Capitolul 1

STRUCTURI DE DATE

1. ORGANIZAREA DATELOR

STUDIU DE CAZ ***Compania Eficient***

Pentru reducerea cheltuielilor cu amenajarea spațiilor comerciale, reclamă și personal, compania Eficient selectează tineri distribuitori, absolvenți de liceu. Oferta companiei este foarte atrăgătoare; de aceea, numărul solicitanților depășește numărul de posturi oferite (n). Selecția candidaților se face în ordinea înregistrării scrisorii de intenție și a CV-ului. Dosarele candidaților sunt păstrate într-un fișet, unul peste altul, în ordinea angajării. Periodic, compania trimite un angajat la cursuri de formare; este trimis întotdeauna, ultimul angajat (fig. 1). Angajații sunt plătiți în funcție de numărul de produse distribuite (vândute) zilnic. Săptămânal, managerul companiei ține evidența pe zile a produselor distribuite de fiecare angajat. În orice moment, managerul poate determina angajatul cu cea mai bună activitate într-o zi sau ziua în care a fost distribuit cel mai mic număr de produse. La sfârșitul săptămânii, după ce aplică bonusuri sau penalizări, managerul centralizează aceste date într-un registru de evidență a vânzărilor realizate de către fiecare angajat.

Managerul companiei dorește să prelucreze cu calculatorul datele pentru selecția candidaților, evidența angajaților și evidența vânzărilor.

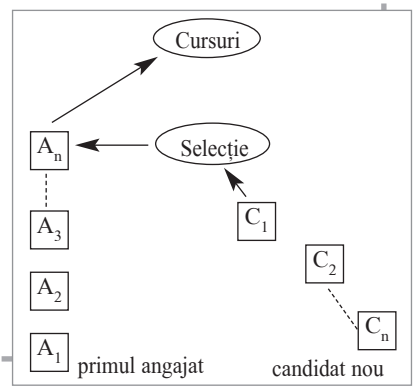

Figura 1

1.1. Analiza problemei

☑ datele despre candidați; din CV-ul fiecărui candidat vor fi reținute următoarele informații:
- numele,
- anul nașterii,
- media la examenul de bacalaureat;

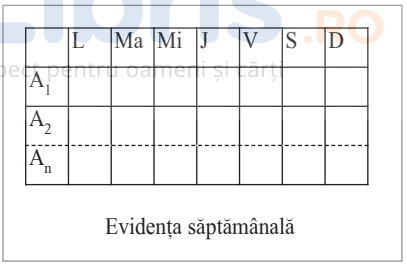

Figura 2

☑ selecția candidaților se face în ordinea înregistrării datelor personale;

☑ după examinarea unui candidat (selecție), datele acestuia nu mai sunt necesare; va intra în selecție candidatul următor;

☑ întotdeauna, datele unui nou candidat sunt așezate după datele ultimului candidat care așteaptă pentru selecție;

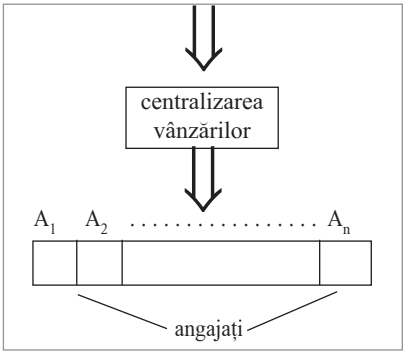

Figura 3

☑ datele angajaților sunt păstrate într-o ordine care să permită numirea rapidă a ultimului angajat în vederea trimiterii sale la cursuri;

☑ pentru evidența săptămânală a vânzărilor, se reține numărul de produse distribuite (vândute) zilnic de către fiecare angajat (fig. 2);

☑ pentru evidența centralizată a vânzărilor, totalul vânzărilor realizate într-o săptămână de fiecare angajat se adaugă la vânzările realizate de acel agajat până la momentul respectiv (fig. 3).

1.2. Soluția problemei

☑ problema propusă de managerul companiei Eficient necesită prelucrări matematice cu un grad mic de dificultate: centralizarea săptămânală, prin însumare, a valorilor reprezentând vânzările realizate de către fiecare angajat;

☑ datele specifice problemei trebuie organizate avantajos, astfel încât folosirea calculatorului să înlocuiască fișetele sau mapele în care sunt păstrate dosarele candidaților/angajaților și să respecte cerințele de așteptare sau prioritate specifice problemei.

1.3. Organizarea datelor

- Din analiza problemei urmărind *semnificația și complexitatea datelor*, rezultă două categorii de date:

☒ *date elementare*, spre exemplu: numărul de produse vândute de un angajat la un moment dat;

☒ *date grupate (structurate):*

– *date cu aceeași semnificație* (de același tip) – numite și *date omogene* –, spre exemplu: evidența săptămânală a vânzărilor pe zile și angajați, evidența vânzărilor pe angajați, datele despre toți angajații/candidații;

– *date cu semnificații diferite* – numite și *date neomegene* – spre exemplu, datele prin care este descrisă o persoană (candidat sau angajat): nume, an, medie.

De reținut!

O persoană este descrisă prin mai multe tipuri de date, ceea ce determină aspectul neomogen al grupului de date.

Candidații sau angajații formează grupuri de același tip – *persoană* – de unde rezultă aspectul omogen al acestui grup de date.

- Din analiza problemei urmărind *restricțiile de intrare-ieșire* (așteptare sau prioritate), după care sunt organizate datele dintr-un grup (structură), rezultă două categorii de date:

☒ date organizate după disciplina specifică unei *cozi* sau *fir de așteptare*; spre exemplu, candidații care așteaptă pentru selecție. Într-o astfel de structură, întotdeauna, un element nou este așezat (intră) după ultimul element. Dintr-o astfel de structură, iese, întotdeauna, primul element – în exemplul nostru, candidatul aflat „la rând".

☒ date organizate după disciplina specifică așezării obiectelor unul peste altul, în *stivă*; spre exemplu, dosarele angajaților. Într-o astfel de structură, întotdeauna, un element nou este așezat deasupra celorlalte, în *vârful* stivei. Dintr-o astfel de structură, iese, întotdeauna, elementul din *vârful* stivei – în exemplul nostru, va fi prelucrat dosarul ultimului angajat.

Concluzie

- organizarea datelor specifice unei probleme se poate face în locații de memorie independente – *date elementare* – sau în locații grupate – *structuri de date;*
- structurile de date pot fi:
  – *structuri omogene* (date cu aceeași semnificație/tip),
  – *structuri neomegene* (date cu semnificații/tipuri diferite);
- structurile omogene se numesc *tablouri;*
- într-un tablou, datele pot fi organizate astfel:

– după un singur criteriu – *tablouri unidimensionale* sau *vectori*; spre exemplu, tabloul pentru evidența vânzărilor realizate de fiecare angajat,

– după două criterii – *tablouri bidimensionale* sau *matrice*; spre exemplu, tabloul pentru evidența vânzărilor realizate zilnic (criteriul 1 – zilele săptămânii) de către fiecare angajat (criteriul 2 – angajații);

- într-un tablou unidimensional, datele pot fi organizate astfel încât să respecte o disciplină de intrare/ieșire de tip *coadă* sau *stivă*;
- structurile neomogene se numesc *articole* sau *înregistrări*; mai multe articole care descriu aceeași entitate (obiect, persoană etc.) reprezintă un grup de date cu aceeași semnificație și poate fi organizat într-o structură omogenă de tip tablou unidimensional (*vector de articole*).

