TRƯỜNG CAO ĐẲNG CỘNG ĐỒNG KON TUM

KHOA SƯ PHẠM

----❣----

BÀI GIẢNG
HỌC PHẦN: LẬP TRÌNH

GV: TRẦN THỊ NGỌC HÀ

Kon Tum, 01/2018

LỜI MỞ ĐẦU

Nhằm đáp ứng yêu cầu học tập của sinh viên bước đầu làm quen với công việc lập trình, chúng tôi đã biên soạn bài giảng **Lập trình Pascal** nhằm giúp cho sinh viên có một tài liệu học tập, rèn luyện tốt khả năng lập trình, tạo nền tảng vững chắc cho các môn học tiếp theo trong chương trình đào tạo.

Giáo trình bao gồm lý thuyết và rất nhiều bài tập từ đơn giản đến phức tạp. Bên cạnh đó, chúng tôi cũng bổ sung một số bài tập dựa trên cơ sở một số thuật toán chuẩn với các cấu trúc dữ liệu được mở rộng nhằm nâng cao kỹ năng, phương pháp lập trình cho sinh viên.

Nội dung của giáo trình được chia thành 11 chương. Trong mỗi chương đều có phần lý thuyết, phần bài tập mẫu và cuối cùng là phần bài tập tự giải để sinh viên tự mình kiểm tra những kiến thức đã học. Trong phần bài tập mẫu, đối với những bài tập khó hoặc có thuật toán phức tạp, chúng tôi thường nêu ra ý tưởng và giải thuật trước khi viết chương trình cài đặt.

Chương 1 TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PASCAL

1. Nguồn gốc:

Pascal là một ngôn ngữ lập trình bậc cao do Niklaus Wirth, giáo sư điện toán trường Đại học kỹ thuật Zurich (Thụy Sĩ) đề xuất năm 1970. Ông lấy tên Pascal để kỷ niệm nhà toán học và nhà triết học người Pháp nổi tiếng Blaise Pascal.

2. Các tập tin cần thiết khi lập trình với Turbo Pascal

Để lập trình được với Turbo Pascal, tối thiểu cần 2 file sau:

TURBO.EXE: Dùng để soạn thảo và dịch chương trình.

TURBO.TPL: Thư viện chứa các đơn vị chuẩn để chạy với TURBO.EXE.

Ngoài ra, muốn lập trình đồ hoạ thì phải cần thêm các tập tin:

GRAPH.TPU: Thư viện đồ hoạ.

***.BGI**: Các file điều khiển các loại màn hình tương ứng khi dùng đồ hoạ.

***.CHR**: Các file chứa các font chữ đồ họa.

3. Các bước cơ bản khi lập một chương trình Pascal

Bước 1: Soạn thảo chương trình.

Bước 2: Dịch chương trình (nhấn phím **F9**), nếu có lỗi thì phải sửa lỗi.

Bước 3: Chạy chương trình (nhấn phím **Ctrl-F9**).

4. Các thành phần cơ bản của ngôn ngữ Pascal

4.1. Từ khóa

Từ khoá là các từ mà Pascal dành riêng để phục vụ cho mục đích của nó. (Chẳng hạn như: **BEGIN, END, IF, WHILE,...**)

Chú ý: Với Turbo Pascal 7.0 trở lên, các từ khoá trong chương trình sẽ được hiển thị khác màu với các từ khác.

4.2. Tên (định danh)

Định danh là một dãy ký tự dùng để đặt tên cho các hằng, biến, kiểu, tên chương trình con... Khi đặt tên, ta phải chú ý một số điểm sau:

Không được đặt trùng tên với từ khoá

Ký tự đầu tiên của tên không được bắt đầu bởi các ký tự đặc biệt hoặc chữ số.

Không được đặt tên với ký tự space,các phép toán.

Ví dụ: Các tên viết như sau là sai

1XYZ	Sai vì bắt đầu bằng chữ số.
#LONG	Sai vì bắt đầu bằng ký tự đặc biệt.
FOR	Sai vì trùng với từ khoá.
KY TU	Sai vì có khoảng trắng (space).
LAP-TRINH	Sai vì dấu trừ (-) là phép toán.

4.3. Dấu chấm phẩy (;)

Dấu chấm phẩy được dùng để ngăn cách giữa các câu lệnh. Không nên hiểu dấu chấm phẩy là dấu kết thúc câu lệnh.

Ví dụ:

FOR i:=1 TO 10 DO Write(i);

Trong câu lệnh trên, lệnh Write(i) được thực hiện 10 lần. Nếu hiểu dấu chấm phẩy là kết thúc câu lệnh thì lệnh Write(i) chỉ thực hiện 1 lần.

