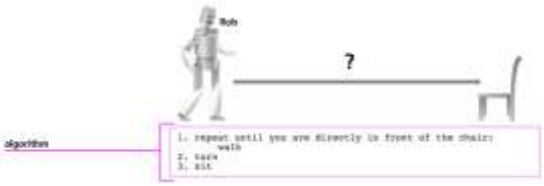
Bước 2 – thiết kế giải pháp *(design solution)*
Cấu trúc lặp

- Trường hợp 1: số lần lặp biết trước



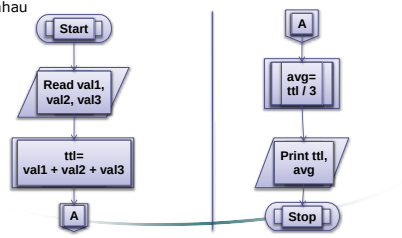
Bước 2 – thiết kế giải pháp *(design solution)*
Cấu trúc lặp

- Trường hợp 2: số lần lặp không biết trước
- Các lệnh trong vòng lặp được thực hiện cho đến khi điều kiện lặp không còn đúng nữa.
  - Điều kiện phải được kiểm tra trước: các lệnh trong vòng lặp có thể không được thực hiện lần nào
  - Khác đi, các lệnh trong vòng lặp có thể được thực hiện ít nhất một lần



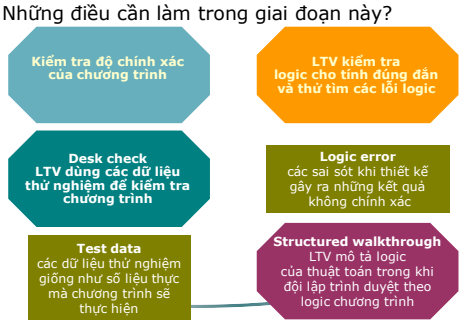
Ví dụ

- Flowchart (biểu đồ luồng) là công cụ để phát triển một giải pháp thành một giải thuật
  - Mô tả giải thuật một cách trực quan
  - Sử dụng ít ký hiệu để định nghĩa giải thuật với độ khó khác nhau



Bước 3 – Hợp thức hóa thiết kế
(validate design)

- Những điều cần làm trong giai đoạn này?

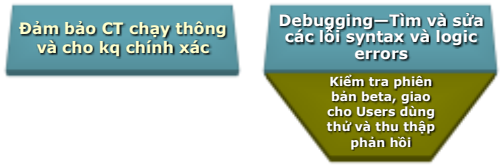


Bước 4: cài đặt thiết kế
(implement design)

- 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**

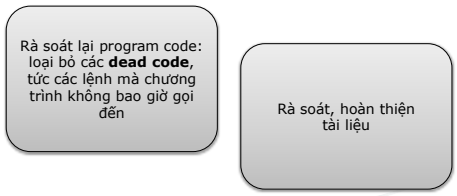
Bước 5 – kiểm tra giải pháp
(test solution)

- Những việc cần làm ?



Bước 6 – viết tài liệu cho giải pháp
(document solution)

- Là bước không kém quan trọng
- **2 hoạt động**



Tóm lại

III. CÁC MÔ THỨC LẬP TRÌNH

1. Mở đầu
2. Hướng mệnh lệnh
3. Hướng chức năng
4. Logic
5. Hướng đối tượng

1. Mở đầu

- · Programming paradigm: mô thức lập trình
  - Tập các khái niệm được dùng như các khuôn mẫu để lập trình
  - Đại diện cho các NNLT có cùng những đặc trưng cơ bản
- · Programming technique: kỹ thuật lập trình
  - Liên quan đến các ý tưởng thuật toán để giải quyết một lớp vấn đề tương ứng
  - Ví dụ:
    - 'Divide and conquer'
    - 'Program development by stepwise refinement'
- · Programming style: phong cách lập trình
  - Cách chúng ta trình bày trong 1 computer program
  - Phong cách tốt giúp cho chương trình dễ hiểu, dễ đọc, dễ kiểm tra -> dễ bảo trì, cập nhật, gỡ rối, tránh bị lỗi
- · Programming culture: văn hóa lập trình
  - Tổng hợp các hành vi lập trình, thường liên qua đến các dòng ngôn ngữ lập trình
  - Tổng thể của mô thức, phong cách và kỹ thuật lập trình
  - Nhân cách đạo đức trong lập trình cũng như khai thác các CT

1. Mở đầu

- Có nhiều mô thức lập trình
  - Imperative paradigm
  - Functional paradigm
  - Logical paradigm
  - Object-oriented paradigm
  - Visual paradigm
  - Parallel paradigm
  - Concurrent paradigm
  - Distributed paradigm
  - Service-oriented paradigm
- Chỉ xét 3 mô thức lập trình: hướng mệnh lệnh, hướng thủ tục và hướng đối tượng

