Tipuri de date structurate

Problema 1

Să se citească un șir *A* de la tastatură, citirea șirului se termina la introducerea valorii 0. Să se construiască și să se tipărească șirul *B* de perechi (*element*, *frecvență*) care memorează frecvența de apariție a fiecărui element din șirul *A* (*element* reprezintă elementul din șirul *A*, iar *frecvență* reprezintă frecvența de apariție a acestuia). Să se elimine din șirul A cele mai frecvente 2 numere prime (în cazul în care mai mult de două numere au aceeași frecvență, se vor elimina aleator două dintre ele), apoi să se tipărească șirul rezultat pe ecran.

Exemple

- Pentru șirul A={1,2,3,4,5,6,1,2,3,1,2,3,4}

  Se va tipări B={(2,3),(3,3),(5,1)}

  Pe ecran se va tipări șirul A modificat: 1,4,5,6,1,1,4

- Pentru șirul A={1,2,4,6,8,10}

  Se va tipări B={(2,1)}

  Pe ecran se va tipări: 1,4,6,8,10

Se vor scrie subprograme pentru:

- citirea șirului *A*;
- determinare frecvență de apariție în șir pentru un număr dat;
- construirea șirului de frecvențe pentru toate numerele prime din șirul *A*;
- sortarea șirului *B* descrescător după frecvențe;
- verificarea dacă un număr este sau nu prim;
- eliminarea din șirul *A* a tuturor aparițiilor unei valori *v* date;
- afișarea șirului *B*;
- afișarea șirului *A*

Rezolvare

```
Program problema1;
const MAX_ELEMENTE = 100;
//Sirul de numere
type Sir =record
    elems: array [1..MAX_ELEMENTE] of integer;
    lg: integer;
end;

//pereche numar, numar de aparitii
type Frecventa = record
    nr:integer;
    nrAparitii:integer;
end;

//sirul de frecvente
type SirFrecventa = record
    elems: array [1..MAX_ELEMENTE] of Frecventa;
    lg:integer;
end;

var nrs:Sir;
var frecvente:SirFrecventa;
```

```
{
 Citeste un sir de numere de la tastatura
 returneaza sirul citit
}
function citesteNumere():Sir;
var rez:Sir; var n:integer;
begin
    rez.lg := 1;
    repeat
        writeln('Numarul:');
        readln(n);
        if n<>0 then
        begin
            rez.elems[rez.lg] := n;
            inc(rez.lg);
        end;
    until (n=0);
    dec(rez.lg);
    citesteNumere:=rez;
end;

{
  Tipareste sirul in consola
  nrs - sir de numere
}
procedure tiparesteSir(nrs:Sir);
var i: integer;
begin
    for i := 1 to nrs.lg do
    begin
        write(nrs.elems[i],',');
    end;
    writeln;
end;


{
  Tipareste sirul cu frecvente
  nrs - sir de numere
}
procedure tiparesteFrecvente(nrs:SirFrecventa);
var i:integer;
begin
    for i := 1 to nrs.lg do
        write('(',nrs.elems[i].nr,',',nrs.elems[i].nrAparitii,'),');
    writeln;
end;
{
   calculeaza numarul de aparitii in sir pentru numarul dat
   returneaza perechea numar, numar aparitii
}
function calculeazaFrecventa(nrs:Sir; nr:integer):Frecventa;
    var rez: Frecventa;
    var i:integer;
begin
    rez.nr := nr;
    rez.nrAparitii := 0;
    for i := 1 to  nrs.lg do
    begin
        if (nrs.elems[i] = nr) then
            inc(rez.nrAparitii);
    end;
    calculeazaFrecventa:=rez;
end;
```

```
{
  Varifica daca un numar e prim
  returneaza true daca numarul e prim
}
function ePrim(nr:integer):boolean;
var i :integer;
var aux:boolean;
begin
    aux:=true;
    if (nr <= 1) then
        aux:=false
    else
    if (nr <= 3) then
        aux:= true
    else if ((nr mod 2 = 0) or (nr mod 3 = 0))  then
        aux:= false;
    i := 5;
    while (i * i <= nr) do
    begin
        if ((nr mod i = 0) or (nr mod (i + 2) = 0)) then
            aux:=false;
        i := i + 6;
    end;
    ePrim:=aux;
end;
{
  Sorteaza sirul crescator dupa numarul de aparitii
  modifica sirul primit ca parametru (sirul devine sortat)
}
procedure sorteazaDupaFrecvente(var frecv:SirFrecventa);
var i,j:integer;
var temp:Frecventa;
begin
    for i := 1 to frecv.lg-1 do
        for j := i + 1 to frecv.lg do
            if (frecv.elems[i].nrAparitii < frecv.elems[j].nrAparitii) then begin
                temp := frecv.elems[i];
                frecv.elems[i] := frecv.elems[j];
                frecv.elems[j] := temp;
        end;
end;

{
  Verifica daca pentru numarul dat avem deja frecventa in sirul fvs
  return true daca in sirul fvs avem deja un element pentru numarl nr
}
function apare(fvs:SirFrecventa; nr:integer):boolean;
var i:integer;
var aux:boolean;
begin
    aux:= false;
    for i := 1 to fvs.lg do
        if (fvs.elems[i].nr = nr) then
            aux:= true;
    apare:=aux;
end;
```

```
{
 * Sirul de frecente pentru numerele prime
 * returneaza sir de frecvente
}
function determinaSirFrecventePrime(nrs:Sir):SirFrecventa;
    var rez:SirFrecventa;
    var i:integer;
begin
    rez.lg := 1;
    for  i := 1 to nrs.lg do
        if ( (not apare(rez, nrs.elems[i])) and (ePrim(nrs.elems[i])) then
        begin
            rez.elems[rez.lg] := calculeazaFrecventa(nrs, nrs.elems[i]);
            inc(rez.lg);
        end;
    dec(rez.lg);
    determinaSirFrecventePrime := rez;
end;

{
 * Sterge elementul de pe pozitie data
 * Modifica sirul, elimina elementul de pe pozitia poz
}
procedure stergeDePePozitie(var nrs:Sir; poz:integer);
var i:integer;
begin
    for i := poz to nrs.lg - 1 do
        nrs.elems[i] := nrs.elems[i + 1];
    dec(nrs.lg);
end;

{
 * Sterge din sir toate aparitiile numarului dat
 * Modifica sirul, elimina toate aparitiile lui nr
}
procedure stergeToateAparitiile(var nrs:Sir; nr:integer);
var poz:integer;
begin
    poz := 1;
    while (poz <= nrs.lg) do
        if (nrs.elems[poz] = nr) then
            stergeDePePozitie(nrs, poz)
        else
            inc(poz);
end;

begin
    nrs := citesteNumere();
    tiparesteSir(nrs);

    frecvente := determinaSirFrecventePrime(nrs);
    sorteazaDupaFrecvente(frecvente);
    tiparesteFrecvente(frecvente);

    stergeToateAparitiile(nrs, frecvente.elems[1].nr);
    if (frecvente.lg > 1) then
        //ma asigur ca am avut cel putin 2 elemente distincte in sir
        stergeToateAparitiile(nrs, frecvente.elems[2].nr);

    tiparesteSir(nrs);

end.
```

