Giáo trình bài tập Pascal

MỤC LỤC

LỜI MỞ ĐẦU

Theo khung chương trình của Bộ Giáo Dục và Đào Tạo, **Ngôn ngữ Lập trình Pascal** là một phần quan trọng trong học phần Tin học Đại cương thuộc các khối ngành Khoa học Tự nhiên, đặc biệt là ngành Công nghệ Thông tin.

Nhằm đáp ứng yêu cầu học tập của học sinh, sinh viên bước đầu làm quen với công việc lập trình, chúng tôi đã biên soạn bộ **Giáo Trình Bài tập 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 Cử nhân Công nghệ Thông tin .

Giáo trình bai gồm rất nhiều bài tập từ đơn giản đến phức tạp. Các bài tập này được biên soạn dựa trên khung chương trình giảng dạy môn **Tin học Đại cương**. 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 10 chương. Trong mỗi chương đều có phần tóm tắt lý thuyết, phần bài tập mẫu và cuối cùng là phần bài tập tự giải để bạn đọc tự mình kiểm tra những kiến thức và kinh nghiệm 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.

Xin chân thành cảm ơn các đồng nghiệp ở Khoa Công nghệ Thông tin Trường Đại học Khoa học Huế đã giúp đỡ, đóng góp ý kiến để hoàn chỉnh nội dung giáo trình này.

Chúng tôi hy vọng sớm nhận được những ý kiến đóng góp, phê bình của bạn đọc về nội dung, chất lượng và hình thức trình bày để giáo trình này ngày một hoàn thiện hơn.

Huế, Tháng 07 Năm 2004

CÁC TÁC GIẢ

Chương 1
CÁC THÀNH PHẦN CƠ BẢN CỦA NGÔN NGỮ LẬP TRÌNH PASCAL

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.

1. 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.

2. 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**).

3. 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.
```

4. 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.

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

5.1. 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.

5.2. 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.

6. Các thành phần cơ bản của ngôn ngữ Pascal

6.1. 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*.

6.2. 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.

6.3. 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.

6.4. 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 ngoạc { 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 *)
```

BÀI TẬP THỰC HÀNH

1. Khởi động Turbo Pascal.
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.
```

3. Dịch và chạy chương trình trên.
4. Lưu chương trình vào đĩa với tên BAI1.PAS.
5. Thoát khỏi Pascal.
6. Khởi động lại Turbo Pascal.
7. Mở file BAI1.PAS.
8. Chèn thêm vào dòng: **CLRSCR;** vào sau dòng **BEGIN**
9. Dịch và chạy thử chương trình.
10. Lưu chương trình vào đĩa.
11. Thoát khỏi Pascal.
12. Viết chương trình in ra màn hình các hình sau:

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

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

Chương 2

CÁC KIỂU DỮ LIỆU CƠ BẢN
KHAI BÁO HẰNG, BIẾN, KIỂU, BIỂU THỨC VÀ CÂU LỆNH

I. CÁC KIỂU DỮ LIỆU CƠ BẢN

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 $\Leftrightarrow$ a $\times$ 2^n
- SHR (phép dịch phải): a SHR n $\Leftrightarrow$ a DIV 2^n

3. Kiểu số thực

3.1. Các kiểu số thực

:

Tên kiểu	Phạm vi	Dung lượng
Single	$1.5\times 10^{-45} \rightarrow 3.4\times 10^{+38}$	4 byte
Real	$2.9\times 10^{-39} \rightarrow 1.7\times 10^{+38}$	6 byte
Double	$5.0\times 10^{-324} \rightarrow 1.7\times 10^{+308}$	8 byte
Extended	$3.4\times 10^{-4932} \rightarrow 1.1\times 10^{+4932}$	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'.

II. 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:

```
CONSTMax = 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 ODDORD
PTR ROUND PRED SUCC SIZEOF SWAP TRUNC

III. 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:

```
CONSTx: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).

IV. 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;
```

V. 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

VI. CÂU LỆNH

6.1. 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.

6.2. Câu lệnh có cấu trúc

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

6.3. Các lệnh xuất nhập dữ liệu

6.3.1. 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

6.3.2. 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.*

6.4. 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 $\in$ [0,15].
- Thủ tục **TEXTBACKGROUND(color:Byte)**: Thiết lập màu nền cho màn hình.

BÀI TẬP MẪU

Bài tập 2.1: 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 = $\frac{1}{2}a.b.\sin(\theta)$ với a,b là độ dài 2 cạnh và θ 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 2.2: 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 2.3: 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 2.4: 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/ $\dfrac{x+y}{2+\dfrac{x}{y}}$ b/ $\dfrac{(a+4)(b-2c+3)}{\dfrac{r}{2h}-9(a-1)}$ c/ x^y , x>0 d/ $e^{\sqrt{|a+\sin^2(x)-x|}}$

Bài tập 2.5: Viết chương trình tính siện tích tam giác theo công thức sau:

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

Bài tập 2.6: 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 = $\dfrac{A.x_i + B.y_i + C}{\sqrt{A^2 + B^2}}$

Bài tập 2.7: 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 2.8: 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 2.9: 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 3
CÁC CÂU LỆNH CÓ CẤU TRÚC

I. CÂU LỆNH RẼ NHÁNH

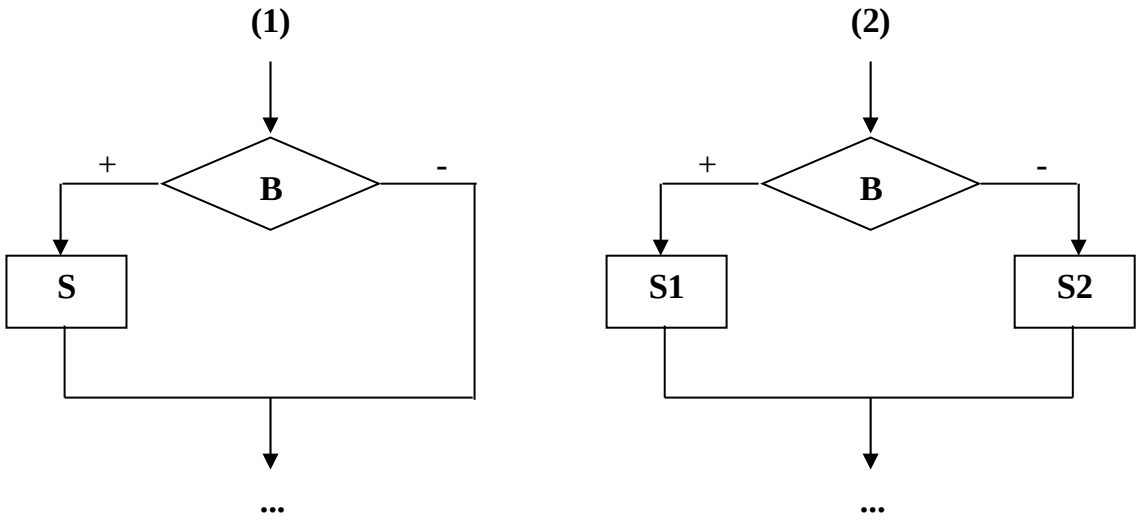
1.1. Lệnh IF

Cú pháp:

(1) IF B THEN S;

(2) IF B THEN S1 ELSE S2;

Sơ đồ thực hiện:



Chú ý: *Khi sử dụng câu lệnh IF thì đứng trước từ khoá ELSE không được có dấu chấm phẩy (;).*

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}.

II. CÂU LỆNH LẶP

2.1. Vòng lặp xác định

Có hai dạng sau:

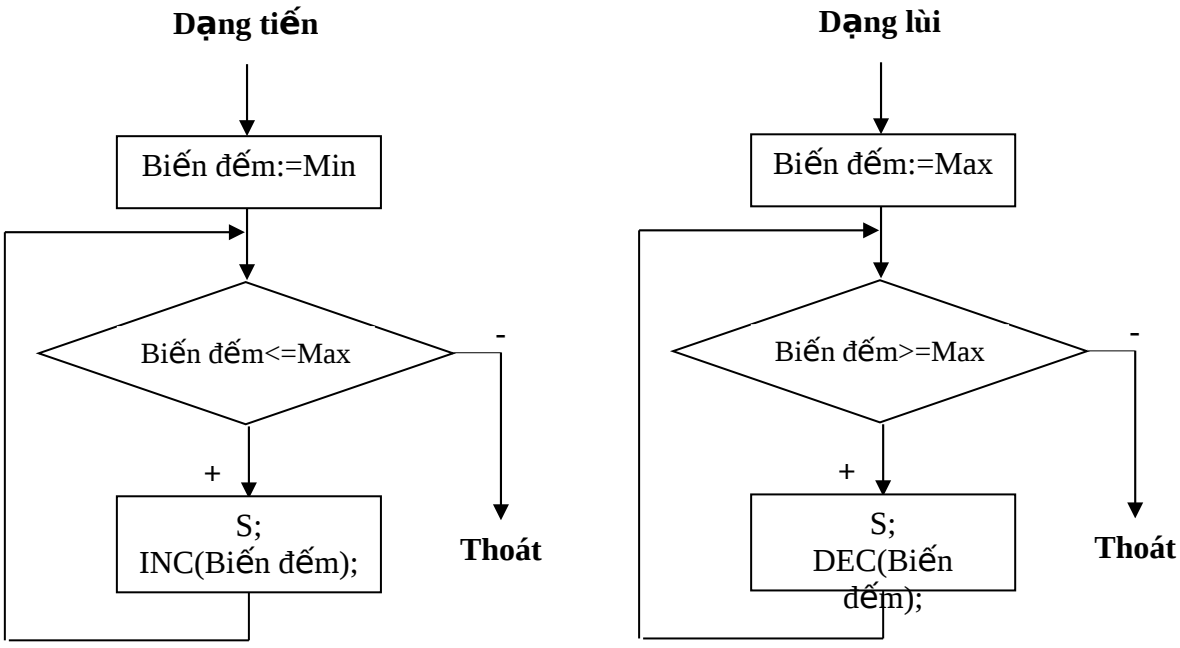
❶ ***Dạng tiến***

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:



Chú ý: Khi sử dụng câu lệnh lặp FOR cần chú ý các điểm sau:

- *Không nên tuỳ tiện thay đổi giá trị của biến đếm bên trong vòng lặp FOR vì làm như vậy có thể sẽ không kiểm soát được biến đếm.*

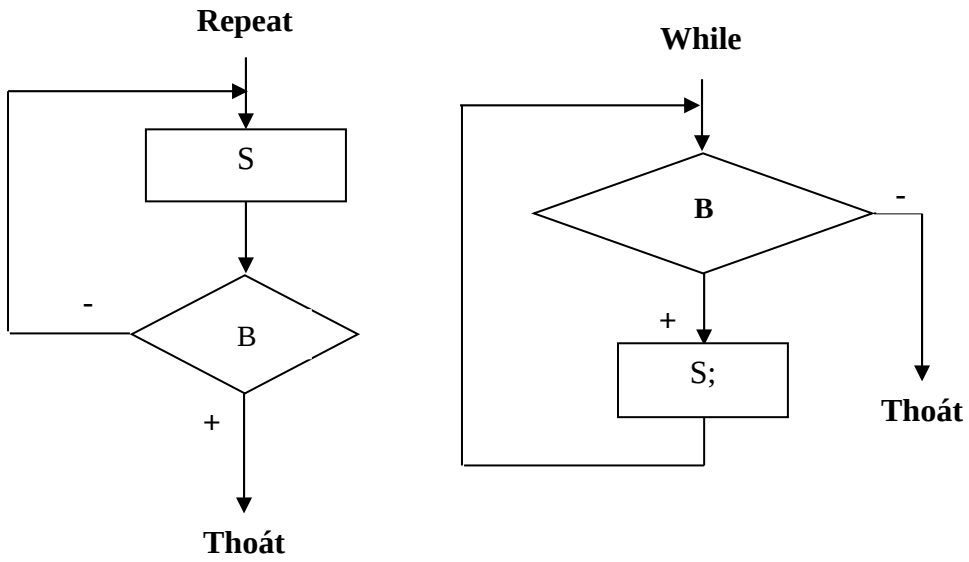
- *Giá trị Max và Min trong câu lệnh FOR sẽ được xác định ngay khi vào đầu vòng lặp. Do đó cho dù trong vòng lặp ta có thay đổi giá trị của nó thì số lần lặp cũng không thay đổi.*

5.3.2. Vòng lặp không xác định

Dạng REPEAT	Dạng WHILE
Repeat **S;** **Until B;**	**While B Do S;**

Ý nghĩa:

- ***Dạng REPEAT***: Lặp lại công việc S cho đến khi biểu thức B=TRUE thì dừng.
- ***Dạng WHILE***: Trong khi biểu thức B=TRUE thì tiếp tục thực hiện công việc S.



BÀI TẬP MẪU

Bài tập 3.1: Viết chương trình nhập vào một số nguyên và kiểm tra xem số vừa nhập là số chẵn hay số lẻ.

```
Uses crt;
Var  x:integer;
Begin
  Write('Nhap vao mot so nguyen : ');  Readln(x);
  If x MOD 2=0 Then
    Writeln('So vua nhap vao la so chan')
  Else
    Writeln('So vua nhap vao la so le');
```

```
  Readln;
End.
```

Bài tập 3.2: Viết chương trình giải phương trình bậc nhất ax+b=0

```
Uses Crt;
Var  a,b,x : real;
Begin
  Write('a = '); Readln(a);
  Write('b = '); Readln(b);
  If a = 0 Then    { Nếu a bằng 0 }
    If b = 0 Then  { Trường hợp a = 0 và b = 0 }
      Writeln('Phuong trinh co vo so nghiem')
    Else   { Trường hợp a=0 và b ≠ 0 }
      Writeln('Phuong trinh vo nghiem')
  Else  { Trường hợp a ≠ 0 }
    Begin
      x:= -b/a;
      Writeln('Phuong trinh co nghiem la :',x:0:2);
    End;
  Readln;
End.
```

Bài tập 3.3: Viết chương trình nhập vào tuổi của một người và cho biết người đó là thiếu niên, thanh niên, trung niên hay lão niên. Biết rằng: nếu tuổi nhỏ hơn 18 là thiếu niên, từ 18 đến 39 là thanh niên, từ 40 đến 60 là trung niên và lớn hơn 60 là lão niên.

```
Uses crt;
Var tuoi:Byte;
Begin
  Write(Nhap vao tuoi cua mot nguoi:');Readln(tuoi);
  Case tuoi Of
    1..17: Writeln(Nguoi nay la thieu nien');
    18..39:Writeln(Nguoi nay la thanh nien');
    40..60:Writeln(Nguoi nay la trung nien');
    Else   Writeln(Nguoi nay la lao nien');
  End;
```

```
  Readln;
End.
```

Bài tập 3.4: Viết chương trình tính tổng S = 1+2+...+N.

Cách 1: Dùng vòng lặp FOR.

```
Program TinhTong;
Uses crt;
Var N,i,S:integer;
Begin
   Clrscr;
   Write('Nhap vao gia tri cua N :'); Readln(N);
   S:=0;
   For i:=1 to N do S:=S+i;
   Writeln('Ket qua la :',S);
   Readln;
End.
```

Cách 2: Dùng vòng lặp REPEAT.

```
Program TinhTong;
Uses crt;
Var N,i,S:integer;
Begin
   Clrscr;
   Write('Nhap vao gia tri cua N :'); Readln(N);
   S:=0; i:=1;
   Repeat
      S:=S+i;
      i:=i+1;
   Until i>N;
   Writeln('Ket qua la :',S);
   Readln;
End.
```

Cách 3: Dùng vòng lặp WHILE.

```
Program TinhTong;
Uses crt;
Var N,i,S:integer;
Begin
```

```
   Clrscr;
   Write('Nhap vao gia tri cua N :'); Readln(N);
   S:=0; i:=1;
   While i<=N Do
      Begin
         S:=S+i;
         i:=i+1;
      End;
   Writeln('Ket qua la :',S);
   Readln;
End.
```

Bài tập 3.5: Viết chương trình nhập vào N số nguyên từ bàn phím. Hãy tính và in ra màn hình tổng của các số vừa được nhập vào.

Ý tưởng:

Dùng phương pháp cộng dồn. Cho vòng lặp FOR chạy từ 1 tới N, ứng với lần lặp thứ i, ta nhập vào số nguyên X và đồng thời cộng dồn X vào biến S.

```
Program  Tong;
Uses crt;
Var N,S,i,X : Integer;
Begin
   Clrscr; S:=0;
   For i:=1 To n Do
      Begin
         Write('Nhap so nguyen X= '); Readln(X);
         S:=S+X;
      End;
   Writeln('Tong cac so duoc nhap vao la: ',S);
   Readln;
End.
```

Bài tập 3.6: Viết chương trình nhập vào các số nguyên cho đến khi nào gặp số 0 thì kết thúc. Hãy đếm xem có bao nhiêu số chẵn vừa được nhập vào.

Ý tưởng:

Bài toán này không biết chính xác số lần lặp nên ta không thể dùng vòng lặp FOR. Vì phải nhập vào số nguyên N trước, sau đó mới kiểm tra xem N=0? Do đó ta nên dùng vòng lặp REPEAT.

```
Program  Nhapso;
Uses crt;
Var N,dem : Integer;
Begin
   Clrscr; dem:=0;
   Repeat
      Write('Nhap vao mot so nguyen N= '); Readln(N);
      If N MOD 2 = 0 Then dem:=dem+1;
   Until N=0;
   Writeln('Cac so chan duoc nhap vao la: ',dem);
   Readln;
End.
```

Bài tập 3.7: Viết chương trình tính số Pi với độ chính xác Epsilon, biết:

Pi/4 = 1-1/3+1/5-1/7+...

Ý tưởng:

Ta thấy rằng, mẫu số là các số lẻ có qui luật: 2*i+1 với i=1,...,n. Do đó ta dùng i làm biến chạy.

Vì tính số Pi với độ chính xác **Epsilon** nên không biết trước được cụ thể số lần lặp, do đó ta phải dùng vòng lặp WHILE hoặc REPEAT. Có nghĩa là phải lặp cho tới khi t=4/(2*i+1) ≤ Epsilon thì dừng.

```
Uses Crt;
Const Epsilon=1E-4;
Var Pi,t:real;
   i,s:Integer;
Begin
   Pi:=4;  i:=1; s:=-1;
   t:=4/(2*i+1);
   While t>Epsilon Do
      Begin
         Pi:=Pi+s*t;
         s:=-s; i:=i+1;
         t:=4/(2*i+1);
      End;
   Writeln('So Pi = ',Pi:0:4);
   Readln;
End.
```

Bài tập 3.8: Viết chương trình nhập vào số nguyên N. In ra màn hình tất cả các ước số của N.

Ý tưởng:

Cho biến i chạy từ 1 tới N. Nếu N MOD i=0 thì viết i ra màn hình.

```
Uses Crt;
Var N,i : Integer;
Begin
   Clrscr;
   Write('Nhap so nguyen N= '); Readln(N);
   For i:=1 To N Do
      If N MOD i=0 Then  Write(i:5);
   Readln;
End.
```

Bài tập 3.9: Viết chương trình tìm USCLN và BSCNN của 2 số a, b được nhập vào từ bàn phím.

Ý tưởng:

- Tìm USCLN: Lấy số lớn trừ số nhỏ cho đến khi a=b thì dừng. Lúc đó: USCLN=a.
- BSCNN(a,b) = a*b DIV USCLN(a,b).

```
Uses crt;
Var a,b,aa,bb:integer;
Begin
   Write('Nhap a : '); Readln(a);
   Write('Nhap b : '); Readln(b);
   aa:=a; bb:=b;
   While aa<>bb Do
      Begin
         If aa>bb Then aa:=aa-bb Else bb:=bb-aa;
      End;
   Writeln('USCLN= ',aa);
   Writeln('BSCNN= ',a*b DIV aa);
   Readln;
End.
```

Bài tập 3.10: Viết chương trình tìm các số có 3 chữ số $\overline{abc}$ sao cho: $\overline{abc} = a^3 + b^3 + c^3$.

Ý tưởng:

Dùng phương pháp vét cạn. Ta biết rằng: a có thể có giá trị từ 1→9 (vì a là số hàng trăm), b,c có thể có giá trị từ 0→9. Ta sẽ dùng 3 vòng lặp FOR lồng nhau để duyệt qua tất cả các trường hợp của a,b,c.

Ứng với mỗi bộ abc, ta sẽ kiểm tra: Nếu $100.a + 10.b + c = a^3 + b^3 + c^3$ thì in ra bộ abc đó.

```
Uses crt;
Var a,b,c : Word;
Begin
   For a:=1 To 9 Do
      For b:=0 To 9 Do
         For c:=0 To 9 Do
            If (100*a + 10*b + c)=(a*a*a + b*b*b + c*c*c) ThenWriteln(a,b,c);
   Readln;
End.
```

Bài tập 3.11: Viết chương trình nhập vào số tự nhiên N rồi thông báo lên màn hình số đó có phải là số nguyên tố hay không.

Ý tưởng:

N là số nguyên tố nếu N không có ước số nào từ 2 →N div 2. Từ định nghĩa này ta đưa ra giải thuật:

- Đếm số ước số của N từ 2 →N div 2 lưu vào biến d.
- Nếu d=0 thì N là số nguyên tố.

```
Uses crt;
Var N,i,d : Word;
Begin
   If N<2 Then Writeln(N,' khong phai la so nguyen to')
   Else
   Begin
      {Đếm số ước số}
      d:=0;
      For i:=2 To N div 2 Do
         If N MOD i=0 Then  d:=d+1;
      {Kiểm tra}
      If d=0 Then Writeln(N,' la so nguyen to')
      Else Writeln(N,' khong phai la so nguyen to');
   End;
```

```
   Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 3.12: Viết chương trình giải phương trình bậc hai: $ax^2 + bx + c = 0, a \neq 0$.

Gợi ý:

- Tính Delta=b*b-4*a*c.
- Biện luận:

  Delta<0: Phương trình vô nghiệm.

  Delta=0: Phương trình có nghiệm kép: x = -b/(2*a).

  Delta>0: Phương trình có 2 nghiệm phân biệt: $x_{1,2}$ = (-b± SQRT(Delta))/(2*a).

Bài tập 3.13: Viết chương trình nhập vào từ bàn phím: giờ, phút, giây. Cộng thêm một số giây cũng được nhập từ bàn phím. Hãy in ra kết quả sau khi cộng xong.

Gợi ý:

- Gọi số giây được cộng thêm là: ss. Gán giây:=giây+ss.
- Nếu giây≥ 60 thì: phút:=phút + giây DIV 60 và giây:=giây MOD 60.
- Nếu phút≥ 60 thì: giờ:=giờ + phút DIV 60 và phút:=phút MOD 60.

Bài tập 3.14: Viết chương trình tìm Max, Min của 4 số: a, b, c, d.

Bài tập 3.15: Viết chương trình nhập vào ngày, tháng, năm. Máy sẽ hiện lên ngày, tháng, năm hôm sau.

Gợi ý:

Biện luận theo tháng. Gom tháng thành 3 nhóm: tháng có 31 ngày (1,3,5,7,8,10,12), tháng có 30 ngày (4,6,9,11) và tháng 2 (có 28 hoặc 29 ngày tùy theo năm nhuận).

Dùng lệnh lựa chọn:

```
CASE thang OF
1,3,5,7,8,10,12: .....
4,6,9,11: .....
2: .....
END;
```

Bài tập 3.16: Viết chương trình in ra màn hình các giá trị của bảng mã ASCII từ 0→255.

Gợi ý:

Cho biến i chạy từ 0 →255. In ra màn hình i và CHR(i).

Bài tập 3.17: Viết chương trình in ra màn hình các số nguyên từ 1 đến 100 sao cho cứ 10 số thì xuống dòng.

Gợi ý:

Cho biến i chạy từ 1 $\rightarrow$ 100. In ra màn hình i và kiểm tra: nếu i MOD 10=0 thì WRITELN.

Bài tập 3.18: Viết chương trình in ra màn hình bảng cữu chương.

Gợi ý:

Dùng 2 vòng lặp FOR lồng nhau: i là số bảng cữu chương (2...9), j là số thứ tự trong từng bảng cữu chương (1...10).

```
For i:=2 To 9 Do
    For j:=1 To 10 Do Writeln(i,'x',j,'=',i*j);
```

Bài tập 3.19: Viết chương trình tính các tổng sau:

S0 = n! = 1*2*...*n {***n giai thừa***}

S1 = 1 + 1/2 + ... + 1/n

S2 = 1 + 1/2! + ... + 1/n!

S3 = 1 + x + x2/2! + x3/3! + ... + xn/n!

S4 = 1 - x + x2/2! - x3/3! + ... + $(-1)^n x^n/n!$

S5 = 1 + sin(x) + $\sin^2(x)$ + ... + $\sin^n(x)$.

Bài tập 3.20: Viết chương trình để tìm lời giải cho bài toán sau:

Trong giỏ vừa thỏ vừa gà,
Một trăm cái cẳng bốn ba cái đầu.
Hỏi có mấy gà mấy thỏ?

Bài tập 3.21: Viết chương trình để tìm lời giải cho bài toán sau:

Trăm trâu trăm bó cỏ
Bó lại cho tròn
Trâu đứng ăn năm
Trâu nằm ăn ba
Năm trâu nghé ăn một.
Hỏi có bao nhiêu trâu đứng, trâu nằm, trâu nghé?

Bài tập 3.22: Viết chương trình nhập vào các số nguyên từ bàn phím cho đến khi nào gặp số nguyên tố thì kết thúc nhập. Tính tổng các số chẵn và trung bình cộng các số lẻ.

Gợi ý:

Dùng vòng lặp REPEAT ... UNTIL NTo; để nhập. Trong đó, NTo là biến kiểu Boolean để kiểm tra số được nhập vào có phải là số nguyên tố hay không.

Bài tập 3.23: Viết chương trình nhập vào một số nguyên dương. Hãy thông báo lên màn hình số đó có bao nhiêu chữ số và tổng các chữ số của số đó.

Gợi ý:

Dùng vòng lặp WHILE. Trong khi N>0 thì: lấy ra chữ số cuối cùng của N để tính bằng phép toán MOD 10, sau đó bỏ bớt đi chữ số cuối cùng của N bằng phép toán DIV 10.

Bài tập 3.24: Viết chương trình in ra màn hình tất cả các số nguyên tố từ 2 đến N. Với N được nhập từ bàn phím.

Bài tập 3.25: Viết chương trình phân tích một số ra thừa số nguyên tố. Ví dụ: N=100 sẽ in ra màn hình:

```
100 | 2
 50 | 2
 25 | 5
  5 | 5
  1 |
```

Bài tập 3.26: Số hoàn thiện là số tự nhiên có tổng các ước của nó (không kể chính nó) bằng chính nó. Viết chương trình kiểm tra xem một số được nhập vào từ bàn phím có phải là số hoàn thiện hay không? Ví dụ: 6, 28 là các số hoàn thiện.

Gợi ý:

- Tính tổng các ước số của N: từ 1 $\rightarrow$N div 2 lưu vào biến S.
- Nếu S=N thì N là số hoàn thiện.

Bài tập 3.27: Viết chương trình in ra các số nguyên từ 1 đến N^2 theo hình xoắn ốc với N được nhập vào từ bàn phím. Ví dụ, với N=5 ta có:

```
 1   2   3   4   5
16  17  18  19   6
15  24  25  20   7
14  23  22  21   8
13  12  11  10   9
```

Chương 4
CHƯƠNG TRÌNH CON: THỦ TỤC VÀ HÀM

I. KHÁI NIỆM VỀ CHƯƠNG TRÌNH CON

Chương trình con (CTC) là một đoạn chương trình thực hiện trọn vẹn hay một chức năng nào đó. Trong Turbo Pascal, có 2 dạng CTC:

- Thủ tục (PROCEDURE): Dùng để thực hiện một hay nhiều nhiệm vụ nào đó.
- Hàm (FUNCTION): Trả về một giá trị nào đó (có kiểu vô hướng, kiểu string hoặc kiểu con trỏ). Hàm có thể sử dụng trong các biểu thức.

Ngoài ra, trong Pascal còn cho phép các CTC lồng vào nhau.

II. CẤU TRÚC CHUNG CỦA MỘT CHƯƠNG TRÌNH CÓ SỬ DỤNG CTC

```
PROGRAM  Tên_chương_trình;
USES CRT;
CONST  .....;
TYPE     .....;
VAR      .....;

PROCEDURE  THUTUC[(Các tham số)];
[Khai báo Const, Type, Var]
BEGIN
   .............
END;

FUNCTION  HAM[(Các tham số)]:<Kiểu dữ liệu>;
[Khai báo Const, Type, Var]
BEGIN
   .............
   HAM:=<Giá trị>;
END;

BEGIN {Chương trình chính}
   ...............
   THUTUC[(...)];
   ...............
   A:= HAM[(...)];
   ...............
END.
```

Chú ý: Trong quá trình xây dựng CTC, khi nào thì nên dùng thủ tục/hàm?

Dùng hàm	Dùng thủ tục
- Kết quả của bài toán trả về ***1 giá trị duy nhất (kiểu vô hướng, kiểu string hoặc kiểu con trỏ).*** - Lời gọi CTC cần nằm trong các biểu thức tính toán.	- Kết quả của bài toán ***không trả về giá trị nào*** hoặc trả về ***nhiều giá trị*** hoặc ***trả về kiểu dữ liệu có cấu trúc (Array, Record, File).*** - Lời gọi CTC không nằm trong các biểu thức tính toán.

Ví dụ 1: Viết CTC để tính n! = 1.2...n.
Ý tưởng: Vì bài toán này trả về 1 giá trị duy nhất nên ta dùng hàm.