4.4. Lời giải thích

Các lời bàn luận, lời chú thích có thể đưa vào bất kỳ chỗ nào trong chương trình để cho người đọc dễ hiểu mà không làm ảnh hưởng đến các phần khác trong chương trình. Lời giải thích được đặt giữa hai dấu ngoặc { và } hoặc giữa cụm dấu (* và *).

Ví dụ:

Var a,b,c:Rea; {*Khai báo biến*}

Delta := b*b – 4*a*c; (* *Tính delta để giải phương trình bậc 2* *)

5. Cấu trúc chung của một chương trình Pascal

{ *Phần tiêu đề* }
PROGRAM Tên_chương_trình;
{ *Phần khai báo* }
USES ......;
CONST .....;
TYPE ....;
VAR ....;
PROCEDURE .....;
FUNCTION .....;
.....
{ *Phần thân chương trình* }
BEGIN
.....
END.

Ví dụ 1: Chương trình Pascal đơn giản nhất

```
BEGIN
   Write('Hello World!');
END.
```

Ví dụ 2:

```
Program  Vidu2;
Const  PI=3.14;
Var  R,S:Real;
Begin
   R:=10;          {Bán kính đường tròn}
   S:=R*R*PI;   {Diện tích hình tròn}
   Writeln('Dien tich hinh tron = ', S:0:2);  { In ra màn hình }
   Readln;
End.
```

6. Một số phím chức năng thường dùng

F2: Lưu chương trình đang soạn thảo vào đĩa.

F3: Mở file mới hoặc file đã tồn tại trên đĩa để soạn thảo.

Alt-F3: Đóng file đang soạn thảo.

Alt-F5: Xem kết quả chạy chương trình.

F8: Chạy từng câu lệnh một trong chương trình.

Alt-X: Thoát khỏi Turbo Pascal.

Alt-<Số thứ tự của file đang mở>: Dịch chuyển qua lại giữa các file đang mở.

F10: Vào hệ thống Menu của Pascal.

7. Các thao tác cơ bản khi soạn thảo chương trình

7.1. Các phím thông dụng

Insert: Chuyển qua lại giữa chế độ đè và chế độ chèn.

Home: Đưa con trỏ về đầu dòng.

End: Đưa con trỏ về cuối dòng.

Page Up: Đưa con trỏ lên một trang màn hình.

Page Down: Đưa con trỏ xuống một trang màn hình.

Del: Xoá ký tự ngay tại vị trí con trỏ.

Back Space (←): Xóa ký tự bên trái con trỏ.

Ctrl-PgUp: Đưa con trỏ về đầu văn bản.

Ctrl-PgDn: Đưa con trỏ về cuối văn bản.

Ctrl-Y: Xóa dòng tại vị trí con trỏ.

7.2. Các thao tác trên khối văn bản

Chọn khối văn bản: **Shift + <Các phím >**

Ctrl-KY: Xoá khối văn bản đang chọn

Ctrl-Insert: Đưa khối văn bản đang chọn vào Clipboard

Shift-Insert: Dán khối văn từ Clipboard xuống vị trí con trỏ.

BÀI TẬP THỰC HÀNH

1.1. Khởi động Turbo Pascal.

1.2. Nhập vào đoạn chương trình sau:

```
Uses Crt;
Begin
   Writeln('************************************************************');
```

```
    Writeln('*  CHUONG TRINH PASCAL DAU TIEN CUA TOI  *');
    Writeln('*                    Oi! Tuyet voi!...                    *);
    Writeln('************************************************************');
    Readln;
  End.
```

1.3. Viết chương trình in ra màn hình các hình sau:

```
        *              ********              *******
       ***             **      **          **      **
      ** **            **      **          **
     **   **           ********            * *
    *********           **      **          **
   **       **         **      **          **      **
  **         **        ********              ********
```

Chương 2 CÁC KIỂU DỮ LIỆU CHUẨN

1. Kiểu logic

- Từ khóa: **BOOLEAN**
- miền giá trị: **(TRUE, FALSE)**.
- Các phép toán: phép so sánh (=, <, >) và các phép toán logic: AND, OR, XOR, NOT.

Trong Pascal, khi so sánh các giá trị boolean ta tuân theo qui tắc: FALSE < TRUE.

Giả sử A và B là hai giá trị kiểu Boolean. Kết quả của các phép toán được thể hiện qua bảng dưới đây:

A	B	A AND B	A OR B	A XOR B	NOT A
TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE	TRUE

2. Kiểu số nguyên

2.1. Các kiểu số nguyên

Tên kiểu	Phạm vi	Dung lượng
Shortint	-128 127	1 byte
Byte	0 255	1 byte
Integer	-32768 32767	2 byte
Word	0 65535	2 byte
LongInt	-2147483648 2147483647	4 byte

2.2. Các phép toán trên kiểu số nguyên

2.2.1. Các phép toán số học:

+, -, *, / (phép chia cho ra kết quả là số thực).

