CÁC THUẬT TOÁN VỀ SỐ

THUẬT TOÁN KIỂM TRA SỐ NGUYÊN TỐ

Thuật toán của ta dựa trên ý tưởng: nếu n >1 không chia hết cho số nguyên nào trong tất cả các số từ 2 đến $\sqrt{n}$ thì n là số nguyên tố. Do đó ta sẽ *kiểm tra tất cả các số nguyên từ 2 đến có round(sqrt(n)), nếu n không chia hết cho số nào trong đó thì n là số nguyên tố.*

Nếu thấy biểu thức *round(sqrt(n))* khó viết thì ta có thể kiểm tra từ *2* đến *n* ***div*** *2*.

Hàm kiểm tra nguyên tố nhận vào một số nguyên n và trả lại kết quả là true (đúng) nếu n là nguyên tố và trả lại false nếu n không là số nguyên tố.

```
function ngto(n:integer):boolean;
var i:integer;
begin
   ngto:=false;
   if n<2 then exit;
   for i:=2 to trunc(sqrt(n)) do
      if n mod i=0 then exit; {nếu n chia hết cho i thì n không là nguyên tố =>
thoát luôn}
   ngto:=true;
end;
```

Chú ý: Dựa trên hàm kiểm tra nguyên tố, ta có thể tìm các số nguyên tố từ 1 đến n bằng cách cho i chạy từ 1 đến n và gọi hàm kiểm tra nguyên tố với từng giá trị i.

THUẬT TOÁN TÍNH TỔNG CÁC CHỮ SỐ CỦA MỘT SỐ NGUYÊN

Ý tưởng là ta chia số đó cho 10 lấy dư (**mod**) thì được chữ số hàng đơn vị, và lấy số đó **div** 10 thì sẽ được phần còn lại. Do đó sẽ chia liên tục cho đến khi không chia được nữa (số đó bằng 0), mỗi lần chia thì được một chữ số và ta cộng dồn chữ số đó vào tổng.

Hàm tính tổng chữ số nhận vào 1 số nguyên n và trả lại kết quả là tổng các chữ số của nó:

```
function tongcs(n:integer): integer;
var s : integer;
begin
        s := 0;
        while n <> 0 do begin
                s := s + n mod 10;
                n := n div 10;
        end;
```

```
        tongcs := s;
end;
```

Chú ý: Tính tích các chữ số cũng tương tự, chỉ cần chú ý ban đầu gán s là 1 và thực hiện phép nhân s với n **mod** 10.

THUẬT TOÁN EUCLIDE TÍNH UCLN

Ý tưởng của thuật toán Euclide là UCLN của 2 số a,b cũng là UCLN của 2 số b và a **mod** b, vậy ta sẽ đổi a là b, b là a **mod** b cho đến khi b bằng 0. Khi đó UCLN là a.

Hàm UCLN nhận vào 2 số nguyên a,b và trả lại kết quả là UCLN của 2 số đó.

```
function UCLN(a,b: integer): integer;
var r : integer;
begin
        while b<>0 do begin
                r := a mod b;
                a := b;
                b := r;
        end;
        UCLN := a;
end;
```

Chú ý: Dựa trên thuật toán tính UCLN ta có thể kiểm tra được 2 số nguyên tố cùng nhau hay không. Ngoài ra cũng có thể dùng để tối giản phân số bằng cách chia cả tử và mẫu cho UCLN.

THUẬT TOÁN TÍNH TỔNG CÁC ƯỚC SỐ CỦA MỘT SỐ NGUYÊN

Để tính tổng các ước số của số n, ta cho i chạy từ 1 đến n **div** 2, nếu n chia hết cho số nào thì ta cộng số đó vào tổng. (Chú ý cách tính này chưa xét n cũng là ước số của n).

```
function tongus(n : integer): integer;
var i,s : integer;
begin
        s := 0;
        for i := 1 to n div 2 do
                if n mod i = 0 then s := s + i;
        tongus := s;
end;
```

Chú ý: Dựa trên thuật toán tính tổng ước số, ta có thể kiểm tra được 1 số nguyên có là số hoàn thiện không: số nguyên gọi là số hoàn thiện nếu nó bằng tổng các ước số của nó.

CÁC THUẬT TOÁN VỀ VÒNG LẶP

THUẬT TOÁN TÍNH GIAI THỪA MỘT SỐ NGUYÊN

Giai thừa n! là tích các số từ 1 đến n. Vậy hàm giai thừa viết như sau:

```
function giaithua(n : integer) : longint;
var i : integer; s : longint;
begin
        s := 1;
        for i := 2 to n do s := s * i;
        giaithua := s;
end;
```

THUẬT TOÁN TÍNH HÀM MŨ

Trong Pascal ta có thể tính a^b bằng công thức exp(b*ln(a)). Tuy nhiên nếu a không phải là số dương thì không thể áp dụng được.