Exemple

Date de intrare	Rezultat
Numarul:1 Numarul:2 Numarul:3 Numarul:4 Numarul:3 Numarul:0	1,2,3,4,3, (3,2),(2,1), 1,4,
Numarul:4 Numarul:6 Numarul:8 Numarul:10 Numarul:0	4,6,8,10, 4,6,8,10,
Numarul:11 Numarul:6 Numarul:8 Numarul:11 Numarul:10 Numarul:11 Numarul:0	11,6,8,11,10,11, (11,3), 6,8,10,
Numarul:11 Numarul:12 Numarul:13 Numarul:15 Numarul:23 Numarul:23 Numarul:23 Numarul:0	11,12,13,15,23,23,23, (23,3),(13,1),(11,1), 11,12,15,

Problema 2

Se citeşte o matrice $A(n,m)=(a_{ij})_{i=\overline{1,n},j=\overline{1,m}}$ de numere naturale nenule, unde $1 \le n,m \le 100$, $1 \le a_{ij} \le 30000$. Să se scrie un program care elimină din A coloanele j având proprietatea că minimul şi maximul de pe coloana respectivă sunt numere *apropiate*. Se va forma un şir R cu numerele distincte eliminate, în ordine descrescătoare a cifrei lor maxime (numerele cu aceeaşi cifră maximă vor apărea pe poziţii consecutive în şir, fără să conteze ordinea lor). Spunem ca două numere naturale sunt *apropiate* dacă scrierile celor două numere în baza 10 conţin cel puţin 2 cifre distincte comune (Ex.: 1313 şi 33112 SUNT *apropiate*, iar 1231 şi 6333 NU sunt *apropiate*). La sfârşit se cere tipărirea matricei A şi a şirului R. Şirul R se va construi direct ordonat, fără ordonarea ulterioară a acestuia.