Phép chia lấy phần nguyên: **DIV** (Ví dụ : 34 DIV 5 = 6).

Phép chia lấy số dư: **MOD** (Ví dụ: 34 MOD 5 = 4).

2.2.2. Các phép toán xử lý bit:

Trên các kiểu ShortInt, Integer, Byte, Word có các phép toán:

NOT, AND, OR, XOR.

A	B	A AND B	A OR B	A XOR B	NOT A
1	1	1	1	0	0
1	0	0	1	1	0
0	1	0	1	1	1
0	0	0	0	0	1

SHL (phép dịch trái): a SHL n a 2^n

SHR (phép dịch phải): a SHR n a DIV 2^n

3. Kiểu số thực

3.1. Các kiểu số thực:

Tên kiểu	Phạm vi	Dung lượng
Single	$1.5 \quad 10^{-45} \quad 3.4 \quad 10^{+38}$	4 byte
Real	$2.9 \quad 10^{-39} \quad 1.7 \quad 10^{+38}$	6 byte
Double	$5.0 \quad 10^{-324} \quad 1.7 \quad 10^{+308}$	8 byte

Extended	$3.4 \ 10^{-4932}$ $1.1 \ 10^{+4932}$	10 byte

Chú ý: Các kiểu số thực Single, Double và Extended yêu cầu phải sử dụng chung với bộ đồng xử lý số hoặc phải biên dich chương trình với chỉ thị **{$N+}** để liên kết bộ giả lập số.

***3.2. Các phép toán trên kiểu số thực:* +, -, *, /**

Chú ý: *Trên kiểu số thực không tồn tại các phép toán DIV và MOD.*

3.3. Các hàm số học sử dụng cho kiểu số nguyên và số thực:

SQR(x): Trả về x^2
SQRT(x): Trả về căn bậc hai của x (x 0)
ABS(x): Trả về |x|
SIN(x): Trả về sin(x) theo radian
COS(x): Trả về cos(x) theo radian
ARCTAN(x): Trả về arctang(x) theo radian
LN(x): Trả về ln(x)
EXP(x): Trả về e^x
TRUNC(x): Trả về số nguyên gần với x nhất nhưng bé hơn x.
INT(x): Trả về phần nguyên của x
FRAC(x): Trả về phần thập phân của x
ROUND(x): Làm tròn số nguyên x
PRED(n): Trả về giá trị đứng trước n
SUCC(n): Trả về giá trị đứng sau n
ODD(n): Cho giá trị TRUE nếu n là số lẻ.
INC(n): Tăng n thêm 1 đơn vị (n:=n+1).
DEC(n): Giảm n đi 1 đơn vị (n:=n-1).

4. Kiểu ký tự

- Từ khoá: **CHAR**.
- Kích thước: 1 byte.
- Để biểu diễn một ký tự, ta có thể sử dụng một trong số các cách sau đây:

  Đặt ký tự trong cặp dấu nháy đơn. Ví dụ 'A', '0'.

  Dùng hàm CHR(n) (trong đó n là mã ASCII của ký tự cần biểu diễn). Ví dụ CHR(65) biểu diễn ký tự 'A'.

  Dùng ký hiệu #n (trong đó n là mã ASCII của ký tự cần biểu diễn). Ví dụ #65.
- Các phép toán: =, >, >=, <, <=,<>.

*** Các hàm trên kiểu ký tự:**

- **UPCASE(ch)**: Trả về ký tự in hoa tương ứng với ký tự ch. Ví dụ: UPCASE('a') = 'A'.
- **ORD(ch)**: Trả về số thứ tự trong bảng mã ASCII của ký tự ch. Ví dụ ORD('A')=65.
- **CHR(n)**: Trả về ký tự tương ứng trong bảng mã ASCII có số thứ tự là n. Ví dụ: CHR(65)='A'.
- **PRED(ch)**: cho ký tự đứng trước ký tự ch. Ví dụ: PRED('B')='A'.
- **SUCC(ch)**: cho ký tự đứng sau ký tự ch. Ví dụ: SUCC('A')='B'.