Ta có thể tính hàm mũ a^n bằng công thức lặp như sau:

```
function hammu(a : real; n : integer): real;
var s : real; i : integer;
begin
        s := 1;
        for i := 1 to n do s := s * a;
        hammu := s;
end;
```

THUẬT TOÁN TÍNH CÔNG THỨC CHUỖI

Thuật toán tính hàm e^x:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ... + \frac{x^n}{n!}$$

Đặt: $s_n = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ... + \frac{x^n}{n!}$ và $r_n = \frac{x^n}{n!}$, ta được công thức truy hồi:

$$\begin{cases} s_0 = 1,\ r_0 = 1 \\ r_i = \dfrac{r_{i-1}.x}{i} \\ s_i = s_{i-1} + r_i \end{cases}$$

Khi đó, ta có thể tính công thức chuỗi trên như sau:

```
function expn(x: real; n : integer): real;
var s,r : real; i : integer;
begin
        s := 1; r := 1;
```

```
        for i := 1 to n do begin
                r := r * x / i;
                s := s + r;
        end;
        expn := s;
end;
```

CÁC BÀI TẬP VỀ MẢNG 1 CHIỀU VÀ 2 CHIỀU

BÀI TẬP 1

Nhập vào một số n (5<=n<=10) và n phần tử của dãy a, $1<a_i<100$ (có kiểm tra dữ liệu khi nhập).

a) In ra các phần tử là số nguyên tố của dãy.

b) Tính ước chung lớn nhất của tất cả các phần tử của dãy.

c) Tính biểu thức sau:

$$S = a_1^1 + a_2^2 + ...a_n^n$$

d) Sắp xếp dãy tăng dần và in ra dãy sau sắp xếp.

HƯỚNG DẪN

Ta nên chia chương trình thành các chương trình con, mỗi chương trình thực hiện một yêu cầu. Ngoài ra ta cũng viết thêm các hàm kiểm tra nguyên tố, hàm mũ, hàm UCLN để thực hiện các yêu cầu đó.

Chương trình như sau:

Khai báo dữ liệu:

```
uses crt;
var n : integer;
    a : array[1..10] of integer; {n<=10 nên mảng có tối đa 10 phần tử}
```

Thủ tục nhập dữ liệu, có kiểm tra khi nhập.

```
procedure nhap;
var i : integer;
begin
        clrscr;
        write('NHAP VAO SO PHAN TU N = ');
        repeat
                readln(n);
                if (5<=n) and (n<=10) then break; {nếu thoã mãn thì dừng vòng
lặp}
```

```
            writeln('Khong hop le (5<=n<=10). Nhap lai!!!'); {ngược lại thì
báo lỗi}
        until false;

        writeln('NHAP VAO N PHAN TU (1<ai<100)');
        for i := 1 to n do begin
            write('a',i,'=');
            repeat
                readln(a[i]);
                if (1<a[i]) and (a[i]<100) then break;
                writeln('Khong hop le. Nhap lai!!!');
            until false;
        end;
end;

function ngto(n : integer): boolean; {hàm kiểm tra nguyên tố, xem giải thích ở
phần trên}
var i : integer;
begin
        ngto := false;
        if n < 2 then exit;
        for i := 2 to round(sqrt(n)) do
            if n mod i = 0 then exit;
        ngto := true;
end;
```

Thủ tục in các số nguyên tố của một mảng

```
procedure inngto;
var i :integer;
begin
        writeln('CAC PHAN TU NGUYEN TO TRONG DAY:');
        for i := 1 to n do                          {duyệt qua mọi phần tử từ 1 đến
n}
            if ngto(a[i]) then writeln(a[i]);       {nếu ai là nguyên tố thì in
ra}
end;

function UCLN(a,b: integer): integer;
var r : integer;
begin
        while b<>0 do begin
            r := a mod b;
            a := b;
```

```
            b := r;
        end;
        UCLN := a;
end;
```

Thủ tục tính UCLN của các phần tử của một mảng

```
procedure TinhUC;
var i,u : integer;
begin
        u := a[1];                              {u là UCLN của các phần tử từ 1 đến i}
        for i := 2 to n do u := UCLN(u,a[i]);   {là UCLN của các phần tử từ 1 đến i-1 và ai}
        writeln('UCLN cua ca day la:',u);
end;

function hammu(a : real; n : integer): real; {hàm mũ tính a^n}
var s : real; i : integer;
begin
        s := 1;
        for i := 1 to n do s := s * a;
        hammu := s;
end;
```

Thủ tục tính tổng các phần tử có lấy mũ:

```
procedure tong;
var s : real; i : integer; {s phải khai báo là số thực để tránh tràn số}
begin
        s := 0;
        for i := 1 to n do s := s + hammu(a[i],i); {s := s + (ai)^i}
        writeln('Tong can tinh:',s:10:0);
end;
```

Thủ tục sắp xếp tăng dần các phần tử của một mảng:

```
procedure sxep;
var i,j,tg : integer;
begin
        for i := 1 to n-1 do
                for j := i + 1 to n do
                        if a[i] > a[j] then begin
                                tg := a[i]; a[i] := a[j]; a[j] := tg;
                        end;
        writeln('DAY SAU KHI SAP XEP TANG DAN:');
```

```
        for i := 1 to n do writeln(a[i]);
end;
```

Chương trình chính: lần lượt gọi từng thủ tục

```
BEGIN
        nhap;
        inngto;
        tinhuc;
        tong;
        sxep;
END.
```

BÀI TẬP 2

Tìm phần tử nhỏ nhất, lớn nhất của một mảng (cần chỉ ra cả vị trí của phần tử).