```
Function GiaiThua(n:Word):Word;
Var P, i:Word;
Begin
   P:=1;
   For i:=1 To n Do P:=P*i;
   GiaiThua:=P;
End;
```

Ví dụ 2: Viết chương trình con để tìm điểm đối xứng của điểm (x,y) qua gốc tọa độ.
Ý tưởng: Vì bài toán này trả về tọa độ điểm đối xứng (xx,yy) gồm 2 giá trị nên ta dùng thủ tục.

```
Procedure DoiXung(x,y:Integer; Var xx,yy:Integer);
Begin
   xx:=-x;
   yy:=-y;
End;
```

CHÚ Ý: Trong 2 ví dụ trên:

- **n, x, y** được gọi là **tham trị** (*không có từ khóa* **var** *đứng trước*) vì sau khi ra khỏi CTC giá trị của nó ***không bị thay đổi***.
- **xx, yy** được gọi là **tham biến** (*có từ khóa* **var** *đứng trước*) vì sau khi ra khỏi CTC giá trị của nó ***bị thay đổi***.

III. BIẾN TOÀN CỤC VÀ BIẾN ĐỊA PHƯƠNG

- **Biến toàn cục**: là các biến được khai báo trong chương trình chính. Các biến này có tác dụng ở mọi nơi trong toàn bộ chương trình.
- **Biến địa phương**: là các biến được khai báo trong các CTC. Các biến này chỉ có tác dụng trong phạm vi CTC đó mà thôi.

Chú ý: Trong một CTC, nếu biến toàn cục trùng tên với biến địa phương thì biến địa phương được ưu tiên hơn.

Ví dụ:

```
Program  KhaoSatBien;
Var a,b: Integer;  {biến toàn cục}

Procedure  ThuBien;
Var  a: Integer;{biến địa phương}
Begin
   a:=10;
   Writeln('A=',a,'B=',b);
End;

Begin
   a:=50;
   b:=200;
   ThuBien;                  {A=10  B=200}
   Writeln('A=',a,'B=',b);   {A=50  B=200}
End.
```

IV. ĐỆ QUI

4.1. Khái niệm đệ qui

Trong một chương trình, một CTC có thể gọi một CTC khác vào làm việc. Nếu như CTC đó ***gọi lại chính nó*** thì gọi là sự đệ qui.

4.2. Phương pháp thiết kế giải thuật đệ qui

- Tham số hóa bài toán
- Tìm trường hợp suy biến.
- Phân tích các trường hợp chung (đưa về các bài toán cùng loại nhưng nhỏ hơn).

Ví dụ: Viết hàm đệ qui để tính n! = 1.2...n.

- Tham số hóa: n! = Factorial(n);
- Factorial(0) = 1 (trường hợp suy biến)
- Factorial(n) = n*Factorial(n-1) (trường hợp chung)

```
Function  Factorial(N:integer):Longint;
Begin
   If N=0 Then  Factorial:=1
   Else Factorial:=N*factorial(N-1);   { lời gọi đệ qui }
End;
```

4.3. Giải thuật quay lui

Bài toán:

Hãy xây dựng các bộ giá trị gồm n thành phần (x_1,...,x_n) từ một tập hữu hạn cho trước sao cho các bộ đó thỏa mãn yêu cầu B cho trước nào đó.

Phương pháp chung

Giả sử đã xác định được k-1 phần tử đầu tiên của dãy: x_1,...,x_{k-1}. Ta cần xác định phần tử thứ k. Phần tử này được xác định theo cách sau:

- Giả sử T_k: tập tất cả các giá trị mà phần tử x_k có thể nhận được. Vì tập T_k hữu hạn nên ta có thể đặt n_k là số phần tử của T_k theo một thứ tự nào đó, tức là ta có thể thành lập một ánh xạ 1-1 từ tập T_k lên tập {1, 2, ..., n_k}.

- Xét j∈ {1, 2, ..., n_k}. Ta nói rằng "***j chấp nhận được***" nếu ta có thể bổ sung phần tử thứ j trong T_k với tư cách là phần tử x_k vào trong dãy x_1,...,x_{k-1} để được dãy x_1,...,x_k.

- Nếu k=n: Bộ (x_1,...,x_k) thỏa mãn yêu cầu B, do đó bộ này được thu nhận.

- Nếu k<n: Ta thực hiện tiếp quá trình trên, tức là phải bổ sung tiếp các phần tử x_{k+1} vào dãy x_1,...,x_k.

Sau đây là thủ tục đệ qui cho giải thuật quay lui:

```
Procedure THU(k:Integer);
Var j:Integer;
Begin
   For j:=1 To nk Do
      If <j chấp nhận được> Then
         Begin
            <Xác định xk theo j>;
            If k=n Then  <Ghi nhận một bộ giá trị>
            Else  THU(k+1);  {Quay lui}
         End;
End;
```

Ví dụ: Liệt kê các dãy nhị phân có độ dài n.

```
Program DayNhiPhan;
Var  b:Array[1..20] Of 0..1;  {Dãy nhị phân có độ dài tối đa là 20}
       n:Byte;

Procedure InKetQua;
Var i:Byte;
Begin
   For i:=1 To n Do  Write(b[i]);
   Writeln;
End;
```

```
Procedure THU(k:Byte);
Var j:Byte;
Begin
   For j:=0 To 1 Do      {Tập giá trị của dãy nhị phân}
      Begin
         b[k]:= j;
         If k=n Then  InKetQua
         Else  THU(k+1);  {Quay lui}
      End;
End;

Begin
   Write('n = '); Readln(n);
   THU(1);
   Readln;
End.
```

V. TẠO THƯ VIỆN (UNIT)

5.1. Cấu trúc của một Unit

```
UNIT  <Tên Unit>;       {phải trùng với tên file}
INTERFACE
   USES .....;
   CONST.....;
   TYPE .....;
   VAR  .....;
   Procedure  <Tên thủ tục>[(Các tham số)];
   Function  <Tên hàm>[(Các tham số)]:<Kiểu hàm>;
IMPLEMENTATION
   Procedure  <Tên thủ tục>[(Các tham số)];
   [Các khai báo]
   Begin
      .....
   End;

   Function  <Tên hàm>[(Các tham số)]:<Kiểu hàm>;
   [Các khai báo]
   Begin
      .....
```

```
   End;
END.
```

Chú ý:

- Tên của Unit phải trùng với tên file.
- Chỉ có những chương trình con được khai báo ở phần INTERFACE mới sử dụng được ở các chương trình khác.
- Các thủ tục và hàm được khai báo ở phần INTERFACE thì bắt buộc phải có trong phần IMPLEMENTATION.

5.2. Ví dụ minh họa

Tạo Unit MYTOOL lưu ở file MYTOOL.PAS.

```
UNIT  MYTOOL;
INTERFACE
   USES CRT;
   VAR  m:Integer;
   Procedure  WriteXY(x,y:Integer; St:String);
   Function  UCLN(a,b:Integer):Integer;
   Function  NGUYENTO(n:Word):Word;
IMPLEMENTATION
   Procedure  WriteXY(x,y:Integer; St:String);
   Var i:Byte;
   Begin
      Gotoxy(x,y); Write(St);
   End;
   Function  UCLN(a,b:Integer):Integer;
   Begin
      While a<>b Do
         Begin
            If a>b Then a:=a-b  Else  b:=b-a;
         End;
      UCLN:=a;
   End;
   Function  NGUYENTO(n:Word):Boolean;
   Var d,i:Word;
   Begin
      d:=0;
      For i:=2 To n DIV 2 Do
```

```
        If n MOD i=0 Then d:=d+1;
      NGUYENTO:=d=0;
   End;
END.
```

Bây giờ, ta có thể viết một chương trình có sử dụng Unit MYTOOL.

```
Uses Crt, MyTool;
Var  a,b:Integer;
Begin
   CLRSCR;
   Write(10,5,'CHUONG TRINH MINH HOA');
   Write('Nhap a = '); Readln(a);
   Write('Nhap b = '); Readln(b);
   Writeln('UCLN cua ',a,' va ',b,' la:',UCLN(a,b));
   Write('Nhap m = '); Readln(m);
   If  NGUYENTO(m) Then
      Writeln(m,' la so nguyen to!')
   Else
      Writeln(m,' khong phai la so nguyen to!')
   Readln;
End.
```

BÀI TẬP MẪU

Bài tập 4.1: Viết hàm tìm Max của 2 số thực x,y.

```
Var a,b:Real;
Function Max(x,y:Real):Real;
Begin
   If x>y Then Max:=x Else Max:=y;
End;
Begin
   Write('Nhap a='); Readln(a);
   Write('Nhap b='); Readln(b);
   Writeln('So lon nhat trong 2 so la: ', Max(a,b));
   Readln;
End.
```

Bài tập 4.2: Viết hàm LOWCASE(c:char):char; để đổi chữ cái hoa c thành chữ thường.

Ý tưởng:

Trong bảng mã ASCII, số thứ tự của chữ cái hoa nhỏ hơn số thứ tự của chữ cái thường là 32. Vì vậy ta có thể dùng 2 hàm CHR và ORD để chuyển đổi.

```
Uses crt;
Var ch:Char;
Function LOWCASE(c:Char):Char;
Begin
   If c IN ['A'..'Z'] Then LOWCASE:=CHR(ORD(c)+32)
   Else LOWCASE:=c;
End;
Begin
   Write('Nhap ký tu ch='); Readln(ch);
   Writeln('Ky tu hoa la: ', LOWCASE(ch));
   Readln;
End.
```

Bài tập 4.3: Viết thủ tục để hoán đổi hai gía trị x,y cho nhau.

```
Var a,b:Real;
Function Swap(Var x,y:Real);
Var Tam:Real;
Begin
   Tam:=x; x:=y; y:=Tam;
End;
Begin
   Write('Nhap a='); Readln(a);
   Write('Nhap b='); Readln(b);
   Swap(a,b);
   Writeln('Cac so sau khi hoan doi:  a=', a:0:2,' b=',b:0:2);
   Readln;
End.
```

Bài tập 4.4: Viết hàm XMU(x:Real;n:Byte):Real; để tính giá trị x^n.

```
Var  x:Real;
     n:Byte;
```

```
Function XMU(x:Real;n:Byte):Real;
Var i:Byte; S:Rea;
Begin
   S:=1;
   For i:=1 To n Do S:=S*x;
   XMU:=S;
End;
Begin
   Write('Nhap x='); Readln(x);
   Write('Nhap n='); Readln(n);
   Writeln('x mu n = ', XMU(x,n):0:2);
   Readln;
End.
```

Bài tập 4.5: Viết thủ tục KHUNG(x1,y1,x2,y2:Integer); để vẽ một khung hình chữ nhật có đỉnh trên bên trái là (x1,y1) và đỉnh dưới bên phải là (x2,y2).

Ý tưởng:

Dùng các ký tự mở rộng trong bảng mã ASCII: │(#179), ─(#196), ┌(#218), └(#192), ┐(#191), ┘(#217).

```
Uses crt;
Procedure Khung(x1,y1,x2,y2:Integer);
Var i,j:Integer;
Begin
  Gotoxy(x1,y1); Write(#218); {Vẽ ┌}
  Gotoxy(x1,y2); Write(#192); {Vẽ └}
 {Vẽ 2 viền ngang của khung}
  For i:=x1+1 To x2-1 do
   Begin
    Gotoxy(i,y1); Write(#196);
    Gotoxy(i,y2); Write(#196);
   End;

  Gotoxy(x2,y1); Write(#191); {Vẽ ┐}
  Gotoxy(x2,y2); Write(#217); {Vẽ ┘}
 {Vẽ 2 viền dọc của khung}
  For j:=y1+1 To y2-1 do
   Begin
    Gotoxy(x1,j); Write(#179);
    Gotoxy(x2,j); Write(#179);
   End;
End;
```

```
Begin
  Clrscr;
  Khung(10,5,40,20);
  Readln;
End.
```

Bài tập 4.6: Viết thủ tục PHANTICH(n:Integer); để phân tích số nguyên n ra thừa số nguyên tố.

```
Uses crt;
Var n:Integer;

Procedure PHANTICH(n:Integer);
Var i:Integer;
Begin
  i:=2;
  While n<>1 Do
   Begin
     While n MOD i=0 Do
      Begin
        Writeln(n:5,'|',i:2);
        n:=n Div i;
      End;
     i:=i+1;
   End;
  Writeln(n:5,'|');
End;

Begin
  Write('Nhap n='); Readln(n);
  PHANTICH(n);
  Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 4.7: Viết 2 hàm tìm Max , min của 3 số thực.

Bài tập 4.8: Viết hàm PERFECT(n:Word):Boolean; để kiểm tra số nguyên n có phải là số hoàn thiện hay không?

Bài tập 4.9: Viết thủ tục FILL(x1,y1,x2,y2:Integer; ch:Char); để tô một vùng màn hình hình chữ nhật có đỉnh trên bên trái là (x1,y1) và đỉnh dưới bên phải là (x2,y2) bằng các ký tự ch.

Bài tập 4.10: Viết hàm tìm BSCNN của 2 số nguyên a,b được khai báo như sau:
Function BSCNN (a,b:word):word ;

Bài tập 4.11: Viết thủ tục để tối giản phân số a/b , với a, b là 2 số nguyên.

Bài tập 4.12: Viết các hàm đệ quy để tính:
$S_1 = 1+2+3+.....+n$;
$S_2 = 1+1/2 + ...+ 1/n$;
$S_3 = 1-1/2 +.....+ (-1)^{n+1}\ 1/n$
$S_4 = 1 + \sin(x) + \sin^2(x) + ...+ \sin^n(x)$

Bài tập 4.13: Viết hàm đệ quy để tính C^k_n biết :
$C^n_n = 1$, $C^0_n = 1$, $C^k_n = C^{k-1}_{n-1} + C^k_{n-1}$.

Bài tập 4.14: Cho m , n nguyên dương . Lập hàm đệ quy tính:

$$A(m,n) = \begin{cases} n+1 & , \quad m = 0 \\ A(m-1,1) & , \quad n = 0 \\ A(m-1, A(m, n-1)) & , \quad m > 0 \wedge n > 0 \end{cases}$$

Bài tập 4.15: Lập hàm đệ qui để tính dãy Fibonaci:

$$F(n) = \begin{cases} 1 & , \quad n = 1 \vee n = 2 \\ F(n-1) + F(n-2) & , \quad n > 2 \end{cases}$$

Bài tập 4.16: Viết hàm đệ qui tìm USCLN của 2 số.

Bài tập 4.17: Viết thủ tục để in ra màn hình số đảo ngược của một số nguyên cho trước theo 2 cách: đệ qui và không đệ qui.

Bài tập 4.18: Viết chương trình in ra màn hình các hoán vị của n số nguyên đầu tiên.

Bài tập 4.19: Xây dựng một Unit SOHOC.PAS chứa các thủ tục và hàm thực hiện các chức năng sau:
- Giải phương trình bặc nhất.
- Giải phương trình bặc hai.
- Tìm Max/Min của 2 số a,b.
- Tìm USCLN và BSCNN của 2 số nguyên a,b.
- Kiểm tra số nguyên dương n có phải là số nguyên tố hay không?
- Kiểm tra số nguyên dương n có phải là số hoàn thiện hay không?
- Đổi một số nguyên dương n sang dạng nhị phân.
- In ra màn hình bảng cữu chương từ $2 \rightarrow 9$.

Sau đó, tự viết các chương trình có sử dụng Unit SOHOC vừa được xây dựng ở trên.

Chương 5
DỮ LIỆU KIỂU MẢNG (ARRAY)

I. KHAI BÁO MẢNG

Cú pháp:

TYPE <Kiểu mảng> = ARRAY [chỉ số] OF <Kiểu dữ liệu>;

VAR <Biến mảng>:<Kiểu mảng>;

hoặc khai báo trực tiếp:

VAR <Biến mảng> : ARRAY [chỉ số] OF <Kiểu dữ liệu>;

Ví dụ:

```
TYPE  Mangnguyen = Array[1..100] of Integer;
      Matrix = Array[1..10,1..10] of Integer;
      MangKytu = Array[Byte] of Char;
VAR   A: Mangnguyen;
      M: Matrix;
      C: MangKytu;
```

hoặc:

```
VAR   A: Array[1..100] of Integer;
      C: Array[Byte] of Char;
```

II. XUẤT NHẬP TRÊN DỮ LIỆU KIỂU MẢNG

- Để truy cập đến phần tử thứ k trong mảng một chiều A, ta sử dụng cú pháp: A[k].
- Để truy cập đến phần tử (i,j) trong mảng hai chiều M, ta sử dụng cú pháp: M[i,j].
- Có thể sử dụng các thủ tục READ(LN)/WRITE(LN) đối với các phần tử của biến kiểu mảng.

BÀI TẬP MẪU

Bài tập 5.1: Viết chương trình tìm giá trị lớn nhất của một mảng chứa các số nguyên gồm N phần tử.

Ý tưởng:

- Cho số lớn nhất là số đầu tiên: Max:=a[1].
- Duyệt qua các phần tử a[i], với i chạy từ 2 tới N: Nếu a[i]>Max thì thay Max:=a[i];

```
Uses Crt;
```

```
Type Mang = ARRAY[1..50] Of Integer;
Var  A:Mang;
     N,i,Max:Integer;
Begin
   {Nhập mảng}
   Write('Nhap N='); Readln(N);
   For i:=1 To N Do
      Begin
         Write('A[',i,']='); Readln(A[i]);
      End;
   {Tìm phần tử lớn nhất}
   Max:=A[1];
   For i:=2 To N Do
      If Max<A[i] Then Max:=A[i];
   {In kết quả ra màn hình}
   Writeln('Phan tu lon nhat cua mang: ', Max);
   Readln;
End.
```

Bài tập 5.2: Viết chương trình tính tổng bình phương của các số âm trong một mảng gồm N phần tử.

Ý tưởng:

Duyệt qua tất cả các phần tử A[i] trong mảng: Nếu A[i]<0 thì cộng dồn $(A[i])^2$ vào biến S.

```
Uses Crt;
Type Mang = ARRAY[1..50] Of Integer;
Var  A:Mang;
     N,i,S:Integer;
Begin
   {Nhập mảng}
   Write('Nhap N='); Readln(N);
   For i:=1 To N Do
      Begin
         Write('A[',i,']='); Readln(A[i]);
      End;
   {Tính tổng}
   S:=0;
```

```
   For i:=1 To N Do
      If A[i]<0 Then S:=S+A[i]*A[i];
   {In kết quả ra màn hình}
   Writeln('S= ', S);
   Readln;
End.
```

Bài tập 5.3: Viết chương trình nhập vào một mảng gồm N số nguyên. Sắp xếp lại mảng theo thứ tự tăng dần và in kết quả ra màn hình.

Ý tưởng:

Cho biến i chạy từ 1 đến N-1, đồng thời cho biến j chạy từ i+1 đến N: Nếu A[i]>A[j] thì đổi chổ A[i], A[j].

```
Uses Crt;
Type Mang = ARRAY[1..50] Of Integer;
Var  A:Mang;
     N,i,j,Tam:Integer;
Begin
   {Nhập mảng}
   Write('Nhap N='); Readln(N);
   For i:=1 To N Do
      Begin
         Write('A[',i,']='); Readln(A[i]);
      End;
   {Sắp xếp}
   For i:=1 To N-1 Do
      For j:=i+1 To N Do
         If A[i]>A[j] Then
            Begin
               Tam:=A[i]; A[i]:=A[j]; A[j]:=Tam;
            End;
   {In kết quả ra màn hình}
   Writeln('Ket qua sau khi sap xep:');
   For i:=1 To N Do Write(A[i]:5);
   Readln;
End.
```

Bài tập 5.4: Viết chương trình nhập vào một mảng A gồm N số nguyên và nhập thêm vào một số nguyên X. Hãy kiểm tra xem phần tử X có trong mảng A hay không?

Ý tưởng:

Dùng thuật toán tìm kiếm tuần tự. So sánh x với từng phần tử của mảng A. Thuật toán dừng lại khi x=A[i] hoặc i>N.

Nếu x=A[i] thì vị trí cần tìm là i, ngược lại thì kết quả tìm là 0 (không tìm thấy).

```
Uses Crt;
Type Mang = ARRAY[1..50] Of Integer;
Var  A:Mang;
     N,i,x:Integer;

Function TimKiem(x, N: Integer; A:Mang):Integer;
Var i:Integer;
  Begin
    I:=1;
    While (I <= N) and (X<>A[I]) do I:=I+1;
    If I <= N Then Timkiem:=I  Else Timkiem:=0;
  End;

Begin
  {Nhập mảng}
  Write('Nhap N='); Readln(N);
  For i:=1 To N Do
    Begin
      Write('A[',i,']='); Readln(A[i]);
    End;
  Write('Nhap X='); Readln(x);
  {Kết quả tìm kiếm}
  If TimKiem(X,N,A)<>0 Then
    Writeln('Vi tri cua X trong mang la:', TimKiem(X,N,A))
  Else Writeln('X khong co trong mang.');
  Readln;
End.
```

Bài tập 5.5: Giả sử mảng A đã được sắp xếp theo thứ tự tăng dần. Viết hàm để kiểm tra xem phần tử X có trong mảng A hay không?

Ý tưởng:

So sánh x với phần tử ở giữa mảng A[giua]. Nếu x=A[giua] thì dừng (vị trí cần tìm là chỉ số của phần tử giữa của mảng). Ngược lại, nếu x>A[giua] thì tìm ở đoạn sau của mảng [giua+1,cuoi], ngược lại thì tìm ở đoạn đầu của mảng [dau,giua-1].
Sau đây là hàm cài đặt cho thuật toán này:

```
Function TimKiemNhiPhan(X, N: Integer; A: Mang):Integer;
Var  dau,cuoi,giua:Integer;
     Found:Boolean;
Begin
  dau:=1; {điểm mút trái của khoảng tìm kiếm}
  cuoi:=N; {điểm mút phải của khoảng tìm kiếm}
  Found:=False; {chưa tìm thấy}
  While (dau <=cuoi) and (Not Found) Do
    Begin
      giua:=(dau + cuoi) Div 2;
      If  X = A[giua] Then Found:=True {đã tìm thấy}
      Else
        If X > A[giua] Then dau:=giua+1
        Else cuoi:=giua-1;
    End;
  If Found Then TimKiemNhiPhan:= giua Else TimKiemNhiPhan:=0;
End;
```

Bài tập 5.6: Viết chương trình tìm ma trận chuyển vị của ma trận A.

Ý tưởng:

Dùng mảng 2 chiều để lưu trữ ma trận. Gọi B là ma trận chuyển vị của ma trận A, ta có: $B_{ij} = A_{ji}$.

```
Uses Crt;
Type Mang = ARRAY[1..10,1..10] Of Integer;
Var  A,B:Mang;
     m,n,i,j:Integer;
Begin
  {Nhập ma trận}
  Write('Nhap số dòng m='); Readln(m);
  Write('Nhap số cột n='); Readln(n);
  For i:=1 To m Do
    For j:=1 To n Do
      Begin
        Write('A[',i,j,']='); Readln(A[i,j]);
      End;
```

```
   {Tìm ma trận chuyển vị}
   For i:=1 To m Do
      For j:=1 To n Do  B[i,j]:=A[j,i];
   {In ma trận chuyển vị ra màn hình}
   For i:=1 To m Do
      Begin
         For j:=1 To n Do  Write(B[i,j]:5);
         Writeln;
      End;
   Readln;
End.
```

Bài tập 5.7: Cho một mảng 2 chiều A cấp mxn gồm các số nguyên và một số nguyên x. Viết chương trình thực hiện các công việc sau:

a/ Đếm số lần xuất hiện của x trong A và vị trí của chúng.

b/ Tính tổng các phần tử lớn nhất của mỗi dòng.

```
Uses Crt;
Type Mang = ARRAY[1..10,1..10] Of Integer;
Var  A:Mang;
      m,n,i,j,x,dem,S,max:Integer;
Begin
   {Nhập ma trận}
   Write('Nhap số dòng m='); Readln(m);
   Write('Nhap số cột n='); Readln(n);
   For i:=1 To m Do
      For j:=1 To n Do
         Begin
            Write('A[',i,j,']='); Readln(A[i,j]);
         End;
   {Nhập x}
   Write('Nhap x='); Readln(x);
   {Đếm số lãn xuất hiện của x và vị trí của x}
   dem:=0;
   Writeln('Vi tri cua x trong mang A: ');
   For i:=1 To m Do
      For j:=1 To n Do
         If x=A[i,j] Then
```

```
            Begin
               Write(i,j,' ; ');
               dem:=dem+1;
            End;
   Writeln('So lan xuat hien cua x trong mang A la: ',dem);
   {Tính tổng các phần tử lớn nhất của mỗi dòng}
   S:=0;
   For i:=1 To m Do {duyệt qua từng dòng}
      Begin
         {Tìm phần tử lớn nhất của dòng thứ i}
         Max:=A[i,1];
         For j:=2 To n Do  {duyệt từng phần tử của dòng thứ i}
            If max<A[i,j] Then max:=A[i,j];
         {Cộng max vào biến S}
         S:=S+max;
      End;
   Writeln('Tong cac phan tu lon nhat cua moi dong la: ',S);
   Readln;
End.
```

Bài tập 5.8: Giải phương trình bằng phương pháp chia nhị phân.

Ý tưởng:

Giả sử cần tìm nghiệm của phương trình f(x)=0 trên đoạn [a,b] với y=f(x) đồng biến và đơn trị trên đoạn [a,b]. Ta giải như sau:

Gọi m là trung điểm của đoạn [a,b]. Nếu f(m)*f(a)<0 thì giới hạn đoạn tìm nghiệm thành [a,m]. Tương tự đối với đoạn [m,b]. Quá trình này lặp lại cho đến khi f(m)<ε, lức này ta có 1 nghiệm gần đúng là m.

