CHƯƠNG V CHƯƠNG TRÌNH CON VÀ LẬP TRÌNH CÓ CẤU TRÚC

Bài 16. CHƯƠNG TRÌNH CON VÀ PHÂN LOẠI VÍ DỤ VỀ CHƯƠNG TRÌNH CON

I. Khái niệm chương trình con

1. ***Bài toán :*** Viết chương trình tính tổng:

$$\mathbf{TLuyThua = a^n + b^m + c^p + d^q}$$

Input: ***a, b, c, d, m, n, p,q;***	Biến nhập: ***a, b, c, d, m, n, p, q***
	Biến trung gian: ***LT1, LT2, LT3, LT4, i, j, k, l***
Output: ***TLuyThua;***	Biến xuất: ***TLuyThua***

– Đoạn lệnh **tính a^n** ***LT1:=1;***

$a^n = \underbrace{a.a.a \ \dots \ a}_{n \text{ lần}}$ ***For i:=1 to n do***

LT1:= LT1 * a;

– Đoạn lệnh **tính b^m** ***LT2:=1;***

$b^m = \underbrace{b.b.b \ \dots \ b}_{m \text{ lần}}$ ***For j:=1 to m do***

LT2:= LT2 * b;

– Đoạn lệnh **tính c^p** ***LT3:=1;***

$c^p = \underbrace{c.c.c \ \dots \ c}_{p \text{ lần}}$ ***For k:=1 to p do***

LT3:= LT3 * c;

– Đoạn lệnh **tính d^q** ***LT4:=1;***

$d^q = \underbrace{d.d.d \ \dots \ d}_{q \text{ lần}}$ ***For l:=1 to q do***

LT4:= LT4 * d;

– Đoạn lệnh tính **Tluythua**:

TLuyThua:=LT1+LT2+LT3+LT4;

❖ *Nhận xét:*

- Bốn đoạn lệnh LT1, LT2, LT3, LT4 có cách viết ***tương tự*** nhau
- Việc lặp lại những đoạn lệnh làm chương trình ***vừa dài vừa khó theo dõi***
- Để nâng cao hiệu quả lập trình, các ngôn ngữ lập trình bậc cao cần xây dựng ***chương trình con*** tính biểu thức tổng quát x^k
- Khi cần tính a^n, b^m, c^p, d^q ta chỉ cần ***gọi chương trình con tổng quát*** và thay thế ***(x,k)*** bằng giá trị cụ thể tương ứng là (a, n) hoặc (b, m) hoặc (c, p) hoặc (d, q).

2. Khái niệm

Chương trình con là một ***dãy lệnh*** mô tả một số thao tác nhất định và có thể được thực hiện (được gọi) từ ***nhiều vị trí*** trong chương trình.

3. Lợi ích của việc sử dụng chương trình con

- Tránh được việc ***viết lặp lại*** cùng một dãy lệnh.
- Hỗ trợ việc thực hiện các ***chương trình lớn***
- Phục vụ cho quá trình trừu tượng hóa
- ***Mở rộng*** khả năng ***ngôn ngữ***
- Thuận tiện cho ***phát triển , nâng cấp*** chương trình.

II. Phân loại và cấu trúc chương trình con

1. Phân loại

- Hàm (**Function**) là chương trình con thực hiện ***một số thao tác nào đó*** và ***trả về 1 giá trị qua tên của nó***

VD: ***length, copy, sqr***

- Thủ tục (**Procedure**) là chương trình con thực hiện các thao tác nhất định nhưng ***không trả về giá trị nào qua tên của nó***

VD: ***Delete, write/ writeln, read/ readln, clrscr, ...***

2. Cấu trúc chương trình con

a) Hàm (Function)

Function <Tên hàm> [(<D. sách tham số>)] **:** <***kiểu dữ liệu*** **>;**

[<***Phần khai báo***>]

Begin

[<Dãy các lệnh>]

<***Tên hàm***> **:=** <***Biểu thức***.>**;**

End;

b) Thủ tục (Procedure)

Procedure <Tên thủ tục> [(<Danh sách tham số>)]**;**

[<***Phần khai báo***>]

Begin

[<Dãy các lệnh>]

End;

❖ *Nhận xét về sự khác nhau giữa hai cấu trúc chương trình con (Hàm và thủ tục)*

Hàm	Thủ tục
– Phần đầu của chương trình con có ***khai báo kiểu dữ liệu của hàm trả về***.	– ***Không có***.
– Trước **End;** có câu lệnh gán ***<tên hàm>:= <biểu thức>***	– ***Không có***.