HƯỚNG DẪN

Giả sử phần tử min cần tìm là phần tử k. Ban đầu ta cho k=1. Sau đó cho i chạy từ 2 đến n, nếu a[k] > a[i] thì rõ ràng a[i] bé hơn, ta gán k bằng i. Sau khi duyệt toàn bộ dãy thì k sẽ là chỉ số của phần tử min. (Cách tìm min này đơn giản vì từ vị trí ta cũng suy ra được giá trị).

```
procedure timmin;
var i, k : integer;
begin
        k := 1;
        for i := 2 to n do
                if a[k] > a[i] then k := i;
        writeln('Phan tu nho nhat la a[',k,']=',a[k]);
end;
```

Tìm max cũng tương tự, chỉ thay dấu so sánh.

```
procedure timmax;
var i, k : integer;
begin
        k := 1;
        for i := 2 to n do
                if a[k] < a[i] then k := i;
        writeln('Phan tu lon nhat la a[',k,']=',a[k]);
end;
```

Chú ý:

1. Nếu áp dụng với mảng 2 chiều thì cũng tương tự, chỉ khác là để duyệt qua mọi phần tử của mảng 2 chiều thì ta phải dùng 2 vòng **for**. Và vị trí một phần tử cũng gồm cả dòng và cột.

Ví dụ 1. **Tìm phần tử nhỏ nhất và lớn nhất của mảng 2 chiều và đổi chỗ chúng cho nhau**:

```
procedure exchange;
var i,j,i1,j1,i2,j2,tg : integer;
begin
        i1 := 1; j1 := 1; {i1,j1 là vị trí phần tử min}
        i2 := 1; j2 := 1; {i2,j2 là vị trí phần tử max}
        for i := 1 to m do
                for j := 1 to n do begin
                        if a[i1,j1] > a[i,j] then begin {so sánh tìm min}
                                i1 := i; j1 := j;       {ghi nhận vị trí min mới}
                        end;
                        if a[i2,j2] < a[i,j] then begin {so sánh tìm max}
                                i2 := i; j2 := j; {ghi nhận vị trí max mới}
                        end;
                end;
        tg := a[i1,j1]; a[i1,j1] := a[i2,j2]; a[i2,j2] := tg; {đổi chỗ}
end;
```

2. Nếu cần tìm phần tử lớn nhất / nhỏ nhất hoặc sắp xếp 1 dòng (1 cột) của mảng 2 chiều thì ta cũng coi dòng (cột) đó như 1 mảng 1 chiều. Chẳng hạn tất cả các phần tử trên dòng k đều có dạng chỉ số là a[k,i] với i chạy từ 1 đến n (n là số cột).

Ví dụ 2. **Tìm phần tử lớn nhất của dòng k và đổi chỗ nó về phần tử đầu dòng.**

```
procedure timmax(k : integer);
var i, vt, tg : integer;
begin
        vt := 1; {vt là vị trí của phần tử min dòng k}
        for i := 1 to n do
                if a[k,i] > a[k,vt] then vt := i; {các phần tử dòng k có dạng a[k,i]}
        tg := a[k,1]; a[k,1] := a[k,vt]; a[k,vt] := tg;
end;
```

Ví dụ 3. **Sắp xếp giảm dần cột thứ k.**

```
procedure sapxep(k: integer);
var i,j,tg : integer;
begin
```

```
        for i := 1 to m-1 do {mỗi cột có m phần tử, vì bảng có m dòng}
                for j := i+1 to m do
                        if a[i,k] > a[j,k] then begin {các phần tử cột k có dạng a[i,k]}
                                tg := a[i,k]; a[i,k] := a[j,k]; a[j,k] := tg;
                        end;
end;
```

BÀI TẬP 3

Tìm các phần tử thoả mãn 1 tính chất gì đó.

HƯỚNG DẪN

Nếu tính chất cần thoả mãn là cần kiểm tra phức tạp (chẳng hạn: nguyên tố, hoàn thiện, có tổng chữ số bằng 1 giá trị cho trước…) thì ta nên viết một hàm để kiểm tra 1 phần tử có tính chất đó không. Còn tính chất cần kiểm tra đơn giản (chẵn / lẻ, dương / âm, chia hết, chính phương…) thì không cần.

Sau đó ta duyệt qua các phần tử từ đầu đến cuối, phần tử nào thoả mãn tính chất đó thì in ra.

Ví dụ 1. In ra các số chính phương của một mảng:

Để kiểm tra n có chính phương không, ta lấy căn n, làm tròn rồi bình phương và so sánh với n. Nếu biểu thức sqr(round(sqrt(n))) = n là true thì n là chính phương.

Vậy để in các phần tử chính phương ta viết:

```
for i := 1 to n do begin
   if sqr(round(sqrt(a[i]))) = a[i] then writeln(a[i]);
```

Ví dụ 2. In ra các số hoàn thiện từ 1 đến n:

Để kiểm tra số có hoàn thiện ta dùng hàm tổng ước (đã có ở phần đầu).

```
for i := 1 to n do begin
   if tongus(i) = i then writeln(i);
```

Ví dụ 3. In ra các phần tử của mảng chia 3 dư 1, chia 7 dư 2:

```
for i := 1 to n do begin
   if (a[i] mod 3=1) and (a[i] mod 7=2) then writeln(a[i]);
```

Ví dụ 4. In ra các số có 3 chữ số, tổng chữ số bằng 20, chia 7 dư 2.