Giả sử f(x) là một đa thức: $f(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$. Lúc này, ta có thể dùng mảng một chiều để lưu trữ các hệ số a_i của đa thức.

```
Uses Crt;
Type HESO=Array[0..20] Of Real;
Var  a:HESO;
     n:Byte;
     Min,Max,epsilon:Real;

Procedure NhapDaThuc;
Var i:Byte;
Begin
  Write('Bac cua da thuc: n= '); Readln(n);
```

```
  Writeln('Nhap cac he so cua da thuc:');
  For i:=0 To n Do
   Begin
     Write('a[',i,']='); Readln(a[i]);
   End;
  Writeln('Nhap doan tim nghiem:[a,b]');
  Write('a= '); Readln(Min);
  Write('b= '); Readln(Max);
  Write('Nhap sai so cua phuong trinh: '); Readln(epsilon);
End;

{Tính giá trị của đa thức}
Function f(x:Real):Real;
Var   S,tam:Real;
        i:Byte;
Begin
    S:=a[0]; tam:=1;
    For i:=1 To n Do
        Begin
            tam:=tam*x;
            S:=S+a[i]*tam;
        End;
    f:=S;
End;

Procedure TimNghiem(Min,Max:real);
Var m:Real;
Begin
  If f(Min)*f(Max)>0 Then Writeln('Phuong trinh vo nghiem.')
  Else If abs(f(Min))<epsilon Then Writeln('Nghiem la x=',min:0:2)
       Else If abs(f(Max))<epsilon Then Writeln('Nghiem la x=',max:0:2)
            Else
              Begin
                m:=(Min+Max)/2;
                If abs(f(m))<=epsilon Then Writeln('Nghiem la x=',m:0:2)
                Else If f(Min)*f(m)<0 Then TimNghiem(Min,m)
                        Else TimNghiem(m,Max);
              End;
End;

Begin
 NhapDaThuc;
 TimNghiem(Min,Max);
 Readln;
End.
```

Bài tập 5.9: Viết chương trình nhập vào số tự nhiên N (N lẻ), sau đó điền các số từ 1 đến n^2 vào trong một bảng vuông sao cho tổng các hàng ngang, hàng dọc và 2 đường chéo đều bằng nhau (bảng này được gọi là Ma phương).

Ví dụ: Với N=3 và N=5 ta có

2	7	6
9	5	1
4	3	8

Bắc

	3	16	9	22	15
	20	8	21	14	2
Tây	7	25	13	1	19
	24	12	5	18	6
	11	4	17	10	23

Đông

Nam

Phương pháp:

Xuất phát từ ô bên phải của ô nằm giữa. Đi theo ***hướng đông bắc*** để điền các số 1, 2, ...

Khi điền số, cần chú ý một số nguyên tắc sau:

- Nếu vượt ra phía ngoài bên phải của bảng thì quay trở lại cột đầu tiên.
- Nếu vượt ra phía ngoài bên trên của bảng thì quay trở lại dòng cuối cùng.
- Nếu số đã điền k chia hết cho N thì số tiếp theo sẽ được viết trên cùng một hàng với k nhưng cách 1 ô về phía bên phải.

```
Uses Crt;
Var A:Array[1..20,1..20] Of Word;
    n,i,j,k:Word;
Begin
 Write('Nhap N= '); Readln(n);
 Clrscr;
{Định vị ô xuất phát}
 i:=n DIV 2 + 1;
 j:=n DIV 2 + 2;

{Điền các số k từ 1 đến n*n}
 For k:=1 To n*n Do
  Begin
   A[i,j]:=k;
   If k MOD n=0 Then j:=j+2
   Else Begin
           {Đi theo hướng đông bắc}
            j:=j+1; i:=i-1;
        End;
   If j>n Then j:=j MOD n;
   If i=0 Then i:=n;
  End;
```

```
 {In kết quả ra màn hình}
  For i:=1 To n Do
   Begin
     For j:=1 To n Do write(a[i,j]:4);
     Writeln;
   End;
  Readln;
End.
```

Bài tập 5.10: Viết chương trình nhập vào 2 mảng số nguyên A, B đại diện cho 2 tập hợp (không thể có 2 phần tử trùng nhau trong một tập hợp). Trong quá trình nhập, phải kiểm tra: nếu phần tử vừa nhập vào đã có trong mảng thì không bổ sung vào mảng. In ra màn hình các phần tử là giao của 2 tập hợp A, B.

Ý tưởng:

Duyệt qua tất cả các phần tử $a_i \in A$. Nếu $a_i \in B$ thì viết a_i ra màn hình.

```
Uses Crt;
Type Mang=ARRAY[1..50] Of Integer;

Var A,B:Mang;
    n,m:Byte;

Function KiemTra(x:Integer; n:Byte; A:Mang):Boolean;
Var i:Byte; Found:Boolean;
Begin
  Found:=False;
  i:=1;
  While (i<=n) AND (not Found) Do
    If x=A[i] Then Found:=True Else i:=i+1;
  KiemTra:=Found;
End;

Procedure NhapMang(Var n:Byte; Var A:Mang);
Var ch:Char;
    x:Integer;
Begin
  n:=0;
  Repeat
    Write('x='); Readln(x);
    If not KiemTra(x,n,A) Then
     Begin
       n:=n+1; A[n]:=x;
     End;
    Writeln('An ESC de ket thuc nhap!');
```

```
    ch:=Readkey;
  Until ch=#27;
End;

Procedure GiaoAB(n:Byte; A:Mang;m:Byte; B:Mang);
Var i:Byte;
Begin
  For i:=1 To n Do
   If KiemTra(A[i],m,B) Then Write(A[i]:4);
End;

Begin
  Clrscr;
  Writeln('Nhap mang A: ');
  NhapMang(n,A);
  Writeln('Nhap mang B: ');
  NhapMang(m,B);
  Writeln('Giao cua 2 mang A&B la: ');
  GiaoAB(n,A,m,B);
  Readln;
End.
```

Bài tập 5.11: Cho một mảng số nguyên gồm n phần tử. Tìm dãy con gồm m phần tử (m≤ n) sao cho dãy con này có tổng lớn nhất. (Dãy con là dãy các phần tử liên tiếp nhau trong mảng).

```
Uses Crt;
Type Mang=ARRAY[1..50] Of Integer;

Var A:Mang;
    n,m,i,j,k:Byte;
    S,Max:Integer;
Begin
  Write('So phan tu cua mang: n= '); Readln(n);
  For i:=1 To n Do
   Begin
     Write('a[',i,']='); Readln(a[i]);
   End;
  Write('Nhap so phan tu cua day con: m= '); Readln(m);

  k:=1; {Vị trí phần tử đầu tiên của dãy con}

 {Giả sử m phần tử đầu tiên của mảng A là dãy con có tổng lớn nhất}
  Max:=0;
  For i:=1 To m Do Max:=Max+A[i];
```

```
 {Tìm các dãy con khác}
  For i:=2 To n-m+1 Do
   Begin
    {Tính tổng của dãy con thứ i}
     S:=0;
     For j:=i To i+m-1 Do S:=S+A[j];
     If S>Max Then {Nếu dãy con tìm được có tổng lớn hơn dãy con trước}
       Begin
         Max:=S; {Thay tổng mới}
         k:=i;       {Thay vị trí đầu tiên của dãy con mới}
       End;
   End;

  Writeln('Day con co tong lon nhat la:');
  For i:=k To k+m-1 Do Write(A[i]:5);
  Readln;
End.
```

Bài tập 5.12: Viết chương trình in ra màn hình tam giác Pascal. Ví dụ, với n=4 sẽ in ra hình sau:

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
```

Ý tưởng:

Tam giác Pascal được tạo ra theo qui luật sau:

+ Mỗi dòng đều bắt đầu và kết thúc bởi số 1.

+ Phần tử thứ j ở dòng k nhận được bằng cách cộng 2 phần tử thứ j-1 và j ở dòng thứ k-1.

```
Uses Crt;
Var Dong:Array[0..20] Of Byte;
    n,i,j:Byte;
Begin
  Write('n= '); Readln(n);
  Clrscr;
  Dong[0]:=1;
  Writeln(Dong[0]:4);

  {Khoi tao gia tri cua dong}
  For i:=1 To n Do Dong[i]:=0;
```

```
 {Voi moi dong i}
 For i:=1 To n Do
  Begin
    For j:=i DownTo 1 Do
     Begin
       Dong[j]:=Dong[j-1]+Dong[j];
       Write(Dong[j]:4);
     End;
    Writeln(Dong[i]:4);
  End;
 Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 5.13: Viết chương trình nhập vào một dãy số thực và số thực x. Thông báo lên màn hình số lượng các phần tử trong dãy bằng x và vị trí của chúng.

Bài tập 5.14: Nhập vào một mảng các số nguyên.

a/ Xếp lại mảng đó theo thứ tự giảm dần.

b/ Nhập vào một số nguyên từ bàn phím. Chèn số đó vào mảng sao cho mảng vẫn có thứ tự giảm dần. (không được xếp lại mảng)

Gợi ý:

- Tìm vị trí cần chèn: i.
- Đẩy các phần tử từ vị trí i tới n sang phải 1 vị trí.
- Gán: A[i]=x;

Bài tập 5.15: Cho 2 mảng số nguyên: Mảng A có m phần tử, mảng B có n phần tử.

a/ Sắp xếp lại các mảng đó theo thứ tự giảm dần.

b/ Trộn 2 mảng đó lại thành mảng C sao cho mảng C vẫn có thứ tự giảm dần (Không được xếp lại mảng C).

Gợi ý:

- Dùng 2 chỉ số i,j để duyệt qua các phần tử của 2 mảng A, B và k là chỉ số cho mảng C.
- Trong khi (i<=m) và (j<=n) thì:

  {Tức là khi đồng thời cả 2 dãy A, B đều chưa duyệt hết}

  + Nếu A[i]>B[j] thì: C[k]:=A[i]; i:=i+1;
  + Ngược lại: C[k]:=B[j]; j:=j+1;
- Nếu dãy nào hết trước thì đem phần còn lại của dãy kia bổ sung vào cuối dãy C.

Bài tập 5.16: Viết chương trình tính tổng và tích 2 ma trận vuông A, B cấp n.

Gợi ý:

Công thức tính tổng 2 ma trận: $\mathbf{C_{ij}} = \mathbf{A_{ij}} + \mathbf{B_{ij}}$

Công thức tính tích 2 ma trận: $\mathbf{C_{ij}} = \sum_{k=1}^{n} A_{ik} * B_{kj}$

Bài tập 5.17: Viết chương trình nhập vào 2 dãy số nguyên $(a)_n$ và $(b)_m$, $m \leq n$. Kiểm tra xem dãy {b} có phải là dãy con của dãy {a} không?

Bài tập 5.18: Viết chương trình nhập vào một dãy số nguyên $a_1, a_2, ..., a_n$. Tìm trong dãy {a} một dãy con tăng dần dài nhất (có số phần tử lớn nhất) và in ra màn hình dãy con đó.

Bài tập 5.19: Cho mảng 2 chiều A cấp mxn. Viết chương trình sắp xếp lại mảng A theo yêu cầu sau:

a/ Các phần tử trên mỗi dòng được sắp xếp theo thứ tự giảm dần.

b/ Các dòng được sắp xếp lại theo thứ tự tăng dần của tổng các phần tử trên mỗi dòng.

Bài tập 5.20: Viết chương trình để kiểm tra một dãy các số nguyên được nhập vào từ bàn phím đã được sắp theo thứ tự tăng dần hay chưa theo 2 cách: Đệ qui và không đệ qui.

Gợi ý:

- Nếu dãy có 1 phần tử thì dãy tăng dần.
- Ngược lại:
  + Nếu A[n-1]>A[n] thì dãy không tăng dần.
  + Ngược lại: Gọi đệ qui với dãy có n-1 phần tử (bỏ bớt đi phần tử cuối cùng).

Bài tập 5.21: Viết chương trình nhập vào 2 mảng số nguyên A, B đại diện cho 2 tập hợp (không thể có 2 phần tử trùng nhau trong một tập hợp). Trong quá trình nhập, phải kiểm tra: nếu phần tử vừa nhập vào đã có trong mảng thì không bổ sung vào mảng.

a/ In ra màn hình hợp của 2 tập hợp A, B.

b/ In ra màn hình hiệu của 2 tập hợp A, B.

Gợi ý:

a/ - In ra màn hình tất cả các phần tử của tập hợp A.

- Duyệt qua tất cả các phần tử $b_i \in B$. Nếu $b_i \notin A$ thì in b_i ra màn hình.

b/ Duyệt qua tất cả các phần tử $a_i \in A$. Nếu $a_i \notin B$ thì in a_i ra màn hình.

Bài tập 5.22: Viết chương trình tính tổng của 2 đa thức h(x) = f(x) + g(x). Trong đó, mỗi đa thức có dạng: $a_0 + a_1x + a_2x^2 + ... + a_nx^n$.

Gợi ý:

Dùng các mảng A, B, C để lưu trữ các hệ số a_i của các đa thức f(x), g(x) và h(x).

Bài tập 5.23: Viết chương trình để tìm các phương án đặt 8 quân hậu trên bàn cờ vua (ma trận 8x8) sao cho các quân hậu không ăn được nhau.

Gợi ý:

Dùng giải thuật quay lui.

Bài tập 5.24: Viết chương trình tính định thức của ma trận vuông cấp n.

Gợi ý:

Dùng cách tính định thức theo phương pháp GAUSE.

Chương 6
XÂU KÝ TỰ (STRING)

I. KHAI BÁO KIỂU STRING

TYPE TênKiểu = STRING[Max];
VAR Tên biến : TênKiểu;

hoặc khai báo biến trực tiếp:

VAR Tên biến : STRING[Max];

Trong đó Max là số ký tự tối đa có thể chứa trong chuỗi (Max ∈ [0,255]). Nếu không có khai báo [Max] thì số ký tự mặ mặc định trong chuỗi là 255.

Ví dụ:

```
Type    Hoten  = String[30];
        St80 = String[80];
Var     Name : Hoten;
        Line : St80;
        St : String; {St có tối đa là 255 ký tự}
```

II. TRUY XUẤT DỮ LIỆU KIỂU STRING

- Có thể sử dụng các thủ tục xuất nhập Write, Writeln, Readln để truy xuất các biến kiểu String.
- Để truy xuất đến ký tự thứ k của xâu ký tự, ta sử dụng cú pháp sau: **Tênbiến[k]**.

III. CÁC PHÉP TOÁN TRÊN XÂU KÝ TỰ

3.1. Phép nối xâu: +

3.2. Các phép toán quan hệ: =, <>, <, <=, >, >=.

Chú ý: *Các phép toán quan hệ được so sánh theo thứ tự từ điển.*

IV. CÁC THỦ TỤC VÀ HÀM VỀ XÂU KÝ TỰ

4.1. Hàm lấy chiều dài của xây ký tự

LENGTH(St : String):Integer;

4.2. Hàm COPY(St : String; Pos, Num: Byte): String;

Lấy ra một xâu con từ trong xâu St có độ dài Num ký tự bắt đầu từ vị trí Pos .

4.3. Hàm POS(SubSt, St :String):Byte;

Kiểm tra xâu con SubSt có nằm trong xâu St hay không? Nếu xâu SubSt nằm trong xâu St thì hàm trả về vị trí đầu tiên của xâu con SubSt trong xâu St, ngược lại hàm trả về giá trị 0.

4.4. Thủ tục DELETE(Var St:String; Pos, Num: Byte);

Xoá trong xâu St Num ký tự bắt đầu từ vị trí Pos.

4.5. Thủ tục INSERT(SubSt: String; Var St: String; Pos: Byte);

Chèn xâu SubSt vào xâu St bắt đầu tại vị trí Pos.

4.6. Thủ tục STR(Num; Var St:String);

Đổi số nguyên hay thực Num thành dạng xâu ký tự, kết quả lưu vào biến St.

4.7. Thủ tục VAL(St:String; Var Num; Var Code:Integer);

Đổi xâu số St thành số và gán kết quả lưu vào biến Num. Nếu việc chuyển đổi thành công thì biến Code có giá trị là 0, ngược lại biến Code có giá trị khác 0 (vị trí của lỗi).

BÀI TẬP MẪU

Bài tập 6.1: Viết chương trình nhập vào một xâu ký tự từ bàn phím. Đổi xâu ký tự đó sang chữ in hoa rồi in kết quả ra màn hình.

Ví dụ :Xâu abcdAbcD sẽ cho ra xâu ABCDABCD.

```
Uses Crt;
Var  St:String;
     i:Byte;
Begin
  Write('Nhap xau St: '); Readln(St);
  For i:=1 to length(St) do St[i]:=Upcase(St[i]);
  Write('Xau ket qua: ', St);
  Readln;
End.
```

Bài tập 6.2: Viết chương trình nhập vào một xâu ký tự từ bàn phím. Đổi xâu ký tự đó sang chữ thường rồi in kết quả ra màn hình.

Ví dụ :Xâu abCdAbcD sẽ cho ra xâu abcdabcd.

```
Uses Crt;
Var  St:String;
     i:Byte;
Begin
  Write('Nhap xau St: '); Readln(St);
  For i:=1 to length(St) do
    If St[i] IN ['A'..'Z'] Then St[i]:=CHR(ORD(St[i])+32);
  Write('Xau ket qua: ', St);
  Readln;
End.
```

Bài tập 6.3: Viết chương trình đếm số ký tự chữ số trong một xâu ký tự được nhập vào từ bàn phím.

```
Uses Crt;
Var  St:String;
     i,d:Byte;
```

```
Begin
   Write('Nhap xau St: '); Readln(St);
   For i:=1 to length(St) do
      If St[i] IN ['0'..'9'] Then d:=d+1;
   Write('So ky tu chu so trong xau: ', d);
   Readln;
End.
```

Bài tập 6.4: Viết chương trình nhập một xâu từ bàn phím. In ra xâu đó sau khi xóa hết các ký tự trắng thừa trong xâu. (Ký tự trắng thừa là các ký tự trắng đầu xâu, cuối xâu và nếu ở giữa xâu có 2 ký tự trắng liên tiếp nhau thì có 1 ký tự trắng thừa).

```
Uses Crt;
Var  St:String;
Procedure  XoaTrangThua(Var St:String);
Begin
   {Xóa các ký tự trắng ở đầu xâu}
   While St[1]=#32 Do Delete(St,1,1);
   {Xóa các ký tự trắng ở cuối xâu}
   While St[Length(St)]=#32 Do Delete(St,Length(St),1);
   {Xóa các ký tự trắng ở giữa xâu}
   While POS(#32#32,St)<>0 Do Delete(St,POS(#32#32,St),1);
End;

Begin
   Write('Nhap xau St: '); Readln(St);
   XoaTrangThua(St);
   Write('Xau sau khi xoa cac ky tu trang thua: ', St);
   Readln;
End.
```

Bài tập 6.5: Viết chương trình liệt kê các từ của một xâu ký tự được nhập vào từ bàn phím, mỗi từ phải được viết trên một dòng.

```
Uses Crt;
Var  St:String;
Procedure  XoaTrangThua(Var St:String);
Begin
   {Xóa các ký tự trắng ở đầu xâu}
   While St[1]=#32 Do Delete(St,1,1);
   {Xóa các ký tự trắng ở cuối xâu}
   While St[Length(St)]=#32 Do Delete(St,Length(St),1);
```

```
   {Xóa các ký tự trắng ở giữa xâu}
   While POS(#32#32,St)<>0 Do Delete(St,POS(#32#32,St),1);
End;

Begin
   Write('Nhap xau St: '); Readln(St);
   XoaTrangThua(St);
   St:=St+#32;
   Writeln('Liet ke cac tu trong xau: ');
   While POS(#32,St)<>0 Do
      Begin
         Writeln(Copy(St,1,POS(#32,St)));
         Delete(St,1,POS(#32,St));
      End;
   Readln;
End.
```

Bài tập 6.6: Viết chương trình nhập vào một xâu ký tự từ bàn phím. Tìm xâu đảo ngược của xâu đó rồi in kết quả ra màn hình theo 2 cách: Đệ qui và không đệ qui.

Ý tưởng:

- Nếu xâu St có 1 ký tự thì xâu đảo = St.
- Ngược lại: Xâu đảo = Ký tự cuối + Đệ qui(Phần còn lại của xâu St).

```
Uses Crt;
Var  St:String;

{Giải thuật không đệ qui}
Function XauDao(St:String):String;
Var  S:String;
     i:Byte;
Begin
   S:='';
   For i:=Length(St) DowTo 1 Do S:=S+St[i];
   XauDao:=S;
End;

{Giải thuật đệ qui}
Function DeQui(St:String):String;
Begin
   If Length(St)<=1 Then DeQui:=St
   Else DeQui:=St[Length(St)] + DeQui(Copy(St,1,Length(St)-1));
End;
```

```
Begin
   Write('Nhap xau St: '); Readln(St);
   Write('Xau dao nguoc: ', XauDao(St));
   Readln;
End.
```

Bài tập 6.7: Viết chương trình nhập vào một xâu ký tự từ bàn phím. Thông báo lên màn hình các chữ cái có trong xâu và số lượng của chúng (Không phân biệt chữ hoa hay chữ thường).

Ý tưởng:

- Dùng một mảng dem với chỉ số là các chữ cái để lưu trữ số lượng của các chữ cái trong xâu.
- Duyệt qua tất cả các ký tự của xâu St: Nếu ký tự đó là chữ cái thì tăng ô biến mảng dem[St[i]] lên 1 đơn vị.

```
Uses Crt;
Var  St:String;
       dem: Array['A'..'Z'] Of Byte;
       i:Byte;
       ch:Char;
Begin
   Write('Nhap xau St: '); Readln(St);
   {Khởi tạo mảng}
   For ch:='A' To 'Z' Do dem[ch]:=0;
   {Duyệt xâu}
   For i:=1 To Length(St) Do
      If Upcase(St[i]) IN ['A'..'Z'] Then Inc(dem[Upcase(St[i])]);
   {Liệt kê các ký tự ra màn hình}
   For ch:='A' To 'Z' Do
      If dem[ch]>0 Then Writeln(ch,' : ',dem[ch]);
   Readln;
End.
```

Bài tập 6.8: Viết chương trình xóa các ký tự chữ số trong một xâu ký tự được nhập vào từ bàn phím.

```
Uses Crt;
Var  St:String;
```

```
{Hàm POSNUM kiểm tra xem trong xâu St có ký tự chữ số hay không? Nếu có,
hàm trả về vị trí đầu tiên của ký tự chữ số, ngược lại hàm trả về giá trị 0}
Function POSNUM(St:String):Byte;
Var  OK:Boolean;
     i:Byte;
Begin
   OK:=False;
   i:=1;
   While (i<=Length(St)) AND (Not OK) Do
      If St[i] IN ['0'..'9'] Then OK:=True
      Else i:=i+1;
   If OK Then POSNUM:=i Else POSNUM:=0;
End;

Begin
   Write('Nhap xau St: '); Readln(St);
   While POSNUM(St)<>0 Do Delete(St,POSNUM(St),1);
   Write('Xau sau khi xoa: ',St);
   Readln;
End.
```

Bài tập 6.9: Viết chương trình để mã hoá và giải mã một xâu ký tự bằng cách đảo ngược các bit của từng ký tự trong xâu.

```
Uses crt;
Var st:string;

{Hàm đảo bit ký tự c}
Function DaoBit(c:char):char;
Var n,i,s,bitcuoi,Mask:byte;
Begin
 {Đổi ký tự sang số}
  n:=ORD(c);
 {s: kết quả đảo bit, Mask: mặt nạ dùng để bật bit thứ i}
  s:=0;
  Mask:=128;

  For i:=1 To 8 Do {duyệt qua 8 bit của n}
   Begin
     {Lấy bit cuối cùng của n: bit cực phải}
     bitcuoi:=n AND 1;
     n:=n shr 1; {loại bỏ bit cuối cùng: n:=n DIV 2}
     {Bật bit thứ i lên: từ trái sang phải}
     if bitcuoi=1 then s:=s OR Mask;
```

```
    Mask:=Mask shr 1; { Mask:= Mask DIV 2}
   End;

  DaoBit:=CHR(s);
End;

Function MaHoa(st:string):string;
Var i:Byte;
Begin
 {Đảo bit từng ký tự trong xâu st}
  For i:=1 To Length(st) Do st[i]:=DaoBit(st[i]);
  Mahoa:=st;
End;

Begin
 Write('Nhap xau: '); Readln(st);
 st:=MaHoa(st);
 Writeln('Xau sau khi ma hoa: ',st);
 Readln;
 st:=MaHoa(st);
 Writeln('Xau sau khi giai ma: ',st);
 Readln;
End.
```

Bài tập 6.10: Viết chương trình thực hiện phép cộng 2 số tự nhiên lớn (không quá 255 chữ số).

```
Uses crt;
Var so1,so2,kqua:string;

Procedure LamDayXau(Var st1,st2:string);
{Them so 0 vao truoc xau ngan}
var i:Byte;
Begin
  If Length(st1)>Length(st2) Then
    For i:=1 To Length(st1)-Length(st2) Do st2:='0'+st2
  Else
    For i:=1 To Length(st2)-Length(st1) Do st1:='0'+st1;
End;

Function Cong(st1,st2:string):string;
Var i,a,b,c,sodu:Byte;
    code:integer;
    st,ch:string;
Begin
  st:=''; sodu:=0;
```

```
  LamDayXau(st1,st2);
  {Lấy từng số của 2 xâu: từ phải sang trái}
  For i:=Length(st1) DownTo 1 Do
   Begin
     {Đổi ký tự sang số nguyên}
     Val(st1[i],a,code);
     Val(st2[i],b,code);
     {Tính tổng của 2 số a,b vừa lấy ra cho vào biến c}
     c:=(a+b+sodu) MOD 10;
     {Lấy phần dư của tổng a+b}
     sodu:=(a+b+sodu) DIV 10;
     {Đổi số nguyên c sang xâu ký tự ch}
     str(c,ch);
     {Cộng xâu ch vào bên trái xâu kết quả st}
     st:=ch+st;
   End;

    {Xử lý trường hợp số dư cuối cùng >0}
   If sodu>0 Then
    Begin
     str(sodu,ch);
     st:=ch+st;
    End;
   Cong:=st;
End;

Begin
  Write('Nhap so thu nhat: '); Readln(so1);
  Write('Nhap so thu hai: '); Readln(so2);
  kqua:=Cong(so1,so2);
  Writeln('Tong= ',kqua);
  Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 6.11: Viết chương trình nhập vào một xâu ký tự từ bàn phím. Tìm và in ra màn hình một từ có độ dài lớn nhất trong xâu.

Gợi ý:

Tách từng từ để so sánh (xem bài tập 5).

Bài tập 6.12: Viết chương trình nhập một xâu ký tự St từ bàn phím và một ký tự ch. In ra màn hình xâu St sau khi xóa hết các ký tự ch trong xâu đó.

Gợi ý:

While POS(ch,st)<>0 Do Delete(st,POS(ch,st),1);

Bài tập 6.13: Viết chương trình nhập một xâu vào từ bàn phím và thông báo lên màn hình xâu đó có phải đối xứng không theo 2 cách: Đệ qui và không đệ qui. (Ví dụ: abba, abcba là các xâu đối xứng).

Gợi ý:

- Nếu xâu Length(st)<=1 thì st là xâu đối xứng
- Ngược lại:
  + Nếu st[1]<>st[Length(st)] thì st không đối xứng
  + Ngược lại: Gọi đệ qui với xâu st sau khi bỏ đi ký tự đầu và ký tự cuối.

Bài tập 6.14: Viết chương trình đảo ngược thứ tự các từ trong một xâu được nhập vào từ bàn phím.

Ví dụ: Xâu ***Nguyen Van An*** sẽ thành ***An Van Nguyen***.

Gợi ý:

Tách từng từ nối vào đầu xâu mới (xem bài tập 5).

Bài tập 6.15: Viết chương trình nhập vào 2 xâu ký tự s1 và s2. Kiểm tra xem xâu s2 xuất hiện bao nhiêu lần trong xâu s1. (Lưu ý: length(s2)<= length(s1)).

Gợi ý:

Dùng hàm POS để kiểm tra và thủ tục DELETE để xóa bớt sau mỗi lần kiểm tra.

Bài tập 6.16: Viết chương trình nhập vào một dòng văn bản, hiệu chỉnh văn bản theo những yêu cầu sau đây và in văn bản sau khi hiệu chỉnh ra màn hình:

a. Xóa tất cả các ký tự trắng thừa.

b. Trước các dấu câu không có các ký tự trắng, sau các dấu câu có một ký tự trắng.

c. Đầu câu in hoa.

Bài tập 6.17: Viết chương trình thực hiện phép nhân 2 số nguyên lớn.

Gợi ý:

- Viết hàm để nhân một số lớn với số có 1 chữ số.
- Áp dụng hàm tính tổng 2 số lớn (xem bài tập 10).

Bài tập 6.18: Viết chương trình để nén và giải nén một xâu ký tự .

Ví dụ: Xâu ‘AAAABBBCDDDDDDDDEEF’ sau khi nén sẽ trở thành ‘4A3BC7D2EF’.

Bài tập 6.19: Viết chương trình nhập vào họ tên đầy đủ của các học viên một lớp học (không quá 50 người). Hãy sắp xếp lại họ tên của các học viên đó theo thứ tự Alphabet (Nếu tên trùng nhau thì xếp thứ tự theo họ lót, nếu họ lót cũng trùng nhau thì xếp thứ tự theo họ). In ra màn hình danh sách của lớp học sau khi đa sắp xếp theo thứ tự Alphabet.

Gợi ý:

- Dùng mảng xâu ký tự để lưu trữ họ tên học viên.
- Đảo ngược các từ của họ tên trước khi sắp xếp.

Bài tập 6.20: Viết chương trình liệt kê ra màn hình tất cả các hoán vị của một xâu ký tự.

Gợi ý:

Dùng giải thuật quay lui.

Chương 7
KIỂU BẢN GHI (RECORD)

I. KHAI BÁO DỮ LIỆU KIỂU RECORD

```
TYPE   TênKiểu = RECORD
            Field1 : Kiểu1;
            Field2 : Kiểu2;
            ...
            FieldN: KiểuN;
       END;
VAR    Biến : TênKiểu;
```

Ví dụ:

```
TYPE   HocSinh  = Record
            Hoten : String[20];
            Tuoi : Integer;
            DiemTB : real;
       End;
VAR  HS : HocSinh;
```

II. XUẤT NHẬP DỮ LIỆU KIỂU RECORD

Không thể dùng các thủ tục xuất/nhập, các phép toán so sánh đối với các biến kiểu record mà chỉ có thể sử dụng thông qua từng trường của biến record đó.

2.1. Truy nhập trực tiếp: TênbiếnRecord.Field

2.2. Sử dụng câu lệnh WITH

```
WITH  TênbiếnRecord DO
   BEGIN
      Xử lý Field1;
      Xử lý Field2;
      ...
      Xử lý FieldN;
   END;
```

2.3. Gán biến Record: Ta có thể gán 2 biến Record cùng kiểu với nhau.

BÀI TẬP MẪU

Bài tập 7.1: Viết chương trình thực hiện phép cộng 2 số phức.