TEME

1. Explicați deosebirea dintre analiza datelor din punct de vedere al complexității și analiza datelor din punct de vedere al regulilor de organizare (intrare/ieșire) într-o structură (grup de date).
2. Formulați un exemplu care să evidențieze diferența dintre datele omogene și datele neomegene.
3. Formulați un exemplu care să necesite organizarea după mai multe criterii a datelor cu aceeași semnificație.
4. Formulați un exemplu care să necesite organizarea datelor cu aceeași semnificație în structuri de tip *coadă*.
5. Formulați un exemplu care să necesite organizarea datelor cu aceeași semnificație în structuri de tip *stivă*.
6. Explicați disciplina structurii de date de tip coadă (FIFO – First Input First Output = *primul intrat, primul ieșit*).
7. Explicați disciplina structurii de date de tip stivă (LIFO – Last Input First Output = *ultimul intrat, primul ieșit).*
8. Formulați exemple de oganizare a datelor după alte reguli decât cele specifice structurilor de tip *coadă* sau *stivă*.

TEMĂ de GRUP

Densitatea straturilor geologice

*Un grup de geologi studiază densitatea straturilor geologice. În acest scop, s-a extras câte o probă pentru fiecare dintre cele **n** straturi geologice studiate. Fiecare probă este transmisă la un laborator specializat; pentru fiecare probă se fac mai multe teste, cel mult **m**. Întrucât nu există suficiente aparate, analiza de laborator durează. La sfârșitul activității, geologii păstrează probele în ordinea naturlă a straturilor geologice* (fig. 4)*. Rezultatele testelor se păstrează astfel încât să se poată determina, cu ușurință:*

– zăcământul la care s-au obținut aceleași valori la toate testele;

*– zăcământul cu cea mai mare densitate medie la cele **m** teste;*

*– zăcământul la care s-a obținut cea mai mare diferență între densitatea maximă și densitatea minimă din cele **m** teste.*

Cerințe:

1. Identificați formele de organizare a datelor specifice problemei propuse.

2. Fiecare membru al grupului descrie una dintre formele de organizare a datelor identificate (justificare, necesitate, proprietăți, prelucrări specifice și alte aspecte).

3. Grupul compune un scenariu pentru prelucrarea cu calculatorul a datelor specifice acestei probleme; scenariul poate fi prezentat narativ sau organizat pe secvențe (pași).

4. Grupul realizează o prezentare PowerPoint care să ajute la susținerea temei.

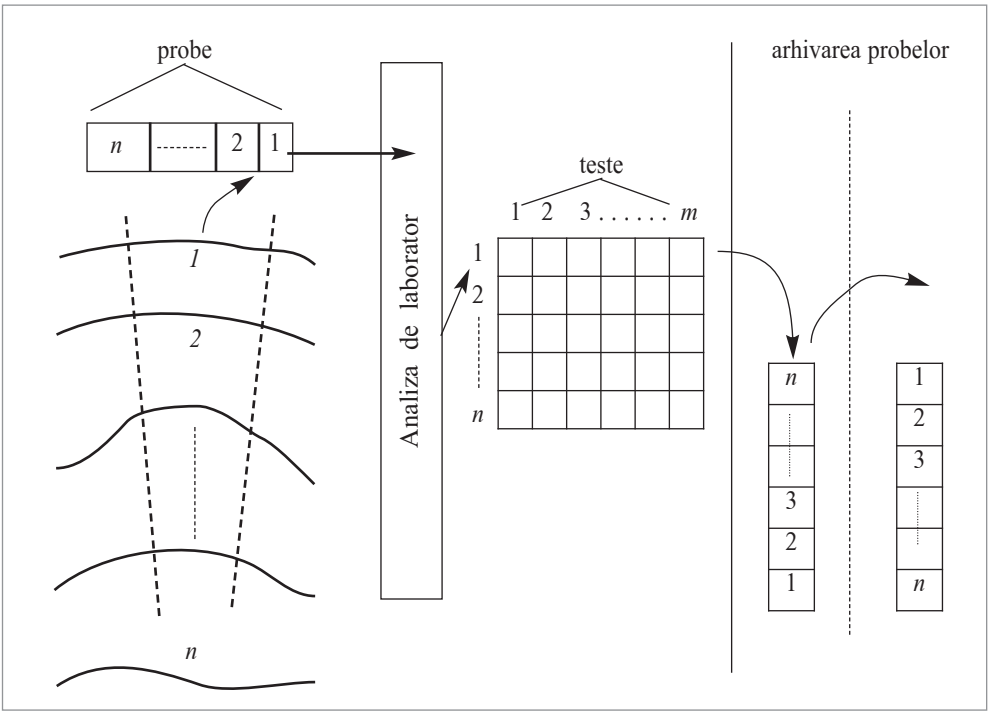

Figura 4

SUGESTIE DE PREZENTARE:

Prezentările pot fi expuse și analizate în laboratorul de Informatică; se va urmări corectitudinea rezolvării, creativitatea prezentării și eficiența lucrului în grup.

2. ORGANIZAREA DATELOR CU ACEEAȘI SEMNIFICAȚIE ÎN TABLOURI BIDIMENSIONALE

APLICAȚIA 1 FLOARE DE COLȚ

Membrii clubului „Floare de colț" participă la o tabără de vară, timp de o săptămână, într-o zonă montană greu accesibilă. Scopul lor este să înregistreze în fiecare zi temperatura aerului și să determine temperatura minimă, temperatura maximă și zilele în care s-au atins aceste temperaturi.

1. *Analiza problemei*

- *Date de intrare*
  – 7 valori reprezentând temperatura înregistrată în fiecare zi a săptămânii
- *Date de ieșire*
  – temperatura minimă (**tmin**);
  – temperatura maximă (**tmax**);
  – ziua/zilele în care s-au atins **tmin** și **tmax.**

2. *Organizarea datelor*

Datele de intrare au aceeași semnificație și vor fi înregistrate într-un vector (**T**) cu 7 elemente.

De exemplu:

T	18	15	12	14	18	15	12

tmin = 12 înregistrată miercuri și duminică
tmax = 18 înregistrată luni și vineri

3. *Raționamentul problemei*

Se înregistrează temperaturile zilnice în vectorul **T**. Se determină, printr-o singură parcurgere, valoarea minimă și cea maximă.

Pentru afișarea zilei/zilelor se mai fac două parcurgeri ale vectorului. Ziua va fi afișată ca indice (1, 2 etc.). Pentru afișarea zilei prin nume (luni, marți etc.) se recomandă introducerea structurii selective după **zi**.

