Programmation

Problème des N reines

Il faut placer 8 reines dans un échiquier (matrice 8X8) sans qu'aucune d'entre-elles ne soit en prise par une autre." Deux reines sont en prise, si elles se trouvent sur une même ligne, une même colonne ou une même diagonale).

L'algorithme peut être amélioré en considérant uniquement les positions où deux reines ne sont ni sur la même ligne ni sur la même colonne.

De cette manière, on peut représenter tout l'échiquier par un vecteur où les indices désignent les lignes et les contenus les colonnes.

Dans le programme suivant, V contient le placement des reines.

Col , Diagp, Diagn contiennent respectivement les colonnes, diagonales positives et négatives déjà sélectionnées. I_col , I_diagp , I_diagn sont les indices respectifs et donc aussi les nombres d'éléments dans chaque table. Les actions Ajouter_col , Ajouter_diagp , Ajouter_diagn ont pour rôle de mettre à jour les tables Col , Diagp, Diagn.
La fonction Appartient teste l'appartenance d'une colonne, d'une diagonale positive ou négative dans la table correspondante.

Le programme marche pour N reines. Il suffit de changer les dimensions des tableaux.

Il fournit tous les placements possibles.

```
SOIT
   V UN TABLEAU ( 8 ) ;
   Col UN TABLEAU ( 8 ) ;
   Diagp UN TABLEAU ( 8 ) ;
   Diagn UN TABLEAU ( 8 ) ;
   I_col , I_diagp , I_diagn : ENTIERS ;
   Ajouter_col , Ajouter_diagp , Ajouter_diagn DES ACTIONS ;
   Placer UNE ACTION ;
   Appartient UNE FONCTION ( BOOLEEN ) ;
   N UN ENTIER ;
   Compt UN ENTIER ;
   Nb_appels UN ENTIER ;
   Nb_iter : ENTIER ;
DEBUT
   N := 8 ;
   Compt := 0 ;
   Nb_iter := 0 ;
   Nb_appels := 0 ;
   I_col := 0 ;
   I_diagp := 0 ;
   I_diagn := 0 ;
   APPEL Placer ( 0 ) ;
   ECRIRE ( 'Compte=' , Compt ) ;
   ECRIRE ( 'Nombre d''appels du module Placer =' , Nb_appels ) ;
   ECRIRE ( 'Nombre d''itérations de la boucle Pour =' , Nb_iter ) ;
FIN
```

```
{Ajouter une colonne}
ACTION Ajouter_col ( Colonne )
SOIT
   Colonne UN ENTIER ;

DEBUT
   I_col := I_col + 1 ;
   AFF_ELEMENT ( Col [ I_col ] , Colonne )
FIN

{Ajouter une diagonale positive}
ACTION Ajouter_diagp ( Num_diagp )
SOIT
   Num_diagp UN ENTIER ;
DEBUT
   I_diagp := I_diagp + 1 ;
   AFF_ELEMENT ( Diagp [ I_diagp ] , Num_diagp )
FIN

{Ajouter une diagonale positive}
ACTION Ajouter_diagn ( Num_diagn )
SOIT
   Num_diagn UN ENTIER ;

DEBUT
   I_diagn := I_diagn + 1 ;
   AFF_ELEMENT ( Diagn [ I_diagn ] , Num_diagn )
FIN

{Appartenance d'un élément à une table}
FONCTION Appartient ( Num , Tab , I_tab ) : BOOLEEN
SOIT
   I , Num DES ENTIERS ;
   Trouv : BOOLEEN ;
   Tab UN TABLEAU ( 8 ) ;
   I_tab UN ENTIER ;
DEBUT
   I := 1 ;
   Trouv := FAUX ;
   TQ ( I <= I_tab ) ET NON Trouv
      SI ELEMENT ( Tab [ I ] ) = Num
         Trouv := VRAI
      SINON
         I := I + 1
      FSI
   FTQ ;
   Appartient := Trouv
FIN
```

```
{Placement des reines : il fournit toutes les possibilités}
ACTION Placer ( K )
SOIT
   J , K DES ENTIERS ;

DEBUT
   SI K = N
      Compt := Compt + 1 ;
      ECRIRE ( 'V=' , V ) ;
      {Retour}
      l_col := l_col - 1 ;
      l_diagp := l_diagp - 1 ;
      l_diagn := l_diagn - 1 ;

   SINON
      POUR J := 1 , N
         Nb_iter := Nb_iter + 1 ;
         SI NON Appartient ( J , Col , l_col ) ET
            NON Appartient ( K + 1 - J , Diagp , l_diagp ) ET
            NON Appartient ( K + 1 + J , Diagn , l_diagn )
            AFF_ELEMENT ( V [ K + 1 ] , J ) ;
            APPEL Ajouter_col ( J ) ;
            APPEL Ajouter_diagp ( K + 1 - J ) ;
            APPEL Ajouter_diagn ( K + 1 + J ) ;
            Nb_appels := Nb_appels + 1 ;
            APPEL Placer ( K + 1 ) ;

         FSI
      FPOUR ;
      {Retour}
      l_col := l_col - 1 ;
      l_diagp := l_diagp - 1 ;
      l_diagn := l_diagn - 1 ;

   FSI
FIN
```