```
Uses Crt;
```

```
Type  Complex =    Record
                     a,b:Real;
                  End;
Var   c1,c2,c3:Complex;
      dau:string;
Begin
   Writeln('Nhap so phuc c1:');
   Write('Phan thuc a = '); Readln(c1.a);
   Write('Phan ao b = '); Readln(c1.b);

   Writeln('Nhap so phuc c2:');
   Write('Phan thuc a = '); Readln(c2.a);
   Write('Phan ao b = '); Readln(c2.b);

   {Tính tổng 2 số phức}
   c3.a := c1.a + c2.a;
   c3.b := c1.b + c2.b;

   {In kết quả ra màn hình}
   Writeln('Tong cua 2 so phuc:');
   If c1.b>=0 Then dau:='+i' else dau:='-i';
   Writeln('c1 = ', c1.a:0:2, dau, abs(c1.b):0:2); {Số phức c1}
   If c2.b>=0 Then dau:='+i' else dau:='-i';
   Writeln('c2 = ', c2.a:0:2, dau, abs(c2.b):0:2); {Số phức c2}
   Writeln('La so phuc:');
   If c3.b>=0 Then dau:='+i' else dau:='-i';
   Writeln('c3 = ', c3.a:0:2, dau, abs(c3.b):0:2); {Số phức c3}
   Readln;
End.
```

Bài tập 7.2: Viết chương trình quản lý điểm thi Tốt nghiệp của sinh viên với 2 môn thi: Cơ sở và chuyên ngành. Nội dung công việc quản lý bao gồm:

- Nhập điểm cho từng sinh viên.
- In danh sách sinh viên ra màn hình.
- Thống kê số lượng sinh viên thi đậu.
- In ra màn hình hình danh sách những sinh viên bị thi lại.

```
Uses Crt;
```

```
Const Max=200;
Type SinhVien=Record
               Hoten:string[30];
               DiemCS,DiemCN:Byte;
             End;

Var SV:ARRAY[1..Max] Of SinhVien;
      n:Byte;
      c:Char;

Procedure NhapDanhSach;
Var ch:Char;
Begin
  Clrscr;
  Writeln('NHAP DANH SACH SINH VIEN');
  n:=0;
  Repeat
    n:=n+1;
    With SV[n] Do
     Begin
       Write('Ho ten: '); Readln(Hoten);
       Write('Diem co so: '); Readln(DiemCS);
       Write('Diem chuyen nganh: '); Readln(DiemCN);
     End;
    Writeln('Nhan phim bat ky de nhap tiep/Nhan <ESC> de ket thuc!');
    ch:=Readkey;
  Until ch=#27;
End;

Procedure InDanhSach;
Var ch:Char;
      i:Byte;
Begin
  Clrscr;
  Writeln('DIEM THI TOT NGHIEP SINH VIEN');
  Writeln;
  WRITELN('STT       Ho ten        Diem Co so     Diem Chuyen nganh');
  For i:=1 To n do
   With SV[i] Do
    Begin
     Writeln(i:3,'.',Hoten:20,DiemCS:5,DiemCN:20);
    End;
  ch:=ReadKey;
End;

Procedure DanhSachSVThilai;
```

```
Var ch:Char;
      i:Byte;
Begin
  Clrscr;
  Writeln('DANH SACH SINH VIEN THI LAI');
  Writeln;
  WRITELN('STT       Ho ten        Diem Co so     Diem Chuyen nganh');
  For i:=1 To n do
   With SV[i] Do
    Begin
     If (DiemCS<5)OR(DiemCN<5) Then
       Writeln(i:3,'.',Hoten:20,DiemCS:5,DiemCN:20);
    End;
  ch:=ReadKey;
End;

Procedure ThongKeSVThiDau;
Var S,i:Byte;
      ch:Char;
Begin
  S:=0;
  For i:=1 To n Do
   If (SV[i].DiemCS>=5)AND(SV[i].DiemCN>=5) Then S:=S+1;
  Writeln('So sinh vien thi dau la: ',s);
  ch:=Readkey;
End;

Begin
  Repeat
    Clrscr;
    Writeln('CHUONG TRINH QUAN LY DIEM THI TOT NGHIEP SINH VIEN');
    Writeln('1. Nhap danh sach sinh vien');
    Writeln('2. In danh sach sinh vien');
    Writeln('3. Thong ke so sinh vien thi dau');
    Writeln('4. danh sach sinh vien thi lai');
    Writeln('<ESC>: Thoat');
    c:=Readkey;
    Case c Of
    '1': NhapDanhSach;
    '2': InDanhSach;
    '3': ThongKeSVThiDau;
    '4': DanhSachSVThilai;
    End;
  Until c=#27;
End.
```

Bài tập 7.3: Viết chương trình nhập vào n đỉnh của một đa giác lồi S.

a/ Tính diện tích của S biết:

$$dt(S)=\frac{1}{2}\left|\sum_{i=1}^{n}(x_i y_{i+1} - x_{i+1} y_i)\right|$$

trong đó: (x_i,y_i) là tọa độ đỉnh thứ i của đa giác S.

b/ Nhập vào thêm một điểm P(x,y). Hãy kiểm tra xem P nằm trong hay ngoài đa giác S.

Ý tưởng:

Nối P với các đỉnh của đa giác S thì ta được n tam giác: $S_i = PP_iP_{i+1}$, với $P_{n+1}=P_1$.

Nếu $\sum_{i=1}^{n} dt(S_i) = dt(S)$ thì $P \in S$.

```
Uses Crt;
Type Toado=Record
           x,y:integer;
         end;
     Mang=array[0..30] of Toado;

Var n:Byte;
    A:Mang;
    P:ToaDo;

Procedure NhapDinh(var n:Byte; Var P:Mang);
Var i:Byte;
Begin
  Write('Nhap so dinh cua da giac n = '); readln(n);
  For i:=1 to n do
    Begin
      Write('P[',i,'].x = ');readln(P[i].x);
      Write('P[',i,'].y = ');readln(P[i].y);
    End;
End;

Function DienTichDaGiac(n:Byte;P:Mang):real;
Var i,j:integer;
    s:real;
Begin
  s:=0;
  for i:= 1 to n do
    begin
      if i=n then j:=1 else j:=i+1;
      s:=s+((P[i].x*P[j].y-P[j].x*P[i].y));
    end;
```

```
  DienTichDaGiac:=abs(s)/2;
end;

Function DienTichTamGiac(A,B,C:ToaDo):real;
Begin
  DienTichTamGiac:=abs(A.x*B.y-B.x*A.y+B.x*C.y-C.x*B.y+C.x*A.y-A.x*C.y)/2;
End;

Function KiemTra(PP:ToaDo;n:Byte;P:Mang):Boolean;
Var i,j:integer;
    s:real;
begin
  s:=0;
  For i:=1 to n do
    begin
      if i=n then j:=1 else j:=i+1;
      s:=s+DienTichTamGiac(PP,P[i],P[j]);
    end;
  If round(s)=round(DienTichDaGiac(n,P)) then KiemTra:=true
  else KiemTra:=false;
end;

Begin
  NhapDinh(n,A);
  Writeln('S=',DienTichDaGiac(n,A):0:2);
  Readln;
  Writeln('Nhap diem P:');
  Write('P.x = ');readln(P.x);
  Write('P.y = ');readln(P.y);
  If KiemTra(P,n,A) Then Writeln('Diem P nam trong da giac S.')
  Else Writeln('Diem P nam ngoai da giac S.');
  Readln;
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 7.4: Viết chương trình nhân hai số phức c1, c2.

Bài tập 7.5: Viết chương trình quản lý điểm thi học phần của sinh viên bao gồm các trường sau: Họ tên, Điểm Tin, Điểm ngoại ngữ, Điểm trung bình, Xếp loại. Thực hiện các công việc sau:

a/ Nhập vào danh sách sinh viên của một lớp (không quá 30 người), bao gồm: Họ tên, Điểm Tin, Điểm Ngoại ngữ. Tính Điểm trung bình và Xếp loại cho từng sinh viên.

b/ In ra màn hình danh sách sinh viên của lớp đó theo dạng sau:

Họ tên	Điểm Tin	Điểm Ngoại ngữ	Điểm T.Bình	Xếp loại
Trần Văn An Lê Thị Béo	8 7	9 5	8.5 6.0	Giỏi T.Bình .....

c/ In ra màn hình danh sách những sinh viên phải thi lại (nợ một trong hai môn).

d/ In ra danh sách những sinh viên xếp loại Giỏi.

e/ Tìm và in ra màn hình những sinh viên có điểm trung bình cao nhất lớp.

f/ Sắp xếp lại danh sách sinh viên theo thứ tự Alphabet.

g/ Sắp xếp lại danh sách sinh viên theo thứ tự giảm dần của điểm trung bình.

h/ Viết chức năng tra cứu theo tên không đầy đủ của sinh viên. Ví dụ: Khi nhập vào tên **Phuong** thì chương trình sẽ tìm và in ra màn hình thông tin đầy đủ của những sinh viên có tên **Phuong** (chẳng hạn như: **Pham Anh Phuong, Do Ngoc Phuong, Nguyen Nam Phuong...**).

Bài tập 7.6: Viết chương trình quản lý sách ở thư viện gồm các trường sau: Mã số sách, Nhan đề, Tên Tác giả, Nhà Xuất bản, Năm xuất bản.

a/ Nhập vào kho sách của thư viện (gồm tất cả các trường).

b/ In ra màn hình tất cả các cuốn sách có trong thư viện.

c/ Tìm một cuốn sách có mã số được nhập vào từ bàn phím. Nếu tìm thấy thì in ra màn hình thông tin đầy đủ của cuốn sách đó, ngược lại thì thông báo không tìm thấy.

c/ Tìm và in ra màn hình tất cả các cuốn sách có cùng tác giả được nhập vào từ bàn phím.

d/ Lọc ra các cuốn sách được xuất bản trong cùng một năm nào đó.

e/ Tìm và in ra màn hình các cuốn sách mà nhan đề có chứa từ bất kỳ được nhập vào từ bàn phím.

Chương 8
DỮ LIỆU KIỂU FILE

I. KHAI BÁO

Type <Tên kiểu File> = **File of** <Kiểu phần tử>;
Var <Tên biến File> : <Tên kiểu File>;
hoặc khai báo trực tiếp:
Var <Tên biến File> : **File of** <Kiểu phần tử>;

Ví dụ:

```
Type   SanPham = File of Record
                    Ten: String[20];
                    SoHieu: Byte;
                 End;
Var f,g: SanPham;
```

hoặc khai báo trực tiếp:

```
Var f,g: File of  Record
                    Ten: String[20];
                    SoHieu: Byte;
                 End;
```

Chú ý:

- Pascal theo dõi các thao tác truy nhập thông qua con trỏ file. Mỗi khi một phần tử nào đó được ghi vào hay đọc từ file, con trỏ của file này được tự động chuyển đến phần tử tiếp theo.
- Các biến kiểu file không được phép có mặt trong phép gán hoặc trong các biểu thức.

II. CÁC THỦ TỤC VÀ HÀM CHUẨN

2.1. Các thủ tục chuẩn

2.1.1. Gán tên file

Cú pháp: **Assign(F, Filename);**

Chức năng: Gán một file trên đĩa có tên là Filename cho biến file F, mọi truy xuất trên file cụ thể được thực hiện thông qua biến file này.

Chú ý:

Filename bao gồm cả tên ổ đĩa và đường dẫn nếu file không nằm trong ổ đĩa, thư mục hiện thời.

2.1.2. Mở file mới

Cú pháp: **Rewrite(F);**

Chức năng: Tạo file mới có tên đã gán cho biến file F. Nếu file đã có trên đĩa thì mọi dữ liệu trên đó sẽ bị xoá và con trỏ file trỏ ở vị trí đầu tiên của file.

2.1.3. Mở file đã có trên đĩa

Cú pháp: **Reset(F);**

Chức năng: Mở file có tên đã gán cho biến file F. Nếu file chưa có trên đĩa thì chương trình sẽ dừng vì gặp lỗi xuất/nhập.

***Chú ý:* Kiểm tra khi mở file**

{$I+}: Mở việc kiểm tra. Khi gặp lỗi Vào/ra chương trình sẽ báo lỗi và dừng lại

{$I-}: Không kiểm tra Vào/ra, chương trình không dừng lại nhưng treo các thủ tục Vào/ra khác cho đến khi hàm **IOresult** (hàm chuẩn của PASCAL). Hàm trả về giá trị **true** nếu việc mở file xảy ra tốt đẹp.

Ví dụ:

```
Procedure MoFile;
Var  ok:Boolean;
     St:String;
     F:Text;
Begin
  Repeat
     Write('Nhập tên tệp: ');readln(st);
     Assign(F,st);
     {$I-} (*Chuyển việc kiểm tra vào ra cho người dùng*)
     Reset(F);
     Ok:=IOResult;
     {$I+}
     if not OK then writeln('Không mở được ');
  Until OK;
End;
```

2.1.4. Đọc dữ liệu từ file

Cú pháp: **Read(F, x);**

Chức năng: Đọc một phần tử dữ liệu từ file F ở vị trí con trỏ file và gán cho các biến x.

2.1.5. Ghi dữ liệu lên file

Cú pháp: **Write(F, Value);**

Chức năng: Ghi giá trị Value vào file F tại vị trí hiện thời của con trỏ file.

2.1.6. Di chuyển con trỏ file

Cú pháp: **Seek(F, n);**

Chức năng: Di chuyển con trỏ file đến phần tử thứ n (phần tử đầu tiên có thứ tự là 0).

2.1.7. Đóng file

Cú pháp: **Close(F);**

Chức năng: Cập nhật mọi sửa đổi trên file F và kết thúc mọi thao tác trên file này.

2.1.8. Xoá file

Cú pháp: **Erase(F);**

Chức năng: Xoá file trên đĩa có tên gán đã được gán cho biến file F (file cần xoá là file đang đóng).

2.1.9. Đổi tên file

Cú pháp: **Rename(F, NewFile);**

Chức năng: Đổi tên của file đang gán cho biến file F thành tên file mới là NewFile.

2.2. Các hàm chuẩn

2.2.1. Hàm trả về vị trí con trỏ file

Cú pháp: **Filepos(F);**

Chú ý: Con trỏ ở đầu file tương ứng vị trí 0.

2.2.2. Hàm kiểm tra cuối file

Cú pháp: **EOF(F);**

Chức năng: Hàm trả về giá trị ***True*** nếu con trỏ file đang ở cuối file, ngược lại hàm trả về giá trị ***False.***

2.2.3. Hàm trả về kích thước của file

Cú pháp: **FileSize(F);**

Chức năng: Hàm trả về ***số lượng phần tử*** có trong file.

III. FILE VĂN BẢN (TEXT FILE)

Thành phần cơ bản là ký tự, song có thể được cấu trúc thành các dòng, mỗi dòng được kết thúc bởi CR và LF, CR có mã ASCII là 13 và LF có mã 10. Cuối file sẽ có dấu kết thúc file Ctrl-Z có mã là 26.

Do các dòng có độ dài thay đổi nên không tính trước được vị trí của một dòng trong file. Vì vậy file dạng Text chỉ có thể đệoc xử lý một cách tuần tự.

3.1. Khai báo

Var **<Tên biến file>: Text;**

3.2. Các thủ tục và hàm chỉ tác động trên file dạng text

3.2.1. Thủ tục Append

Cú pháp: **Append(F);**

Chức năng: Mở file đã tồn tại để bổ sung nội dung vào cuối file.

3.2.2. Thủ tục Readln

Cú pháp: **Readln(F,x);**

Chức năng: Đọc một dòng từ vị trí con trỏ file và gán cho biến **x**. Thực hiện xong, con trỏ file sẽ chuyển về đầu dòng tiếp theo. Biến **x** có thể nhận các kiểu: Char, String hoặc kiểu số.

3.2.3. Thủ tục Writeln

Cú pháp: **Writeln(F, x);**

Chức năng: Ghi giá trị x vào file ở vị trí con trỏ file. Kết thúc thủ tục, con trỏ file sẽ chuyển về đầu dòng sau.

Chú ý:

Máy in được xem là một file dạng text, và biến được mở sẵn trong Unit Printer cho file này là LST. Vì vậy để in một dòng St ra máy in ta có thể dùng lệnh Writeln(LST,St).

3.2.4. Thủ tục Flush

Cú pháp: **Flush(F);**

Chức năng: Cập nhật nội dung của file có tên gán cho biến file F mà không cần dùng thủ tục Close và vẫn có thể thao tác trên file.

3.2.5. Thủ tục SetTextBuf

Cú pháp: **SetTextBuf(F, x);**

Chức năng: Thay đổi vùng nhớ đệm dành cho file dạng text với kích thước cho bởi biến x. Mặc định vùng nhớ này là 128 byte.

Chú ý:

Thủ tục này phải được gọi trước các thủ tục mở file: Reset, Rewrite, Append.

3.2.6. Hàm EOLn

Cú pháp: **EOLn(F);**

Chức năng: Hàm trả về giá trị ***True*** nếu con trỏ đang ở cuối một dòng, ngược lại hàm trả về giá trị ***False***.

***Chú ý*:**

- Các thủ tục và hàm không sử dụng được đối với file dạng text: ***Seek, FilePos, FileSize***.
- Sau đây là các thao tác cơ bản khi xuất nhập file:

Ghi dữ liệu vào file	Đọc dữ liệu từ file

ASSIGN(f,FileName);
REWRITE(f);
...
WRITE(f,value);
...
CLOSE(f);
	ASSIGN(f,FileName);
	RESET(f);
	...
	While Not EOF(f) Do
	Begin
	READ(f,x);
	...
	End;
	...
	CLOSE(f);

IV. FILE KHÔNG ĐỊNH KIỂU (FILE VẬT LÝ)

4.1. Khái niệm

File không định kiểu là file không xác định kiểu của mỗi thành phần trong file, mà được hiểu là một dãy byte, mỗi phần tử có kích thước k byte, quy định bởi người lập trình. File không định kiểu tương hợp với mọi kiểu file.

4.2. Khai báo

Var **<Tên biến File>: File;**

4.3. Các thủ tục và hàm có thể thao tác trên file không đinh kiểu

4.3.1. Mở file

Mở file chưa có trên đĩa: **Rewrite(F, k);**

Mở file đã có trên đĩa: **Reset(F, k);**

Giá trị k mô tả số lượng byte sẽ được đọc ghi trong một thao tác. Kích thước của file phải là bội số của k.

4.3.2. Xuất/ nhập dữ liệu

Cú pháp: **BlockRead(F, x, n [,Kq]);**
BlockWrite(F, x, n [,Kq]);

Chức năng:

- Đọc/ Ghi n "bản ghi". Mỗi "bản ghi" được hiểu là một phần tử k byte.
- x chứa nội dung đọc/ghi
- Kq là số lương"bản ghi" được thực hiện.

Chú ý:

File không định kiểu thường được dùng trong các thao tác sao chép với tốc độ cao.

BÁI TẬP MẪU

Bài tập 8.1: Tạo một file SINHVIEN.DAT để lưu thông tin của một lớp sinh viên. Mỗi sinh viên cần những thông tin sau: Họ tên, Ngày sinh, Quê quán, Điểm trung bình, Xếp loại (trường xếp loại do chương trình tự tính lấy dựa vào điểm trung bình như sau: nếu điểm trung bình < 5 thì xếp loại 'D', nếu 5 <= điểm trung bình < 6.5 thì xếp loai 'C', nếu 6.5 <= điểm trung bình < 8 thì xếp loại 'B', trường hợp còn lại xếp loại 'A').

```
Program Vi_du_1;
Type
   St20 = String[20];
   St10 = String[10];
   SinhVien = record
            Hoten: St20;
            Ngaysinh,Quequan: St10;
            DiemTb: real;
            Xeploai: Char;
        end;
Var
  f: File of SinhVien;
  filename:String;
  Sv: sinhvien;
  Bhoten:st20;
  i:word;
Begin
```

```
    write('Nhap ten file: ');
    readln(filename);
    assign(f,filename);
    rewrite(f);
    i:=1;
    repeat
        writeln('Nhap thong tin cua cac sinh vien');
        writeln('Thong tin cua sinh vien thu ', i);
        write('Ho ten: ');
        readln(Bhoten);
        if Bhoten <> '' then
          begin
            sv.hoten:= Bhoten;
            write('Ngay sinh (dd/mm/yyyy): ');
            readln(sv.ngaysinh);
            write('Quequan: ');
            readln(sv.quequan);
            write('Diem trung binh: ');
            readln(sv.diemtb);
            if sv.diemtb<5 then
              sv.xeploai:='D'
            else
              if sv.diemtb<6.5 then
                sv.xeploai:='C'
              else
                if sv.diemtb<8 then
                  sv.xeploai:='B'
                else
                  sv.xeploai:='A';
            write(f,sv);
          end;
        inc(i);
    until Bhoten = '';
    close(f);
end.
```

Bài tập 8.2: In toàn bộ nội dung của file SINHVIEN.DAT ra màn hình, nếu có, ngược lại thì thông báo “File khong ton tai”.

```
Program Vi_du_2;
Type
   St20 = String[20];
   St10 = String[10];
   SinhVien = record
                Hoten: St20;
                Ngaysinh,Quequan: St10;
```

```
                DiemTb: real;
                Xeploai: Char;
            end;
Var
  f: File of SinhVien;
  Sv: sinhvien;
  Bhoten:st20;
  i:word;
Begin
   assign(f,'Sinhvien.dat');
   {$I-}
   reset(f);
   {$I+}
   if IOResult <> 0 then
     Begin
        writeln('File khong ton tai');
        exit;
     End;
   writeln(#32:10, 'DANH SACH SINH VIEN');
   writeln(#32:6,'HO TEN',#32:8,'NGAY SINH',#32:4,'QUE QUAN  DTB');
   while not eof(f) do
       begin
          read(f,sv);
          with sv do
         writeln(hoten,#32:20,length(hoten),ngaysinh,#32:2,quequan,#32:10-
length(quequan),Diemtb:5:2);
       end;
   close(f);
   readln;
End.
```

Bài tập 8.3: In danh sách tất cả sinh viên có thông tin lưu trong file SINHVIEN.DAT xếp loại khá (‘B’) trở lên.

```
Program Vi_du_3;
Type
   St20 = String[20];
   St10 = String[10];
   SinhVien = record
                Hoten: St20;
                Ngaysinh,Quequan: St10;
                DiemTb: real;
                Xeploai: Char;
            end;
Var
  f: File of SinhVien;
```

```
  filename:String;
  Sv: sinhvien;
  Bhoten:st20;
  n:word;
Begin
    assign(f,'sinhvien.dat');
    {$I-}
    reset(f);
    {$I+}
    if IOResult <>0 then
      begin
          writeln('File khong ton tai');
          exit;
      end;
    n:=0;
    writeln('Danh sach sinh vien dat loai kha tro len');
    while not Eof(f) do
        begin
            read(f,sv);
            with sv do
                if xeploai <= 'B' then { (xeploai = 'B') or (xeploai = 'A') }
                  begin
                      writeln(hoten,ngaysinh,quequan,diemtb);
                      inc(n);
                  end;
        end;
   close(f);
        writeln('Danh sach nay gom ',n,' sinh vien');
    readln;
end.
```

Bài tập 8.4: Thông tin về điểm của sinh viên có họ tên là Bhoten, ngày sinh là Bngay và quê quán là Bquequan bị sai lệch. Hãy sữa điểm và xếp loại của sinh viên này với dữ liệu nhập từ bàn phím.

```
Program Vi_du_4;
Type
   St20 = String[20];
   St10 = String[10];
   SinhVien = record
                Hoten: St20;
                Ngaysinh,Quequan: St10;
                DiemTb: real;
                Xeploai: Char;
              end;
Var
```

```
  f: File of SinhVien;
  filename:String;
  Sv: sinhvien;
  Bhoten:st20;
  Bngaysinh,Bquequan:St10;
Begin
    assign(f,'sinhvien.dat');
    {$I-}
    reset(f);
    {$I+}
    if IOResult <>0 then
      begin
          writeln('File khong ton tai');
          exit;
      end;
    write('Ho ten sinh vien: ');
    readln(bhoten);
    write('Ngay sinh: ');
    readln(Bngaysinh);
    write('Que quan: ');
    readln(bquequan);
    while not Eof(f) do
        begin
            read(f,sv);
            with sv do
                if (hoten=bhoten) and ((ngaysinh=bngaysinh) and
                (quequan=bquequan))  then
              begin
                  write('Nhap dtb can sua: ');
                  readln(diemtb);
                  if diemtb <5 then
                    xeploai:='D'
                  else
                      if diemtb <6.5 then
                        xeploai:='C'
                      else
                          if diemtb <8 then
                            xeploai:='B'
                          else
                              xeploai:='A';
                  n:=filepos(f);
                  seek(f,n-1);
                  write(f,sv);
                  exit;
              end;
        end;
```

```
   Close(f);
        readln;
End.
```