4. *Reprezentarea algoritmului*

```
început temperaturi1
alocă T[7]
pentru zi = 1, 7 execută citește T[zi] sfârșit pentru
tmin ← T[1]
tmax ← T[1]
pentru zi = 2, 7 execută
    bloc
        dacă T[zi] < tmin atunci tmin ← T[zi]
        sfârșit dacă
        dacă T[zi] > tmax atunci tmax ← T[zi]
        sfârșit dacă
    sfârșit bloc
```

```
sfârșit pentru
scrie tmin
pentru zi = 1, 7 execută
    dacă T[zi] = tmin atunci scrie zi
    sfârșit dacă
sfârșit pentru
scrie tmax
pentru zi = 1, 7 execută
    dacă T[zi] = tmax atunci scrie zi
    sf dacă
sfârșit pentru
sfârșit temperaturi1
```

APLICAȚIA 2 FLOARE DE COLȚ

Măsurarea temperaturii aerului în zonele greu accesibile a fost foarte apreciată de ecologiști. Numărul voluntarilor dornici să participe la astfel de acțiuni a crescut. Instructorul clubului a organizat zece echipe care să măsoare temperatura zilnică în diverse zone de interes.

Prelucrarea valorilor înregistrate se va face astfel:

– se afișează temperatura minimă și temperatura maximă înregistrate în cele zece zone pe parcursul săptămânii;

– se afișează temperatura medie, pentru oricare dintre zone, la solicitarea celui interesat;

– se afișează zona în care s-a atins cea mai mică, respectiv cea mai mare temperatură, pentru orice zi a săptămânii solicitată de cel interesat.

1. *Analiza problemei*

Datele problemei au aceeași semnificație – temperatura zilnică –, dar se referă la zone diferite; de aceea, pentru organizarea lor se introduce și criteriul **zonă**. Rezultă un tablou (**T**) cu două dimensini: **zona** (linie) și **ziua** (coloană) (fig. 5).

T

Zona							
1						21	
2						18	
3	18	15	12	14	18	15	12
4						16	
5						17	
6						19	
7						20	
8						20	
9						17	
10						18	
	1	2	3	4	5	6	7

Ziua

Figura 5

Cum organizăm datele în tablouri

1. Dacă într-o problemă este necesară memorarea mai multor valori cu aceeași semnificație, aceste valori vor fi grupate într-un ansamblu de tip **tablou**.
2. În funcție de cerințele problemei și de semnificația datelor, acestea sunt organizate în tablouri cu o singură dimensiune, numite **vectori**, sau în tablouri cu două dimensiuni, numite **matrice**.
3. Elementele unui tablou se identifică printr-o adresă formată din numele tabloului și câte un indice pentru fiecare dimensiune.
4. La tablourile cu două dimensiuni, primul indice din adresă reprezintă linia, iar cel de-al doilea coloana tabloului.
5. Operațiile care se repetă pentru fiecare element din tablou pot fi grupate în structuri repetitive cu contor. Contorul generează chiar indicele de adresă.

3. IMPLEMENTAREA TABLOURILOR BIDIMENSIONALE

Pentru memorarea și prelucrarea datelor organizate ca tablouri bidimensionale, înainte de scrierea unui program trebuie să cunoaștem următoarele elemente:

– tipul elementelor din tablou (datele cu aceeași semnificație),

– numărul maxim de elemente din tablou (capacitatea tabloului).

capacitatea tabloului = nrmax _linii * nrmax _coloane

Aceste elemente rezultă din analiza problemei și sunt folosite de compilator pentru determinarea zonei de memorie alocată tabloului.

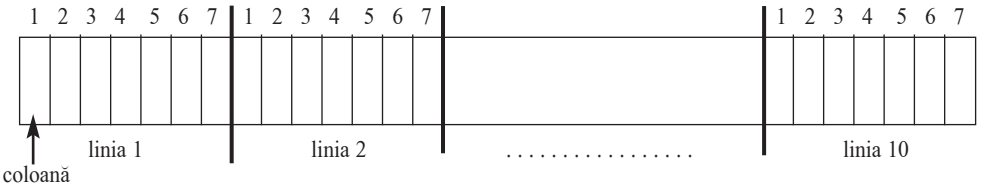

Figura 6 – Liniarizarea tabloului bidimensional

Tabloul ocupă în memoria calculatorului o „suprafață" (*array*) compusă din locații învecinate (adiacente). În fiecare locație este memorată valoarea unui element. Adresa locațiilor începe de la o *valoare de referință* specifică limbajului de programare (corespunzătoare primei linii) și se construiește prin valori succesive (corespunzătoare coloanelor de pe linie), pentru fiecare element din tablou. În memoria calculatorului, tabloul este liniarizat: elementele sunt memorate în locații adiacente, în ordinea liniilor (fig. 6). Adresarea unui element se face printr-o pereche de indici corespunzători liniei și coloanei pe care se află elementul respectiv.

În consecință, tipul variabilelor folosite ca indice de adresă trebuie să accepte valori în domeniile:

(1) [*valoare de referință, nrmax-linii*] pentru indicele de linie și

(2) [*valoare de referință, nrmax-coloane*] pentru indicele de coloană.

Adresarea elementelor se face, cel mai frecvent, prin două structuri repetitive cu contor, imbricate:

```
pentru  indice_linie = valoare-de referință la nr-linii execută
        pentru  indice_coloana = valoare-de referință la nr-coloane execută
          // prelucrarea elementelor din tablou în ordinea așezării pe linii
        sfârșit pentru
sfârșit pentru
```

Pentru indicele de linie sau indicele de coloană, se pot folosi tipuri diferite de date, ceea ce permite o referire asemănătoare cu cea întâlnită la jocul de șah; spre exemplu: T[2][C] reprezintă elementul de pe tabla de șah aflat pe linia 2 în coloana C (fig. 7).

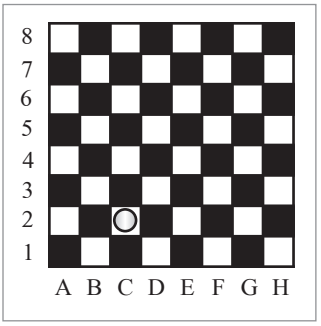

Figura 7

Elementele de sintaxă specifice tablourilor bidimensionale sunt prezentate în Tabelul 1.

***Tabelul* 1. Tablouri bidimensionale – elemente de sintaxă**

PASCAL	**C / C++**
Declararea tabloului	
var *tablou* **array** [1...*nl, 1..nc*] **of** *tipelement;*	*tipelement tablou*[*nl*][nc];
Declararea indicelui de adresă	
var *l,c:tipindice;*	*tipindice l,c;*
Adresarea unui element din tablou	
tablou [l,c] adresa de referință este 1.	*tablou* [*l*][*c*] adresa de referință este 0.
Exemplu: pentru tabloul *temperaturi* cu 100 de elemente de tip întreg distribuite pe 10 linii și 10 coloane: se inițializează cu zero elementele de pe primele 5 linii	

var *temperaturi* : **array** [1..10,1..10] **of integer**;
l,c: **byte;**
begin
for l:=1 to 5 do
for c:=1 to 10 do
temperaturi [l,c] =0;
end
	int *temperaturi* [10] [10];
	short l,c;
	{
	for (l=0; i< 5; i++)
	for (c=0; i< 10; i++)
	
	temperaturi [l][c]=0;
	}

Datele organizate în tablouri bidimensionale sunt prelucrate element cu element. Iată câteva dintre cele mai frecvente prelucrări:

- introducerea valorilor direct de la tastatură sau dintr-un fișier;
- afișarea valorilor pe ecran sau într-un fișier;
- verificarea unor proprietăți;
- compararea valorilor (aflarea elementului maxim sau minim);
- simularea unor situații reale.