III. Biến và tham số trong chương trình con

a) Biến

Biến toàn cục	Biến cục bộ
– Là biến được khai báo ở ***chương trình chính*** có tác dụng ***mọi nơi*** trong chương trình.	– Là biến được khai báo ở ***chương trình con*** và chỉ có tác dụng trong ***chương trình con*** đó

– Giá trị của biến toàn cục có thể ***thay đổi*** nhờ vào các câu lệnh trong ***chương trình chính*** cũng như trong tất cả các ***chương trình con***	– Giá trị của biến cục bộ chỉ có thể ***thay đổi*** nhờ vào các câu lệnh trong ***chương trình con*** đó.
– Được cấp phát khi bộ nhớ khi ***chương trình bắt đầu thực hiện***	– Được cấp phát bộ nhớ khi ***chương trình con đó được gọi đến***
– Bị xóa khi ***chương trình chính được thực hiện xong***	– Bị xóa khi ***chương trình con đó kết thúc***

b) Tham số

- Tham số hình thức
  - Có hai loại: Tham số giá trị và tham số biến
  - Là các biến được ***khai báo*** cho dữ liệu ***vào ra*** của chương trình con
  - Khi khai báo, tham số hình thức được đặt ở phần đầu của chương trình con

Tham số giá trị (Tham trị)	**Tham số biến (Tham biến)**
– Là tham số hình thức được khai báo ***không có*** từ khóa ***VAR*** ở trước.	– Là tham số hình thức được khai báo ***có*** từ khóa ***VAR*** ở trước.
– Là các biến được dùng để ***chứa dữ liệu vào*** chương trình con.	– Là các biến được dùng để ***chứa dữ liệu ra*** (kết quả tính toán hoặc cần lấy ra) của chương trình con.
– Khi chương trình con kết thúc, các giá trị của tham số thực sự ứng tham trị ***vẫn giữ nguyên giá trị trước khi gọi***	– Khi chương trình con kết thúc, các giá trị của tham số thực sự ứng với tham biến ***sẽ có thể thay đổi so với trước khi gọi*** .....

- Tham số thực sự
  - Là các biến ***chứa*** trong ***lời gọi*** chương trình con.

IV. Ví dụ minh họa

Hãy viết chương trình tính tổng sau: **TLuyThua = $a^n + b^m + c^p + d^q$**

```
program TinhTongLuyThua;
uses crt;
var  a,b,c,d : real;
     n,m,p,q : integer;
     TLuythua : real;
  (* KHAI BAO VA DINH NGHIA CHUONG TRINH CON *)
  { CTC Nhap he so va so mu}
 Procedure Nhap (var x : real; var k : integer);
  begin
       write('Nhap he so: '); readln(x);
 write('Nhap so mu: '); readln(k);
  end;
 { CTC Tinh tung luy thua}
 Function  Luythua (x : real; k : integer) : real;
  var  i : integer;
       kq : real ;
  begin
     kq := 1;
     for  i := 1 to k do
                 kq := kq*x;
              Luythua := kq;
end;
```

```
{ CTC Tinh tong 4 luy thua}
procedure TinhTong;
  begin
    TLuythua:=Luythua(a,n) + Luythua(b,m)
              + Luythua(c,p) + Luythua(d,q);
  end;

(* CHUONG TRINH CHINH *)
begin
    clrcr;
    Nhap(a,n);
    Nhap(b,m);
    Nhap(c,p);
    Nhap(d,q);
    ……..;
    TinhTong;
    writeln('Tong luy thua = ',TLuythua:5:2);
    readln
end.
```

Lệnh Gọi chương trình con để nhập cơ số và số mũ

Lệnh Gọi chương trình con để tính tổng lũy thừa

……

……

……

……

……

……