Ta dùng hàm tổng chữ số đã có ở trên:

```
for i := 100 to 999 do begin {duyệt qua mọi số có 3 chữ số}
   if (tongcs(i)=20) and (i mod 7=2) then writeln(i);
```

Chú ý: Nếu áp dụng với mảng 2 chiều thì cũng tương tự, chỉ khác là để duyệt qua mọi phần tử của mảng 2 chiều thì ta phải dùng 2 vòng **for**.

Ví dụ, để in các phần tử nguyên tố của 1 mảng 2 chiều:

```
for i := 1 to m do begin
for j := 1 to n do begin
   if ngto(a[i,j]) then writeln(a[i,j]);
```

BÀI TẬP 4

Nhập và in mảng 2 chiều dạng ma trận (m dòng, n cột).

HƯỚNG DẪN

Để nhập các phần tử của mảng 2 chiều dạng ma trận, ta cần dùng các lệnh sau của unit CRT (nhớ phải có khai báo user crt ở đầu chương trình).

GotoXY(a,b): di chuyển con trỏ màn hình đến vị trí (a,b) trên màn hình (cột a, dòng b). Màn hình có 80 cột và 25 dòng.

whereX: hàm cho giá trị là vị trí cột của con trỏ màn hình.

whereY: hàm cho giá trị là vị trí dòng của con trỏ màn hình.

Khi nhập 1 phần tử ta dùng lệnh readln nên con trỏ màn hình sẽ xuống dòng, do đó cần quay lại dòng của bằng lệnh GotoXY(j * 10, whereY -1), nếu ta muốn mỗi phần tử của ma trận ứng với 10 cột màn hình.

```
procedure nhap;
var i,j : integer;
begin
        clrscr;
        write('Nhap m,n =  '); readln(m,n);
        for i := 1 to m do begin
                for j := 1 to n do begin
                        write('A[',i,',',j,']=');   readln(a[i,j]); {nhập xong thì xuống dòng}
                        gotoXY(j*10,whereY-1); {di chuyển về dòng trước, vị trí tiếp theo}
                end;
                writeln; {nhập xong 1 hàng thì xuống dòng}
        end;
end;
```

Để in bảng dạng ma trận thì đơn giản hơn, với mỗi dòng ta sẽ in các phần tử trên 1 hàng rồi xuống dòng:

```
procedure inbang;
var i,j : integer;
begin
        for i := 1 to m do begin                    {viết các phần tử của hàng i }
```

```
            for j := 1 to n do write(a[i,j]:6); {mỗi phần tử chiếm 6 ô để căn phải cho thẳng cột và không sít nhau}
            writeln;        {hết 1 hàng thì xuống dòng}
        end;
end;
```

CÁC BÀI TẬP VỀ XÂU KÍ TỰ

BÀI TẬP 1

Nhập vào một xâu s khác rỗng và thực hiện chuẩn hoá xâu, tức là:

a) **Xoá các dấu cách thừa**
b) **Chuyển những kí tự đầu từ thành chữ hoa, những kí tự khác thành chữ thường.**

HƯỚNG DẪN

Chương trình như sau:

```
var s : string;
procedure chuanhoa(var s : string); {s là tham biến để có thể thay đổi trong chương trình con}
var i : integer;
begin
    while s[1]=' ' do delete(s,1,1); {xoá các kí tự cách thừa ở đầu xâu}
    while s[length(s)]=' ' do delete(s,length(s),1); {xoá các kí tự cách thừa ở cuối xâu}
{xoá các kí tự cách thừa ở giữa các từ: nếu s[i-1] là cách thì s[i] là dấu cách là thừa. Phải dùng vòng lặp for downto vì nếu trong quá trình xoá ta làm giảm chiều dài của xâu, nếu for to sẽ không dừng được.}
    for i := length(s) downto 2 do
        if (s[i]=' ') and (s[i-1]=' ') then delete(s,i,1);
{Chuyển kí tự đầu xâu thành chữ hoa}
    s[1] := Upcase(s[1]);
    for i := 2 to length(s) do
        if s[i-1]=' ' then s[i] := Upcase(s[i]) {Chuyển s[i] là kí tự đầu từ thành chữ hoa.}
        else
            if s[i] in ['A'..'Z'] then    {s[i] là kí tự chữ hoa không ở đầu một từ}
              s[i] := chr(ord(s[i]) + 32); {thì phải chuyển thành chữ thường}
end;

BEGIN
  write('Nhap vao 1 xau s:');
  readln(s);
```

```
  chuanhoa(s);
  writeln('Xau s sau khi chuan hoa:',s);
  readln;
END.
```

BÀI TẬP 2

Nhập vào một xâu x khác rỗng và thông báo xâu đó có phải là xâu đối xứng hay không?

HƯỚNG DẪN

Xâu đối xứng nếu nó bằng chính xâu đảo của nó. Vậy cách đơn giản nhất là ta sẽ xây dựng xâu đảo của ***x*** và kiểm tra xem nó có bằng ***x*** không. Để xây dựng xâu đảo của x, cách đơn giản nhất là cộng các kí tự của x theo thứ tự ngược (từ cuối về đầu).

