Problema damelor

Fiind dată o tablă de şah, de dimensiune n, x_n, se cer toate soluţiile de aranjare a n dame, astfel încât să nu se afle două dame pe aceeaşi linie, coloană sau diagonală (dame să nu se atace reciproc).

Exemplu: Presupunând că dispunem de o tablă de dimensiune 4x4, o soluţie ar fi următoarea:

	D		
			D
D			
		D	

Observăm că o damă trebuie să fie plasată singură pe linie. Plasăm prima damă pe linia 1, coloana 1.

D			

A doua damă nu poate fi aşezată decât în coloana 3.

D			
		D	

Observăm că a treia damă nu poate fi plasată în linia 3. Încercăm atunci plasarea celei de-a doua dame în coloana 4.

D			
			D

A treia damă nu poate fi plasată decât în coloana 2.

D			
			D
	D		

În această situaţie dama a patra nu mai poate fi aşezată.

Încercând să avansăm cu dama a treia, observăm că nu este posibil să o plasăm nici în coloana 3, nici în coloana 4, deci o vom scoate de pe tablă. Dama a doua nu mai poate avansa, deci şi ea este scoasă de pe tablă. Avansăm cu prima damă în coloana 2.

D			

A doua damă nu poate fi aşezată decât în coloana 4.

	D		
			D

Dama a treia se aşează în prima coloană.

	D		
			D
D			

Acum este posibil să plasăm a patra damă în coloana 3 si astfel am obţinut o soluţie a problemei.

	D		
			D
D			
		D	

Algoritmul continuă în acest mod până când trebuie scoasă de pe tablă prima damă.

Pentru reprezentarea unei soluţii putem folosi un vector cu n componente (având în vedere că pe fiecare linie se găseşte o singură damă).

Exemplu pentru soluţia găsită avem vectorul ST ce poate fi asimilat unei stive.

Două dame se găsesc pe aceeaşi diagonală dacă si numai dacă este îndeplinită condiţia: |st(i)-st(j)|=|i-j| (diferenţa, în modul, între linii si coloane este aceeaşi).

3	ST(4)
1	ST(3) În general ST(i)=k semnifică faptul că pe linia i dama ocupă poziţia k.
4	ST(2)
2	ST(1)

Exemplu: în tabla 4 x4 avem situaţia:

	D		
			D
D			
		D	

st(1)= 1 i = 1
st(3)= 3 j = 3
|st(1) - st(3)| = |1 – 3| = 2
|i – j| = |1 – 3| = 2

sau situaţia

	D		
			D
D			
		D	

st(1) = 3 i = 1
st(3) = 1 j = 3
|st(i) - st(j)| = |3 – 1| = 2
|i – j| = |1 – 3| = 2

Întrucât doua dame nu se pot găsi în aceeaşi coloană, rezultă că o soluţie este sub formă de permutare. O primă idee ne conduce la generarea tuturor permutărilor si la extragerea soluţiilor pentru problema ca două dame să nu fie plasate în aceeaşi diagonală. A proceda astfel, înseamnă că lucrăm conform strategiei ***backtracking***. Aceasta presupune ca imediat ce am găsit două dame care se atacă, să reluăm căutarea.

lată algoritmul, conform strategiei generate de ***backtracking***:

- În prima poziţie a stivei se încarcă valoarea 1, cu semnificaţia că în linia unu se aşează prima damă în coloană.

- Linia 2 se încearcă aşezarea damei în coloana 1, acest lucru nefiind posibil întrucât avem doua dame pe aceeaşi coloană.

- În linia 2 se încearcă aşezarea damei în coloana 2 , însă acest lucru nu este posibil, pentru că damele se găsesc pe aceiaşi diagonală (|st(1)-st(2)|=|1-2|);
- Aşezarea damei 2 în coloana 3 este posibilă.
- Nu se poate plasa dama 3 în coloana 1, întrucât în liniile 1-3 damele ocupa acelaşi coloană.
- Şi această încercare eşuează întrucât damele de pe 2 şi 3 sunt pe aceeaşi diagonală.
- Damele de pe 2-3 se găsesc pe aceeaşi coloană.
- Damele de pe 2-3 se găsesc pe aceeaşi diagonală.
- Am coborât în stivă mutând dama de pe linia 2 şi coloana 3 în coloana 4.

Algoritmul se încheie atunci când stiva este vidă. Semnificaţia procedurilor utilizate este următoarea:

INIT - nivelul k al stivei este iniţializat cu 0;

ABSOLUT - returnează valoarea absolută a unui întreg;

VALID - validează valoarea pusă pe nivelul k al stivei, verificând dacă nu avem două dame pe aceeaşi linie (st(k)=st(i)), sau dacă nu avem două dame pe aceeaşi diagonală (st(k)-st(i)=|k-i|) caz în returnează 1; în caz contrar, în returnează 0;

SOLUTIE - verifică dacă stiva a fost completată până la nivelul n inclusiv;

TIPAR - tipăreşte o soluţie.

O soluție implementată în C++ este:

```
#include <iostream>

using namespace std;

int st[100], n, nrsol;

int absolut(int x)
{
   if(x<0) x=-x;
   return x;
}

int valid(int k)
{
   int i;
   for (i=1; i<=k-1; i++)
      if ( (st[i]==st[k]) || (absolut( (st[k]-st[i]) ) == (k-i) ) )
         return 0;
```

```
    return 1;
}

int solutie ( int k)
{
    if (k==n)
        return 1;
    return 0;
}

void init (int k)
{
    st[k]=0;
}

void tipar ()
{
    int i,j;
    nrsol++;
    cout<<"\nSolutia "<<nrsol<<" este\n";
    for(i=1;i<=n;i++)
    {
        cout<<endl;
        for(j=1;j<=n;j++)
            if (st[j]==i)
                cout<<" D ";
            else
                cout<<" + ";
    }
    cout<<endl;
}



void back(int k)
{
    init(k);
    int i;
    for (i=1;i<=n;i++)
    {
        st[k]=i;
        if (valid(k)==1)
        {
```

```
            if (solutie(k)==1)
                tipar();
            else
                back(k+1);
        }
    }
}

int main()
{
    cout<<"Numarul de dame (dimensiunea tablei): ";
    cin>>n;

    back(1);
    cout<<"\nSunt "<<nrsol<<" solutii";
    return 0;
}
```