Lập trình và ngôn ngữ lập trình (8)

Nguyễn Thanh Bình
Khoa Công nghệ Thông tin
Trường Đại học Bách khoa
Đại học Đà Nẵng

Lập trình

- kỹ năng cá nhân
  - năng lực cá nhân
  - hiểu biết các công cụ lập trình
- lập trình viên cần
  - nguyên tắc lập trình
  - kinh nghiệm
- lập trình viên tốt
  - viết chương trình
    - đúng đắn
    - dễ hiểu
    - dễ bảo trì, phát triển

Ngôn ngữ lập trình

- Có nhiều phương pháp lập trình khác nhau
  - nhiều ngôn ngữ lập trình khác nhau
- Điểm chung của các ngôn ngữ lập trình (NNLT)
  - dễ diễn đạt
  - dễ hiểu
  - dễ thực thi trên máy tính
- Một số tính chất của NNLT
  - kiểu và kiểm tra kiểu
  - mô-đun hóa

3

Kiểu

- Hầu hết các NNLT đều có *khái niệm kiểu*
  - kiểu số, kiểu lô-gíc...
  - một biến có kiểu dữ liệu xác định
-
- *Kiểm tra kiểu*
  - đảm bảo một toán tử/hàm chỉ áp dụng cho những toán tử/tham số có kiểu cho phép

4

Kiểu

- Ngôn ngữ định kiểu (types languages)
  - có hệ thống kiểu
  - cho phép kiểm tra sử dụng kiểu phù hợp mà không cần thực thi chương trình
    - kiểm tra tĩnh
- Ngôn ngữ định kiểu cho phép
  - phát hiện sớm một số lỗi liên quan đến kiểu
- Ngôn ngữ định kiểu
  - C, Java, C++...

5

Đa hình

- Ưu điểm của hệ thống kiểu và kiểm tra kiểu
  - chặt chẽ
  - dễ kiểm tra
- Tuy nhiên
  - hệ thống kiểu phải mềm dẽo trong sử dụng
    - đa hình

6

Đa hình

- Một số tình huống đa hình
  - Viết hàm áp dụng cho các mảng có số phần tử khác nhau
    - kiểu mảng được kiểm tra khi biên dịch
    - số phần tử của mảng được kiểm tra khi thực thi
  - Áp dụng hàm cho các kiểu dữ liệu khác nhau
    - xây dựng nhiều phiên bản của hàm tương ứng với các kiểu khác nhau
    - hoặc chỉ xây dựng một phiên bản của hàm, xử lý khác nhau được thực hiện khi thực thi
      - template (C++), generic (Java)

7

Đa hình

- Một số tình huống đa hình
  - Viết hàm áp dụng cho các mảng có số phần tử khác nhau
    - kiểu mảng được kiểm tra khi biên dịch
    - số phần tử của mảng được kiểm tra khi thực thi
  - Áp dụng hàm cho các kiểu dữ liệu khác nhau
    - xây dựng *nhiều phiên bản của hàm* tương ứng với các kiểu khác nhau
    - hoặc chỉ xây dựng *một phiên bản của hàm*, xử lý khác nhau được thực hiện khi thực thi
      - template (C++), generic (Java)
    - hoặc sử dụng *kiểu con/lớp con*
      - đa hình trong ngôn ngữ lập trình hướng đối tượng

8

Mô-đun hóa

- Xuất hiện vào những năm 70
- Đóng vai trò quan trọng để tạo ra phần mềm chất lượng
- Thiết kế hướng mô-đun
  - phần mềm = tập hợp các mô-đun và quan hệ giữa chúng
- Hầu hết các NNLT đều hỗ trợ mô-đun hóa

9

Mô-đun hóa

- Một mô-đun gồm hai phần
  - Phần giao diện
    - giao tiếp với bên ngoài mô-đun ay mô-đun khác
  - Phần thân
    - nội dung của mô-đun
    - cục bộ đối với mỗi mô-đun, che dấu đối với mô-đun khác