Chương trình:

```
var x : string;
(***********************************************)
function doixung(x : string) : boolean; {hàm kiểm tra xâu đối xứng}
var y : string;
   i : integer;
begin
    y := '';
{xây dựng y là xâu đảo của x, bằng cách cộng dần các kí tự của x vào y theo
thứ tự ngược}
    for i := length(x) downto 1 do y := y + x[i];
{so sánh x và xâu đảo của nó}
    if x=y then doixung := true else doixung := false;
end;
BEGIN
    write('Nhap vao 1 xau:');
    readln(x);
    if doixung(x) then
      writeln('Xau doi xung!')
    else
      writeln('Xau khong doi xung!');
    readln;
END.
```

BÀI TẬP 3

Nhập vào một xâu s và đếm xem nó có bao nhiêu từ. Từ là một dãy các kí tự, cách nhau bởi dấu cách?

HƯỚNG DẪN

Cách đếm từ đơn giản nhất là đếm dấu cách: nếu s[i] là kí tự khác cách và s[i-1] là kí tự cách thì chứng tỏ s[i] là vị trí bắt đầu của một từ. Chú ý là từ đầu tiên của xâu không có dấu cách đứng trước.

Chương trình:

```
var s : string;
{Hàm đếm số từ của một xâu}
function sotu(s : string) : integer;
var i, dem : integer;
begin
{cộng thêm dấu cách phía trước xâu để đếm cả từ đầu tiên}
    s := ' ' + s; dem := 0;
    for i := 2 to length(s) do {s[i] là vị trí bắt đầu 1 từ}
       if (s[i-1]=' ') and (s[i]<>' ') then dem := dem + 1;
    sotu := dem;
end;
BEGIN
    write('Nhap vao 1 xau:');
    readln(s);
    writeln('So tu trong xau la:',sotu(s));
    readln;
END.
```

BÀI TẬP 4

Nhập vào một xâu s và in ra các từ của nó (Từ là một dãy các kí tự, cách nhau bởi dấu cách). Xâu có bao nhiêu từ là đối xứng?

HƯỚNG DẪN

Có nhiều cách để tách một xâu thành các từ. Cách đơn giản nhất tiến hành như sau:

1) Bỏ qua các dấu cách cho đến khi gặp một kí tự khác cách (hoặc hết xâu).
2) Ghi các kí tự tiếp theo vào xâu tạm cho đến khi gặp dấu cách hoặc hết xâu, khi đó ta được 1 từ.
3) Nếu chưa hết xâu thì quay lại bước 1.

Mỗi khi tìm được một từ, ta ghi luôn nó ra màn hình, nếu từ đó là đối xứng thì tăng biến đếm. Ta cũng có thể lưu các từ tách được vào một mảng nếu bài tập yêu cầu dùng đến những từ đó trong các câu sau.

Chương trình:

```
var s : string;
   dem : integer;
{Hàm kiểm tra từ đối xứng}
function doixung(x : string) : boolean;
var y : string;
   i : integer;
begin
    y := '';
    for i := length(x) downto 1 do y := y + x[i];
    if x=y then doixung := true else doixung := false;
end;
{Thủ tục thực hiện tách từ}
procedure tach;
var i, len : integer;
   t : string;
begin
    writeln('Cac tu trong xau:');
    i := 1; len := length(s);
    repeat
{B1: bỏ qua các dấu cách cho đến khi hết xâu hoặc gặp 1 kí tự khác cách:}
        while (s[i]=' ') and (i<=len) do inc(i);
        if i>=len then break; {nếu hết xâu thì dừng}
        t := '';                {t là biến tạm lưu từ đang tách}
{B2: lấy các kí tự khác cách đưa vào biến tạm cho đến khi hết xâu hoặc gặp 1
kí tự cách:}
        while (s[i]<>' ') and (i<=len) do begin
            t := t + s[i];
            inc(i);
        end;
{in ra từ vừa tách được và kiểm tra đối xứng}
        writeln(t);
        if doixung(t) then inc(dem);
    until i >= len;
    writeln('So tu doi xung trong xau:',dem);
end;
(*********************************************)
BEGIN
    write('Nhap vao 1 xau:');
```

```
    readln(s);
    tach;
END.
```

BÀI TẬP 5

Một số nguyên gọi là palindrom nếu nó đọc từ trái sang cũng bằng đọc từ phải sang. Ví dụ 121 là một số palindrom. Nhập một dãy n phần tử nguyên dương từ bàn phím, 5<= n<=20 và các phần tử có 2 đến 4 chữ số. In ra các số là palindrom trong dãy.

HƯỚNG DẪN

Một số là palindrom thì xâu tương ứng của nó là xâu đối xứng. Ta sẽ xây dựng một hàm kiểm tra một số có phải là palindrom không bằng cách chuyển số đó thành xâu và kiểm tra xâu đó có đối xứng không?