Tabelul 2. **Operații la nivel de matrice**

PASCAL	C / C++
A. Introducerea valorilor în ordinea „pe linii”	

var A: array[1..10, 1..10] of integer;
l,c,m,n : byte;
begin
write (‘ numarul de linii=’); readln (m);
write (‘ numarul de coloane=’); readln (n);
for l:=1 to m do
  for c:=1 to n do
  readln (A[l,c]);
end;
	int A [10] [10] ;
	int l,c,m,n ;
	{
	cout<<” numarul de linii=” ; cin>>m;
	cout<<” numarul de coloane=” ; cin>>n
	for (l=0; l< m; l++)
	  for (c=0; c< n; c++)
	  cin >>A[l][c];
	}

B. Afișarea valorilor în ordinea „pe coloane” se păstrează declarațiile de la secvența A	

begin
for c:=1 to n do
 begin
  writeln;
  for l:=1 to m do
  write (A[l,c], ‘ ’);
  end;
end;
	for (c=0; c< n; c++)
	{
	  cout<<endl;
	  for (l=0; l< m; l++)
	  cout <<A[l][c]<< “ ”;
	}

C. Determinarea elementului minim de pe o linie oarecare, k se păstrează declarațiile de la secvența A	

var k:byte; min: integer;
begin
  write (‘specificati linia=’); readln (k);
 min := A[k,1];
 for c:=2 to n do
  if A[k,c] < min then
  min:= A[k,c];
write (‘min. de pe linia’, k, ‘=’, min);
end;
	int k, min;
	{
	cout<<” specificati linia=” ; cin>>k;
	min=A[k][0];
	  for (c=1; c< n; c++)
	  if A[k][c] < min
	  min= A[k][c];
	cout<<”min. de pe linia”<<k<<”=”<<min;
	}

PASCAL	C / C++
D. Simularea unor situații reale.	
Exemplu: memorarea relațiilor de prietenie dintre membrii unui grup de n persoane (n<=10)	

var R : array [1..10, 1..10] of byte;
l,c,i,j,d,n : byte;
begin
write (‘ numarul de persoane=’);
readln (n);
d:=1;
repeat
write (‘prietenie intre: ’); readln (i,j);
{relatia de pritenie este reciproca}
R[i,j] :=1; R[j,i] :=1;
write (‘mai sunt prieteni? ’);
readln (d);
{ da (d=1)/ nu (d=0)}
until d=0;
for l:=1 to n do
begin
writeln;
for c:=1 to n do
write (R[l,c], ‘ ’);
end;
end;
	int R[10] [10] ;
	int l,c,i,j,d=1,n ;
	{
	cout<<” numarul de persoane=” ; cin>>n;
	while (d)
	{
	cout<<”prietenie intre:” ; cin>>i>>j;
	// relatia de prietenie este reciproca
	R[i][j]=1; R[j][i]=1;
	cout<<” mai sunt prieteni?” ; cin>>d;
	// da (d=1)/ nu (d=0)
	}
	for (l=0; l< n; l++)
	{
	cout<<endl;
	for (c=0; c< n; c++)
	cout>>R[l][c]<<” ”;
	}
	}	

Reluăm aplicația Floare de colț

Din analiza problemei, a reieșit necesitatea organizării datelor într-un tablou bidimensional. Continuăm rezolvarea problemei.

2. *Raționamentul problemei*

Pentru determinarea temperaturii minime și a temperaturii maxime înregistrate în cele zece zone, pe parcursul săptămânii, se prelucrează toate elementele din tablou.

Pentru determinarea temperaturii medii, **tmed**, se solicită zona și se prelucrează doar elementele de pe linia corespunzătoare.

Pentru determinarea zonei în care s-a atins temperatura minimă sau maximă, se solicită ziua și se prelucrează doar coloana corespunzătoare.

```
începput temperaturi2
alocă T[10, 7]
pentru zona = 1, 10 execută
    pentru zi = 1, 7 execută
        citește T[zona, zi]
    sfârșit pentru
sfârșit pentru
tmin ← T[1, 1]
tmax ← T[1, 1]
pentru zona = 1, 10 execută
    pentru zi = 1, 7 execută
        dacă T[zona, zi] < tmin
            atunci tmin ← T[zona, zi]
        sfârșit dacă
        dacă T[zona, zi] > tmax
            atunci tmax ← T[zona, zi]
        sfârșit dacă
    sfârșit pentru
sfârșit pentru
scrie tmin
scrie tmax
scrie "ce zonă vă interesează?"
citește zona
tmed ← 0
pentru zi = 1, 7 execută
    tmed ← tmed + T[zona, zi]
sfârșit pentru
tmed ← tmed/7
scrie zona, tmed
scrie "ce zi vă interesează?"
citește zi
tmin ← T[1, zi]
zmin ← 1
tmax ← T[1, zi]
zmax ← 1
pentru zona = 2, 10 execută
    bloc
        dacă T[zona, zi] < tmin
            atunci
                bloc
                    tmin ← T[zona, zi]
                    zmin ← zona
                sfârșit bloc
        sfârșit dacă
        dacă T[zona, zi] > tmax
            atunci
                bloc
                    tmax ← T[zona, zi]
                    zmax ← zona
                sfârșit bloc
        sfârșit dacă
    sfârșit bloc
sfârșit pentru
scrie zmin, tmin, zi
scrie zmax, tmax, zi
sfârșit temperaturi2
```

TEME

1. La cabinetul medical, se calculează înălțimea medie a tuturor băieților din școală. Precizați care este forma de organizare a datelor corespunzătoare acestei situații.

2. La cabinetul medical, se calculează înălțimea medie a tuturor băieților din școală, pe grupe de vârstă (ani de studiu). Precizați care este forma de organizare a datelor corespunzătoare acestei situații.

3. La cabinetul medical, se calculează înălțimea medie a tuturor băieților din școală pe grupe de vârstă (ani de studiu), pentru fiecare clasă în parte (9A,..., 12 C,...). Precizați care este forma de organizare a datelor corespunzătoare acestei situații.

4. Administratorul unui bloc cu 12 etaje și 20 de apartamente pe etaj calculează cheltuielile de întreținere, în funcție de numărul de persoane din fiecare apartament (toate apartamentele au același număr de camere).

Realizați un program care să răspundă următoarelor cerințe:

a) înregistrarea numărului de persoane din fiecare apartament, pe etaje, de la parter până la ultimul etaj;

b) afișarea numărului de persoane din fiecare apartament, pe etaje, de la ultimul etaj până la parter;

c) cunoscând numărul unui apartament, introdus de la tastatură, să se afișeze etajul la care se află apartamentul și numărul de persoane care locuiesc în apartamentul respectiv.

