Câu 1: **What Is Principle Of Programming Language?**  
Nó là một bộ quy tắc được điều chỉnh để truyền đạt các hướng dẫn đến một máy, đặc biệt là máy tính.

Câu 2: **Mục tiêu của các nguyên tắc của ngôn ngữ lập trình là gì?**  
Mục tiêu là:

* Giới thiệu một số mô hình lập trình khác nhau
* Để có kinh nghiệm với các mô hình này bằng cách sử dụng các ngôn ngữ lập trình ví dụ
* Để hiểu các khái niệm về cú pháp, dịch thuật, trừu tượng hóa và thực hiện

Câu 3: **Các mô hình lập trình là gì?**

Một số mô hình là-

* Ví dụ về thủ tục : C, Pascal, Basic, Fortran
* Ví dụ chức năng : Lisp, ML
* Các  
  ví dụ hướng đối tượng : C ++, Java, Smalltalk
* Ví dụ dựa trên quy tắc (hoặc Logic) : Prolog

Câu 4: **Liệt kê các mô hình tính toán của ngôn ngữ?**

Mô hình là:

* Máy RAM  
  thủ tục
* Đồ thị theo chu kỳ được định hướng  
  Mô hình Smalltalk của OO
* Hàm đệ quy từng phần  
  Lisp và ML
* Thuật toán Markov  
  Prolog dựa trên những điều này

Câu 5: **What Is Translation?**

Dịch là giao tiếp chuyển đổi mã nguồn thành mã đích.

Câu 6:

X: Địa chỉ gián tiếp 1: Vòng lặp

Y: Địa chỉ ngay lập tức 2: Con trỏ

Z: Tự động giảm địa chỉ 3. Hằng số

1. X—3 Y—2 Z—1  
   (b) X—1 Y—3 Z—2  
   (c) X—2 Y—3 Z—1  
   (d) X—3 Y—1 Z—2

Real : Số được chuyển đổi thành một chuỗi.

Unit : Một số nguyên chỉ định chế độ đơn vị tuyến tính, theo hàm rtos được xây dựng.

Prec: Một số nguyên chỉ định độ chính xác, theo hàm rtos được xây dựng.

NGÔN NGỮ LISP

* Giới thiệu:

Ðược J. MAC CARTHY viết năm 1958, LISP là một trong những ngôn ngữ lập trình sớm nhất. Ðầu năm những năm 80, LISP được phát triển mạnh nhờ những áp dụng trong lĩnh vực trí tuệ nhân tạo. LISP có các ưu điểm chính như sau:

* Cú pháp đơn giản. Trong LISP chỉ có một cấu trúc duy nhất là cấu trúc danh sách (LISP là ngôn ngữ xử lý danh sách: LISP = LISt Processing language), không có lệnh, không có từ khóa, tất cả các hàm đều được viết dưới dạng danh sách.
* Là một ngôn ngữ mạnh nhờ tính tương đương giữa dữ liệu và chương trình: dữ liệu và chương trình đều là danh sách, chúng có thể thao tác nhờ chung một công cụ.
* Mềm dẻo và dễ phát triển.
* Các khái niệm cơ bản
* Nguyên tử (atom)

Nguyên tử là một đối tượng cơ bản của LISP, nguyên tử có thể là số hoặc ký hiệu.

* Số. Dữ liệu số trong LISP cũng giống như trong một số ngôn ngữ lập trình khác như Pascal, C…

Ví dụ về các hằng số: 5, -17, 5.35, 3/4, 118.2E+5,...

* Ký hiệu (symbol) là một chuỗi các ký tự (trừ các ký tự đặc biệt, dấu ngoặc và khoảng trống). Các hằng ký hiệu được viết mở đầu bằng dấu nháy đơn ‘.

Ví dụ về các hằng ký hiệu: ‘a, ‘anh, ‘anh\_ba,...

Một số ký hiệu được định nghĩa trước như: T (về mặt logic, được hiểu là TRUE), NIL (về mặt logic, được hiểu là FALSE).

Hằng ký hiệu số được xem như là một số, chẳng hạn ‘5 = 5.

* Danh sách

Danh sách là một dãy có phân biệt thứ tự của các phần tử cách nhau ít nhất một khoảng trắng và đặt nằm trong cặp dấu ngoặc đơn ().

Phần tử của danh sách có thể là một nguyên tử hoặc là một danh sách.

Hằng danh sách được mở đầu bằng dấu nháy đơn ‘.