Chương **3**

KHAI BÁO HẰNG, BIẾN, KIỂU, BIỂU THỨC VÀ CÂU LỆNH

1. KHAI BÁO HẰNG

- Hằng là một đại lượng có giá trị không thay đổi trong suốt chương trình.
- Cú pháp:

CONST <Tên hằng> = <Giá trị>;

hoặc:

CONST <Tên hằng>: = <Biểu thức hằng>;

Ví dụ:

```
CONST Max = 100;
      Name = 'Tran Van Hung';
      Continue = FALSE;
      Logic = ODD(5);    {Logic =TRUE}
```

Chú ý: *Chỉ các hàm chuẩn dưới đây mới được cho phép sử dụng trong một biểu thức hằng*:

ABS CHR HI LO LENGTH ODD ORD
PTR ROUND PRED SUCC SIZEOF SWAP TRUNC

2. KHAI BÁO BIẾN

- Biến là một đại lượng mà giá trị của nó có thể thay đổi trong quá trình thực hiện chương trình.
- Cú pháp:

VAR <Tên biến>[,<Tên biến 2>,...] : <Kiểu dữ liệu>;

Ví dụ:

```
VAR   x, y: Real;  {Khai báo hai biến x, y có kiểu là Real}
      a, b: Integer;  {Khai báo hai biến a, b có kiểu integer}
```

Chú ý: *Ta có thể vừa khai báo biến, vừa gán giá trị khởi đầu cho biến bằng cách sử dụng cú pháp như sau*:

CONST <Tên biến>: <Kiểu> = <Giá trị>;

Ví dụ:

```
CONST x:integer = 5;
```

Với khai báo biến x như trên, trong chương trình giá trị của biến x có thể thay đổi. (Điều này không đúng nếu chúng ta khai báo x là hằng).

3. ĐỊNH NGHĨA KIỂU

- Ngoài các kiểu dữ liệu do Turbo Pascal cung cấp, ta có thể định nghĩa các kiểu dữ liệu mới dựa trên các kiểu dữ liệu đã có.
- Cú pháp:

TYPE <Tên kiểu> = <Mô tả kiểu>;
VAR <Tên biến>:<Tên kiểu>;

Ví dụ:

```
TYPE  Sothuc = Real;
      Tuoi = 1..100;
      ThuNgay = (Hai,Ba,Tu, Nam, Sau, Bay, CN)
VAR   x :Sothuc;
      tt : Tuoi;
```

Day: ThuNgay;

4. BIỂU THỨC

Biểu thức (expression) là công thức tính toán mà trong đó bao gồm các phép toán, các hằng, các biến, các hàm và các dấu ngoặc đơn.

Ví dụ: (x +sin(y))/(5-2*x) biểu thức số học

(x+4)*2 = (8+y) biểu thức logic

Trong một biểu thức, thứ tự ưu tiên của các phép toán được liệt kê theo thứ tự sau:

Lời gọi hàm.

Dấu ngoặc ()

Phép toán một ngôi (NOT, -).

Phép toán *, /, DIV, MOD, AND.

Phép toán +, -, OR, XOR

Phép toán so sánh =, <, >, <=, >=, <>, IN

5. CÂU LỆNH

5.1. Câu lệnh đơn giản

- ***Câu lệnh gán (:=)***: <Tên biến>**:=**<Biểu thức>;
- Các lệnh xuất nhập dữ liệu: **READ/READLN, WRITE/WRITELN.**
- Lời gọi hàm, thủ tục.

5.2. Câu lệnh có cấu trúc

- Câu lệnh ghép: **BEGIN ... END;**
- Các cấu trúc điều khiển: **IF.., CASE..., FOR..., REPEAT..., WHILE...**

Chương 4
VÀO RA DỮ LIỆU

1. Lệnh xuất dữ liệu

Để xuất dữ liệu ra màn hình, ta sử dụng ba dạng sau:

(1)WRITE(<tham số 1> [, <tham số 2>,...]);

(2)WRITELN(<tham số 1> [, <tham số 2>,...]);

(3)WRITELN;

Các thủ tục trên có chức năng như sau:

(1) Sau khi xuất giá trị của các tham số ra màn hình thì con trỏ không xuống dòng.

(2) Sau khi xuất giá trị của các tham số ra màn hình thì con trỏ xuống đầu dòng tiếp theo.

(3) Xuất ra màn hình một dòng trống.

Các tham số có thể là các hằng, biến, biểu thức. Nếu có nhiều tham số trong câu lệnh thì các tham số phải được phân cách nhau bởi dấu phẩy.

Khi sử dụng lệnh WRITE/WRITELN, ta có hai cách viết: ***không qui cách*** và ***có qui cách***:

- ***Viết không qui cách***: dữ liệu xuất ra sẽ được canh lề ở phía bên trái. Nếu dữ liệu là số thực thì sẽ được in ra dưới dạng biểu diễn khoa học.

Ví dụ:

WRITELN(x); WRITE(sin(3*x));

- ***Viết có qui cách***: dữ liệu xuất ra sẽ được canh lề ở phía bên phải.