5. Se consideră tabloul **M** cu următoarele elemente:

1	2	3	4
5	6	7	8
9	10	11	12

Precizați ce valori afișează secvența de mai jos:

```
pentru c = 1, 3 execută
    pentru l = 1, 2 execută
        scrie M[l, c]
    sfârșit pentru
sfârșit pentru
```

a) 1, 5, 9, 2, 6, 10, 3, 7, 11; b) 1, 2, 3, 5, 6, 7; c) 1, 5, 2, 6, 3, 7.

6. Ce realizează următoarea secvență de operații:

```
alocă M[10, 10]
pentru i = 1, 10 execută
    M[i, i] ← i
sfârșit pentru
```

a) atribuie valori de la 1 la 10 elementelor din vectorul **M**;

b) atribuie fiecărui element de pe diagonala matricei **M** o valoare egală cu linia pe care aceasta se află;

c) atribuie valori de la 1 la 10 primelor 10 elemente din matricea **M**.

7. Care dintre următoarele secvențe de operații memorează în colțurile matricei **M** valori citite de la tastatură; matricea are 5 linii și 5 coloane.

a)
```
pentru i = 1, N execută
    pentru j = 1, N execută
        citește M[i, j]
    sfârșit pentru
sfârșit pentru
```

b)
```
pentru i = 1, 5 execută
    pentru j = 1, 5 execută
        dacă (i = 1) și (j = 5)
            atunci citește M[i, i]
            altfel citește M[j, j]
        sfârșit dacă
    sfârșit pentru
sfârșit pentru
```

c)
```
l ← 1
c ← 5
pentru x = 1, 4 execută
    citește M[l, c]
    c ← l
    l ← c
sfârșit pentru
```

d)
```
l ← 1
c ← 5
citește M[l, l], M[l, c], M[c, l], M[c, c]
```

4. TABLOURI BIDIMENSIONALE – CAZURI PARTICULARE

Matrice pătrată

Rezolvarea problemelor cu calculatorul necesită găsirea soluțiilor de organizare și memorare a datelor, astfel încât acestea să păstreze semnificațiile reale atât din punct de vedere al valorilor proprii, cât și al relațiilor cu alte date. Spre exemplu, dacă membrii unui grup (persoane) împrumută bani unii de la alții, interesează atât suma de bani primită/datorată, cât și cine/de la cine a împrumutat. În acest caz, dacă în grup sunt *n* persoane, un tablou bidimensional, *G*, cu *n* linii și *n* coloane, este suficient pentru păstrarea atât a valorilor împrumutate, cât și a relațiilor de împrumut (fig. 8).

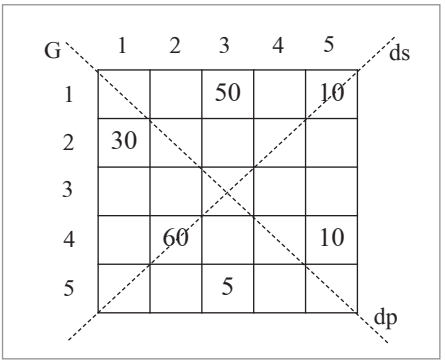

Figura 8

Fiecare element din tablou reprezintă valoarea unui împrumut; liniile și coloanele reprezintă persoanele din grup care dau bani unei alte persoane sau primesc bani de la altă persoană din grup. Fie *i* și *j* două persoane: *G[i,j]* reprezintă suma de bani pe care i-a împrumutat-o lui *j,* adică suma de bani pe care *j* a primit-o de la *i*. Pentru exemplul din figura 8, *G[4,2]* reprezintă suma de 60 lei pe care persoana 4 a împrumutat-o persoanei 2.

Tabloul folosit are o particularitate: numărul de linii este egal cu numărul de coloane, de aceea este numit tablou pătrat sau, mai simplu, *matrice pătrată.*

Într-o matrice pătrată deosebim următoarele elemente specifice (fig. 8):
– diagonala principală (*dp*);
– diagonala secundară (*ds*);
– triunghiurile formate de cele două diagonale;
– direcțiile paralele cu fiecare dintre cele două diagonale.

Matrice binară

Reluăm situația grupului de persoane care împrumută bani unii de la alții, dar urmărim numai relația de împrumut: cine/de la cine a primit bani. În acest caz, nu se mai păstrează valoarea împrumutului. Fie *i* și *j* două persoane: *G[i,j]* are semnificația *i* a împrumutat bani lui *j* echivalent cu *j* a primit bani de la *i* (fig. 9). Elementele matricei nu pot avea decât două valori alese convențional (spre exemplu 0 sau 1), cu semnificația:

G	1	2	3	4	5
1	0	0	1	0	1
2	1	0	0	0	0
3	0	0	0	0	0
4	0	1	0	0	1
5	0	0	1	0	0

Figura 9

$$G[i,j] = \begin{cases} 1 & \text{dacă } i \text{ împrumută bani lui } j \\ 0 & \text{dacă } i \text{ nu împrumută bani lui } j \end{cases}$$

Matricea ale cărei elemente au valori în mulțimea {0,1}, cu semnificații logice complementare, se numește *matrice binară.*

Matrice simetrică

Dacă în grupul de *n* persoane urmărim relațiile de prietenie, acestea pot fi memorate tot într-o matrice binară ale cărei elemente au următoarea semnificație:

$$P[i,j] = \begin{cases} 1 \text{ dacă } i \text{ este prieten cu } j \\ 0 \text{ dacă } i \text{ nu este prieten cu } j \end{cases}$$

Prietenia este o relație reciprocă; acest aspect se regăsește în proprietatea de *simetrie* a matricei binare asociată grupului de persoane: *P [i, j] = P [j, i]* (fig. 10).

P	1	2	3	4	5
1			1	1	
2			1		1
3	1	1		1	
4	1		1		1
5		1		1	

Figura 10

Matricea punctelor unui plan

Pe ecranul monitorului, în modul de lucru text, caracterele sunt afișate pe rânduri, de sus în jos, de la stânga la dreapta, pe fiecare rând. Se poate spune că ecranul monitorului este o suprafață plană ale cărei puncte sunt distribuite pe linii și coloane (cel mai frecvent, 24 de linii și 80 de coloane). În acest exemplu, regăsim modelul bidimensional de organizare a datelor: oricărei suprafețe plane îi poate fi asociată o *matrice a punctelor*. Valorile atribuite elementelor din matrice au semnificația specifică problemei modelate. Pentru exemplul suprafeței-ecran, dacă elementele matricei sunt de tip caracter, în matrice poate fi reținut textul de pe un ecran.

Un alt exemplu: o imagine – fotografie – este tot o reprezentare în plan. Punctele planului formează obiecte distincte, dacă sunt evidențiate diferit de la un obiect la altul, prin culoare. Dacă toate punctele unei imagini (suprafață) sunt colorate la fel, imaginea este formată dintr-un singur obiect.