Exemplu

Pentru n=4, m=4 şi matricea

$$A=\begin{pmatrix} 15 & 4 & 156 & 58 \\ 13 & 18 & 240 & 37 \\ 18 & 220 & 131 & 69 \\ 131 & 83 & 330 & 13 \end{pmatrix}$$

Se va tipări

$A=\begin{pmatrix} 4 & 58 \\ 18 & 37 \\ 20 & 196 \\ 33 & 133 \end{pmatrix}$ şi de ex. R =(182, 1316, 156, 15, 240, 13, 330)

Se vor scrie subprograme pentru:

a). citirea unei matrice $A(n,m)$
b). verificarea dacă două numere sunt *apropiate*
c). verificarea dacă maximul şi minimul de pe o coloană j a matricei A sunt numere *apropiate*
d). eliminarea unei coloane j din matricea $A(n,m)$
e). inserarea unui număr în şirul R (conform cerinţelor problemei)
f). adăugarea elementelor de pe o coloană j din matricea A în şirul R
g). tipărirea unei matrice
h). tipărirea unui şir

```
program mat;
//sir de numere
type Sir = record
     elems:array[1..100] of integer;
     lg: integer;
end;

//matrice
type Matrice = record
    elems:array[1..100,1..100] of integer;
    linii: integer;
    coloane: integer;
end;

//sirul caracteristic
type ApareCifra = array[0..9] of boolean;
```

```
// Calculeaza vectorul caracteristic al unui numar
procedure caracteristic(a:integer;var apare:ApareCifra);
var i:integer;
    ultimaCifra:integer;
begin
   for i:=0 to 9 do
     apare[i] :=false;
   while (a>0) do begin
     ultimaCifra:= a mod 10;
     apare[ultimaCifra] := true;
     a:=a div 10;
   end;
end;

//returneaza cifra maxima a numarului nr
function cifraMaxima(nr:integer):integer;
  var apare:ApareCifra;
      max:integer;
begin
  caracteristic(nr,apare);
  max := 9;
  while (not (apare[max])) do
      max := max -1;
  cifraMaxima:=max;
end;
//adauga elementul in sir
//mentine sirul ordonat dupa cifra maxima
procedure adaugaSortat(var l:Sir; elem:integer);
   var cif,i,j:integer;
       duplicat:boolean;
begin
  cif:=cifraMaxima(elem);
  i:=1;
  while (i<=l.lg) and (cif<=cifraMaxima(l.elems[i])) do
  begin
     i:=i+1;
     if (l.elems[i]=elem) then duplicat:=true;
  end;
  if not(duplicat) then begin
     //inserez pe pozitia i
     for j:=l.lg+1 downto i+1 do
        l.elems[j]:=l.elems[j-1];
     l.elems[i]:=elem;
     l.lg:=l.lg+1;
  end;
end;


//tipareste sirul de numere
procedure tiparesteSir(l:Sir);
var i:integer;
begin
  for i:=1 to l.lg do
    write(l.elems[i],',');
  writeln;
end;

//citeste matrice
procedure citesteMat(var m:Matrice);
var i,j:integer;
begin
   write('Nr linii:');
   readln(m.linii);
   write('Nr coloane:');
   readln(m.coloane);
```

```
   for i:=1 to m.linii do begin
     for j:=1 to m.coloane do begin
        write('m[',i,'][',j,']=');
        readln(m.elems[i][j]);
     end;
   end;
end;

//tipareste matricea
procedure tiparesteMat(m:Matrice);
var i,j:integer;
begin
  writeln;
  for i:=1 to m.linii do begin
      for j:=1 to m.coloane do
         write(m.elems[i][j],',');
      writeln;
  end;
end;

//verifica daca a, b sunt apropiate
//returneaza true daca a si b au cel putin 2 cifre distincte in comun
function apropiate(a:integer; b:integer):boolean;
  var apareA,apareB:ApareCifra;
      comune,i:integer;
begin
  caracteristic(a,apareA);
  caracteristic(b,apareB);
  comune:=0;
  for i:=0 to 9 do
     if (apareA[i]) and (apareB[i]) then
         comune:=comune+1;
  apropiate:=comune>=2;
end;

//verifica daca minimul si maximul de pe coloana data sunt apropiate
//returneaza true daca min,max sunt apropiate
function minMaxApropiate(a:Matrice; coloana:integer):boolean;
  var min,max, i:integer;
begin
  min:=a.elems[1][coloana];
  max:=a.elems[1][coloana];
  for i:=2 to a.linii do begin
     if (a.elems[i][coloana]<min) then
         min:= a.elems[i][coloana];
     if (a.elems[i][coloana]>max) then
         max:=a.elems[i][coloana];
  end;
  minMaxApropiate:=apropiate(min,max);
end;

//elimina din matrice coloana specificata
//adauga elementele eliminate in sir
procedure eliminaColoana(var a:Matrice;coloana:integer;var l:Sir);
  var lin, col: integer;
begin
  for lin:=1 to a.linii do begin
     adaugaSortat(l,a.elems[lin][coloana]);
     for col:=coloana downto a.coloane-1 do
         a.elems[lin][col] := a.elems[lin][col+1];
  end;
  a.coloane:=a.coloane-1;
end;
```

```
//programul principal
var a:Matrice;
    r:Sir;
    col:integer;
begin
  citesteMat(a);
  tiparesteMat(a);
  r.lg:=0;
  for col:=a.coloane downto 1 do
     if (minMaxApropiate(a,col)) then
        eliminaColoana(a,col,r);
  tiparesteMat(a);
  tiparesteSir(r);
  readln;
end.
```