Ví dụ về các hằng danh sách:

* ‘()Danh sách rỗng, tương đương ký hiệu NIL.
* ‘(a 5 c)Danh sách gồm 3 phần tử.
* ‘(3 (b c) d (e (f g)))Danh sách gồm 4 phần tử, trong đó phần tử thứ 2 và phần tử thứ 4 lại là các danh sách.
* (setq a 3) ; Gán số 3 cho biến có tên a =3

 (+ 5 3 6)

= 14

* Chú ý: Nếu biểu thức dùng hàm QUOTE hoặc dấu nháy đơn sẽ không được đánh giá

Ví dụ:

> ‘(+ 1 2)

= (+ 1 2)

(MOD 14 3)=2

+ (CAR L) nhận vào danh sách L, trả về phần tử đầu tiên của L.

+ (CDR L) nhận vào danh sách L, trả về một danh sách bằng phần còn lại của danh sách L sau khi bỏ đi phần tử đầu tiên.

* Các hàm so sánh các số <, >, <=, >=, = và /=, cho kết quả là T hoặc NIL

Ví dụ:

>(< 4 5)

= T

>(> 4 (\* 2 3))

= NIL

* (EQ s1 s2) so sánh xem hai ký hiệu s1 và s2 có giống nhau hay không?

Ví dụ:

>(eq ‘tuong ‘tuong)

= T

>(eq ‘tuong ‘duong)

= NIL

* (EQUAL o1 o2) so sánh xem đối tượng bất kỳ o1 và o2 có giống nhau hay không?

Ví dụ:

>(equal ‘(a b c) ‘(a b c))

= T

>(equal ‘(a b c) ‘( b a c))

= NIL

+ (LIST E1 E2 ... En) nhận vào n biểu thức E1, E2, ..., En, trả về danh sách bao gồm n phần tử V1, V2, ..., Vn, trong đó Vi là giá trị của biểu thức Ei (i=1..n) .

* (ATOM a) xét xem a có phải là một nguyên tử.
* (NUMBERP n) xét xem n có phải là một số.
* (LISTP L) xét xem L có phải là một danh sách.
* (SYMBOLP S) xét xem S có phải là một ký hiệu.
* (NULL L) nhận vào 1 danh sách L. Nếu L rỗng thì trả về kết quả là T, ngược lại thì trả về kết quả là NIL.
* **Cú pháp định nghĩa hàm là:**
* (defun <tên hàm> <danh sách các tham số hình thức>
* <biểu thức>
* )

VD:

(defun binh\_phuong (a)

(\* a a)

)

Sau khi nạp hàm này cho LISP, ta có thể sử dụng như các hàm đã được định nghĩa trước.

>(binh\_phuong 5)

= 25

##### Biến toàn cục

Biến toàn cục (global variables) là biến mà phạm vi của nó là tất cả các hàm. Biến toàn cục sẽ tự động giải phóng khi chương trình dịch LISP kết thúc.

* Hàm (SETQ <tên biến> <biểu thức>)

Gán trị của <biểu thức> cho <tên biến> và trả về kết quả là giá trị của <biểu thức>.

Ví dụ:

>(setq x (\* 2 3))

= 6

> x ; biến x vẫn còn tồn tại và có giá trị là 6

= 6

* **Biến cục bộ**

Biến cục bộ (local variables) là biến mà phạm vi của nó chỉ nằm trong hàm mà nó được tạo ra. Biến cục bộ sẽ tự động giải phóng hàm tạo ra nó kết thúc.

* (LET ( (var1 E1) (var2 E2) ... (vark Ek)) Ek+1 ... En)

Ta thấy hàm này có 2 phần: phần gán trị cho các biến và phần định trị các biểu thức.

Gán trị của biểu thức Ei cho biến cục bộ vari tương ứng và thực hiện (PROGN Ek+1 ... En).

Ví dụ:

>(Let ((a 3) (b 5)) (\* a b) (+ a b))

= 8

#### Ðệ quy

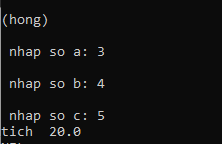
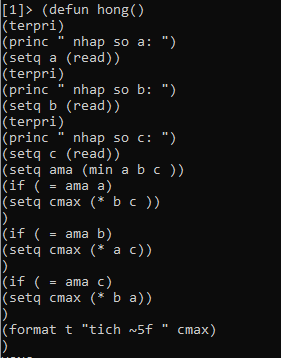
Một hàm đệ quy là một hàm có lời gọi chính nó trong biểu thức định nghĩa hàm. Mô tả một đệ quy bao gồm:

* Có ít nhất một trường hợp “dừng” để kết thúc việc gọi đệ quy.
* Lời gọi đệ quy phải bao hàm yếu tố dẫn đến các trường hợp “dừng”.

Vd: (if(> a b )

(Format t”~d be hơn” )

* Ví dụ về tính tích của 2 số lớn nhất



Vd: tinh tong