Oricărei imagini îi poate fi asociată o matrice a punctelor; valorile atribuite elementelor din matrice au semnificația culorii fiecărui punct. În figura 11 este reprezentată matricea asociată unei imagini, în care s-au folosit 3 coduri de culori cu semnificația: 0 – alb, 1– negru, 2 – roșu.

Cu ajutorul matricelor de tip plan, pot fi modelate și situații de joc: așezarea pieselor pe tabla de șah, configurația unui labirint, „X și O", „avioane" și altele.

0	0	0	1	0	0	0	0
0	1	1	1	1	1	0	0
1	2	2	2	2	0	0	0
1	2	2	2	1	0	0	0
1	2	2	1	2	0	0	0
0	1	1	2	1	1	1	0
0	0	1	2	1	2	1	0
0	1	1	1	0	1	1	0

Figura 11

TEME

1. Determinați proprietățile elementelor aflate pe diagonala principală, într-o matrice pătrată. Scrieți un program pentru afișarea acestor elemente.

2. Determinați proprietățile elementelor aflate pe diagonala secundară, într-o matrice pătrată. Scrieți un program pentru afișarea acestor elemente.

3. Scrieți un program care să verifice dacă o matrice pătrată este simetrică.

4. Formulați un exemplu de problemă care să necesite organizarea datelor într-o matrice binară.

5. Formulați un exemplu de problemă care să necesite organizarea datelor într-o matrice de tip plan.

6. Determinați condițiile de amplasare pe tabla de șah a două piese de joc – dame –, astfel încât acestea să nu se atace. (Indicație: două piese de șah – dame – se atacă dacă sunt amplasate pe aceeași linie, pe aceeași coloană sau pe aceeași diagonală.)

7. Construiți tabloul de vecinătate pentru țările situate pe harta din figura 12.

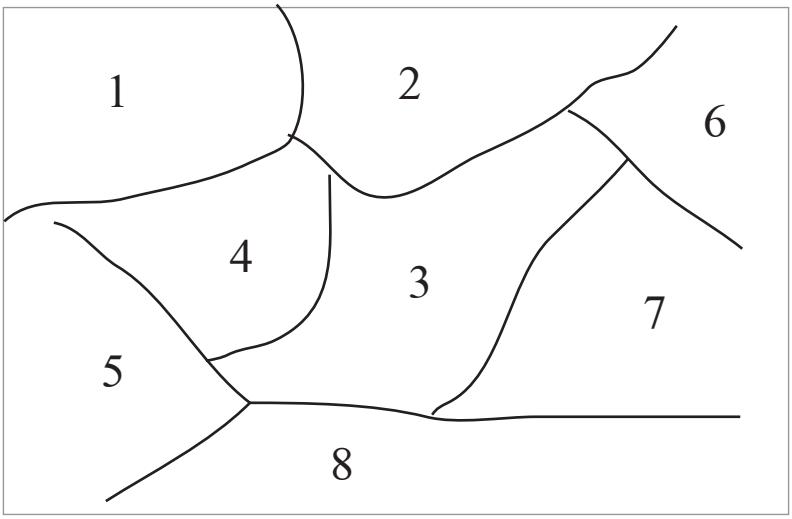

Figura 12

8. Să se verifice dacă o matrice pătrată este „tablou magic”. Într-un tablou magic, suma elementelor de pe oricare linie este egală cu suma elementelor de pe oricare dintre coloane precum și cu suma elementelor de pe oricare dintre diagonale.

Exemplu:

3	2	7
8	4	0
1	6	5

5. PRELUCRAREA TABLOURILOR BIDIMENSIONALE

5.1. Localizarea elementelor cu aceeași proprietate

Formațiuni geografice

Se dorește determinarea configurației unui teren dreptunghiular după formațiunile geografice din Tabelul 3 și figura 13.

Analiza terenului se face prin secționarea acestuia pe orizontală și pe verticală.

Punctele aflate la intersecția dintre secțiunile orizontale și secțiunile verticale sunt cotate față de nivelul mării; cotele sunt valori numerice întregi și pozitive.

*Terenul este secționat prin **n** secțiuni orizontale și **m** secțiuni verticale.*

O formațiune geografică este formată din cel puțin trei puncte.

Tabelul 3. **Formațiuni geografice**

FORMAȚIUNEA GEOGRAFICĂ	
Pantă	a)
Râpă	b)
Deal	c)
Vale	d)
Platou	e)
Teren denivelat	f)
Punct de tip *șa_xy*: punct aflat la cota maximă pe secțiunea orizontală *x* și la cota minimă pe secțiunea verticală *y*; se poate defini și punct de tip *șa_yx*	g)

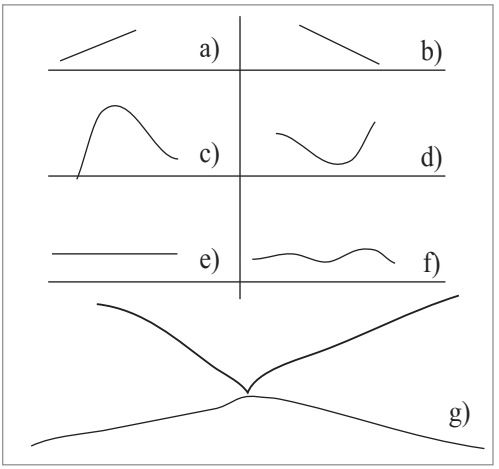

Figura 13

Exemplu:

În Tabelul 4 este reprezentat un teren pe care s-au făcut patru secțiuni orizontale și șapte secțiuni verticale.

Tabelul 4. **Înregistrarea datelor într-un teren secționat**

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	25	72	69	69	69	73	40
2	20	40	50	56	30	19	100
3	15	30	35	40	39	20	10
4	10	55	0	60	42	50	19

Rezultatele analizei pe secțiuni sunt prezentate în tabelul Tabelul 5.

Tabelul 5. **Analiza pe secțiuni**

Secțiuni orizontale		Secțiuni verticale	
Numărul secțiunii	**Formațiunea geografică**	**Numărul secțiunii**	**Formațiunea geografică**
1	Platou la cota 69	1	Râpă
2	Deal cu vârf la cota 56	2	Vale cu punct minim la cota 30
3	Deal cu vârf la cota 40	3	Râpă
4	Teren denivelat	4	Vale cu punct minim la cota 40
		5	Teren denivelat
		6	Vale cu punct minim la cota 19
		7	Teren denivelat
Puncte de tip șa_xy: punctul de coordonate [3,4] la cota 40			

Sugestie de rezolvare

Pentru determinarea formațiunilor geografice, cotele vor fi înregistrate într-un tablou bidimensional cu semnificația: linii – secțiuni orizontale, coloane – secțiuni verticale.

Pentru fiecare secțiune (linie sau coloană) se va studia monotonia șirurilor de valori (cotele înregistrate pe o linie sau pe o coloană) – Tabelul 6.

Rezolvarea problemei conduce la determinarea elementelor cu aceeași proprietate dintr-un tablou bidimensional.