Ví dụ:

WRITELN(x:5); WRITE(sin(13*x):5:2);

Câu lệnh	Kết quả trên màn hình
Writeln('Hello');	Hello
Writeln('Hello':10);	Hello
Writeln(500);	500
Writeln(500:5);	500
Writeln(123.457)	1.2345700000E+02
Writeln(123.45:8:2)	123.46

2. Nhập dữ liệu

Để nhập dữ liệu từ bàn phím vào các biến có kiểu dữ liệu chuẩn (trừ các biến kiểu BOOLEAN), ta sử dụng cú pháp sau đây:

READLN(<biến 1> [,<biến 2>,...,<biến n>]);

Chú ý: *Khi gặp câu lệnh* **READLN;** *(không có tham số), chương trình sẽ dừng lại chờ người sử dụng nhấn phím ENTER mới chạy tiếp.*

3. Sự kết hợp giữa thủ tục Write vụ Readln:

Kết hợp hai câu lệnh Write và Readln để chỉ dẫn về cách nạp dữ liệu vào.

```
Write('nhập giá trị cho biến); readln(x,y,z);
Write('nhap so a:'); readln(a);
```

4. Các hàm và thủ tục thường dùng trong nhập xuất dữ liệu

Hàm **KEYPRESSED**: Hàm trả về giá trị TRUE nếu như có một phím bất kỳ được nhấn, nếu không hàm cho giá trị là FALSE.

Hàm **READKEY**: Hàm có chức năng đọc một ký tự từ bộ đệm bàn phím.

Thủ tục **GOTOXY(X,Y:Integer)**: Di chuyển con trỏ đến cột X dòng Y.

Thủ tục **CLRSCR**: Xoá màn hình và đưa con trỏ về góc trên bên trái màn hình.

Thủ tục **CLREOL**: Xóa các ký tự từ vị trí con trỏ đến hết dòng.

Thủ tục **DELLINE**: Xoá dòng tại vị trí con trỏ và dồn các dòng ở phía dưới lên.

Thủ tục **TEXTCOLOR(color:Byte)**: Thiết lập màu cho các ký tự. Trong đó color [0,15].

Thủ tục **TEXTBACKGROUND(color:Byte)**: Thiết lập màu nền cho màn hình.

5. Một số thủ tục trình bày màn hình

5.1 Viết ra kiểu số nguyên:

a. ***Cách viết không quy cách:***

Giá trị được viết ra màn hình đúng bằng giá trị thực của nó.

Cú pháp:

Write(I) với I là biến số nguyên

b. ***Cách viết có quy cách:***

Ta có thể ấn định số chỗ để viết ra số nguyên, bằng cách thêm dấu hai chấm (:) theo sau là số để viết. Máy sẽ bố trí sẽ bố trí số nguyên từ phải qua trái. Nếu thừa số nó sẽ bỏ trắng ở bên trái.

Cú pháp: Writeln(I:m);

Giải thích:

- I: Giá trị của I cần in ra.
- m: Số chỗ dành cho số i.

Vídụ: Writeln(123);
Writeln(-23:5);
➔ 123
-23

5.2 Viết ra kiểu số thực:

a. ***Viết không quy cách:***

Cú pháp:

Write(R); với R là biến số thực.

Trong mode bình thường, giá trị của biểu thức R được in ra dưới dấu phẩy động trên 17 vị trí như sau (10 chữ số phần phân trong phần định trị) có dạng:

X.X . .X E XX (đối với số dương)
-X.X . . X E XX(đối với số âm)

Ví dụ:

```
r:=123.456
Writeln(R);
Writeln(3.14);
```

➔ 1.2345600000E+02 3.1400000000E+00

Giải thích:

Hai dấu cách ở đầu, một số phần nguyên, dấu chấm, 10 chỗ phần thập phân (nếu không có giá trị cụ thể thì nó thêm vào số 0), chữ E biểu diễn số mũ, dấu của phần mũ, hai số biểu diễn giá trị phần mũ. Tổng số chỗ mà số thực chiếm là 18.

b. ***Cách viết có quy cách:***

Mẫu1:

Write hay Writeln(R: m)

Giải thích:

+ R: là biến kiểu số thực.
+ m: lá số chỗ cả số thực viết dưới dạng E mũ.
+ Giá trị của n được in ra dưới dạng dấu phẩy động.

Số in ra dạng:

X.X. .X E XX

- XX . .X E XX

Ví dụ:

Writeln(12.345786345:12)

➔ 1.2345786 E 01

Giải thích ví dụ: Số thực in ra dưới dạng E mũ (phần mũ chiếm 4 chỗ), toàn bộ số chỗ là n, thì phần còn lại để viết định trị là n-4 chỗ.

Mẫu 2:

Write(R:m:n)

Giải thích:

R: là biến số thực.

m: là số chỗ cho cả số thực

n: là số chỗ cho phần thập phân.