2. Mô thức lập trình hướng mệnh lệnh

first do this and next do that

- Vấn đề:
  - Làm thế nào để thực thi các nhiệm vụ tính toán ?
  - Làm thế nào để biết được sự thay đổi trạng thái của chương trình khi tính toán ?
- Cách giải quyết: dùng dãy các lệnh (*statement*) để miêu tả việc tính toán; các lệnh này gây ra các ảnh hưởng có thể nhận biết được đến trạng thái của chương trình.
  - Declarative statement – Lệnh khai báo:
  - Assigment statement – Lệnh gán:
  - Program flow control statements – Các lệnh điều khiển cấu trúc chương trình:
  - Nested statement – Lệnh ghép:

2. Mô thức lập trình hướng mệnh lệnh

first do this and next do that

- Đặc trưng:
  - Nguyên lý và ý tưởng: Công nghệ số hóa phần cứng + ý tưởng của Von Neumann
  - Lệnh đặc trưng: Assignment, IO, procedure calls
  - Các thủ tục và hàm chính là hình ảnh về sự trừu tượng: che dấu các lệnh trong CT con, có thể coi CT con là 1 lệnh
  - Các ngôn ngữ đại diện: Fortran, Algol, Pascal, Basic, C
  - Tương ứng với cách mô tả các công việc hàng ngày như là trình tự nấu ăn hay sửa chữa xe cộ
  - Còn gọi là "Procedural programming"

3. Mô thức lập trình hướng chức năng

evaluate an expression and use the resulting value for something

- Nguồn gốc: lý thuyết hàm số → đơn giản và rõ ràng hơn mô thức lập trình hướng mệnh lệnh
- Ngôn ngữ lập trình: miêu tả
  - Tập hợp các kiểu dữ liệu có cấu trúc
  - Tập hợp các hàm định nghĩa trên các kiểu dữ liệu đó

3. Mô thức lập trình hướng chức năng

evaluate an expression and use the resulting value for something

- Đặc trưng cơ bản: Chú trọng đến việc mô-đun hóa chương trình
  - Một chức năng là biểu diễn trừu tượng của một biểu thức
  - Giải thuật thực hiện theo từng bước
  - Các giá trị trả về là không thể biến đổi
  - Không thể thay đổi cấu trúc dữ liệu của một giá trị, nhưng có thể sao chép lại các thành phần tạo nên giá trị đó
  - Tính toán bằng cách gọi các chức năng

3. Mô thức lập trình hướng chức năng

evaluate an expression and use the resulting value for something

- Ví dụ:

```
function GT(n: longint): longint;
var x: longint;
Begin
 x:=1;
 while (n > 0) do begin
    x:= x * n;
    n:= n - 1;
 end;
 GT:= x;
End;
```

4. Mô thức lập trình logic

Answer a question via search for a solution

- Mô hình này đặc biệt phù hợp với những lĩnh vực liên quan đến việc trích rút thông tin từ những sự kiện và mối quan hệ giữa các sự kiện – lĩnh vực trí tuệ nhân tạo.
- Đặc trưng:
  - Về nguyên tắc và ý tưởng: Tự động kiểm chứng trong trí tuệ nhân tạo
  - Dựa trên các chân lý- tiên đề axioms,các quy luật suy diễn - inference rules, và các truy vấn queries.
  - Chương trình thực hiện từ việc tìm kiếm có hệ thống trong 1 tập các sự kiện, sử dụng 1 tập các luật để đưa ra kết luận

5. Mô thức lập trình hướng đối tượng

Send messages between objects to simulate the temporal evolution of a set of real world phenomena

- Nguyên lý và ý tưởng: Các khái niệm và mô hình tương tác trong thế giới thực
- Dữ liệu cũng như các thao tác trên dữ liệu được bao gói trong các đối tượng
- Cơ chế che dấu thông tin nội bộ được sử dụng để tránh những tác động từ bên ngoài

5. Mô thức lập trình hướng đối tượng

Send messages between objects to simulate the temporal evolution of a set of real world phenomena

- Các Objects tương tác với nhau qua việc truyền thông điệp, đó là phép ẩn dụ cho việc thực hiện các thao tác trên 1 object
- Trong phần lớn các NNLT HĐT, objects được nhóm lại trong classes
  - Objects trong classes có chung các thuộc tính, cho phép lập trình trên lớp, thay vì lập trình trên từng đối tượng riêng lẻ
  - Classes đại diện cho các khái niệm còn objects đại diện cho hiện tượng
  - Classes có tính kế thừa, cho phép mở rộng hay chuyên biệt hóa lớp

Kết luận

- Chương trình phần mềm được viết bằng một ngôn ngữ lập trình, theo một chu trình phát triển với các bước chính sau:
  - Đặc tả và phân tích yêu cầu
  - Thiết kế
  - Cài đặt
  - Kiểm thử
- Những tính chất cần có với các chương trình phần mềm là:
  - Tính mềm dẻo scalability / Khả năng chỉnh sửa modifiability
  - Khả năng tích hợp integrability / Khả năng tái sử dụng reusability
  - Tính chuyển đổi, linh hoạt, độc lập phần cứng –portability
  - Hiệu năng cao –performance
  - Độ tin cậy – reliability
  - Dễ xây dựng
  - Rõ ràng, dễ hiểu
  - Ngắn gọn, xúc tích
- Các mô thức lập trình chính là:
  - Hướng mệnh lệnh
  - Hướng chức năng
  - Hướng đối tượng
  - Logic