Chương trình:

```
uses crt;
var  n : integer;
    a : array[1..20] of integer;
{Thủ tục nhập dữ liệu}
procedure nhap;
var i : integer;
begin
    clrscr;
    repeat
         write('n= '); readln(n);
         if (n<=20) and (n>=5) then break; {nếu đã thoả mãn thì thoát khỏi vòng
lặp}
         writeln('Yeu cau 5<=n<=20. Nhap lai!');
    until false;
    for i := 1 to n do
        repeat
            write('A[',i,']='); readln(a[i]);
            if (a[i]<=9999) and (a[i]>=10) then break; {a[i] có 2 đến 4 chữ số}
            writeln('Yeu cau cac phan tu co 2 den 4 chu so. Nhap lai!');
        until false;
end;
{Hàm kiểm tra bằng các kiểm tra xâu đối xứng}
function palindrom(k : integer): boolean;
var x,y : string;
   i : integer;
begin
```

```
    str(k,x);   {chuyển k thành xâu x}
    y := '';
    for i := length(x) downto 1 do y := y + x[i];
{nếu x là đối xứng thì k là palindrom}
    if x=y then palindrom := true else palindrom := false;
end;
{In kết quả:}
procedure palin;
var i : integer;
begin
    writeln('Cac so la palindrom trong day:');
    for i := 1 to n do
       if palindrom(a[i]) then writeln(a[i]);
    readln;
end;
(* Chương trình chính *)
BEGIN
    nhap;
    palin;
END.
```

CÁC BÀI TẬP VỀ TỆP

BÀI TẬP 1

Nhập một mảng 2 chiều m dòng, n cột từ file BANGSO.TXT. Cấu trúc file như sau: dòng đầu là 2 số m và n, cách nhau bằng dấu cách, m dòng sau, mỗi dòng n số nguyên.

a) **Hãy in ra những số là số nguyên tố của mảng.**
b) **Tìm vị trí phần tử lớn nhất trong mảng.**
c) **Sắp xếp mỗi dòng của mảng tăng dần và in ra mảng dạng ma trận.**

HƯỚNG DẪN

Ta khai báo một mảng 2 chiều và nhập dữ liệu từ file vào mảng. Quá trình nhập từ file văn bản giống như nhập từ bàn phím, không cần thực hiện kiểm tra dữ liệu.

Để sắp xếp mảng theo yêu cầu, ta thực hiện sắp xếp từng dòng của mảng bằng cách viết một thủ tục sắp xếp (kiểu đổi chỗ cho đơn giản) coi mỗi dòng của mảng như 1 mảng 1 chiều.

Chương trình:

```
var m,n : integer;
```

```
   a : array[1..100,1..100] of integer;
(* Nhập dữ liệu *)
procedure nhap;
var f : text;
   i,j : integer;
begin
    assign(f,'BANGSO.TXT'); reset(f);
    readln(f,m,n);
    for i := 1 to m do
       for j := 1 to n do read(f,a[i,j]);
    close(f);
end;

function ngto(k : integer): boolean;
var i : integer;
begin
    ngto := false;
    if k < 2 then exit;
    for i := 2 to round(sqrt(k)) do
       if k mod i = 0 then exit;
    ngto := true;
end;

procedure inngto;
var i,j : integer;
begin
    writeln('Cac phan tu nguyen to cua mang:');
    for i := 1 to m do
       for j := 1 to n do
          if ngto(a[i,j]) then write(a[i,j],' ');
    writeln;
end;

procedure timmax;
var max,i,j,im,jm : integer;
begin
    max := a[1,1]; im := 1; jm := 1; {im, jm lưu toạ độ phần tử đạt max}
    for i := 1 to m do
       for j := 1 to n do
          if max < a[i,j] then begin
            max := a[i,j];    {mỗi lần gán max thì gán toạ độ luôn}
            im := i; jm := j;
          end;
```

```
    writeln('Phan tu lon nhat bang la A[',im,',',jm,']=',max);
end;
{Thủ tục thực hiện sắp xếp tăng dần dòng thứ k. Các phần từ dòng k có dạng
a[k,i]}
procedure xepdong(k: integer);
var i,j, tg : integer;
begin
    for i := 1 to n do
       for j := i+1 to n do
          if a[k,i] > a[k,j] then begin
            tg := a[k,i]; a[k,i] := a[k,j]; a[k,j] := tg;
          end;
end;

procedure sapxep;
var i,j : integer;
begin
    for i := 1 to m do xepdong(i); {sắp xếp từng dòng}
    writeln('Mang sau khi sap xep:');
    for i := 1 to m do begin                {in dạng ma trận}
       for j := 1 to n do write(a[i,j] : 5); {in các phần tử trên 1 dòng}
       writeln;                {in hết 1 dòng thì xuống dòng}
    end;
end;
BEGIN
    nhap;
    inngto;
    timmax;
    sapxep;
END.
```

BÀI TẬP 2

Nhập 2 số m, n từ bàn phím, sau đó sinh ngẫu nhiên m×n số nguyên ngẫu nhiên có giá trị từ 15 đến 300 để ghi vào file BANG.TXT. Sau đó thực hiện các yêu cầu sau:

a) In m×n số đã sinh dạng ma trận m dòng, n cột.

b) In ra các số chính phương.

Yêu cầu: không được dùng mảng 2 chiều để lưu trữ dữ liệu.

