Problema reginelor

(Backtracking recursiv)

Se consideră o tablă de șah de dimensiune **n** ($4 \leq n \leq 10$). Să se plaseze pe tablă **n** regine, astfel încât să nu existe două regine care să se atace (două regine se atacă dacă se află pe aceeași linie, aceeași coloană sau aceeași diagonală).

```
#include <iostream>
#include <cmath>
using namespace std;

//n reprezinta dimensiunea tablei (nxn) si numarul de regine
//x este vectorul solutie
//x[k] retine linia pe care se afla regina de pe coloana k
int n, x[12];


//intoarce true daca s-au completat toate cele n coloane
bool eSolutie(int k)
{
   //daca am ajuns la coloana n+1, inseamna ca avem coloanele de la 1 la n completate corect
   if(k==n+1) return true;
   return false;
}


//verificam daca regina de pe coloana k nu se ataca cu cele deja plasate in stanga ei
bool eOk(int k)
{
   for(int i=1; i<k; i++)
   {
       //verificare pe linie: doua regine nu pot fi pe aceeasi linie
       if(x[k]==x[i])
          return false;
      //verificare pe diagonala: reginele se ataca pe diagonala daca diferenta dintre coloane este egala cu diferenta dintre linii în modul
      if(k-i==abs(x[k]-x[i]))
          return false;
   }
   return true;
}
```

```
//afisam o solutie gasita sub forma de matrice/tabla de sah
void print()
{
   //parcurgem tabla linie cu linie (i) si coloana cu coloana (j)
   for(int i=1; i<=n; i++)
   {
      for(int j=1; j<=n; j++)
      {
         //daca pe coloana j, regina se afla pe linia i, o afisam cu *
         if(x[j]==i)
            cout<<"*"<<" ";
         else
            cout<<"-"<<" "; //spatiu gol, fara regina
      }
      cout<<endl; //trecem la linia urmatoare a tablei
   }
   cout<<endl; //spatiu intre solutii diferite
}


//functia recursiva de Backtracking care completeaza coloana k
void BKT (int k)
{
   //daca am gasit o solutie valida (toate cele n regine sunt plasate)
   if(eSolutie(k))
   {
      print(); //afisam solutia
   }
   else
   {
      //incercam sa plasam regina de pe coloana k pe fiecare linie de la 1 la n
      for(int i=1; i<=n; i++)
      {
         x[k]=i; //punem provizoriu regina pe coloana k, linia i
         if(eOk(k)) //daca pozitia nu genereaza atacuri cu reginele plasate anterior
            BKT(k+1); //trecem la pasul urmator: incercam sa plasam regina pe coloana k+1
      }
   }
}


int main()
{
   //citim dimensiunea tablei de sah
   cin>>n;
   //pornim algoritmul de Backtracking de la prima coloana (k=1)
   BKT(1);

   return 0;
}
```