Tabelul 6. **Monotonia șirurilor de valori**

Formațiunea geografică	Monotonia șirului de valori
Pantă/ Râpă	Șir crescător/ descrescător
Deal	Există un punct de tip vârf, astfel încât toate punctele dispuse la stânga acestuia formează un șir monoton crescător, iar toate punctele dispuse la dreapta acestuia formează un șir monoton descrescător.
Vale	Există un punct de cota minimă, astfel încât toate punctele dispuse la stânga acestuia formează un șir monoton descrescător, iar toate punctele dispuse la dreapta acestuia formează un șir monoton crescător.
Platou	Există cel puțin trei puncte consecutive aflate la aceeași cotă.

Pentru exemplificarea rezolvării, prezentăm, în pseudocod, secvența prelucrărilor pentru determinarea formațiunii de tip *platou* și a punctelor de tip *șa_xy* .

Determinarea formațiunii de tip platou

```
început platou
// căutarea unei formațiuni platou pe linia k
//semnificația variabilelor de lucru
//M[100,100]  tabloul cotelor cu  m secțiuni verticale
// lp  lungimea platoului
// cp  cota platoului
// inițalizare lungime platou
lp←1
cp←M[k,1]
pentru c=2 la m execută
// se verifică dacă punctul M[k,c] aparține platoului
    dacă  M[k,c] =cp
            atunci lp ← lp+1   // crește  lungimea platoului
            altfel
                dacă  lp>=3     // există platou la copta cp
                    atunci  scrie platou la cota cp
                sfârșit dacă
// inițializări pentru determinarea unui nou platou
            lp←1
            cp←M [k,c]
    sfârșit dacă
sfârșit pentru
// se verifică dacă linia  k se termină cu platou
dacă lp>=3
   atunci  scrie platou la cota cp
sfârșit dacă
sfârșit platou
```

Determinarea punctelor de tip șa_xy

- Raționamentul de rezolvare

Pasul 1

Se parcurge tabloul pe linii: pentru fiecare linie, se determină elementul maxim și se păstrează coloana acestuia în vectorul *max_linii*.

Pentru exemplul dat, vectorul *max_linii* are următoarele valori: 6, 7, 4, 4.

Pasul 2

Se parcurge tabloul pe coloane; pentru fiecare coloană, se determină elementul minim și se păstrează linia acestuia în vectorul *min_coloane*.

Pentru exemplul dat, vectorul *min_coloane* are următoarele valori: 4, 3, 4, 3, 2, 2, 3.

Pasul 3

Se parcurge vectorul *max_linii*: în variabila de lucru *cmax,* se reține valoarea unui element *max_linii[k]*

$$cmax = max\_linii[k]$$

se verifică proprietatea de punct șa_xy:

$$min\_colane[cmax] = k$$

Pentru exemplul dat, urmărim datele din Tabelul 7.

Există punct șa_xy pe linia 3, coloana 4.

Tabelul 7. **Determinarea punctului șa_xy**

linia	Cmax	min-coloane[cmax]
1	6	2
2	7	3
3	4	3
4	4	3

- Secvența pseudocod a prelucrărilor pentru determinarea punctelor de tip *șa_xy*:

```
începeut   punct_șa_xy
// căutarea unui punct șa_xy
//semnificația variabilelor de lucru
//M[100,100]  tabloul cotelor cu  m secțiuni orizontale  și n secțiuni verticale
// max_linii [100]  vector  cu m elemente în care se reține coloana  elementului maxim de pe
        fiecare linie
// min_coloane[100]  vector  cu n elemente în care se reține linia  elementului minim de pe
        fiecare coloană
//max    valoarea maximă   pe o linie;  cm coloana pe care se află  max
//min    valoarea minimă   pe o coloană;  lm linia  pe care se află  min

// se determină elementul maxim de pe fiecare linie
pentru k=1 la m execută
  max ← M[k,1]
  cm  ← 1
  pentru c=2  la  n execută
   dacă  M[k,c] >max
              atunci
                    bloc
                    max← M[k,c]
                    cm ← c
                   sfârșit bloc
    sfârșit dacă
 sfârșit pentru
```

```
// în linia k, elementul maxim se află pe coloana cm
  max_linii[k] ← cm
sfârșit pentru

// se determină elementul minim de pe fiecare coloană
pentru c=1 la n execută
  min ← M[1,c]
  lm ← 1
  pentru k=2 la m execută
   dacă M[k,c] <min
            atunci
                 bloc
                 min ← M[k,c]
                 lm ← k
                sfârșit bloc
    sfârșit dacă
 sfârșit pentru
// în coloana c, elementul minim se află pe linia lm
min_coloane[c] ← lm
sfârșit pentru
// se parcurge vectorul max_linii
pentru k=1 la m execută
 cmax ← max_linii [k]
// se verifică proprietatea de punct șa_xy
 dacă min_coloane[cmax]=k
            atunci
                    scrie punct sa de coordonate k, cmax
 sfârșit dacă
sfârșit pentru
sfârșit punct _sa_xy
```

TEME

1. Construiți exemple numerice pentru următoarele formațiuni geografice:

a) platou pe coloana 3, c) râpă pentru coloana 5, e) punct șa_xy,
b) deal pentru linia 1, d) vale pentru linia 4, f) punct șa_yx.

2. Construiți expresii pentru relațiile de monotonie corespunzătoare fiecărei formațiuni geografice.

3. Codificați, în limbajul de programare studiat, secvența pseudocod pentru determinarea formațiunii geografice de tip platou.

4. Codificați, în limbajul de programare studiat, secvența pseudocod pentru determinarea punctelor de tip șa_xy.

5. Realizați, în limbajul de programare studiat, un program pentru determinarea punctelor de tip șa_yx.

6. Scrieți secvențele de instrucțiuni pentru determinarea următoarelor formațiuni geografice:
a) râpă, b) deal, c) vale.

7. Realizați și testați programul pentru derminarea configurației unui teren ale cărui coordonate și cote se citesc din fișierul *teren.in* cu următoarea structură:
– pe prima linie valorile *n* și *m* reprezentând: *n* numărul de secțiuni orizontale și *m* numărul de secțiuni verticale;
– pe următoarele *n* linii câte *m* valori reprezentând cotele aflate pe fiecare secțiune orizontală.

5.2. Prelucrarea elementelor distribuite pe aceleași direcții (linii, coloane, diagonale)

Aranjamente florale

Un grădinar are mai multe soiuri de plante pe care dorește să le planteze atât în aranjamente clasice, cât și în forme noi; spre exemplu, în locul rondurilor, grădinarul vrea să compună careuri florale. Întrucât timpul de creștere și înflorire al plantelor nu poate fi întârziat, grădinarul și-a propus să testeze modelele cu calculatorul.

Analiza problemei

Pentru rezolvarea cu calculatorul, soiurile de plante decorative vor fi codificate. În Tabelul 8 se prezintă un exemplu de codificare.

Tabelul 8. **Codificarea plantelor decorative**

Planta decorativă/culoare	Codul plantei
Iarbă/verde	1
Trandafir imperial/roșu	2
Narcise/albe	6
Narcise/galbene	7
Crizanteme/mov	8
Crizanteme/albe	9
Tuia/verde	11

Careul pe care va fi probat modelul este format din *n***m* sau *n***n* puncte; în fiecare punct, poate fi sădită o plantă. În Tabelul 9 sunt prezentate câteva modele după care se vor testa aranjamentele florale.