Résultats

```
V= 1  5  8  6  3  7  2  4
V= 1  6  8  3  7  4  2  5
V= 1  7  4  6  8  2  5  3
V= 1  7  5  8  2  4  6  3
V= 2  4  6  8  3  1  7  5
V= 2  5  7  1  3  8  6  4
V= 2  5  7  4  1  8  6  3
V= 2  6  1  7  4  8  3  5
V= 2  6  8  3  1  4  7  5
V= 2  7  3  6  8  5  1  4
V= 2  7  5  8  1  4  6  3
V= 2  8  6  1  3  5  7  4
V= 3  1  7  5  8  2  4  6
V= 3  5  2  8  1  7  4  6
V= 3  5  2  8  6  4  7  1
V= 3  5  7  1  4  2  8  6
```

V= 3 5 8 4 1 7 2 6
V= 3 6 2 5 8 1 7 4
V= 3 6 2 7 1 4 8 5
V= 3 6 2 7 5 1 8 4
V= 3 6 4 1 8 5 7 2
V= 3 6 4 2 8 5 7 1
V= 3 6 8 1 4 7 5 2
V= 3 6 8 1 5 7 2 4
V= 3 6 8 2 4 1 7 5
V= 3 7 2 8 5 1 4 6
V= 3 7 2 8 6 4 1 5
V= 3 8 4 7 1 6 2 5
V= 4 1 5 8 2 7 3 6
V= 4 1 5 8 6 3 7 2
V= 4 2 5 8 6 1 3 7
V= 4 2 7 3 6 8 1 5
V= 4 2 7 3 6 8 5 1
V= 4 2 7 5 1 8 6 3
V= 4 2 8 5 7 1 3 6
V= 4 2 8 6 1 3 5 7
V= 4 6 1 5 2 8 3 7
V= 4 6 8 2 7 1 3 5
V= 4 6 8 3 1 7 5 2
V= 4 7 1 8 5 2 6 3
V= 4 7 3 8 2 5 1 6
V= 4 7 5 2 6 1 3 8
V= 4 7 5 3 1 6 8 2
V= 4 8 1 3 6 2 7 5
V= 4 8 1 5 7 2 6 3
V= 4 8 5 3 1 7 2 6
V= 5 1 4 6 8 2 7 3
V= 5 1 8 4 2 7 3 6
V= 5 1 8 6 3 7 2 4
V= 5 2 4 6 8 3 1 7
V= 5 2 4 7 3 8 6 1
V= 5 2 6 1 7 4 8 3
V= 5 2 8 1 4 7 3 6
V= 5 3 1 6 8 2 4 7
V= 5 3 1 7 2 8 6 4
V= 5 3 8 4 7 1 6 2
V= 5 7 1 3 8 6 4 2
V= 5 7 1 4 2 8 6 3
V= 5 7 2 4 8 1 3 6
V= 5 7 2 6 3 1 4 8
V= 5 7 2 6 3 1 8 4
V= 5 7 4 1 3 8 6 2
V= 5 8 4 1 3 6 2 7
V= 5 8 4 1 7 2 6 3
V= 6 1 5 2 8 3 7 4
V= 6 2 7 1 3 5 8 4
V= 6 2 7 1 4 8 5 3
V= 6 3 1 7 5 8 2 4
V= 6 3 1 8 4 2 7 5
V= 6 3 1 8 5 2 4 7
V= 6 3 5 7 1 4 2 8

```
V= 6  3  5  8  1  4  2  7
V= 6  3  7  2  4  8  1  5
V= 6  3  7  2  8  5  1  4
V= 6  3  7  4  1  8  2  5
V= 6  4  1  5  8  2  7  3
V= 6  4  2  8  5  7  1  3
V= 6  4  7  1  3  5  2  8
V= 6  4  7  1  8  2  5  3
V= 6  8  2  4  1  7  5  3
V= 7  1  3  8  6  4  2  5
V= 7  2  4  1  8  5  3  6
V= 7  2  6  3  1  4  8  5
V= 7  3  1  6  8  5  2  4
V= 7  3  8  2  5  1  6  4
V= 7  4  2  5  8  1  3  6
V= 7  4  2  8  6  1  3  5
V= 7  5  3  1  6  8  2  4
V= 8  2  4  1  7  5  3  6
V= 8  2  5  3  1  7  4  6
V= 8  3  1  6  2  5  7  4
V= 8  4  1  3  6  2  7  5
Compte= 92
Nombre d'appels du module Placer = 2056
Nombre d'itérations de la boucle Pour =15720
```