Nghĩa là giá trị của số thực R được in ra trên ít nhất m vị trí, trong đó có n chữ số phần phân. Khi còn dư thừa vị trí thì số được dồn về bên phải để tạo ra các khoảng trống bên trái.

Ví dụ:

Writeln(12.156789612:12:6);

➔ 12.156789

Giải thích: Phần thập phân được dành 6 chỗ, song nếu không có giá trị cụ thể nó sẽ diền số 0 vào. Còn nếu phân thập phân mà nhiều, nó sẽ làm tròn như ví dụ trên.

5.3 Viết ra màn hình kiểu boolean:

Việc đưa ra giá trị kiểu boolean có thể dùng hai cách giống như kiểu nguyên.

Ví dụ:

```
VAR b:boolean;
BEGIN
   b:=true;
   Writeln(b:2);Writeln(b:7)
END.
```

Ł FALSE
FALSE

5.4 Viết ra màn hình kiểu ký tự: giống kiểu số nguyên

5.5 Đưa dữ liệu ra máy in:

Ta dùng thủ tục Write hay Writeln để đưa dữ liệu ra máy in, bằng cách ta thêm LST và dấu phẩy vào trước các giá trị, các biến cần in ra.

Ví dụ:

```
Writeln(LST, I:4);
```

Chú ý:

Biến LST nằm trong unit PRINTER do vậy ta phải khai báo Printer bằngtừ khoá USES ở đầu chương trình.

BÀI TẬP MẪU

Bài tập 4.1: Viết chương trình nhập vào độ dài hai cạnh của tam giác và góc giữa hai cạnh đó, sau đó tính và in ra màn hình diện tích của tam giác.

Ý tưởng:

Công thức tính diện tích tam giác: S = $\frac{1}{2}a.b.\sin(\ )$ với a,b là độ dài 2 cạnh và là góc kẹp giữa 2 cạnh a và b.

```
Program Tinh_dien_tich_tam_giac;
Var a,b,goc,dientich: Real;
Begin
   Write('Nhap vao do dai canh thu nhat: '); Readln(a);
   Write('Nhap vao do dai canh thu hai: '); Readln(b);
  Write('Nhap vao goc giua hai canh: '); Readln(goc);
   Dientich:=a*b*sin(goc)/2;
   Writeln('Dien tich cua tam giac la: ',Dientich:0:2);
   Readln;
End.
```

Bài tập 4.2: Viết chương trình tính $\sqrt[n]{x}$, x>0.

Ý tưởng:

Ta có: $\sqrt[n]{x} = x^{\frac{1}{n}} = e^{\frac{1}{n}\ln x}$

```
Program Tinh_can_bac_n_cua_x;
Var   x,S: Real;
      n: Word;
Begin
   Write('Nhap vao n= '); Readln(n);
   Write('Nhap vao x= '); Readln(x);
   S:=EXP(1/n*LN(x));
   Writeln('S = ',S:0:2);
   Readln;
End.
```

Bài tập 4.3: Viết chương trình nhập vào 2 số a, b. Sau đó hoán đổi giá trị của 2 số đó:

a/ Cho phép dùng biến trung gian.

```
Program Swap;
Var   a,b,tam: Integer;
Begin
   Write('Nhap vao a= '); Readln(a);
   Write('Nhap vao b= '); Readln(b);
   tam:=a;    {tam lấy giá trị của a}
   a:=b;      {a lấy giá trị của b}
   b:=tam; {b lấy lại giá trị của tam}
```

```
   Writeln('a = ',a,' b = ',b);
   Readln;
End.
```

b/ Không được phép dùng biến trung gian.

```
Program Swap;
Var   a,b: Integer;
Begin
   Write('Nhap vao a= '); Readln(a);
   Write('Nhap vao b= '); Readln(b);
   a:=a+b;     {a lấy tổng giá trị của a+b}
   b:=a-b;  {b lấy giá trị của a}
   a:=a-b;     {a lấy lại giá trị của b}
   Writeln('a = ',a,' b = ',b);
   Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 4.4: Viết chương trình nhập vào các số nguyên: a, b, x, y, ... sau đó in ra màn hình kết quả của các biểu thức sau:

a/ $\frac{x \quad y}{2 \quad \frac{x}{y}}$ b/ $\frac{(a \quad 4)(b \quad 2c \quad 3)}{\frac{r}{2h} \quad 9(a \quad 1)}$ c/ x^y , x>0 d/ $e^{\sqrt{|a \quad \sin^2(x) \quad x|}}$

Bài tập 4.5: Viết chương trình tính siện tích tam giác theo công thức sau:

$S = \sqrt{p(p \quad a)(p \quad b)(p \quad c)}$ với p = $\frac{1}{2}$(a+b+c)

Bài tập 4.6: Viết chương trình tính khoảng cách từ một điểm $I(x_i,y_i)$ đến đường thẳng có phương trình D: Ax + By + C = 0.

Gợi ý:

Công thức tính khoảng cách: $h = \frac{A.x_i \quad B.y_i \quad C}{\sqrt{A^2 \quad B^2}}$

Bài tập 4.7: Viết chương trình tách một số n thành 2 số a, b sao cho tích $P=a*b^2$ đạt cực đại với n được nhập vào từ bàn phím.

Gợi ý:

Gọi x là số thứ hai thì số thứ nhất là: (n-x). Theo đề ta có: $P(x) = x^2.(n-x)$.

Hàm P đạt cực đại khi $P'(x) = -3x^2 + 2nx = 0$ ➔ x = 2n/3.

Bài tập 4.8: Màn hình đồ họa của một máy tính có độ phân giải: 640x480. Biết rằng, mỗi điểm trên màn hình chiếm 1 byte. Hỏi cần bao nhiêu byte để lưu trữ toàn bộ màn hình đồ họa đó?

Có 2 sinh viên viết chương trình tính số byte lưu trữ màn hình đồ họa:

```
Program Sinhvien1;
Var   a,b:integer;
      s:Word;
Begin
```

```
    a:=640; b:=480;
    s:=a*b;
    writeln(s);  readln;
End.

Program Sinhvien2;
Var   a,b:Word;
        s: LongInt;
Begin
    a:=640; b:=480;
    s:=a*b;
    writeln(s);  readln;
End.
```

Hãy cho biết 2 chương trình trên cho kết quả đúng hay sai? Tại sao?

Bài tập 4.9: Màn hình đồ họa của một máy tính có độ phân giải: 640x480. Biết rằng, mỗi điểm trên màn hình chiếm 1 byte. Hỏi cần bao nhiêu byte để lưu trữ một vùng có kích thước bằng 1/10 màn hình đồ họa đó?

Có 2 sinh viên viết chương trình giải bài toán này như sau:

```
Program Sinhvien1;
Var   a,b:Word;
        s: LongInt;
Begin
    a:=640; b:=480;
    s:=a;
    s:=s*b;
    s:=s DIV 10;
    writeln(s);  readln;
End.

Program Sinhvien2;
Var   a,b:Word;
        s: LongInt;
Begin
    a:=640; b:=480;
    s:=a*b DIV 10;
    writeln(s);  readln;
End.
```

Hãy cho biết 2 chương trình trên cho kết quả đúng hay sai? Tại sao?

Chương 5
CẤU TRÚC LỆNH

1. CÂU LỆNH RẼ NHÁNH

1.1. Lệnh IF Cú pháp:

Có hai dạng

Dạng 1:

IF <Bthức lôgic> then <lệnh hoặc dãy câu lệnh >;

Giải thích:

Việc1: Có thể là một lệnh hoặc khối các câu lệnh hợp thành.

Hoạt động:

Nếu <bthứclôgic> có giá trị đúng thì máy sẽ thực hiện <lệnh hoặc câu lệnh > còn không thì máy kết thúc(không thực hiện việc 1), sau đó thoát khỏi câu lệnh IF thực hiện các câu lệnh tiếp theo sau lệnh IF.

Ví dụ: Tìm max của a và b

```
Max:=a;
If  max<b  then  max:=b;
```

Dạng 2:

IF <Bthức lôgic> then < lệnh hoặc dãy câu lệnh 1 >

else < lệnh hoặc dãy câu lệnh >;

Giải thích:

< lệnh hoặc dãy câu lệnh 1 >, < lệnh hoặc dãy câu lệnh >
: Có thể là một lệnh hoặc khối các câu lệnh hợp thành.

Hoạt động:

Nếu <bthứclôgic> có giá trị đúng thì máy sẽ thực hiện < lệnh hoặc dãy câu lệnh 1 > còn nếu sai thì thực hiện < lệnh hoặc dãy câu lệnh 2>, sau đó thoát khỏi câu lệnh IF để thực hiện các câu lệnh tiếp sau câu lệnh IF.

Ví dụ:

Tìm max của hai số nguyên dương a và b

```
If  b<a  then  max :=a
else  max :=b;
```

Chú ý:

- Các If có thể lồng lại với nhau, có nghĩa là bản thân việc1 hoặc việc2 lại có thể chứa các câu lệnh If khác nhưng không được cắt nhau.

- Nếu else ít hơn if thì else được gắn với if không có else gần nhất trước đó(xét ngược từ dưới lên).

Ví dụ: Giải phương trình bậc hai.