HƯỚNG DẪN

Do yêu cầu không được dùng mảng 2 chiều để lưu trữ dữ liệu nên ta sẽ đọc file đến đâu, xử lí đến đấy.

- Để sinh các số ngẫu nhiên từ a đến b, ta dùng biểu thức **a + random(b-a+1)**.
- Để kiểm tra số k có phải là số chính phương không, ta lấy căn bậc 2 của k, làm tròn rồi bình phương. Nếu kết quả bằng k thì k là số chính phương. Tức là kiểm tra **sqr(round(sqrt(k))) = k**.

Chương trình:

```
var m,n : integer;
   f : text;
procedure sinh;
var
   i,j : integer;
begin
    write('Nhap vao 2 so m,n: '); readln(m,n);
    assign(f,'BANG.TXT'); rewrite(f);
    writeln(f,m,' ',n);
    for i := 1 to m do begin
       for j := 1 to n do
          write(f,15 + random(300-15+1) : 6); {sinh số ngẫu nhiên từ 15 đến
300}
       writeln(f);
    end;
    close(f);
end;
{Hàm chính phương}
function cp(k : integer) : boolean;
begin
    if sqr(round(sqrt(k))) = k then cp := true
    else cp := false;
end;

procedure chinhphuong;
var
   i,j,k : integer;
begin
    assign(f,'BANG.TXT'); reset(f);
    readln(f,m,n);
    writeln('CAC SO CHINH PHUONG CUA BANG:');
    for i := 1 to m do begin
       for j := 1 to n do begin
          read(f,k);
          if cp(k) then write(k,' '); {vừa đọc vừa xử lí}
       end;
```

```
    end;
    close(f);
end;

procedure inbang;
var
   i,j,k : integer;
begin
    assign(f,'BANG.TXT'); reset(f); {mở lại để in dạng ma trận}
    readln(f,m,n);
    writeln(#10,'IN BANG DANG MA TRAN:');
    for i := 1 to m do begin
       for j := 1 to n do begin
          read(f,k);
          write(k : 6);        {đọc đến đâu in đến đó}
       end;
       writeln;
    end;
    close(f);
end;

BEGIN
    sinh;
    chinhphuong;
    inbang;
END.
```

CÁC BÀI TẬP VỀ BẢN GHI

BÀI TẬP 1

Viết chương trình quản lí sách. Mỗi cuốn sách gồm tên sách, tên nhà xuất bản, năm xuất bản, giá tiền, số lượng:

a) **Đưa ra danh sách các cuốn sách của nhà xuất bản Giáo dục.**
b) **Tính tổng số tiền sách.**
c) **Sắp xếp danh sách theo năm xuất bản giảm dần và ghi kết quả ra màn hình.**
d) **In ra màn hình các cuốn sách có giá tiền<=10.000đ và xuất bản sau năm 2000.**

HƯỚNG DẪN

Mô tả mỗi cuốn sách là một bản ghi, các thông tin về nó (tên sách, tên tác giả,…) là các trường. Danh sách cuốn sách sẽ là một mảng các bản ghi.

Khai báo kiểu dữ liệu mô tả sách như sau:

```
type
   sach = record
        ten : string[30];              {tên sách}
        nxb  : string[20];             {tên Nhà xuất bản}
        namxb   : integer;    {năm xuất bản}
        soluong : integer;    {số lượng}
        gia : real;                {giá tiền}
   end;
```

Thông tin của tất cả các cuốn sách ta lưu trong một mảng các bản ghi kiểu sach:

```
var
  ds : array[1..100] of sach;
  n : integer;
```

Nhập dữ liệu: ta nhập tên sách trước. Nếu tên sách là xâu rỗng thì đừng nhập, ngược lại lần lượt nhập các thông tin khác:

```
  procedure nhap;
  var t : string;
  begin
      ClrScr;
      writeln('NHAP THONG TIN VE CAC CUON SACH');
      writeln('(nhap ten sach la xau rong neu muon dung)');
      repeat
          write('Ten sach:  ');
          readln(t);
          if t='' then break;
          n := n + 1;
          with ds[n] do begin
            ten := t;
            write('NXB:  ');readln(nxb);
            write('Nam xuat ban:  ');readln(namxb);
            write('So luong:  ');readln(soluong);
            write('Gia tien:  ');readln(gia);
          end;
      until false;
  end;
```

Câu a: ta sẽ duyệt qua toàn bộ danh sách các cuốn sách, kiểm tra nếu tên nhà xuất bản là **Giáo dục** thì in ra tất cả các thông tin của cuốn sách tương ứng:

```
  procedure insach;
```

```
var
  i   : integer;
begin
    Clrscr;
    writeln('CAC CUON SACH CUA NXB GIAO DUC:');
    for i:=1 to n do
       with ds[i] do
          if nxb='Giao duc' then begin
            writeln('Ten:',ten);
            writeln('Nam xuat ban:',namxb);
            writeln('So luong:',soluong);
            writeln('Gia tien:',gia);
          end;
    readln;
end;
```

Câu b: ta cũng duyệt qua toàn bộ các cuốn sách, nhân số lượng và giá tiền rồi cộng dồn vào một biến tổng. Sau đó in ra biến tổng đó:

```
procedure tinh;
var i : integer;
   tong : real;
begin
    tong := 0;
    for i := 1 to n do
       with ds[i] do tong := tong + gia * soluong;
    writeln('TONG GIA TRI CUA TAT CA CAC CUON SACH:', tong:0:3);
end;
```