Bài tập 8.5: In ra màn hình toàn bộ nội dung của một file văn bản, tên file được được nhập từ bàn phím khi thực hiện chương trình.

```
Program Vidu_5;
Var
   f: Text;
   filename,St: String;
Begin
   write('Nhap ten file: ');
   readln(filename);
   assign(f,filename);
   {$I-}
   reaset(f);
   {$I+}
   if IOResult <> 0 then
      begin
         writeln('File khong ton tai');
         halt;
      end;
   writeln('Noi dung cua file ',filename)
   while not Eof(f) do
      begin
         readln(f,st);
         writeln(st);
      end;
   close(f);
   readln;
End.
```

Bài tập 8.6: Đếm số dòng, số ký tự trắng xuất hiện trong một file văn bản đã có trên đĩa, tên file được nhập từ bàn phím khi chạy chương trình.

```
Program Vidu_6;
Var
   f: Text;
   filename,St: String;
   NLines,NStr: word;
   i: byte;
Begin
   write('Nhap ten file: ');
   readln(filename);
   assign(f,filename);
```

```
   reaset(f);
   NBl:=0;
   NStr:=0;
   while not Eof(f) do
      begin
         readln(f,st);
         inc(NStr);
         for i:= 1 to length(St) do
            if St[i] = #32 then
               inc(NBl);
      end;
   Close(f);
       writeln('So dong : ',NStr);
   writeln('So ky tu trang: ', NBl)
   readln;
End.
```

Bài tập 8.7: Sao chép nội dung của file SINHVIEN.DAT vào file văn bản SINHVIEN.TXT sao cho mỗi sinh viên lưu trong một dòng.

```
Program Vidu_7;
Type
   St20 = String[20];
   St10 = String[10];
   SinhVien = record
                 Hoten: St20;
                 Ngaysinh,Quequan: St10;
                 DiemTb: real;
                 Xeploai: Char;
              end;
Var
   f: File of SinhVien;
   g:Text;
   St:String;
   Sv: sinhvien;
   Bdiem: String[5];
Begin
      assign(f,'sinhvien.dat');
      {$I-}
      reset(f);
      {$I+}
      if IOResult <>0 then
         begin
               writeln('File khong ton tai');
               exit;
         end;
```

```
   rewrite(g);
   while not Eof(f) do
      begin
         read(f, Sv);
         with Sv do
            begin
               Str(diemtb,bdiem:5:2);
               St:= hoten+#32+ngaysinh+#32+quequan+#32+Bdiem;
               writeln(g,St);
            end;
      end;
   Close(f);
   Close(g);
   readln;
End.
```

Bài tập 8.8: Một ma trận mxn số thực được chứa trong một file văn bản có tên MT.INP gồm: dòng đầu chứa hai số m, n; m dòng tiếp theo lần lượt chứa m hàng của ma trận. Hãy viết chương trình đọc dữ liệu từ file MT.INP, tính tổng của từng hàng ma trận và ghi lên file văn bản có tên KQ.OUT trong đó, dòng đầu chứa số m, dòng thứ hai chứa m tổng của m hàng (m,n<=200).

```
MT.INP          ⇒       KQ.OUT
5 4                     5
3 8 –1 5                15  4  8  12  12
5 7 –8 0
4 –3 1 6
2 4 –1 7
3 6 8 -5
```

```
Program Vidu_8;
Var
   f,g: Text;
   S:array[byte] of real;
   m,n,i,j: byte;

Begin
   assign(f,'MT.INP');
   reset(f);
   readln(f,m,n);
   fillchar(S,m,0);
   for i:= 1 to m do
      begin
         for j:=1 to n do
            begin
               read(f,x);
               S[i]:=S[i]+x;
```

```
            end;
         readln(f);
      end;
   close(f);
   assign(g,'KQ.OUT');
      rewrite(g);
   writeln(g,m);
   for i:= 1 to m do
      write(g,S[i]:0:2,#32);
   close(g);
End.
```

Chú ý:

- Chương trình trên không kiểm tra sự tồn tại của file 'MT.INP', nếu cần có thể kiểm tra tương tự các ví dụ trên.
- Tổng của mỗi hàng được lưu trong mảng một chiều S (phần tử S[i] lưu tổng của hàng i)

Bài tập 8.9: Cho 3 ma trận số nguyên A = $(a_{ij})_{mxn}$, B = $(b_{jk})_{nxp}$, C = $(c_{kl})_{pxq}$, được chứa trong file MATRIX.INP gồm: dòng đầu chứa 4 số m, n, p, q. m+n+p dòng tiếp theo lần lượt chứa m hàng ma trận A, n hàng ma trận B và p hàng ma trận C. Viết chương trình đọc dữ liệu từ file MATRIX.INP và tính ma trận tích D = AxBxC rồi ghi lên file văn bản có tên MATRIX.OUT trong đó: Dòng đầu chứa m, q; m dòng tiếp theo chứa m hàng của ma trận D.

$$d_{il} = \sum_{j=1}^{n}\sum_{k=1}^{p} a_{ij} * b_{jk} * c_{kl}$$

```
Program Vidu_9;
Var
   f,g: Text;
   A, B, C, D:array[1..100,1..100] of integer;
   m,n,p,q,i,j,k,l,r,s: byte;

Begin
   assign(f,'MATRIX.INP');
   reset(f);
   readln(f,m,n,p,q);
   fillchar(D,mxq,0);
   for i := 1 to m do
      begin
         for j:= 1 to n do  read(f,A[i,j]);
         readln(f);
      end;
   for j:= 1 to n do
      begin
         for k:=1 to p do read(f,B[j,k]);
```

```
            readln(f);
        end;
    for k:= 1 to p do
        begin
            for l:=1 to q do read(f,C[k,l]);
            readln(f);
        end;
    close(f);
    assign(g,'MATRIX.OUT');
    rewrite(g);
    writeln(g,m,#32,q);
    for i:= 1 to m do
        begin
            for l:=1 to q do
                begin
                    for j:= 1 to n do
                      for k:=1 to p do
                            D[i,l] := D[i,l] + A[i,j]*B[j,k]*C[k,l];
                    write(g,D[i,l], #32);
                end;
            writeln(g);
        end;
    close(g);
    readln;
End.
```

Chú ý: Công thức tính giá trị của các phần tử ma trận $D = (d_{il})_{mxq}$ như sau:

Bài tập 8.10: Một ma trận mxn số thực được chứa trong một file văn bản có tên DULIEU.INP gồm: dòng đầu chứa hai số m, n; m dòng tiếp theo lần lượt chứa m hàng của ma trận. Hãy viết chương trình đọc dữ liệu từ file DULIEU.INP, cho biết các hàng của ma trận có tổng phần tử trên hàng đó lớn nhất. Kết quả ghi lên file văn bản có tên DULIEU.OUT , trong đó dòng đầu chứa giá trị lớn nhất của tổng các phần tử trên một hàng, dòng thứ hai chứa chỉ số các hàng đạt giá trị tổng lớn nhất đó (m,n<=100).

Chẳng hạn

DULIEU.INP	⇒	DULIEU.OUT
6 5		34
3 6 8 12 2		2 5 6
7 5 6 10 6		
8 2 4 5 1		
3 5 6 1 3		
10 12 3 1 8		
8 8 8 9 1		

```
Program Vi_du_10;
```

```
Var
   f,g: Text;
   S:array[1..100] of real;
   T: Set of byte;
   GTMax: real;
        m,n,i,j: byte;
Begin
   assign(f,'DULIEU.INP');
   reset(f);
   readln(f,m,n);
   fillchar(S,m,0);
        for i:= 1 to m do
      begin
         S:=0;
            for j:=1 to n do
             begin
                read(f,x);
                S[i]:=S[i]+x;
             end;
         readln(f);
      end;
   close(f);
   T:=[1];
   GTMax:=S[1];
   for i:= 2 to m do
      if  S[i] > GtMax then
         begin
            T:=[i];
            GtMax:= S[i];
         end
      else
         if S[i] = GTMax then
            T:= T+[i];
        assign(g,'DULIEU.OUT');
        rewrite(g);
   writeln(g,GTMax:0:2);
   for i:=1 to 100 do
      if i in T then
         write(g,i,#32);
   readln;
End.
```

Chú ý:

- Chương trình trên dùng mảng S để lưu tổng giá trị các phần tử trên mỗi hàng. Cụ thể, S[i] là tổng giá trị các phần tử trên hàng thứ i của ma trận đã cho.

- Tập T , GTMax lần lượt là tập chứa các chỉ số các hàng và giá trị lớn nhất của các phần tử trên mỗi hàng tại thời điểm đang xét. Xuất phát ta xem hàng thứ nhất có tổng giá trị lớn nhất. Khi xét hàng thứ i có các trường hợp sau:
  - S[i] > GTMax: S[i] mới là tổng lớn nhất và lúc này chỉ có hàng i đạt được giá trị này
  - S[i] = GTMax: có thêm hàng i đạt giá trị lơn nhất.
  - S[i] < GTMax: không có gì thay đổi

Bài tập 8.11: Viết chương trình sao chép nội dung của một file cho trước vào file khác, tên của file nguồn và file đích được nhập từ bàn phím khi chạy chương trình.

```
Program Sao_chep_File;
const
   bufsize = 200;
var
   f,g: file;
   File_nguon, file_dich: String;
   Buf: array[1..63000] of Byte;
   No_read, Temp: integer;
Begin
   write('Nhap ten file nguon: ');
   readln(file_nguon);
   assign(f,file_nguon);
   reset(f);
   write('Nhap ten file dich: ');
   readln(file_dich);
   assign(g,file_dich);
   rewrite(g);
   Temp:= filesize(f);
   while Temp > 0 do
      begin
         if bufsize < =Temp then
            No_read:= bufsize
         else
            No_read:= Temp;
      BlockRead((f, Buf, No_read);
      BlockWrite(g,Buf, No_Read);
      Temp:=Temp – No_read;
   end;
   close(g);
End.
```

BÀI TẬP TỰ GIẢI

Bài tập 8.12: Viết chương trình đổi tên một file đã có trên đĩa.

Gợi ý:
Dùng thủ tục Rename.

Bài tập 8.13: Viết chương trình xóa một file có trên đĩa.
Gợi ý:
Dùng thủ tục Erase.

Bài tập 8.14: Viết chương trình nối 2 file văn bản đã có trên đĩa thành một file thứ 3 với tên file được nhập vào từ bàn phím.
Gợi ý:
- Mở file 1 và file 2 để đọc dữ liệu, mở file 3 để ghi dữ liệu.
- Lần lượt đọc từng phần tử trong file 1 và 2 lưu vào file 3.
- Đóng cả ba file lại.

Bài tập 8.15: Viết chương trình thực hiện các công việc sau:
1. Tạo ra 2 file số nguyên và sắp xếp chúng theo thứ tự tăng dần.
2. Hãy nối 2 file đó lại với nhau thành file thứ 3 sao cho file mới vẫn có thứ tự tăng dần.

Gợi ý:
Xem giải thuật ở bài tập 5.15.

Bài tập 8.16: Cho đa thức $P(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$
Trong đó n là bậc của đa thức và $a_0, a_1, ..., a_n$ là các hệ số của đa thức được lưu trong một file văn bản với qui ước sau:
- Dòng đầu của file văn bản chứa bậc của đa thức và giá trị của x.
- Dòng tiếp theo chứa các hệ số của đa thức.

Ví dụ: $P(x) = 3 + 2x - 5x^2 + 4x^3$, x = 2.5 sẽ được lưu trong file văn bản như sau:
3 2.5
3 2 -5 4
Viết chương trình đọc file văn bản trên để lấy các số liệu rồi tính giá trị của đa thức.
Gợi ý:
- Tổ chức mảng để lưu đa thức.
- Viết thủ tục để đọc file text lưu vào mảng.
- Tham khảo bài tập 5.8.

Bài tập 8.17: Viết chương trình đếm số từ có trong một file văn bản.
Gợi ý:
- Viết hàm COUNT để đếm số từ của 1 dòng.
- Đọc từng dòng của file văn bản, dùng hàm COUNT để cộng dồn vào biến dem.

Bài tập 8.18: Tại một cửa hàng, người ta quản lý các hoạt động MUA/BÁN trong năm bằng cùng một loại hoá đơn. Mỗi hoá đơn là một bản ghi gồm các trường:
SoHoadon (số hoá đơn); Thang (tháng mua/bán); Mahang (mã hàng mua/bán); Loai (nhận một trong hai giá trị 'M'(mua) hoặc 'B' (bán)

Như vậy căn cứ vào trường Loai ta biết đó là hoá đơn mua hay hoá đơn bán. Viết chương trình cho phép nhập vào một dãy các hoá đơn và lưu vào file có tên Hoadon.dat, quá trình nhập dừng khi SoHoadon = 0. Tính số dư trong tháng n (n được nhập từ bàn phím khi thực hiện chương trình) . Biết rằng số dư trong một tháng được tính theo công thức:

Số dư = Tổng bán - Tổng mua,

trong đó tổng bán, tổng mua lần lượt là tổng số tiền bán, mua trong tháng đó.

Yêu cầu:

- Khi nhập chú ý kiểm tra để Loai chỉ nhận một trong hai giá trị 'M' hoặc 'B' và tháng chỉ nhận giá trị từ 1 đến 12.
- Không được sử dụng mảng.

Hướng dẫn: Khai báo file lưu các hoá đơn, mỗi hoá đơn là một bản ghi như sau

```
Type
   Hoadon = record
                SoHoadon: word;
                Thang: byte;
                Mahang: string[5];
                Loai: char;
            end;
Var
   f: file of  hoadon;
```

Bài tập 8.19: Người ta quản lý các đầu sách của một thư viện bằng một bản ghi gồm có các trường: Masach, Tensach, Tentacgia, Nhaxb (nhà xuất bản), Namxb (năm xuất bản), SoLuong. Viết chương trình cho phép thực hiện các thao tác sau:

a. Nhập vào các đầu sách có trong thư viện và lưu vào file có tên Sach.dat, quá trình nhập dừng khi mã sách đưa vào là một xâu rỗng.
b. Duyệt và in ra tên các quyển sách được xuất bản sau năm m (m được nhập từ bàn phím khi thực hiện chương trình).
c. Bổ sung sách vào thư viện theo yêu cầu: nếu sách đã có thì chỉ tăng số lượng sách bổ sung, ngược lại thêm một đầu sách mới vào file.

Chú ý:

- Không được sử dụng mảng
- Khi nhập chú ý kiểm tra để năm xuất bản <= năm hiện tại
- Sau khi in ra danh sách các đầu sách xuất bản sau năm m, cho biết thêm danh sách đó có bao nhiêu đầu sách tất cả.

Hướng dẫn: Khai báo thư viện là một file các đầu sách, mỗi đầu sách là một bản ghi như sau

```
Type
   St5 = String[5];
   St20 = String[20];
   Dausach = Record
                Masach: St5,
                Tensach, Tentacgia, Nhaxb: St20,
                Namxb: word;
```

```
            SoLuong: byte;
        end;
Var
   f: file of DauSach;
```

Bài tập 8.20: Người ta lưu thông tin các cán bộ trong cơ quan vào file có tên CANBO.DAT, mỗi cán bộ là một bản ghi gồm các trường: STT, Hoten, Ngaysinh, Diachi, HSLuong, HSPhucap, SoDT. Hãy viết chương trình thực hiện các yêu cầu sau:

a. Nhập danh sách cán bộ và lưu vào file, quá trình nhập dừng khi họ tên nhập vào là xâu rỗng và trường STT chương trình tự gán.
b. In ra danh sách cán bộ có hệ số lương nằm trong khoảng từ x đến y, x và y là các số thực được nhập từ bàn phím khi thực hiện chương trình.
c. Sao chép thông tin các cán bộ có tuổi trên 50 vào một file khác.
d. In bảng lương của tất cả cán bộ lưu trong file CANBO.DAT ra màn hình gồm các thông tin: STT, Hoten, HSLuong, Luong, trong đó Luong được tính theo công thức Luong = (HSLuong+HSPhucap)*290000, dữ liệu in ra định dạng theo cột. Cuối bảng, in tổng lương của toàn cơ quan.
e. Sao chép nội dung của file CANBO.DAT vào file văn bản CANBO.TXT, mỗi cán bộ tương ứng một dòng.

Hướng dẫn: Khai báo mỗi cán bộ là một bản ghi như sau

```
Type
   St10 = String[10];
   St20 = String[20];
   Canbo = Record
              Hoten, Diachi: St20,
              Ngaysinh: St10; {dd/mm/yyyy}
              HSluong,  HSPhucap: real;
              SoDT: St10; {Số điện thoại }
           end;
Var
   f: file of Canbo;
```

- Khi nhập ngày sinh phải kiểm tra định dạng theo yêu cầu: dd/mm/yyyy
- Tuổi của một cán bộ được tính bằng năm hiện tại trừ cho năm sinh. Năm sinh lấy từ 4 ký tự cuối cùng của ngày sinh và chuyển sang dạng số.

Bài tập 8.21: Viết chương trình nhập vào tên một file văn bản. Kiểm tra file này có tồn tại trên đĩa không? Nếu có, in nội dung của file từ dòng thứ m đến dòng thứ n, trong đó m và n là hai số nguyên dương bất kỳ được nhập từ bàn phím khi thực hiện chương trình.

Hướng dẫn: Mở file bằng thủ tục Reset, rồi chuyển con trỏ về dòng thứ m, đọc và in n dòng (hoặc cho đến hết file).

Bài tập 8.22:Giả sử trong một file văn bản trên đĩa có tên là MATRIX.TXT người ta đã lưu các số liệu về một ma trận A cấp mxn và một vector X n chiều. Cách lưu trữ như sau:

- Dòng đầu tiên chứa hai số m và n

- Dòng thứ hai chứa vector X
- m dòng tiếp theo lần lượt chứa m hàng của ma trận A
- Giữa các số trong một dòng cách nhau một ký tự trắng

Viết chương trình tính giá trị vector Y = AX và đưa kết quả ra màn hình đồng thời lưu vào cuối file MATRIX.TXT (A và X được lấy từ file MATRIX.TXT)

Yêu cầu:

Chương trình phải thiết lập các thủ tục sau

- LayDulieu(A,X,m,n) thực hiện việc đọc dữ liệu từ file MATRIX.TXT và gán cho A, X, m, n
- TinhTich(A,X,m,n,Y) thực hiện việc tính vector Y
- LuuKetqua(Y,m) thực hiện việc in vector Y ra màn hình và lưu vào cuối file MATRIX.TXT
- Thành phần thứ i của vector Y được tính theo công thức

$$Y[i] = \sum_{j=1}^{m} A[i, j] * X[j]$$

Bài tập 8.23: Giả sử trong một file văn bản trên đĩa có tên là DANHBA.TXT lưu danh bạ điện thoại trong thành phố. Cách lưu như sau:

- Dòng đầu lưu hai số nguyên dương m và n, trong đó m là số máy điện thoại thuộc cơ quan nhà nước, còn n là số máy thuộc tư nhân.
- m dòng tiếp theo lưu thông tin lần lượt của m máy điện thoại thuộc cơ quan nhà nước, mỗi dòng ghi số điện thoại, một ký tự trắng và sau đó là tên cơ quan.
- n dòng tiếp theo nữa lưu thông tin lần lượt của n máy điện thoại tư nhân, mỗi dòng ghi số điện thoại, một ký tự trắng và sau đó là họ tên chủ điện thoại.

Viết chương trình đọc dữ liệu từ file DANHBA.TXT và in bảng danh bạ điện thoại ra màn hình theo thứ tự tăng dần của chủ máy điện thoại, các máy điện thoại thuộc cơ quan nhà nước in trước rồi đến các máy điện thoại tư nhân. Danh sách in ra theo 3 cột, cột 1 ghi số điện thoại, cột 2 ghi tên cơ quan hoặc tên chủ máy điện thoại, cột 3 ghi loại là TN (tư nhân) hoặc NN (nhà nước)

Yêu cầu:

- Khai báo kiểu bản ghi là MAYDT bao gồm 3 trường: SoDt, TenChu, Loai
- Thiết lập thủ tục LayDulieu(A,k) để đọc dữ liệu từ file DANHBA.TXT và lưu vào mảng A (mảng các MAYDT) với k là số phần tử của mảng.
- Thiết lập thủ tục SAPXEP(A,k) để sắp xếp mỗi nhóm máy điện thoại nhà nước, tư nhân theo thứ tự tăng dần của tên chủ máy điện thoại trong mảng A.
- Thiết lập thủ tục INKETQUA(A,k) để in ra màn hình danh bạ điện thoại từ mảng A.

Bài tập 8.24: Cho một file văn bản có có tên là MATRIX.TXT với nội dung như sau:

- Dòng đầu tiên của file chứa hai số nguyên dương m và n lần lượt là số hàng và số cột của một ma trận cấp mxn (m,n <=50).
- m dòng tiếp theo mỗi dòng chứa n số nguyên là gía trị các phần tử của mỗi hàng.

Hãy viết chương trình thực hiện các yêu cầu sau:

a. Viết thủ tục LAYDULIEU để đọc dữ liệu từ file MATRIX.TXT và lưu vào mảng hai chiều A.
b. Viết hàm MAXDONG(i:Byte): LongInt trả về giá trị lớn nhất của hàng i.
c. Ghi các giá trị lớn nhất của mỗi hàng vào cuối file MATRIX.TXT.

Bài tập 8.25: Viết chương trình tạo ra hai tập tin lưu các số kiểu word mà các số trong mỗi file đã được sắp thứ tự tăng dần. Hãy tạo tập tin mới chứa tất cả các số của 2 tập tin trên sao cho thứ tự tăng dần vẫn được duy trì.
Chú ý: Không được dùng mảng.

Bài tập 8.26: Giả sử trong một file văn bản trên đĩa có tên là MT.DAT người ta đã lưu các số liệu về hai ma trận A và B cùng cấp mxn. Cách lưu trữ như sau:

- Dòng đầu tiên chứa hai số m và n

m dòng tiếp theo lần lượt chứa m hàng của ma trận A
m dòng tiếp theo nữa lần lượt chứa m hàng của ma trận B
Giữa các số trong một dòng cách nhau một ký tự trắng
Viết chương trình tính ma trận tổng C = A + B và ghi kết quả vào file MT.OUT với cấu trúc: dòng đầu chứa số m, m dòng tiếp theo chứa ma hàng của ma trận C.

Bài tập 8.27: Để có thể sao chép các file có kích thước lớn lên đĩa mềm, người ta chia nhỏ file cần chép thành nhiều file có kích thước nhỏ hơn, sau đó nối các file này lại bằng lệnh copy. Hãy viết chương trình sao chép một file thành hai file có kich thước bằng nhau. Tên của tập tin nguồn và hai tập tin đích được nhập từ bàn phím khi thực hiện chương trình.
Hướng dẫn: Khai báo các file nguồn và đích là các file không định kiểu. Gọi Temp là một nửa kích thước của file nguồn, tính bằng byte. Thực hiện việc sao chép từ byte đầu tiên đến byte thứ Temp vào file đích thứ nhất, sau đó chép phần còn lại của file nguồn vào file đích thứ hai.

Chương 9
DỮ LIỆU KIỂU CON TRỎ

I. KHAI BÁO

```
Type
   <Tên kiểu con trỏ> = ^ <Kiểu của biến động>;
Var
   <Tên biến>:<Tên kiểu con trỏ>;
```

Ví dụ 1:

```
Type
   TroNguyen : ^integer;
Var
   p, q: TroNguyen;
```

Sau khai báo này các biến p và q là các biến con trỏ có thể trỏ đến các biến động có kiểu integer. Chương trình sẽ cấp phát 4 byte cho mỗi biến con trỏ. Còn vùng nhớ của các biến động chưa được cấp phát.

Ví dụ 2:

```
Type
   TroSv = ^ Sinhvien;
   Sinhvien = Record
                 Hoten: String[20];
                 Diem: real;
                 Tiep: TroSv;
              End;
Var
   p: TroSv;
```

Trong ví dụ này, p là biến trỏ có thể trỏ đến các bản ghi có kiểu Sinhvien, trong bản ghi này lại có trường Tiep là một biến trỏ có thể trỏ đến biến động khác cũng có kiểu Sinhvien.

II. LÀM VIỆC VỚI BIẾN ĐỘNG

2.1. Cấp phát vùng nhớ

Dùng thủ tục New theo cú pháp:

New(<biến trỏ>);

Phép gán giữa hai biến trỏ được thực hiện nếu chúng có cùng kiểu. Sau phép gán p:=q; các con trỏ p và q cùng trỏ đến một địa chỉ. Do đó mọi thay đổi của p^ cũng làm thay đổi q^. Như vậy, cần phân biệt hai phép gán p:=q và p^:=q^. Ngoài ra, các con trỏ cùng kiểu có thể được so sánh với nhau bằng các toán tử quan hệ = và <>.

Turbo Pascal cũng khai báo sẵn một con trỏ không trỏ tới một biến động nào gọi là con trỏ **Nil**. Giá trị con trỏ Nil là tương hợp với mọi kiểu con trỏ. Nil có thể được

gán cho biến con trỏ để chỉ ra rằng con trỏ ấy hiện không được sử dụng. Chúng ta cũng có thể sử dụng Nil trong các phép so sánh.

2.2. Giải phóng vùng nhớ

Dùng thủ tục **Dispose(p);**

Trong đó p là một biến con trỏ. Thủ tục Dispose cho phép trả lại bộ nhớ động đã được cấp phát bởi thủ tục New.

III. DANH SÁCH ĐỘNG

3.1. Khái niệm

Chúng ta đã từng làm quen với kiểu mảng, lưu danh sách gồm nhiều thành phần có cùng kiểu. Mỗi thành phần là một biến tĩnh và số lượng thành phần của danh sách là cố định. Ở đây chúng ta đề cập đến một dạng danh sách động theo nghĩa: mỗi thành phần là một biến động và số lượng thành phần của danh sách có thể thay đổi. Mỗi biến động trong danh sách được gọi là một nút.

3.2. Khai báo

Để khai báo một danh sách động trước hết ta khai báo kiểu của mỗi nút trong danh sách.

```
Type <Trỏ nút> = ^ <Nút>;
     <Nút> = Record
                Data: DataType;
                Next: <Trỏ Nút>;
             End;
Var  First: <Trỏ Nút>;
```

First là địa chỉ của nút đầu tiên trong danh sách, dựa vào trường Tiep của nút này ta bết được địa chỉ của nút thứ hai, cứ như vậy ta biết được địa chỉ của tất cả các nút trong danh sách. Danh sách dạng này được gọi là danh sách liên kết đơn.