```
PROGRAM  G_P_T_B2;
VAR
```

```
    a, b, c: real;
   d, x1,x2: real;
BEGIN
   write('nhap a = ');readln(a);
   write('nhap b = ');readln(b);
     write('nhap c = ');readln(c);
   d:=b*b - 4*a*c;
   if d>0 then
      begin
       writeln('phuong trinh co hai nghiem phan biet ');
       x1:=(-b + sqrt(d))/(2*a);
       x2:=(-b - sqrt(d))/(2*a);
       writeln('nghiem x1 = ',x1:6:2);
       writeln('nghiem x2 = ',x2:6:2);
   end;
  if d=0 then
   begin
       writeln('phuong trinh co mot nghiem duy nhat ');
       x1:=-b/(2*a);
       writeln('nghiem duy nhat x = ',x1:6:2);
   end;
  if d<0 then   writeln('phuong trinh vo nghiem ');
  readln;
END.
```

1.2. Lệnh CASE Cú pháp:

Dạng 1	Dạng 2
CASE B OF	**CASE B OF**
Const 1: S_1;	**Const 1: S_1;**
Const 2: S_2;	**Const 2: S_2;**
...	**...**
Const n: S_n;	**Const n: S_n;**
END;	**ELSE S_{n+1};**
	END;

Trong đó:

- B: Biểu thức kiểu vô hướng đếm được như kiểu nguyên, kiểu logic, kiểu ký tự, kiểu liệt kê.
- Const i: Hằng thứ i, có thể là một giá trị hằng, các giá trị hằng (phân cách nhau bởi dấu phẩy) hoặc các đoạn hằng (dùng hai dấu chấm để phân cách giữa giá trị đầu và giá trị cuối).
- Giá trị của biểu thức và giá trị của tập hằng i (i=1¸n) phải có cùng kiểu.

Khi gặp lệnh CASE, chương trình sẽ kiểm tra:

- Nếu giá trị của biểu thức B nằm trong tập hằng const i thì máy sẽ thực hiện lệnh S_i tương ứng.
- Ngược lại:+ Đối với dạng 1: Không làm gì cả.
  + Đối với dạng 2: thực hiện lệnh S_{n+1}.

2. CÂU LỆNH LẶP

2.1. Vòng lặp xác định Có hai dạng sau:

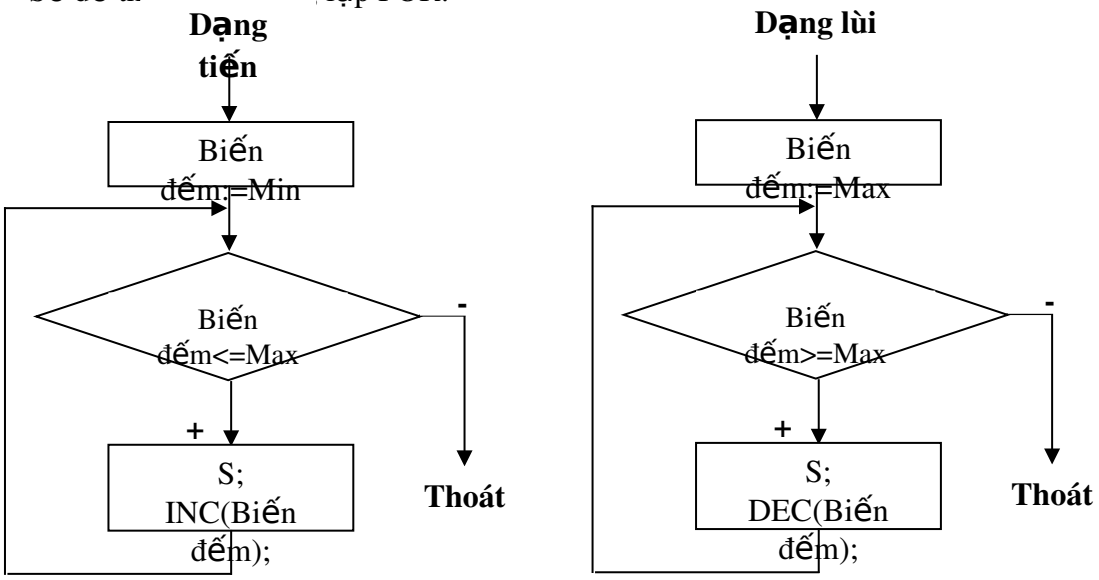
❶ ***Dạng tiến***

FOR <biến đếm>:=<giá trị Min> TO <giá trị Max> DO S;

❷ ***Dạng lùi***

FOR <biến đếm>:=<giá trị Max> DOWNTO <giá trị Min> DO S;

Sơ đồ thực hiện vòng lặp FOR:



Ví dụ 1: Tính tổng s=1+2+3+ . . .+n

```
PROGRAM   TINH_TONG;
Var  s:real;
        I:integer;
BEGIN
        Wrtieln('Nhap so n:');readln(n);
        s:=0;
```