10

Mô-đun hóa

- Các mô-đun chỉ trao đổi dữ liệu qua phần giao diện
  - không sử dụng biến toàn cục
- Nếu thay đổi phần thân thì ít ảnh hưởng (hoặc không ảnh hưởng) đến các mô-đun khác

- Trong ngôn ngữ lập trình cấu trúc
  - mô-đun = hàm
- Trong ngôn ngữ lập trình hướng đối tượng
  - mô-đun = lớp / phương thức

11

Các phương pháp lập trình cơ bản

- Lập trình thủ tục/cấu trúc (procedural programming)
- Lập trình hướng đối tượng (object-oriented programming)
- Lập trình hàm (functional programming)
- Lập trình lô-gíc (logic programming)

12

Lập trình thủ tục

- được sử dụng phổ biến
- lập trình có cấu trúc
- phù hợp với thiết kế hướng chức năng
- NNLT thủ tục
  - Fortran, Ada, Pascal, C...

13

Lập trình hướng đối tượng

- khái niệm cơ bản
  - đối tượng, lớp
  - đóng gói
  - thừa kế
  - đa hình
- xu hướng phát triển của các NNLT hiện đại
- NNLT hướng đối tượng
  - Smalltalk, C++, Java, Delphi...

14

Lập trình hàm

- tính toán các biểu thức
  - hàm tính toán dựa trên các giá trị của tham số
- thao tác trên danh sách
- áp dụng
  - lĩnh vực tính toán
  - trí tuệ nhân tạo
- NNLT hàm
  - LISP, Scheme...

15

Lập trình lô-gíc

- thực hiện các biểu thức lô-gíc
  - khái niệm hợp giải (resolution)
    - tìm kiếm giá trị của các biến sao cho biểu thức lô-gíc có giá trị đúng
- ứng dụng
  - xây dựng hệ chuyên gia
  - xử lý ngôn ngữ tự nhiên
- NNLT lô-gíc
  - Prolog

16

Chọn NNLT

- quyết định quan trọng trong phát triển phần mềm
  - giảm chi phí
  - mã nguồn chất lượng
  - dễ bảo trì, phát triển

17

Chọn NNLT

- dựa vào nhiều yếu tố (1)
  - yêu cầu của khách hàng
    - khách hàng tự bảo trì sản phẩm
  - chương trình dịch
    - cần có chương trình dịch có chất lượng tốt
  - công cụ hỗ trợ
    - dễ dàng quá trình lập trình, bảo trì
  - kinh nghiệm của lập trình viên
    - chọn NNLT mà lập trình làm chủ

18

Chọn NNLT

- dựa vào nhiều yếu tố (2)
  - yêu cầu tính khả chuyển (portability)
    - thực hiện trên nhiều máy tính/plaform khác nhau
  - lĩnh vực ứng dụng
    - hệ thống nhúng: C, Assembly...
    - hệ thống quản lý: .NET, VB, C++...
    - hệ chuyên gia: Prolog
    - mạng: Java, .NET...
    - website: PHP, ASP...
    - không tồn tại ngôn ngữ đa năng cho mọi ứng dụng

19

Phong cách lập trình

- Cần có chương trình dễ hiểu
  - phụ thuộc vào đặc điểm NNLT
  - phong cách của người lập trình
- Phong cách lập trình không do lập trình viên tự đặt ra mà do tổ chức/doanh nghiệp/dự án đặt ra
  - các luật lập trình
  - các quy ước lập trình
- Mục đích
  - mã nguồn dễ hiểu, dễ kiểm thử, dễ bảo trì
  - ít lỗi

20

Phong cách lập trình

- Một số nguyên tắc lập trình
  - đặt tên
    - có ý nghĩa, gợi nhớ
  - trình bày
    - rỏ ràng, dễ hiểu
  - chú thích
    - đầy đủ, dễ đọc
  - hạn chế sử dụng cấu trúc khó hiểu
    - break, continue, goto...
  - ví dụ
    - quy ước lập trình C++

21