3.3. Các thao tác thường gặp trên danh sách liên kết đơn

Trong phần này chúng ta giả thiết rằng mỗi nút trong danh sách có hai trường: trường Info (lưu nội dung của biến động) và trường Next (lưu địa chỉ của nút tiếp theo). ta có khai báo danh sách như sau

```
Type TroNut = ^Nut;
Nut = Record
        Info: data; {data là kiểu dữ liệu đã định nghĩa trước}
        Next: TroNut;
      End;
Var First:TroNut;
```

3.3.1. Khởi tạo danh sách

```
First:=Nil;
```

3.3.2. Bổ sung một nút vào đầu danh sách

```
{1. Tạo ra nút mới}
```

```
   New(p);
   p^.Info:=X;
{2. Bổ sung vào đầu danh sách}
   p^.Next:=First;
   First:=p;
```

3.3.3. Bổ sung một nút vào cuối danh sách

Xuất phát danh sách không có nút nào cả. Nút mới thêm vào sẽ nằm cuối danh sách. Khi đó ta cần hai biến con trỏ First và Last lần lượt trỏ đến các nút đầu và cuối danh sách.

```
Procedure Khoitao;
var p: TroNut;
Begin
   First:= nil; Last:= nil;
   While <còn thêm nút mới vào danh sách> do
      Begin
         New(p);
         Readln(p^.Info);
         p^.Next:= Nil;
         If First = Nil then
               First:= p
         Else
               Last^.next:= p;
         Last:= p;
      End;
End;
```

3.3.4. Duyệt danh sách

Duyệt danh sách là thăm và xử lý từng nút trong danh sách.

```
   Procedure Duyet;
   Var p: Tronut;
   Begin
     p:= First;
     While p <> nil do
        Begin
           <Xử lý p>;
           p:= p^.Next; {duyệt qua nút tiếp theo}
        End;
   End;
```

3.3.5. Bổ sung một nút vào sau nút được trỏ bởi p

Thủ tục sau thực hiện việc bổ sung một nút có nội dung x vào sau nút được trỏ bởi p.

```
   Procedure Bosung(p,x);
```

```
Var   q: TroNut;
Begin
   New(q);
   q^.info:=x;
   if first = nil then
      begin
         q^.next := nil;
         first := q;
      end
   else
      begin
         q^.next:= p^.next;
         p^.next:= q;
      end;
End;
```

3.3.6. Xoá một nút khỏi danh sách

Thủ tục sau thực hiện việc xóa một nút trỏ bởi p ra khỏi danh sách.

```
Procedure Xoa(p);
Var   q: TroNut;
Begin
   if First = nil then
      exit;
   if p = First then
      First := First^.next
   else
      begin
         q:= First;
         While q^.next <> p do
            q:= q^.next;
         q^.next:= p^.next;
      end;
   Dispose(p);
End;
```

BÀI TẬP MẪU

Bài tập 9.1: Trong các bài tập từ 9.1 đến 9.4, dùng danh sách liên kết đơn lưu một dãy số nguyên. Nút đầu tiên trong danh sách được trỏ bởi First. Cho khai báo mỗi nút trong danh sách như sau:

```
Type TroNut = ^ Nut;
   Nfut = Record
      GiaTri: Integer;
      Tiep: TroNut;
      End;
Var   First: TroNut;
```

Viết chương trình thực hiện các yêu cầu sau:
a. Nhập dãy các số nguyên và lưu vào danh sách có nút đầu trỏ bởi First, quá trình nhập dừng khi dữ liệu đưa vào không phải là số nguyên.
b. In giá trị các nút lớn hơn 0.

```
Program Vi_du_1;
TypeTroNut = ^ Nut;
   Nut = Record
      GiaTri: Integer;
      Tiep: TroNut;
      End;
Var  First: TroNut;
   p: pointer;

Procedure Nhap;
Var
   n:integer;
   kq:boolean;
   last,p: tronut;
begin
   first:=nil;
   last:= nil;
   repeat
      write('Nhap gia tri mot nut – Ket thuc bang ky tu Q: ');
      {$I-}
      readln(n);
      {$I+}
      kq:= IOResult=0;
      if kq then
         begin
            new(p);
            p^.Giatri:=n;
            p^.Tiep:=nil;
            if first = nil then
               first:= p;
            else
               last^.Tiep:= p;
            last:=p;
         end;
   until not kq;
end;

Procedure In_so_duong;
Var
   p: Tronut;
begin
```

```
   p:= first;
   while p <> nil do
      begin
         if p^.Giatri > 0 then
            write(p^.Giatri:8);
         p:=p^.Tiep;
      end;
end;

Begin
   Mark(p);
   Nhap;
   In_so_duong;
   Release(p);
   Readln;
End.
```

Bài tập 9.2: Viết thủ tục đếm số nút có giá trị lớn hơn 0 và tính giá trị trung bình cộng của các nút đó.

```
Procedure Nut_duong(var dem: word; tb:real);
Var
   p: Tronut;
   tong:longint;
begin
   dem:=0;
   tong:=0;
   p:= first;
   while p <> nil do
      begin
         if p^.Giatri > 0 then
            begin
               inc(dem);
               tong:=tong+p^.Giatri;
            end;
         p:=p^.tiep;
   if dem = 0 then
      tb:=0
   else
      tb:= tong /dem;
end;
```

Bài tập 9.3: Giả sử dãy giá trị các nút trong danh sách đã được sắp tăng dần. Viết các thủ tục và hàm sau:

a. **Procedure Insert(var first: TroNut; m: integer)** thực hiện việc bổ sung một nút vào danh sách sao cho tính tăng dần được bảo toàn.

```
Procedure Insert(var first: TroNut; m: integer);
Var
   p,q: Tronut;
begin
   new(p);
   p^.Giatri:= m;
   if (first = nil) or (first^.Giatri < m ) then
      begin
         p^.Tiep:=nil;
         first:= p;
      end
   else
      begin
         q:= first;
         while (q^.Tiep <> nil) and ((q^.Tiep)^.Giatri < m) do
            q:= q^.Tiep;
         p^.Tiep:= q^.tiep;
         q^.Tiep:= p;
      end;
end;
```

b. Procedure InitList thực hiện việc tạo danh sách có tính chất trên bằng cách nhập dữ liệu từ bàn phím và quá trinh nhập dừng khi nhấn phím ESC (yêu cầu: sử dụng thủ tục Insert).

```
Procedure InitList;
Var
   m: integer;
Begin
   first:= nil;
   repeat
      write('Nhap gia tri cua mot nut: ');
      readln(m);
      insert(first,m);
   until readkey = #27;
end;
```

c. Procedure List(First: TroNut) in dãy giá trị các nút trong danh sách.

```
Procedure List(First: Tronut);
Var
   p:Tronut;
begin
   p:= first;
   while p <> nil do
```

```
        begin
            write(p^.Giatri);
            p:=p^.Tiep;
        end;
end;
```

d. Procedure DeleteZero(Var First: TroNut) thực hiện việc xoá tất cả các nút có giá trị 0 trong danh sách.

```
Procedure DeleteZero(Var First: TroNut);
var
   p,q: Tronut;
begin
   p:= first;
   while (p <> nil) and (p^.Giatri < 0) do
      begin
         q:= p;
          p:= p^.Tiep;
      end;
   while (p <> nil) and (p^.Giatri = 0) do
      begin
         q^.Tiep:= p^.Tiep;
         dispose(p);
         p:= q^.Tiep;
      end;
end;
```

e. Function TroMax(First: TroNut): TroNut trả về địa chỉ của nút đầu tiên đạt giá trị lớn nhất (tính từ đầu danh sách, nếu có, ngược lại hàm trả về giá trị Nil).

```
Function Tromax(First: TroNut);
var
   p.q: Tronut;
   m:integer;
begin
   if first = nil then
      TroMax:= nil
   else
      begin
         p:= first;
          m:= p^.Giatri;
         q:= p^.Tiep;
         while (q <> nil) do
            begin
              if q^.Giatri > m then
                begin
```

```
                p:= q;
                m:= p^.Giatri;
              end;
            q:= q^.Tiep;
          end;
        TroMax:=p;
        end;
end;
```

Bài tập 9.4: Giả sử danh sách khác rỗng. Viết các thủ tục và hàm sau:

a. Function GiaTriMax(First: TroNut): integer trả về giá trị lớn nhất của nút có trong danh sách.

```
Function GiaTriMax(First: TroNut): integer;
var
   m: integer;
   p, q: Tronut;
begin
   p:= first;
   m:= p^.Giatri;
   q:= p^.Tiep;
   while q<> nil do
      begin
         if q^.Giatri > m then
            m:=q^.Giatri;
         q:= q^.Tiep;
   GiaTriMax:= m;
end;
```

b. Function GiaTriMin(First: TroNut): Integer trả về giá trị nhỏ nhất của nút có trong danh sách.

```
Function GiaTriMax(First: TroNut): integer;
var
   m: integer;
   p,q: Tronut;
begin
   p:= first;
   m:= p^.Giatri;
   q:= p^.Tiep;
   while q<> nil do
      begin
         if q^.Giatri < m then
            m:=q^.Giatri;
         q:= q^.Tiep;
   GiaTriMin:= m;
```

```
end;
```

Bài tập 9.5: Cho danh sách liên kết đơn có nút đầu trỏ bởi First, được khai báo như sau

```
Type
        TroNut = ^nut;
        Nut = Record
                Info: real;
                Next: TroNut;
              End;
Var
        First: Tronut;
```

Viết các thủ tục và hàm sau:

a. Function Search(First: TroNut; k: word): TroNut trả về địa chỉ của nút thứ k (nếu có, ngược lại, hàm trả về giá trị Nil).

```
Function Search(First: TroNut; k: word): Tronut;
Var
        d: word;
        p: Tronut;
Begin
        d:=0;
        p:=first;
        while (p <> nil) do
          begin
            inc(d);
            if d = k then
               break;
            p:= p^.next;
        end;
        Search:= p;
End;
```

b. Procedure Delete_K(Var First: TroNut; k: word) thực hiện việc xoá nút thứ k trong danh sách (nếu có).

```
Procedure Delete_K(Var first: Tronut; k:word);
var
        d: word;
        p,q: Tronut;
begin
        d:=1;
        p:= first;
        while (p<> nil) and (d <k) do
            begin
               q:= p;
```

```
                p:= p^.Next;
                inc(d);
            end;
        if p <> nil then
            begin
                if  p = first then
                    first:= first^.next
                else
                    q^.next:= p^.next;
                dispose(p);
            end;
end;
```

c. Procedure DeleteList thực hiện việc xoá tất cả các nút trong danh sách.

```
Procedure DeleteList;
var
   p: Tronut;
begin
   while first <> nil do
      begin
         p:= first;
         first:= first^.next;
         dispose(p);
      end;
end;
```

Bài tập 9.6: Cho file văn bản có tên NGUYEN.INP lưu các số nguyên, giữa các số trong file cách nhau một ký tự trắng hoặc dấu xuống dòng. Viết chương trình thực hiện các yêu cầu sau:

a. Lấy dữ liệu từ file NGUYEN.INP và lưu vào danh sách liên kết đơn có nút đầu trỏ bởi First.

b. Tính tổng giá trị các nút, tổng giá trị các nút dương, tổng giá trị các nút âm, số nút có giá trị âm, số nút có giá trị dương. Các kết quả tính được sẽ lưu vào file văn bản có tên KETQUA.OUT, dòng đầu chứa 3 giá tri tổng, dòng thứ hai chứa hai giá trị còn lại.

```
Program Vi_du_6;
type
   Contro = ^ Nut;
   Nut = Record
         info: integer;
         next: Contro;
      end;
var
   first: Contro;
```

```
Procedure Lay_du_lieu;
var
   p: Contro;
   so: integer;
   f: text;
Begin
   assign(f, 'NGUYEN.INP');
   reset(f);
   first:= nil;
   while not Eof(f) do
      begin
         read(f, so);
         new(p);
         p^.info:= so;
         p^.next:= first;
         first:= p;
      end;
   close(f);
End;

Procedure Tinh_toan;
var
   f:text;
   p: Contro;
   T, T_duong, T_am: longint;
   N_duong, N_am: word;
begin
   assign(f,'KETQUA.OUT');
   rewrite(f);
   p:= first;
   T:= 0;
   T_duong: =0;
   T_am:= 0;
   N_duong:= 0;
   N_am:= 0;
   while p <> nil do
      begin
         T:= T + p^.info;
         if p^.info > 0 then
            begin
               T_duong:= T_duong + p^.info;
               inc(N_duong);
            end;
         if p^.info < 0 then
            begin
               T_am:= T_am + p^.info;
```

```
                inc(N_am);
            end;
        p:= p^.next;
    end;
  writeln(f, T,#32,T_duong,#32,T_am);
  writeln(f,N_duong,#32,N_am);
  close(f);
end;

Begin
  Lay_du_lieu;
  Tinh_toan;
End.
```

Bài tập 9.7: Người ta lưu thông tin các bệnh nhân của bệnh viện X trong danh sách liên kết đơn có nút đầu trỏ bởi First, mỗi bệnh nhân tương ứng với một nút trong danh sách được khai báo như sau:

```
Type St20 = String[20];
       St5 = String[5];
       St2 = String[2];
  TroBN = ^BenhNhan;
       BenhNhan =    Record
           MaBN: St5;       {Mã bệnh nhân}
                            Hoten: St20;      {Họ tên bệnh nhân}
           Tuoi: byte;    {Tuổi}
           Tiep: TroBN;
        End;
```

Chú ý: *Hai ký tự đầu của mã bệnh nhân là mã của khoa điều trị.*

Viết các thủ tục và hàm sau:

a. Procedure BoSungBN(Var First: TroBN; Bma: St5; Bten: St20; Btuoi: byte) bổ sung bệnh nhân có mã là Bma, họ tên là Bten, tuổi là Btuoi vào cuối danh sách có nút đầu trỏ bởi First (Lưu ý: Kiểm tra Bma chưa có trong danh sách mới bổ sung).

```
Procedure BoSungBN(var First: TroBN; Bma: St5; Bten: St20; Btuoi:byte);
var
  p,q: TroBN;
begin
  p:= first;
  while (p <> nil) and (p^.MaBN <> Bma) do
     begin
        q:= p;
        p:= p^.tiep;
     end;
  if p = nil then
```

```
    begin
       new(p);
       p^.MaBn:= Bma;
       p^.Hoten:= Bten;
       p^.tuoi:= Btuoi;
       p^.Tiep:= nil;
       if first = nil then
          first:= p
       else
          q^.tiep:= p;
end;
```

b. Procedure KhoiTao(Var First: TroBN) nhập dữ liệu cho danh sách có nút đầu trỏ bởi First, quá trình nhập dừng khi mã bệnh nhân đưa vào là xâu rỗng (Yêu cầu sử dụng thủ tục BoSungBN).

```
Procedure KhoiTao(Var First: TroBN);
var
   bma:St5;
   bten: st20;
   btuoi: byte;
begin
   first:= nil;
   repeat
      write('Nhap ma benh nhan: ');
         readln(bma);
      if bma <> '' then
         begin
            write('Ho ten benh nhan: ');
            readln(bten);
               write('Tuoi: ');
            readln(btuoi);
            BosungBN(first, bma, bten, btuoi);
         end;
   until bma = '';
end;
```

c. Function SoBN(First: TroBN; BKhoa: St2): word trả về số lương bệnh nhân điều trị tại khoa có mã BKhoa.

```
Function SoBN(First: TroBN; BKhoa: St2): word;
Var
   p: TroBN;
   dem:word;
Begin
   dem:= 0;
```

```
   p:= first;
   while p <> nil do
      begin
         if copy(p^.MaBN,1,2) = BKhoa then  inc(dem);
         p:= p^.tiep;
      end;
   SoBN:= dem;
End;
```

d. Procedure LietKe(First: TroBN; n: byte) in thông tin của các bệnh nhân có tuổi nhỏ hơn hoặc bằng n.

```
Procedure LietKe(First: TroBN; n: byte);
Var
   p: TroBN;
Begin
   p:= first;
   while p <> nil do
      begin
         with p do
            if tuoi <= n then
                  writeln(mabn, #32,hoten, #32, tuoi);
         p:= p^.tiep;
      end;
End;
```

e. Procedure XoaBN(Var First: TroBN; Bma: St5) xoá bệnh nhân có mã Bma khỏi danh sách.

```
Procedure XoaBN(Var First: TroBN; Bma: St5);
Var
   p,q: TroBN;
Begin
   p:= first;
   while (p <> nil) and (p^.MaBN <> Bma) do
      begin
         q:= p;
         p:= p^.tiep;
      end;
   if p <> nil then
      begin
         if p = first then
            first:= first^.tiep
         else
            q^.tiep:= p^.tiep;
         dispose(p);
```

```
End;
```

Bài tập 9.8: Người ta lưu thông tin của mỗi đại lý trong công ty bởi một nút trong danh sách liên kết đơn và được khai báo như sau:

```
Type St6 = String[6];
   TroDL = ^ DaiLy;
   DaiLy = Record
         SoDT: St6;
         DoanhThu: LongInt;
         Next: TroDL;
      End;
```

Viết các thủ tục và hàm:

a. Procedure BoSung(Var First: TroDL; Tel: St6; m: LongInt) để bổ sung một đại lý có số điện thoại Tel và doanh thu là m vào đầu danh sách có nút đầu trỏ bởi First.

```
Procedure BoSung(Var First: TroDL; Tel: St6; m: LongInt);
Var
   p: TroDL;
Begin
   new(p);
   p^.SoDt:= Tel;
   p^.Doanhthu:= m;
   p^.next:= first;
   first:= p;
End;
```

b. Function DThu(First: TroDL; Tel: St6): LongInt trả về doanh thu của đại lý có số điện thoại là Tel, nếu không có đại lý đó thì hàm trả về giá trị 0.

```
Function DThu(First: TroDL; Tel: St6): LongInt;
Var
   p: TroDL;
Begin
   p:= first;
   while (p <> nil) and (p^.SoDT <> Tel) do
      p:= p^.next;
   if p <> nil then
      Dthu:= p^.doanhthu
   else
      Dthu:= 0;
End;
```

c. Function TongDThu(First: TroDL): Real trả về tổng doanh thu của tất cả các đại lý trong công ty.

```
Function TongDThu(First: TroDL): Real;
Var
   p: TroDl;
   T: real;
Begin
   T:= 0;
   p:= first;
   while p <> nil do
      begin
         T:= T+ p^.Doanhthu;
         p:= p^.next;
      end;
   TongDthu:= T;
End;
```

d. Function DemDL(First: TroDL; m: LongInt): Word trả về số đại lý của công ty có doanh thu lớn hơn m.

```
Function DemDL(First: TroDL; m: LongInt): Word;
Var
   p: TroDL;
   dem: word;
Begin
   dem:= 0;
p:= first;
   while p <> nil do
      begin
         if p^.Doanhthu > m then
            inc(dem);
         p:= p^.next;
      end;
   DemDL:= dem;
End;
```

e. Procedure XoaDL(Var First: TroDL; Tel: St6) xóa đại lý có số điện thoại Tel ra khỏi danh sách.

```
Procedure XoaDL(Var First: TroDL; Tel: St6);
Var
   p,q: TroDL;
Begin
   p:= first;
   while (p <> nil) and (p^.SoDT <> Tel) do
      begin
         q:= p;
         p:= p^.next;
```

```
      end;
   if p <> nil then
      begin
         if p = first then
            first:= first^.next
         else
            q^.next:= p^.next;
         dispose(p);
      end;
End;
```

BÀI TẬP TỰ GIẢI

Bài tập 9.9: Viết một hàm để xác định xem một danh sách liên kết đã cho có thứ tự tăng dần hay không theo 2 cách: Không đệ qui và đệ qui.

Bài tập 9.10: Cho 2 danh sách liên kết đơn đại diện cho 2 tập hợp được trỏ bởi L1 và L2. Viết chương trình để hiển thị:

1. Phần giao của 2 danh sách trên.
2. Phần hợp của 2 danh sách trên.
3. Phần hiệu của 2 danh sách trên.

Bài tập 9.11: Cho 2 danh sách liên kết L1 và L2.

1. Sắp xếp lại 2 danh sách đó theo thứ tự tăng dần.
2. Trộn 2 danh sách đó lại thành danh sách L3 sao cho L3 vẫn có thứ tự tăng dần.

Bài tập 9.12: Dùng danh sách móc nối để biểu diễn một đa thức $P_n(x) = a_n x^n + a_{n-1}x^{n-1} +...+ a_0$. Trong đó, mỗi số hạng của đa thức được xác định bởi 2 thành phần: hệ số a_i và số mũ i.

Như vậy, ta có thể xây dựng cấu trúc dữ liệu cho đa thức như sau:

```
TYPE   DATHUC = ^SOHANG;
       SOHANG = Record
          HeSo: Real;
          SoMu: Integer;
          Next: DATHUC;
       End;
```

Viết chương trình thực hiện các công việc sau:

1. Viết thủ tục nhập vào một đa thức.
2. Viết thủ tục để sắp xếp lại các số hạng của đa thức theo thứ tự số mũ giảm dần.
3. Viết thủ tục/hàm để cộng 2 đa thức.
4. Viết hàm để tính giá trị của đa thức theo giá trị X.

Bài tập 9.13: Cho một file văn bản trong đó có chứa các từ. Các dấu phân cách từ là: ký tự trắng, dấu chấm, dấu phẩy, dấu chấm phẩy, dấu hai chấm, dấu than, dấu hỏi. Mọi từ đều bắt đầu bằng một ký tự trong tập ['A'..'Z'].

1. Viết thủ tục cho phép đọc các từ trong file văn bản đã cho và lưu các từ đó vào mảng các danh sách móc nối:

```
TuDien : ARRAY['A'..'Z'] OF DanhSach;
```

Trong đó kiểu DanhSach được cho như sau:

```
TYPE DanhSach = RECORD
                  Tu : String[10];
                  Next : DanhSach;
                END;
```

Mỗi danh sách móc nối trong từ điển đều phải được sắp thứ tự (tăng dần), và các từ được lưu trong từ điển phải khác nhau.

2. Viết một thủ tục hiển thị tất cả các từ trong từ điển ra màn hình theo thứ tự tăng dần.

3. Viết một thủ tục bổ sung một từ mới vào từ điển bằng cách đọc từ đó từ bàn phím, tìm nó trong từ điển.

- Nếu tìm thấy, hiển thị thông báo:"Từ đã có trong từ điển".
- Nếu không tìm thấy, chèn từ đó vào trong từ điển ở vị trí thích hợp.

Bài tập 9.14: Cho dãy số nguyên sắp theo thứ tự tăng dần và lưu trong một danh sách liên kết đơn có địa chỉ nút đầu danh sách là First.

a. Viết chương trình xoá tất cả các nút trong danh sách có giá trị 0.

b. Viết chương trình in ra các giá trị phân biệt của danh sách.

Bài tập 9.15: Cho hai dãy số thực lưu trong hai danh sách liên kết đơn, có địa chỉ của các nút đầu danh sách lần lượt là First1 và First2. Giả sử trong mỗi danh sách giá trị các nút đã được sắp tăng dần. Hãy viết chương trình tạo một danh sách liên kết đơn có nút đầu trỏ bởi List, chứa tất cả các phần tử của hai danh sách trên, danh sách mới này cũng được sắp thứ tự.

Bài tập 9.16: Một công ty du lịch quản lý tất cả các xe ô tô của họ bằng một danh sách liên kết, mỗi nút của danh sách được khai báo như sau:

```
Type
   TroXe = ^Xe;
   St6 = String[6];
   St20 = String[20];
   Xe = Record
      TaiXe: St20; { họ tên tài xế }
      BienSo: St6; { biển số xe }
      Socho: Byte; { số chỗ ngồi }
      Tiep: TroXe;
       end;
Var
   First: TroXe;
```

a. Viết thủ tục Procedure Print(First: TroXe; n:byte); in họ tên tài xế, biển số xe của tất cả các xe có n chỗ ngồi được lưu trong danh sách.

b. Viết hàm Function SoChoNgoi(First: TroXe; Bso: St6); trả về số chỗ của xe có biển số Bso.

Bài tập 9.17: Người ta quản lý các sách trong thư viện bằng một danh sách liên kết, sắp theo thứ tự của mã sách. Mỗi đầu sách tương ứng với một nút trong danh sách có khai báo như sau:

```
Type
   TroSach = ^Sach;
   St4 = String[4];
   St20 = String[20];
   Sach = Record
          Ma: St4;
          Ten, Tacgia: St20;
          NamXb: word;
          Soluong: Byte;
          Next: TroSach;
       end;
Var
   First: TroSach;
```

Chú ý: Các đầu sách được săp theo thứ tự mã sách.

- **a.** Viết thủ tục Procedure CapNhat(Var First: TroSach; Bma:St4; Bten, BTgia: St20; Bnam: word; n: byte); bổ sung vào thư viện đầu sách có mã là Bma, tên sách Bten, tác giả BTgia và số lượng bổ sung là n theo yêu cầu: Nếu đầu sách có mã là Bma đã có trong thư viện thì chỉ tăng số lượng lên n, ngược lại thêm một đầu sách mới vào thư viện với số lượng n và bảo toàn thứ tự của mã sách.
- **b.** Viết thủ tục Procedure LietKeNam(First: TroSach; Nam: word) in danh sách các đầu sách xuất bản vào năm Nam.
- **c.** Viết hàm Function So_Dau_Sach(First: TroSach; BTgia: St20) trả về số đầu sách của tác giả BTgia.
- **d.** Viết thủ tục Procedure LietKeten(First: TroSach; Bten: St20) in danh sách tất cả các đầu sách có tên sách là Bten.

Bài tập 9.18: Một cửa hàng kinh doanh vật liệu xây dựng quản lý lượng hàng tồn kho bằng một danh sách liên kết. Mỗi loại vật liệu tương ứng với một nút trong danh sách và có khai báo như sau:

```
Type
   St3 = String[3];
   St10 = String[10];
   TroVT = ^Vattu;
   Vattu = Record
          Ma: St3;
          Ten: St10;
          DVTinh: St10;   { đơn vị tính}
          Soluong: word;
          Tiep: TroVattu;
       End;
```

Var
First: TroVattu;

a. Viết thủ tục Procedure XuatKho(Var First: TroVattu; Bma: St3; Bdonvi: St10; n: word); lấy ra khỏi kho n bdonvi loại vật tư có mã là Bma theo yêu cầu sau:
- Nếu vật tư có mã Bma không có trong kho thì thông báo kho không có loại vât tư này và kết thúc thực hiện.
- Ngược lại, kiểm tra Bdonvi có trùng với DVTinh của loại vật tư này không, nếu không trùng thì yêu cầu đổi theo đơn vị tính lưu trong danh sách trước khi thực hiện xuất kho.
- Nếu số lượng trong kho của vật tư có mã Bma nhỏ hơn n thì đưa ra thông báo và hỏi lại có muốn xuất không, nếu muốn thì xuất với số lượng bao nhiêu?
- Sau khi xuất n đơn vị loại vật tư theo yêu cầu, giảm số lượng vật tư trong kho cho phù hợp và nếu số lượng của vật tư này bằng 0 thì xoá nó ra khỏi danh sách.

b. Viết thủ tục Procedure NhapKho(Var First: TroVattu; Bma: St3; Bten, Bdv: St10; n: word); nhập vào kho n Bdv laọi vật tư có mã Bma theo yêu cầu:
- Nếu loại vật tư có mã Bma đã có trong kho thì chỉ tăng số lượng vật tư này trong kho, nhớ kiểm tra đơn vị tính như câu a.
- Ngược lại, bổ sung một loại vật tư mới vào kho với mã là Bma, tên vật tư là Bten, đơn vị tính là Bdv và số lương tương ứng là n.

c. Viết hàm Function SoVattu(First: TroVattu; Bma:St3); trả về số lượng vật tư có mã là Bma còn tồn trong kho, nếu không có vật tư này, hàm trả về giá trị 0.

d. Viết thủ tục Procedure ThongKe(First: TroVattu); in ra thông tin của tất cả các loại vật tư hiện có trong kho.

Bài tập 9.19: Cho danh sách các số nguyên được tổ chức như sau:

```
Type  TRO = ^PT;
   PT = Record
            X: Integer;
            Link: TRO;
        End;
```

1. Viết thủ tục **NHAP(Var Dau:TRO);** để nhập vào một danh sách các số nguyên có nút đầu tiên được trỏ bởi con trỏ Dau.
2. Viết thủ tục **LIETKE(Dau:TRO);** để in ra màn hình giá trị của tất cả các nút trong danh sách được trỏ bởi con trỏ Dau.
3. Giả sử Dau là con trỏ trỏ đến đầu của một danh sách chưa được sắp xếp, còn L được gán bằng NIL. Hãy viết thủ tục **SAPCHON(Var Dau,L:TRO);** cho phép chọn các phần tử trong danh sách Dau từ giá trị lớn đến bé, đưa vào danh sách L để cuối cùng Dau=Nil, còn L sẽ trở thành một danh sách đã được sắp xếp theo thứ tự tăng dần.
4. Viết hàm **TANG(Dau:TRO):Boolean;** để xác định xem danh sách được trỏ bởi Dau đã có thứ tự tăng dần hay không theo 2 cách: Không đệ qui và đệ qui.
5. Viết thủ tục **DAO(Var Dau:TRO);** để đảo ngược các con trỏ của danh sách Dau.