Câu c: Sắp xếp danh sách giảm dần theo năm xuất bản bằng phương pháp nổi bọt (2 vòng **for**). *Chú ý biến trung gian trong đổi chỗ phải có kiểu* ***sach*** *thì mới gán được*.

```
procedure sxep;
var i,j : integer;
   tg : sach;
begin
    for i := 1 to n do
       for j := i + 1 to n do
          if ds[i].namxb < ds[j].namxb then begin
            tg := ds[i]; ds[i] := ds[j]; ds[j] := tg;
          end;
    for i:=1 to n do
```

```
      with ds[i] do begin
         writeln('Ten:',ten);
         writeln('Nam xuat ban:',namxb);
         writeln('So luong:',soluong);
         writeln('Gia tien:',gia);
      end;
   readln;
end;
```

Câu d: ta làm tương tự việc in danh sách các sách của NXB Giáo dục:

```
procedure inds;
var i : integer;
begin
   writeln('CAC CUON SACH GIA RE HON 10000 VA XUAT BAN TU
NAM 2000:');
   for i := 1 to n do
      with ds[i] do
         if (gia <= 10000) and (namxb >= 2000) then writeln(ten);
end;
```

Chương trình chính: Lần lượt gọi các chương trình con theo thứ tự:

```
BEGIN
   nhap;
   insach;
   tinh;
   sxep;
   inds;
   readln;
END.
```

BÀI TẬP 2

Viết chương trình quản lí cán bộ. Thông tin về cán bộ gồm tên, tuổi, hệ số lương, phụ cấp, thu nhập.

a) Nhập thông tin cán bộ từ file văn bản CANBO.TXT. Các thông tin gồm tên, tuổi, hệ số lương, phụ cấp, mỗi thông tin trên một dòng.

Tính thu nhập = hệ số lương × 350000đ + phụ cấp

b) Đưa ra danh sách các bộ trẻ (tuổi <= 30), in đầy đủ các thông tin

c) Sắp xếp tên cán bộ theo abc và ghi lên file truy cập trực tiếp SAPXEP.DAT.

d) Đọc danh sách từ file SAPXEP.DAT, in ra màn hình các cán bộ có thu nhập từ 3 triệu trở lên.

HƯỚNG DẪN

Làm tương tự bài 1, chú ý là nhập dữ liệu từ file chứ không phải từ bàn phím. Do đó không cần ghi các thông tin yêu cầu nhập ra màn hình. Hơn nữa, phải tạo trước một file văn bản là CANBO.TXT để chương trình có thể chạy mà không báo lỗi.

Toàn văn chương trình:

```
uses crt;
type
   canbo = record
        ten : string[20];
        tuoi : byte;
        hsl, phucap, thunhap: real;
   end;
var
  ds : array[1..100] of canbo;
  n : integer;
(*********************************************)
procedure nhap;
var f : text;
begin
    assign(f,'CANBO.TXT'); reset(f);
    n := 0;
    while not eof(f) do begin
         n := n + 1;
         with ds[n] do begin
             readln(f,ten);
             readln(f,tuoi);
             readln(f,hsl);
             readln(f,phucap);
             thunhap := hsl * 350000 + phucap;
         end;
    end;
    close(f);
end;
(*********************************************)
procedure in30;
var i : integer;
begin
    writeln('DANH SACH CAC CAN BO TRE:');
```

```
    for i := 1 to n do
       with ds[i] do
          if tuoi <= 30 then begin
            writeln('Ten:',ten);
            writeln('Tuoi:',tuoi);
            writeln('He so luong:',hsl :0 :3);
            writeln('Phu cap:',phucap :0 :3);
            writeln('Thu nhap:',thunhap :0 :3);
          end;
end;
(*********************************************)
procedure sxep;
var i,j : integer;
   tg : canbo;
begin
    for i := 1 to n do
       for j := i + 1 to n do
          if ds[i].ten > ds[j].ten then begin
            tg := ds[i]; ds[i] := ds[j]; ds[j] := tg;
          end;
end;
(*********************************************)
procedure ghitep;
var f : file of canbo;
   i : integer;
begin
    assign(f,'SAPXEP.DAT'); rewrite(f);
    for i := 1 to n do write(f,ds[i]);
    close(f);
end;
procedure doctep;
var f : file of canbo;
   i : integer;
begin
    assign(f,'SAPXEP.DAT'); reset(f);
    i := 0;
    while not eof(f) do begin
         i := i + 1;
         read(f,ds[i]);
    end;
    n := i;
    close(f);
end;
```

```
(*******************************************)
procedure in3M;
var i : integer;
begin
    writeln('DANH SACH CAC CAN BO CO THU NHAP CAO:');
    for i := 1 to n do
       with ds[i] do
           if thunhap >= 3000000 then begin
             writeln('Ten:',ten);
             writeln('Tuoi:',tuoi);
             writeln('Thu nhap:',thunhap :0 :3);
           end;
end;

(*******************************************)
BEGIN
    nhap;
    in30;
    sxep;
    in3M;
    readln;
END.
```