Tabelul 9. **Modele pentru aranjamente florale**

Denumirea modelului	Exemplu de model
Brazde paralele orizontale	Decor de primăvară cu narcise albe și galbene: 6 6 6 6 6 7 7 7 7 7 6 6 6 6 6
Triunghi	Decor de primăvară cu narcise albe și galbene; aleea centrală (diagonală) cu gazon: 1 7 7 7 7 6 1 7 7 7 6 6 1 7 7 6 6 6 1 7 6 6 6 6 1
Diagonale paralele cu diagonala principală	Decor cu trandafiri imperiali și tuia plantați pe direcții paralele cu aleea centrală (diagonală): 1 2 11 2 11 2 1 2 11 2 11 2 1 2 11 2 11 2 1 2 11 2 11 2 1
Spirală	Decor de toamnă cu crizanteme mov și albe: 8 8 8 8 8 8 8 9 9 9 9 8 8 9 8 8 9 8 8 9 8 8 9 8 8 9 9 9 9 8 8 8 8 8 8 8

Sugestie de rezolvare

Careurile florale vor fi generate într-un tablou bidimensional, G, cu *n* linii și *m* coloane, sau *n* linii și *n* coloane, în funcție de model. Pentru fiecare model, se determină adresele punctelor în care vor fi sădite plantele și se înregistrează la aceste adrese codul plantei corespunzător modelului.

Rezolvarea problemei conduce la umplerea matricei cu valori dispuse pe direcții specifice modelului: linii, coloane, triunghiuri, diagonale, spirală.

În Tabelul 10 sunt prezentate secvențele pseudocod pentru generarea adreselor de umplere corespunzătoare modelelor prezentate.

Tabelul 10. **Secvențe pseudocod pentru generarea adreselor de umplere**

Modelul floral	Direcțiile de umplere	Generarea adreselor

Brazde orizontale	Linii	// brazda de tip linie
		// linia i
		pentru c=1 la m **execută**
		G[i,c] ← cod
		sfârșit pentru

Triunghi	Triunghi stânga	//se lucrează cu o matrice pătrată **n*n**
		// pe linii **i** și coloane **j**
		pentru i=2 la n **execută**
		pentru j=1 la i-1 **execută**
		G[i,j] ← cod
		sfârșit pentru
		sfârșit pentru

Paralele la diagonala principală	Se pot forma n-2 direcții **(d)** paralele cu diagonala principală și situate deasupra acesteia	//se lucrează cu o matrice **n*n**
		// pe linii **i** și coloane **j**
		// pe diagonalele **d**
		
		pentru d=1 la n-2 **execută**
		pentru i=1 la n-d **execută**
		G[i,i+d] ← cod
		sfârșit pentru
		sfârșit pentru

Spirală	Se parcurg cadranele de la exterior – primul cadran-**p**, spre interior, ultimul cadran-**u**	u p←1 u←n
		repetă
		// se parcurge prima linie, **p**, din cadran,
		//de la stânga la dreapta
		pentru j=p **la** u **execută**
		G[p,j] ← cod
		sfârșit pentru
		// se parcurge ultima coloană, **u**, din cadran,
		// de sus în jos
		pentru i=p+1 **la** u **execută**
		G[i,u] ← cod
		sfârșit pentru
		//se parcurge ultima linie, **u**, din cadran

```
//de la dreapta la stânga
pentru j=u -1 la p execută
 G[u,j] ← cod
 sfârșit pentru
// se parcurge prima coloană, p, din cadran,
// de jos în sus
pentru i=u - 1 la p - 1 execută
 G[i,p] ← cod
 sfârșit pentru

// se pregătesc valorile p și u
// pentru cadranul următor
p← p+1
u←u-1
// se testează dacă se mai pot forma cadrane
până când p>u
```

TEME

1. Construiți expresii pentru relațiile care definesc direcțiile de umplere pentru fiecare dintre modelele propuse.

2. Scrieți secvențele de instrucțiuni pentru generarea fiecărui model.

3. Realizați și testați programul pentru generarea fiecărui model floral (pentru fiecare model se va scrie un program).

4. Pentru atractivitatea prezentării, realizați un program care să afișeze modelul floral colorat, corespunzător culorii de cod. (Indicație: modul de lucru text permite setarea atributului de culoare atât pentru fond, cât și pentru text.)

5. Realizați un program care să permită utilizatorului (grădinarul) să aleagă, pe rând, oricare dintre modelele oferite (program cu meniu – fig. 14).

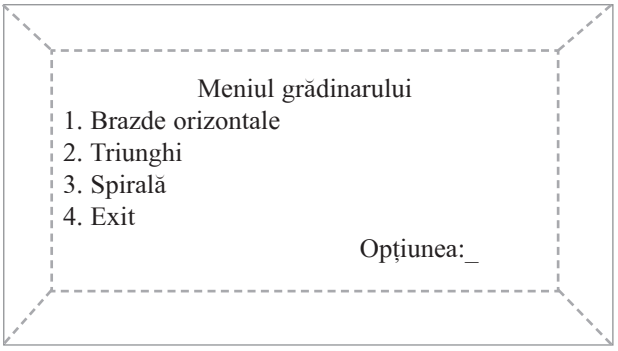

Figura 14

6. Rescrieți secvența umplerii în spirală astfel încât să folosiți cât mai puține structuri repetitive.

7. Compuneți un model floral nou și scrieți secvența de instrucțuni (programul) pentru generarea modelului propus.

8. Analizați următoarele secvențe pseudocod și determinați modelul de umplere generat:

a)
pentru i=2 **la** n **execută**
  pentru j=n, **la** n+2-i **execută**
  G[i,j] ← cod
  sfârșit pentru
sfârșit pentru
	b)
	pentru i= 1 **la** [n/2] **execută**
	  pentru j=i+1 **la** n-i **execută**
	  G[i,j] ← cod
	  sfârșit pentru
	sfârșit pentru

c)
pentru i=n **la** [n/2]+1 **execută**
  pentru j=n **la** n+2-i **execută**
  G[i,j] ← cod
  sfârșit pentru
sfârșit pentru
	d)
	pentru i=1 **la** n **execută**
	  pentru j=1, la **m execută**
	  G[i,j] ← cod
	  sfârșit pentru
	sfârșit pentru

5.3. Simularea unor situații reale

Tabloul de familie

Se consideră o familie formată din părinți (1) și copii (2). Fiecare copil are doi părinți între care există relație de căsătorie. Nu toți membrii familiei sunt căsătoriți; nu toți membrii familiei au copii.

Relațiile dintre membrii familiei sunt păstrate în tabloul de familie (fig. 15).

Figura 15. Tablou pentru o familie formată din 8 persoane

Membrul familiei	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***
1				*1*		*2*		*2*
2					*2*	*1*	*2*	
3					*1*			
4	*1*					*2*		*2*
5			*1*					
6		***1***			***2***		***2***	
7								
8								

CUPRINS