Bài tập 9.20: Cho danh sách các số nguyên được tổ chức như sau:

```
Type  TRO = ^PT;
   PT = Record
            Gtri: Integer;
            Tiep: TRO;
        End;
```

1. Viết hàm **DXUNG(ds:TRO):Boolean;** nhằm kiểm tra các giá trị trong danh dách ds có phải là đối xứng hay không theo 2 cách: Đệ qui và không đệ qui.(Ví dụ: danh sách chứa các dãy số sau đây là đối xứng: 2 3 6 3 2; 5 7 7 5).
2. Viết hàm **KhaDX(ds:TRO):Boolean;** để kiểm tra xem với danh sách ds đã cho, có tồn tại một cách sắp xếp lại các phần tử để cuối cùng nhận được một danh sách đối xứng hay không?
3. Viết thủ tục **SAPLAI(Var ds:TRO);** để đưa ra một cách sắp xếp lại các phần tử để có một danh sách đối xứng (giả thiết điều này có thể làm được nhờ hàm **KhaDX**).

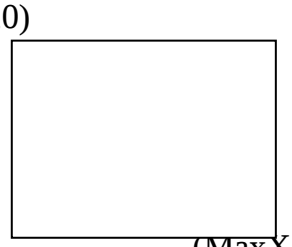
Chương 10
ĐỒ HỌA

I. MÀN HÌNH TRONG CHẾ ĐỘ ĐỒ HỌA (GRAPHIC)

Hình ảnh trong chế độ đồ họa được tạo ra bằng các điểm ảnh (Pixel), số điểm ảnh của màn hình đồ họa tùy thuộc vào từng loại CARD màn hình và MODE qui định cho màn hình đó.

Việc lập trình trong chế độ đồ họa cần phải xác định được loại màn hình đang sử dụng và chương trình phải vận hành được trên nhiều loại màn hình khác nhau.

Tọa độ của một điểm ảnh trên màn hình đồ họa cũng giống như trong chế độ văn bản (TEXT) với điểm ảnh đầu tiên trên góc trái màn hình là (0,0), tọa độ đỉnh dưới phải tùy thuộc vào độ phân giải của màn hình, CARD màn hình và MODE màn hình.



Để dử dụng được chế độ đồ họa trên màn hình, ta cần phải có các File sau:

GRAPH.TPU Chứa các lệnh đồ họa
* .BGI Chứa Font màn hình
* .CHR Chứa Font ký tư

II. KHỞI TẠO VÀ THOÁT KHỎI CHẾ ĐỘ ĐỒ HỌA

2.1. Khởi tạo chế độ đồ họa

Thủ tục **INITGRAPH**(Gd,Gm:Integer; Path:String);

trong đó:

- Gd: Chỉ CARD màn hình.

Thông thường, một chương trình phải được chạy trên nhiều loại màn hình khác nhau nên ta có thể khai báo:

Gd = Detect (= 0)

Với hằng Detect, máy sẽ tự động tìm CARD màn hình tương ứng để chạy chương trình.

- Gm: Chỉ MODE màn hình.

Trong trường hợp khai báo Gd = Detect thì không cần thiết phải khai báo Gm vì máy tính sẽ tự xác định loại CARD màn hình và thiết lập chế độ MODE màn hình tương ứng với CARD màn hình đó.

- Path: Đường dẫn đến nơi chứa các file *.BGI. Nếu Path = ‘’ thì ta hiểu là các file *.BGI nằm trong thư mục hiện hành.

Hàm **GRAPHRESULT**:Integer;

Hàm này trả về kết quả của việc khởi động đồ họa.

= 0 : Thành công.

<>0 : Bị lỗi.

Tên của lỗi được xác định bởi hàm **GRAPHERRORMSG**(Er:Integer):String;

Hàm này cho ra một xâu ký tự thông báo lỗi của đồ họa xác định bởi đối số Er.

* Hằng số GrOK = 0: Việc khởi động đồ họa có lỗi.

Ví dụ:

```
Uses Graph;
Procedure ThietLapDoHoa;
 var gd,gm,Gr:integer;
 Begin
   DetectGraph(Gd,Gm);
   InitGraph(gd,gm,'C:\TP\BGI');
   Gr:=GraphResult;
   If Gr<>GrOK then
     Begin
       writeln('Loi Do hoa: ',GraphErrorMsg(Gr));
       Halt(1);
     End;
 End;
BEGIN
   ThietLapDoHoa;
   . . .
END.
```

Chú ý: Ta có thể khởi tạo mode đồ hoạ với chế độ 256 màu bằng cách sử dụng hàm **InstallUserDriver**(Name:String;Ptr:Pointer):Integer; với điều kiện trên đĩa phải có file **SVGA256.BGI**.

```
Procedure ThietLapDoHoa;
 var gd,gm,Gr:integer;
 Begin
   Gd:= InstallUserDriver('SVGA256',NIL);
   Gm:=2; {Mode 640x480x256}
   InitGraph(gd,gm,'C:\TP\BGI');
 End;
```

2.2. Thoát khỏi chế độ đồ họa

Thủ tục **CLOSEGRAPH;**

Sau đây là cấu trúc chung của một chương trình đồ họa:.

```
Uses Crt,Graph;
Procedure ThietLapDoHoa;
 var gd,gm,Gr:integer;
 Begin
   DetectGraph(Gd,Gm);
   InitGraph(gd,gm,'C:\TP\BGI');
   Gr:=GraphResult;
   If Gr<>GrOK then
```

```
    Begin
      writeln('Loi Do hoa: ',GraphErrorMsg(Gr));
      Halt(1);
    End;
 End;
BEGIN
  ThietLapDoHoa;
  . . .
  CloseGraph;
END.
```

III. TỌA ĐỘ VÀ CON TRỎ TRÊN MÀN HÌNH ĐỒ HỌA

3.1. Hàm GetMaxX:Integer;

Cho tọa độ cột lớn nhất của màn hình.

3.2. Hàm GetMaxY:Integer;

Cho tọa độ dòng lớn nhất của màn hình.

3.3. Thủ tục MOVETO(x,y:Integer);

Di chuyển con trỏ từ vị trí đang đứng đến tọa độ (x,y).

3.4. Thủ tục MOVEREL(dx,dy:Integer);

Di chuyển con trỏ từ vị trí đang đứng đến tọa độ mới cách tọa độ cũ khoảng cách là dx, dy.

3.5. Vẽ một điểm trên màn hình:

Dùng thủ tục PUTPIXEL(x,y:Integer; color:Word);

3.6. Lấy màu của một điểm tại tọa độ x,y:

Hàm GETPIXEL(x,y:Integer):Word;

IV. ĐẶT MÀU TRÊN MÀN HÌNH ĐỒ HỌA

4.1. Đặt màu cho đối tượng cần vẽ

Dùng thủ tục SETCOLOR(Color:Byte);

4.2. Đặt màu nền

Dùng thủ tục SETBKCOLOR(Color:Byte);

V. CỬA SỔ TRONG CHẾ ĐỘ ĐỒ HỌA

5.1. Đặt cửa sổ trên màn hình

Thủ tục SETVIEWPORT(x1,y1,x2,y2:Integer; Clip:Boolean);

Với x1,y1: đỉnh trên trái của cửa sổ.

x2,y2: đỉnh dưới phải của cửa sổ.

Clip = TRUE: những gì vượt khỏi màn hình sẽ bị cắt bỏ.

Clip = FALSE: những gì vượt khỏi màn hình sẽ không bị cắt bỏ.

* Khi tạo cửa sổ thì tọa độ trên màn hình sẽ thay đổi theo.

Tọa độ mới = Tọa độ cũ - Tọa độ đỉnh trên trái.

5.2. Xóa hình ảnh trong cửa sổ

- Xóa hình ảnh trong cửa sổ, ta dùng thủ tục CLEARVIEWPORT;
- Xóa toàn bộ màn hình, ta dùng thủ tục CLEARDEVICE;

VI. VIẾT CHỮ TRONG CHẾ ĐỘ ĐỒ HỌA

6.1. Chọn Font chữ

Ta dùng thủ tục SETTEXTSTYLE(font,Dir,size:Word);

- Các font có thể chứa các hằng sau:
  DefaultFont = 0;TriplexFont = 1;SmallFont = 2;
  SansSerifFont = 3; GothicFont = 4;
- Dir có các hằng sau:
  HorizDir = 0 Từ trái qua phải.
  VetDir = 1 Từ dưới lên trên.
- Size: độ lớn của chữ.

6.2. Chọn phân bố chữ

Dùng thủ tục SETTEXTJUSTIFY(Hz,Vt:Word);

Chọn vị trí của chữ xung quanh tọa độ định sẵn.

- Hz là phân bố chữ theo trục ngang. Có các hằng sau:
  LeftText = 0 Chữ viết nằm bên phải trục đứng.
  CenterText = 1 Chữ viết nằm ở giữa trục đứng.
  RightText = 2 Chữ viết nằm bên trái trục đứng.
- Vt là bố trí chữ theo hướng dọc đối với tọa độ qui định xuất chuỗi. Các hằng liên quan:
  BottomText = 0 Chữ viết nằm bên trên trục ngang.
  CenterText = 1 Chữ viết nằm ở giữa trục ngang.
  TopText = 2 Chữ viết nằm bên dưới trục ngang.

6.3. Viết một xâu ký tự lên màn hình

- Xuất một xâu ký tự tại vị trí con trỏ:
  Dùng thủ tục OUTTEXT(St:String);
- Xuất một xâu ký tự tại tọa độ x,y:
  Dùng thủ tục OUTTEXTXY(x,y:Word; St:String);

Chú ý: Cách xuất chuỗi của hai thủ tục trên được qui định trong thủ tục SETTEXTJUSTIFY và SETTEXTSTYLE.

VII. VẼ CÁC HÌNH CƠ BẢN

7.1. Chọn kiểu đường

Dùng thủ tục SETLINESTYLE(Ls,Pt,Tk:Word);

Thủ tục này xác định kiểu đường được vẽ trong đồ họa.

Ls: kiểu đường vẽ. Ls có các giá trị sau:

0: Đường liền nét
1: Nét đứt
2: Nét chấm gạch

3: Nét gạch
4: Đường do người thiết kế tạo ra.

Pt: xác định màu vẽ.
. Nếu Ls = 0..3 thì Pt=0 (Lấy giá trị Default)
. Nếu Ls = 4 thì Pt là số nguyên chỉ màu của kiểu đường.

Tk: xác định độ dày của đường.
Tk = 1: bình thường.
Tk = 3: đậm nét.

7.2. Vẽ đoạn thẳng

LINE(x1,y1,x2,y2:Integer); vẽ từ điểm (x1,y1) đến điểm (x2,y2)
LINETO(x,y:Integer); vẽ từ vị trí con trỏ đến điểm (x,y)
LINEREL(dx,dy:Integer); vẽ từ vị trí con trỏ đến điểm cách đó một khoảng dx,dy.

7.3. Vẽ hình chữ nhật

Dùng thủ tục RECTANGLE(x1,y1,x2,y2:Integer);

7.4. Vẽ cung tròn

Thủ tục ARC(x,y:Integer; g1,g2,R:Word);
Vẽ cung tròn có tâm (x,y) bán kính R, góc bắt đầu là g1 và góc kết thúc là g2.

7.5. Vẽ đường tròn - Ellip

Thủ tục vẽ đường tròn: CIRCLE(x,y:Integer; R:Word);
Thủ tục ELLIPSE(x,y:integer; g1,g2,Rx,Ry:Word);
Vẽ Ellip có tâm (x,y) bán kính ngang Rx, bán kính dọc Ry, góc bắt đầu là g1 và góc kết thúc là g2.

7.6. Định MODE đường vẽ

Thủ tục SETWRITEMODE(Mode:Integer);
- Định Mode vẽ cho các đường thẳng.
- Ta có thể chọn Mode bằng các hằng:
CopyPut = 0; XORPut = 1;

Trong đó:
. CopyPut là Mode chèn, đường mới sẽ không xóa đường cũ.
. XORPut là Mode xóa, đường mới sẽ xóa đường cũ.

XIII. TÔ MÀU CÁC HÌNH

8.1. Chọn kiểu tô

Thủ tục SETFILLSTYLE(Pt,Cl:Word);
Với:
- Pt: Mẫu tô của hình. Có các hằng từ 0 đến 12.
0: Tô bằng màu nền.
1: Tô bằng màu viền.
2: Tô bằng các dấu ---
.....
- Cl: Màu tô của hình.

8.2. Vẽ hình chữ nhật có tô màu ở bên trong

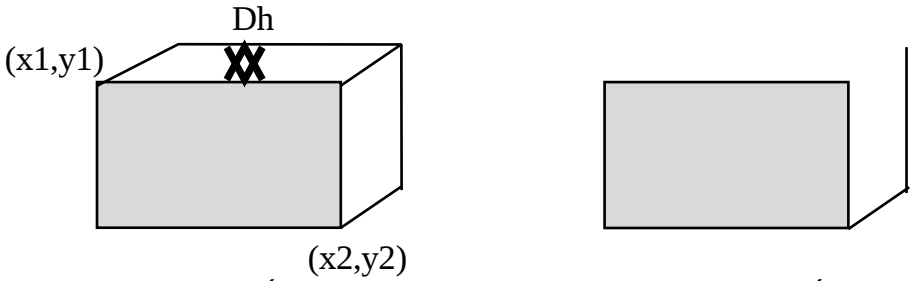
Thủ tục BAR(x1,y1,x2,y2:Integer);

Vẽ hình chữ nhật có tô màu và mẫu tô được xác định bởi thủ tục SETFILLSTYLE.

8.3. Vẽ hình hộp chữ nhật

Thủ tục BAR3D(x1,y1,x2,y2,Dh:Word; Top:Boolean);

Vẽ hình hộp chữ nhật có tọa độ đỉnh trên là (x1,y1), đỉnh dưới là (x2,y2) và chiều dày là Dh.



Có nắp Không có nắp

Top = TRUE: Hình hộp có nắp.

Top = FALSE: Hình hộp không có nắp.

8.4.Vẽ hình Ellip

Thủ tục FILLELLIPSE(x,y:Integer; Rx,Ry:Word);

8.5. Vẽ hình quạt tròn

Thủ tục PIESLICE(x,y:Integer; g1,g2,R:Word);

Vẽ hình quạt tròn có tâm (x,y), góc đầu g1, góc cuối g2, bán kính R.

8.6. Vẽ hình quạt Ellip

thủ tục SECTOR(x,y:Integer; g1,g2,Rx,Ry:Word);

8.7. Làm loang màu một vùng kín

Thủ tục FLOODFILL(x,y:Integer; Color:Word);

Trong đó:

(x,y): điểm nằm trong vùng kín.

Color: màu muốn tô.

8.8. Vẽ đa giác

Đối với một đa giác bất kỳ có N đỉnh, ta phải khai báo N+1 đỉnh để vẽ đường gấp khúc với tọa độ điểm đầu trùng với tọa độ điểm cuối.

Để vẽ đa giác ta dùng thủ tục: DRAWPOLY(Np:Word; Var P);

trong đó:

- Np: số đỉnh của đa giác + 1
- P: chứa tọa độ các đỉnh, là một mảng có Np thành phần có kiểu dữ liệu là PointType được định nghĩa trong Unit Graph như sau:

```
TYPE    PointType = Record
                      x,y: Integer;
                    End;
```

IX. CÁC KỸ THUẬT TẠO HÌNH CHUYỂN ĐỘNG

9.1. Kỹ thuật lật trang màn hình

CARD màn hình có nhiều trang, mỗi trang được đánh số 0,1,2,...

Để vẽ hình lên một trang màn hình, ta dùng thủ tục:

SETACTIVEPAGE(Page:Word);

Trong đó, Page là số của trang màn hình. Thủ tục này được đặt trước khi có lệnh vẽ lên màn hình.

Để đưa trang màn hình ra màn hình, ta dùng thủ tục:

SETVISUALPAGE(Page:Word);

Page: trang màn hình muốn xem.

Thông thường, màn hình sẽ làm việc và hiện ra trên trang 0. Do đó, để vừa xem màn hình vừa vẽ lên trang màn hình khác, ta thường dùng hai thủ tục trên đi kèm với nhau.

Để thực hiện tự động chương trình khi sử dụng cú pháp lật hình này, ta thường theo một giải thuật sau:

- Tạo biến page1,page2:Word;
- Tạo vòng lặp

```
...
Repeat
   SetVisualPage(page1); (* Xem trang màn hình page1 *)
   SetActivePage(page2); (* Vẽ hình lên trang page2 *)
   .......
   < Các thủ tục vẽ hình >
   .......
   (* Hoán vị 2 biến page1, page2 *)
   Temp:=page1;
   page1:=page2;
   page2:=Temp;
Until <Điều kiện thoát>;
```

9.2. Lưu và di chuyển một vùng màn hình

Chúng ta có thể lưu một vùng màn hình vào bộ nhớ rồi sau đó lại dán nó lên màn hình tại một vị trí khác.

Lưu một vùng màn hình vào bộ nhớ được thực hiện bằng thủ tục:

GETIMAGE(x1,y1,x2,y2:Integer; Var P:Pointer);

trong đó P là con trỏ để lưu nội dung của vùng (x1,y1,x2,y2).

Việc đăng ký một vùng nhớ động phải được khai báo dung lượng cần thiết. Dung lượng vùng nhớ được thực hiện bằng hàm:

IMAGESIZE(x1,y1,x2,y2:Integer):Word;

Để hiện hình ảnh từ bộ nhớ ra màn hình, ta dùng thủ tục:

PUTIMAGE(x,y:Integer; P:Pointer; Mode:Word);

trong đó:

(x,y): Tọa độ đỉnh trái hình chữ nhật mà ta muốn đưa ra.

P : Con trỏ lưu vùng hình chữ nhật.

Mode: Hằng số chỉ phương thức hiện ra màn hình. Mode chứa một trong các hằng sau:

NormalPut = 0: Xuất ra như đã lưu (phép MOV)
XORPut = 1: Phép XOR, xóa hình cũ nếu hai hình giao nhau.
ORPut = 2: Phép OR, lấy cả hai hình nếu hai hình giao nhau.
ANDPut = 3: Phép AND, nếu hai hình giao nhau thì lấy phần chung.
NOTPut = 4: Phép NOT, cho ra âm bản.

Về việc thực hiện ta tiến hành như sau:

- Khai báo một biến con trỏ P:Pointer;
- Đăng ký một vùng nhớ động do P qủan lý bằng thủ tục GETMEM(P,ImageSize(x1,y1,x2,y2));
- Lưu hình ảnh bằng thủ tục GETIMAGE(x1,y1,x2,y2,P^);
- Xuất ra màn hình bằng thủ tục PUTIMAGE(x,y,P^,Mode);

BÀI TẬP MẪU

Bài tập 10.1: Viết dòng chữ có bóng trong chế độ 256 màu.

```
Uses crt,Graph;
var gd,gm:integer;

Procedure WriteStr(dx,dy:Integer;st:String);
 Var i,j:Integer;
 Begin
  settextstyle(5,0,8);
  j:=16;
  (* Viet chu bong *)
  for i:=0 to 15 do
    begin
      setcolor(j);
      outtextxy(dx+i,dy+i,st);
      inc(j);
    end;
  setcolor(40);
  outtextxy(dx+i,dy+i,st);
 End;

Begin
 gd:=INSTALLUSERDRIVER('SVGA256',NIL);
 GM:=4;
 initgraph(gd,gm,'c:\bp\BGI');
 WriteStr(1,100,' Pham Anh Phuong');
 readln;
 CloseGraph;
```

```
End.
```

Bài tập 10.2: Vẽ các hình chữ nhật ngẫu nhiên trên màn hình.

```
Uses Crt,Graph;

Procedure ThietLapDohoa;
Var Gd,Gm:Integer;
Begin
   Gd:=0;
   InitGraph(Gd,Gm,'D:\BP\BGI');
End;

Function RandColor:Byte;
  Begin
    RandColor:=Random(MaxColors - 1)+1;
  End;

Procedure DeMo;
   Var  x1,y1,x2,y2:Integer;
   Begin
      Randomize;
      Repeat
    x1:=Random(GetMaxX);
    y1:=Random(GetMaxY);
    x2:=Random(GetMaxX - x1) + x1;
    y2:=Random(GetMaxX - y1) + y1;
    SetColor(RandColor);
    Rectangle(x1,y1,x2,y2);
    Delay(500);
     Until KeyPressed;
   End;

BEGIN
  ThietLapDohoa;
  DeMo;
  CloseGraph;
END.
```

Bài tập 10.3: Vẽ một kim đồng hồ quay quanh tâm O(x0,y0).

```
Uses crt,Graph;
Var x0,y0:word;
     Alpha,Beta,R:real;

Procedure VeDgt(x0,y0:word; R,Alpha:real);
```

```
 Begin
   Line(x0,y0,x0+Round(R*Cos(Pi*Alpha/180)),
                y0-Round(R*Sin(Pi*Alpha/180)));
 End;

BEGIN
 ThietLapDoHoa;
 x0:=GetMaxX div 2;
 y0:=GetMaxY div 2;
 R:=100;
 Alpha:=90;
 Beta:=6;
 SetWriteMode(XORPut);
 VeDgt(x0,y0,R,Alpha);
 Repeat
 VeDgt(x0,y0,R,Alpha);
  Alpha:=Alpha - Beta;
  VeDgt(x0,y0,R,Alpha);
  Delay(250);
 Until KeyPressed;
 CloseGraph;
END.
```

Bài tập 10.4: Viết chương trình tạo Menu cho phép chọn và thực hiện các chức năng bằng cách di chuyển mũi tên trên các hộp sáng, các thủ tục thực hiện xong quay trở lại Menu chính. Nhấn ESC để thoát khỏi chương trình.

```
USES  crt,graph;
Const  mau1 =15;
       mau2 =8;
       maumn=7;
       XTop=200;
       YTop=100;
       Dy=32;
       Dx=250;
Type MANG_MENU=Array[1..20] of string;{dung luu cac dong menu }
    MANG_THUTUC=Array[1..20] of Procedure;{dung luu cac thu tuc}
var  DongMN:MANG_MENU;
    ThuTuc:MANG_THUTUC;
    SoDong:byte;

Procedure Wait;
Var ch:Char;
BEGIN
   ch:=ReadKey;
END;
```

```
{$F+}
Procedure Modun1;
BEGIN
   Line(50,50,200,300);
   Wait;
END;

Procedure Modun2;
BEGIN
   Circle(200,200,100);
   Wait;
END;

Procedure Modun3;
Begin
   Ellipse(200,300,0,360,100,150);
   Wait;
End;

Procedure Modun4;
BEGIN
   Rectangle(50,50,200,300);
   Wait;
END;

Procedure Modun5;
BEGIN
   OutTextXY(50,50,'Chao mung cac ban den voi chuong trinh do hoa');
   Wait;
END;

Procedure Modun6;
BEGIN
   OutTextXY(50,50,'Day la Menu do hoa');
   Wait;
END;

Procedure Thoat;
BEGIN
 Halt;
END;
{$F-}

Procedure ThietLapDoHoa;
 var gd,gm:integer;
```

```
Begin
  Gd:=0;
  InitGraph(gd,gm,'C:\BP\BGI');
End;

Procedure Box(x1,y1,x2,y2:integer; MauVienTren,MauVienduoi,MauNen:byte);
{Ve nut menu}
  Var i:Byte;
begin
  setfillstyle(1,MauNen);
  bar(x1,y1,x2,y2);
  setcolor(MauVienTren);
  For i:=0 to 1 do
    Begin
      moveto(x1-i,y2+i);
      lineto(x1-i,y1-i);
      lineto(x2+i,y1-i);
    End;
 setcolor(MauVienDuoi);
 For i:=0 to 1 do
   Begin
     moveto(x2+i,y1-i);
     lineto(x2+i,y2+i);
     lineto(x1-i,y2+i);
   End;
end;

Procedure Ve_menu(Xdau,Ydau,DeltaX,DeltaY:Word;
                  chon,SoDong:Byte;DongMN:MANG_MENU);
 Var i:Byte;
 Begin
   for i:=1 to SoDong do
     begin
       if i=chon then
           Box(Xdau,Ydau+i*DeltaY+6,Xdau+DeltaX,YDau+i*DeltaY+DeltaY,
                 mau2,mau1,maumn)
       Else
           Box(Xdau,Ydau+i*DeltaY+6,Xdau+DeltaX,YDau+i*DeltaY+DeltaY,
                 mau1,mau2,maumn);
       OutTextxy(Xdau+20,Ydau+15+i*DeltaY,DongMN[i]);
     end;
 End;

Procedure PullDown(x,y,DeltaX,DeltaY:Word;SoDong:Byte;
                   DongMenu:MANG_MENU;ThuTuc:MANG_THUTUC);
 Var sott,LuuSott,Chon,i:Byte;
```

```
    OK:Boolean;

Function Select(Xdau,Ydau,DeltaX,DeltaY:Word;SoDong:Byte):Byte;
  var ch:char; j:Byte;
  Begin
   While True do
    Begin
       If KeyPressed then
       Begin
       ch:=readkey;
       case ch of
          #13: Begin {ENTER}
                select:=Sott;
                Exit;
               End;
          #72:Begin
               LuuSott:=Sott;
               Sott:=Sott-1;
               if Sott<1 then Sott:=SoDong;
               Select:=Sott;
              Box(XTop,YTop+LuuSoTT*DeltaY+6,
                    Xdau+DeltaX,Ydau+LuuSoTT*DeltaY+DeltaY,
                    Mau1,Mau2,maumn);
               Outtextxy(Xdau+20,Ydau+15+LuuSoTT*DeltaY,
                         DongMN[LuuSoTT]);
               Box(Xdau,Ydau+SoTT*DeltaY+6,
                    Xdau+DeltaX,Ydau+SoTT*DeltaY+DeltaY,
                    Mau2,Mau1,maumn);
               Outtextxy(Xdau+20,Ydau+15+SoTT*DeltaY,
                         DongMN[SoTT]);
              End;
          #80:
              Begin
               LuuSott:=Sott;
               Sott:=Sott+1;
               if Sott>SoDong then Sott:=1;
               Select:=Sott;
               Box(Xdau,Ydau+LuuSoTT*DeltaY+6,
                    Xdau+DeltaX,Ydau+LuuSoTT*DeltaY+DeltaY,
                    Mau1,Mau2,maumn);
               Outtextxy(Xdau+20,Ydau+15+LuuSoTT*DeltaY,
                          DongMN[LuuSoTT]);
               Box(Xdau,Ydau+SoTT*DeltaY+6,
                    Xdau+DeltaX,Ydau+SoTT*DeltaY+DeltaY,
                    Mau2,Mau1,maumn);
               Outtextxy(Xdau+20,Ydau+15+SoTT*DeltaY,
```

```
                        DongMN[SoTT]);
            End;
          #27: {ESC}
            Begin
              OK:=False; Exit;
            End;
       end; { of case key }
      End;
    End;
 End;

Begin {PullDown}
  Sott:=1;   OK:=TRUE;
  Ve_menu(X,Y,DeltaX,DeltaY,Sott,SoDong,DongMenu);
  While OK do  { lap khong dieu kien }
    Begin
      Chon:=select(x,y,DeltaX,DeltaY,SoDong);
      For i:=1 to SoDong do
        If (i=Chon)and OK Then
          Begin
            ClearDevice;
            ThuTuc[i];
            ClearDevice;

            Ve_Menu(X,Y,DeltaX,DeltaY,Sott,SoDong,DongMenu);
          End;
    end;{ of While }
End;

BEGIN
  SoDong:=7;
  DongMN[1]:='VE DOAN THANG ';
  DongMN[2]:='VE DUONG TRON';
  DongMN[3]:='VE ELLIPSE';
  DongMN[4]:='VE HINH CHU NHAT';
  DongMN[5]:='VIET LOI CHAO';
  DongMN[6]:='VIET DONG QUANG CAO';
  DongMN[7]:='THOAT KHOI CHUONG TRINH';

  ThuTuc[1]:=Modun1;
  ThuTuc[2]:=Modun2;
  ThuTuc[3]:=Modun3;
  ThuTuc[4]:=Modun4;
  ThuTuc[5]:=Modun5;
  ThuTuc[6]:=Modun6;
  ThuTuc[7]:=Thoat;
```

```
   ThietLapDoHoa;
   SetBKcolor(LightBlue);
   PullDown(XTop,YTop,DX,DY,SoDong,DongMN,ThuTuc);
  CloseGraph;
END.
```

Bài tập 10.5: Vẽ hai hình

Sau đó, viết chương trình thực hiện chuyển động của miệng cá.

```
Uses Crt,Graph;
Type ProType=Procedure;
Var Gd,Gm:integer;
    page1,page2:word;
    Hinh:Array[0..1] of ProType;
    Xc,Yc,r:Integer;
    i:Byte;

{$F+}
Procedure HinhCa1;
Begin
   SetColor(15);
   PieSlice(Xc,Yc,30,330,R); {Ve bung ca}
   SetColor(0);
   Circle(Xc + R div 2,Yc - R div 2,4); {Mat ca}
 End;

Procedure HinhCa2;
 Begin
   SetColor(15);
   PieSlice(Xc,Yc,15,345,R); {Ve bung ca}
   SetColor(0);
   Circle(Xc + R div 2 ,Yc - R div 2,4); {Mat ca}
 End;
{$F-}

Begin
  gd:=4;
  InitGraph(gd,gm,'');
  Xc:=GetMaxX div 2;
  Yc:=GetMaxY div 2;
  R:=50; i:=0;
```

```
  Hinh[0]:=HinhCa1;
  Hinh[1]:=HinhCa2;
  page1:=0; page2:=1;
  Repeat
    SetVisualPage(page1);
    SetActivePage(page2);
    i:=1-i;
    Hinh[i]; Delay(200);
    page1:=1-page1;
    page2:=1-page2;
  Until  KeyPressed;
  CloseGraph;
End.
```

Bài tập 10.6: Viết chương trình tạo một dòng chữ chạy ngang qua màn hình.

```
Uses crt,graph;
Var gd,gm:integer;

Procedure Run(s:string);
  var page:byte;x,y:integer;
  Begin
    page:=1;
    x:=getmaxx;y:=getmaxy div 3;
    Settextjustify(0,1);
    Setwritemode(xorput);
    Setactivepage(page);
    Repeat
      Outtextxy(x,y,s);
      Setvisualpage(page);
      page:=not page;
      setactivepage(page);
      delay(10);
      Outtextxy(x+1,y,s);
      x:=x-1;
      if x<-textwidth(s) then x:=getmaxx;
    Until (keypressed) and (readkey=#27);
  end;

Begin
     gd:=4;
     Initgraph(gd,gm,'C:\BP\bgi');
     setcolor(14);
     settextstyle(1,0,5);
     Run('Pham Anh Phuong');
     Closegraph;
End.
```

Bài tập 10.7: Viết chương trình vẽ mô hình chiếc đĩa bay chuyển động ngẫu nhiên trên màn hình.

```
Uses crt; Graph;
Const r = 20; StartX = 100; StartY = 50;

Procedure ThietLapDohoa;
Var Gd,Gm:Integer;
Begin
   Gd:=0;
   InitGraph(Gd,Gm,'D:\BP\BGI');
End;

Procedure Move(Var x,y:Integer);
   Var Step:Integer;
   Begin
      Step:=Random(2*r);
      If Odd(Step) Then Step:=-Step;
      x:=x+Step;
      Step:=Random(r);
      If Odd(Step) Then Step:=-Step;
      y:=y+Step;
   End;

Procedure VeDiaBay;
   Begin
      Ellipse(StartX,StartY,0,360,r,(r div 3)+2);
      Ellipse(StartX,StartY-4,190,357,r,r div 3);
      Line(StartX+7,StartY-6,StartX+10,StartY-12);
      Line(StartX-7,StartY-6,StartX-10,StartY-12);
      Circle(StartX+10,StartY-12,2);
      Circle(StartX-10,StartY-12,2);
   End;

Procedure Play;
   Var x1,y1,x2,y2,size:Word;
        x,y:Integer;
        P:Pointer;
   Begin
      VeDiaBay;
      x1:=StartX - (r+1);
      y1:=StartY - 14;
      x2:=StartX + r + 1;
      y2:=StartY + (r div 3) + 3;
      (* Lưu và xóa ảnh *)
      size:=ImageSise(x1,y1,x2,y2);
      GetMem(p,size);
```

```
      GetImage(x1,y1,x2,y2,P^);
      PutImage(x,y,P^,XORPut); { Xóa ảnh }
      x:=GetMaxX div 2;
      y:=GetMaxY div 2;
      (* Di chuyển đĩa bay *)
      Repeat
         PutImage(x,y,P^,XORPut); { Vẽ đĩa bay }
         Delay(200);
         PutImage(x,y,P^,XORPut); { Xóa đĩa bay }
         Move(x,y);
      Until KeyPressed;
      FreeMem(p,size);  { Giải phóng vùng nhớ }
   End;
BEGIN
   ThietLapDoHoa;
   Play;
   CloseGraph;
END.
```

Bài tập 10.8: Viết chương trình để vẽ đa giác đều có n đỉnh.

Ý tưởng:

Khi vẽ một đa giác đều N đỉnh, các đỉnh này nằm trên một đường tròn (O,R) đồng thời khoảng cách giữa hai đỉnh và tâm tạo thành một góc nhọn không đổi có giá trị là 2*Pi/N.

Giả sử đỉnh thứ nhất của đa giác nằm trên đường thẳng tạo với tâm một góc 00, đỉnh thứ hai tạo một góc 2*Pi/N và đỉnh thứ i sẽ tạo một góc là 2*Pi(i-1)/N.

Một cách tổng quát, ta tạo một mảng để chứa tọa độ các đỉnh.

```
Const       Max = <Giá trị>;
Type     Mang = ARRAY[1..Max] of PointType;
Var      P:Mang;
```

Giả sử chọn P0: PointType là tọa độ tâm của đa giác thì đỉnh thứ i của đa giác sẽ tạo một góc là: Angle:=2*Pi*(i-1)/N

Nhưng nếu đa giác này có đỉnh đầu tiên tạo một góc bằng A0 thì:

```
Angle:=2*Pi*((i-1)/N + A0/360)
```

Và tọa độ các đỉnh này trên màn hình là:

```
P[i].x := P0.x + R*cos(Angle)
P[i].y := P0.y  - R*sin(Angle)
```

Ta xây dựng thủ tục để tự động lưu các đỉnh của đa giác đều vào mảng P. Trong đó: P0 là tọa độ tâm, A0 là góc bắt đầu, R là bán kính, N là số đỉnh (3<N<Max).

```
Uses Crt,Graph;
Const Max = 10;
Type Mang = Array[1..Max] of PointType;
Var  A0,R:real;
     N:Byte;
```

```
      P0:PointType; P:Mang;

Procedure ThietLapDohoa;
Var Gd,Gm:Integer;
Begin
   Gd:=0;
   InitGraph(Gd,Gm,'D:\BP\BGI');
End;

Procedure TaoDinh(R,A0:real;N:Byte;P0:PointType;Var P:MANG);
 var  i:Byte;
       Angle:real;
 Begin
   If (n<3)or(n>=Max) then
    Begin
      Writeln('Khong tao duoc tap dinh!');
      Exit;
    End;
   For i:=1 to n do
     With P[i] do
      Begin
        Angle:=2*Pi*((i-1)/n + A0/360);
        x:=P0.x + Round(R*Cos(Angle));
        y:=P0.y - Round(R*Sin(Angle));
      End;
   P[n+1]:=p[1];
 End;

BEGIN
   Write('Nhap so dinh cua da giac deu: n= '); Readln(N);
   ThietLapDoHoa;
   P0.x:=GetMaxX div 2;
   P0.y:=GetMaxY div 2;
   A0:=90;
   R:=GetMaxY div 4;
   TaoDinh(R,A0,5,P0,P);
   DrawPoly(5,P);
   CloseGraph;
END.
```

Bài tập 10.9: Viết chương trình vẽ đồ thị hàm số sau: $f(x) = ax^2 + bx + c$.

Ý tưởng:

Bước 1: Xác định đoạn cần vẽ [Min,Max].

Bước 2: Đặt gốc tọa độ lên màn hình (x0,y0).

Chia tỉ lệ vẽ trên màn hình theo hệ sô k.

Chọn số gia dx trên đoạn cần vẽ.

Bước 3: Chọn điểm xuất phát: x = Min, tính f(x).
Đổi qua tọa độ màn hình và làm tròn:
x1:=x0 + Round(x.k);
y1:=y0 - Round(y.k);
Di chuyển đến (x1,y1): MOVETO(x1,y1);
Bước 4: Tăng x lên: x:=x + dx;
Đổi qua tọa độ màn hình và làm tròn:
x2:=x0 + Round(x.k);
y2:=y0 - Round(y.k);
Vẽ đến (x2,y2): LINETO(x2,y2);
Bước 5: Lặp lại bước 4 cho đến khi x > Max thì dừng.

```
Uses Crt,Graph;
 var a,b,c,Max,Min:real;

Procedure ThietLapDohoa;
Var Gd,Gm:Integer;
Begin
   Gd:=0;
   InitGraph(Gd,Gm,'D:\BP\BGI');
End;

Function F(x:real):real;
 Begin
   F:=a*x*x + b*x + c;
 End;

Procedure VeDoThi(Min,Max:real);
 var x1,y1:integer;
     dx,x,k:real;
     x0,y0:word;
 Begin
   x0:=GetMaxX div 2;
   y0:=GetMaxY div 2;
   K:=GetMaxX/30;
   dx:=0.001;
   x:=Min;
   x1:=x0 + Round(x*k);
   y1:=y0 - Round(F(x)*k);
   Moveto(x1,y1);
   While x<Max do
     Begin
       x:=x+dx;
       x1:=x0 + Round(x*k);
       y1:=y0 - Round(F(x)*k);
       LineTo(x1,y1);
     End;
 End;
```

```
BEGIN
 Write('Nhap a= '); Readln(a);
 Write('Nhap b= '); Readln(b);
 Write('Nhap c= '); Readln(c);
 ThietLapDoHoa;
 Min:=-10; Max:=10;
 {Vẽ trục tọa độ}
 Line(GetMaxX Div 2,1,GetMaxX Div 2,GetMaxY);
 Line(1,GetMaxY Div 2,GetMaxX,GetMaxY Div 2);
 VeDoThi(Min,Max);
 Repeat Until KeyPressed;
 CloseGraph;
END.
```

Bài tập 10.10: Vẽ hình bông hoa.

Ý tưởng:

Dùng tọa độ cực. Giả sử ta có tọa độ cực trong đó:

Trục cực: Ox

Góc quay: α

thì tọa độ cực của một điểm trong mặt phẳng là cặp (x,y) với:

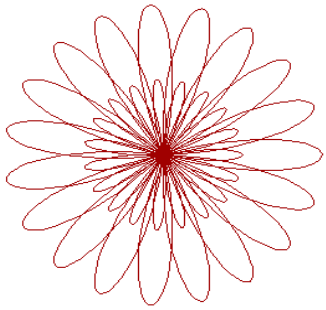
x = f(α).Cos(α)

y = f(α).Sin(α)

Trong đó: f(α) là phương trình do ta qui định.

Ví dụ:

f(α) = k.Cos(nα) : Hình bông hoa.



Hình bông hoa

f(α) = a.α (a>0) : Đường xoắn ốc Acsimet.

f(α) = k.(1+Cos(α)): Hình trái tim.

```
Uses Crt,Graph;
 var R,chuky:real;

Procedure ThietLapDohoa;
Var Gd,Gm:Integer;
Begin
   Gd:=0;
   InitGraph(Gd,Gm,'D:\BP\BGI');
End;
```

```
Function F(R,Alpha:real):real; { Tính hàm f(α) }
 Begin
   F:=R*cos(19*Alpha/3)+5;
 End;

Procedure VeHinh(ChuKy:real);
 var x1,x2,y1,y2:integer;
     a,Alpha,k:real;
     x0,y0:word;
 Begin
   x0:=GetMaxX div 2;
   y0:=GetMaxY div 2;
   K:=GetMaxX/50;
   a:=Pi/180;
   Alpha:=0;
   x1:=x0 + Round(F(R,Alpha)*k*cos(Alpha));
   y1:=y0 - Round(F(R,Alpha)*k*sin(Alpha));
   Moveto(x1,y1);
   While Alpha<ChuKy do
     Begin
       Alpha:=Alpha+a;
       x1:=x0 + Round(F(R,Alpha)*k*cos(Alpha));
       y1:=y0 - Round(F(R,Alpha)*k*sin(Alpha));
       LineTo(x1,y1);
       Delay(10);
     End;
 End;

BEGIN
 ThietLapDoHoa;
 R:=15; chuky:=4*Pi;
 VeHinh(chuky);
 repeat until KeyPressed;
 CloseGraph;
END.
```

Bài tập 10.11: Viết chương trình vẽ cung Koch. Các bước phát sinh của cung Koch được thực hiện trong hình sau:

- Bắt đầu từ đường ngang K_0 có độ dài bằng 1.
- Để tạo cung bậc-1(gọi là K_1), chia đường thành ba phần và thay đoạn giữa bằng tam giác đều có cạnh dài 1/3. Bây giờ, toàn bộ đường cong có độ dài 4/3.

(a) K_0

(b) K_1

(c) K_2

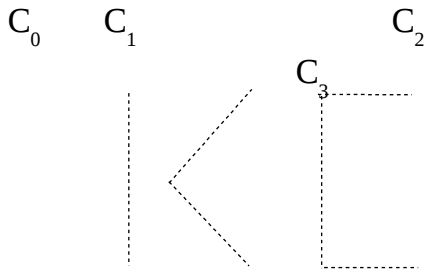
- Cung bậc-2 K_2 có được bằng cánh dựng tiếp các tam giác đều từ 4 đoạn của K1. Vì mỗi đoạn có độ dài tăng 4/3 lần nên toàn bộ cung dài ra 4/3 lần.

Ý tưởng:

Từ hình (b) ta thấy rằng, đầu tiên hướng vẽ quay trái 60^0, rồi quay phải 120^0, cuối cùng quay trái 60^0 để trở về hướng ban đầu.

```
Uses  Crt,Graph;
Var n:Integer;
   Goc,length:real;

Procedure ThietLapDohoa;
Var gd,gm:integer;
Begin
  gd:=0;
  InitGraph(gd,gm,'D:\bp\bgi');
End;

Procedure   Koch(dir,len:real;n:integer);
const  rads=0.017453293;
Begin
  If n>0 Then
    Begin
      Koch(dir,len/3,n-1);
      dir:=dir+60; {Quay phải 60 độ}
      Koch(dir,len/3,n-1);
      dir:=dir-120; {Quay trái 120 độ}
      Koch(dir,len/3,n-1);
      dir:=dir+60; {Quay phải 60 độ}
      Koch(dir,len/3,n-1);
    End
  else LineRel(Round(len*cos(rads*dir)),Round(len*sin(rads*dir)));
end;

Begin
  ThietLapDoHoa;
  n:=4;
  Goc:=180;
  Length:=150;
  Moveto(300,200);
  Koch(Goc,Length,n);
  Repeat until  keypressed;
  Closegraph;
END.
```

Bài tập 10.12: Viết chương trình tạo ra C-cung dựa trên sự tinh chế tương tự của một đoạn thẳng theo hình sau:



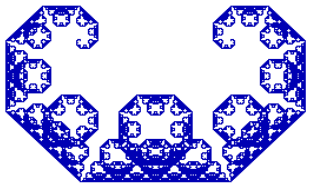
Ý tưởng:

Để có dạng phát sinh kế tiếp, mỗi đoạn thẳng được thay bởi một "hình gãy" gồm 2 đoạn ngắn hơn tạo với nhau một góc 90^0. Các đoạn mới có độ dài bằng $1/\sqrt{2}$ lần đoạn ở bước trước.

Xét hướng vẽ ở một đầu của đoạn thẳng. Để vẽ hình gãy, hướng vẽ quay trái 45^0, vẽ một đoạn, quay phải 90^0, vẽ đoạn thứ hai và sau đó trở về hướng cũ bằng cách quay góc 45^0.

```
Uses  graph,crt;

Procedure ThietLapDohoa;
Var    gd,gm,gr:integer;
Begin
  gd:=0;
  Initgraph(gd,gm,'D:\bp\bgi');
End;

PROCEDURE     VeC_Cung;
Var n:Integer;
      Goc,length:real;
    Procedure   Rong(dir,len:real;n:integer);
     const  d=0.7071067;
            rads=0.017453293;
     begin
       if n>1 then
         begin
           dir:=dir+45;
           Rong(dir,len*d,n-1);
           dir:=dir-90;
           Rong(dir,len*d,n-1);
           dir:=dir+45;
         end
       else LineRel(Round(len*cos(rads*dir)),Round(len*sin(rads*dir)));
     end;
```

```
  Begin
    n:=15;
    Goc:=0;
    Length:=130;
    Moveto(200,200);
    Rong(Goc,Length,n);
    repeat until  keypressed;
  End;

BEGIN
  ThietLapDoHoa;
  VeC_Cung;
  Closegraph;
END.
```



C Cung

Bài tập 10.13: Viết chương trình vẽ tập Mandelbrot - là một hình trong mặt phẳng phức. Tập Mandelbrot được phát sinh theo công thức sau:

$$z \rightarrow z^2 + c \ (*)$$

Tập hợp Mandelbrot là tập bao gồm những số phức c sao cho z^2+c vẫn hữu hạn với mọi lần lặp.

Ý tưởng:

Ta chọn số phức cố định c và tính biểu thức z2+c với z là số phức biến đổi.

Nếu chọn z = 0 thì $z^2+c = c$. Thay z vào công thức (*) ta được c^2+c.

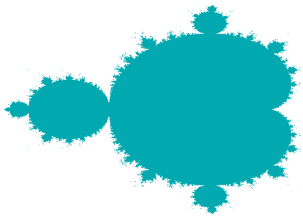
Tiếp tục thay z bằng giá trị mới, ta lại có: (c2+c)2+c, ...

Cứ như vậy, ta thu được một dãy vô hạn các số z.

```
Uses crt,graph;
Const row=1;
        col=1;
Var  x1,y1,x2,y2,kx,ky:real;
       Gioihan:Byte;
       x0,y0:word;
   Diemduoi,Diemtren:Integer;

Procedure ThietLapDohoa;
Var    gd,gm,gr:integer;
Begin
  gd:=0;
  Initgraph(gd,gm,'D:\bp\bgi');
```

```
End;

Procedure KhoiTao;
 Begin
   Diemduoi:=GetMaxX;
   Diemtren:=GetMaxY;
   x1:=-2; y1:=-1.25;
   x2:=0.5; y2:=1.25;
   kx:=(x2-x1)/diemduoi;
   ky:=(y2-y1)/diemtren;
   Gioihan:=50;
 End;

Procedure ManDelbrot;
 var  dong,cot,dem:integer;
      P0,Q0,Modun,x,y,Aux:real;
 Begin
  cot:=0;
  While cot<=diemduoi do
   Begin
     P0:=x1+cot*kx;
     dong:=0;
     While dong<=(diemtren div 2) do
       Begin
        Q0:=y1+dong*ky;
        x:=0; y:=0;
        dem:=1; Modun:=1;
        While (dem<=gioihan)and(modun<4) do
          Begin
            Aux:=x;
            x:=x*x-y*y +P0;
            y:=2*y*Aux + Q0;
            Modun:=x*x + y*y;
            dem:=dem+1;
          End;
        If Modun<4 Then
          Begin
           PutPixel(cot,dong,3);
           PutPixel(cot,diemtren-dong,3);
          End;
        dong:=dong+row;
       End;
     cot:=cot+col;
   End;
 End;
```

```
Begin
 ThietLapDohoa;
 KhoiTao;
 Mandelbrot;
 readln;
 CloseGraph;
End.
```



Tập MandelBrot

Bài tập 10.14: Viết chương trình mô phỏng phép quay một tam giác quanh gốc tọa độ.

Ý tưởng:

Ma trận của phép quay quanh gốc tọa độ: $R = \begin{pmatrix} Cos(\alpha) & Sin(\alpha) \\ -Sin(\alpha) & Cos(\alpha) \end{pmatrix}$

$\Leftrightarrow \begin{cases} x' = x.Cos(\alpha) - y.Sin(\alpha) \\ y' = x.Sin(\alpha) + y.Cos(\alpha) \end{cases}$

```
Uses crt,Graph;
 Type ToaDo=Record
       x,y:real;
      End;
 var k,Alpha,goc:real;
     P,PP,PPP,P1,P2,P3:ToaDo;
     x0,y0:word;
     ch:char;

Procedure ThietLapDohoa;
Var    gd,gm,gr:integer;
Begin
  gd:=0;
  Initgraph(gd,gm,'D:\bp\bgi');
End;

Procedure VeTruc;
  Begin
     Line(GetMaxX div 2,0,GetMaxX div 2,GetMaxY);
     Line(0,GetMaxY div 2,GetMaxX,GetMaxY div 2);
  End;

Procedure VeHinh(P1,P2,P3:ToaDo);
  Begin
```

```
    Line(x0+Round(P1.x*k),y0-Round(P1.y*k),
          x0+Round(P2.x*k),y0- Round(P2.y*k));
    Line(x0+Round(P2.x*k),y0-Round(P2.y*k),
          x0+Round(P3.x*k),y0- Round(P3.y*k));
    Line(x0+Round(P3.x*k),y0-Round(P3.y*k),
          x0+Round(P1.x*k),y0- Round(P1.y*k));
  End;

Procedure QuayDiem(P:ToaDo;Alpha:real; var PMoi:ToaDo);
  Begin
    PMoi.x:=P.x*cos(Alpha)-P.y*sin(Alpha);
    PMoi.y:=P.x*sin(Alpha)+P.y*cos(Alpha);
  End;

Procedure QuayHinh(P1,P2,P3:ToaDo;Alpha:real;
                          var P1Moi,P2Moi,P3Moi:ToaDo);
  Begin
   QuayDiem(P1,Alpha,P1Moi);
   QuayDiem(P2,Alpha,P2Moi);
   QuayDiem(P3,Alpha,P3Moi);
  End;

BEGIN
  ThietLapDoHoa;
  x0:=GetMaxX div 2;
  y0:=GetMaxY div 2;
  k:=GetMaxX/50;
  Vetruc;
  P.x:=5; P.y:=3; PP.x:=2; PP.y:=6; PPP.x:=6; PPP.y:=-4;
  P1:=P; P2:=PP; P3:=PPP;
  Alpha:=0; goc:=Pi/180;
  SetWriteMode(XORPut);
  VeHinh(P,PP,PPP);
  Repeat
    ch:=readkey;
    if ord(ch)=0 then ch:=readkey;
    case Upcase(ch) of
      'K': Begin
            VeHinh(P1,P2,P3);
            Alpha:=Alpha-goc;
            QuayHinh(P,PP,PPP,Alpha,P1,P2,P3);
            VeHinh(P1,P2,P3);
           End;
      'M': Begin
            VeHinh(P1,P2,P3);
            Alpha:=Alpha+goc;
            QuayHinh(P,PP,PPP,Alpha,P1,P2,P3);
```

```
            VeHinh(P1,P2,P3);
           End;
    End;
  Until ch=#27;
  CloseGraph;
END.
```

BÀI TẬP TỰ GIẢI

Bài tập 10.15: Viết chương trình vẽ bàn cờ quốc tế lên màn hình.

Bài tập 10.16: Viết chương trình vẽ một chiếc xe ô tô (theo hình dung của bạn) và cho nó chạy ngang qua màn hình.

Gợi ý:

Dùng kỹ thuật lật trong màn hình hoặc di chuyển vùng màn hình.

Bài tập 10.17: Viết chương trình vẽ lá cờ tổ quốc đang tung bay.

Gợi ý:

Dùng kỹ thuật lật trong màn hình.

Bài tập 10.18: Viết chương trình nhập vào n học sinh của một lớp học bao gồm 2 trường sau: Họ tên, điểm trung bình.

a/ Hãy thống kê số lượng học sinh giỏi, khá, trung bình và yếu.

b/ Vẽ biểu đồ thống kê số lượng học sinh giỏi, khá, trung bình và yếu theo 2 dạng: biểu đồ cột (column) và biểu đồ bánh tròn (Pie).

Bài tập 10.19: Viết chương trình để vẽ đồ thị của các hàm số sau:

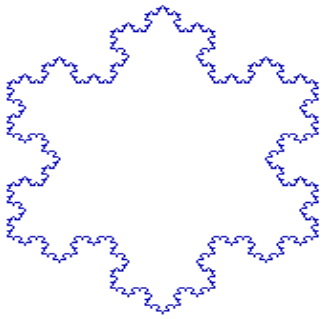
a/ $y = ax^3 + bx^2 + cx + d$

b/ $y = ax^4 + bx^3 + cx^2 + dx + e$

c/ $y = \frac{ax+b}{cx+d}$

d/ $y = \frac{ax^2+bx+c}{dx+e}$

Bài tập 10.20: Hình vẽ cung Koch dựa trên 3 cạnh của tam giác đều như hình sau:



Bài tập 10.21: Viết chương trình để vẽ đường xoắn ốc.

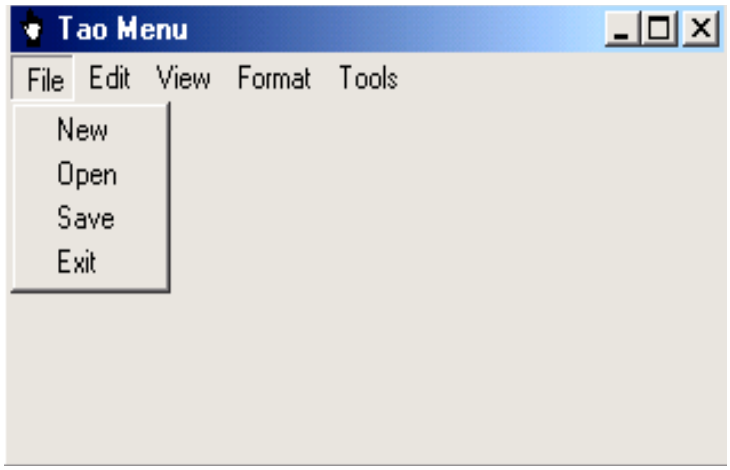
Gợi ý:
Dùng tọa độ cực.

Bài tập 10.22: Viết chương trình vẽ cái đồng hồ đang hoạt động.

Bài tập 10.23: Viết chương trình mô phỏng chuyển động của trái đất xung quanh mặt trời và đồng thời chuyển động của mặt trăng xung quanh trái đất.
Gợi ý:
Dùng ma trận của phép quay.

Bài tập 10.24: Xây dựng một thư viện (Unit) chứa tất cả các bài tập trong chương này.

Bài tập 10.25: Viết chương trình tạo Menu đồ họa giống như các Menu trong môi trường WINDOWS (xem hình).



MỤC LỤC