```
with Ada.Text_Io; use Ada.Text_Io;
with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;

procedure N_Reines is

   Taille : Natural; -- taille de l'échiquier sur lequel on cherche à placer les reines;

   ------------------------------------------------------------------------
   ------------------------------------------------------------------------
   -- cette procedure est la procedure qui est évaluée dans ce programme

   function Calcul_N_Reine (N:Positive) return Natural is
      ----------------------
      -- calcul le nombre de placements de n reines sur l'échiquier,
      --    donnée: N la taille de l'échiquier
      --    résultat : le nombre de solutions

      type Indice is new Integer range 1..N;

   type D1 is array(1-N .. N-1) of Boolean; -- voir ci dessous dans Echiquier
   type D2 is array(2..2*N) of Boolean;     --     sur l'usage de ces types
   type Col is array(Indice) of Boolean;

   type Les_Places is array (Indice) of Indice;

   type Echiquier is record
      Colonne: Les_Places; -- position de la reine dans la ième ligne
      Diag_45_Libre : D1 := (others => True);
      Diag_135_Libre : D2 :=  (others => True);
      Colonne_Libre : Col :=  (others => True);
      -- indiquent si les diagonales et colonnes sont libres; les diagonales sont repérées par
      -- (i+j) = constante pour diag_135 et  (i-j) = constante pour diag_45
   end record;

   Sol: Echiquier;   -- la solution partielle Echiquier en variable globale
   Nbre_Sol : Natural :=0; --  et son nombre d~éléments

   ------------------------------------------------------------------------------
   function Position_Libre (I,J: Indice) return Boolean is
      ------------------------
      -- indique si la position (i,j) est libre sur l'échiquier Sol
      --      données: l'indice ligne (i) et l'indice de colonne (j)
      --      résultat: vrai si la case n'est pas en prise par une reine sur l'échiquier Sol

   begin
      return (Sol.Colonne_Libre(J) and Sol.Diag_45_Libre(Integer(I-J)) and Sol.Diag_135_Libre
(Integer(I+J)) );
   end Position_Libre;

   -- procedure imprime_echiquier (E: echiquier); pas mise en oeuvre

   ------------------------------------------------------------------------
   procedure Explore (K: Natural) is
      ------------------
      -- explore toutes les prolongations d'une solution partielle rangée dans Sol
      --    données : k le nombre de reines déjà placées dans la variable globale SOL représentant
```

```
l'échiquier
      --           : un entier nbre_sol qui sera incrémenté
      --    résultats :
      --           : Nbre_Sol est mis à jour par incrémentation du nombre de solutions prolongeant la solution
      --                partielle dans Sol à l'appel
      --           : l'échiquier Sol est rendu dans l'état avant appel

      I : Indice ; -- la prochaine ligne à considérer
   begin
      if K=N then Nbre_Sol := Nbre_Sol +1;  -- on en tient une
      -- on pourrait l'imprimer aussi, on a tout ce qu'il faut pour le faire
      else -- on essaie de prolonger
                I:= Indice(K+1);
                for J in Indice loop
                   if Position_Libre (I,J) then
                      -- on place une reine en (i,j) :
                      Sol.Colonne(I) :=J;
                      Sol.Colonne_Libre(J) := False;
                      Sol.Diag_45_Libre (Integer(I-J)) := False;
                      Sol.Diag_135_Libre(Integer(I+J)) := False;
                      -- on explore la suite
                      Explore (Integer(I));
                      -- on remet en ordre : la reine est retirée
                      Sol.Colonne_Libre(J) := True;
                      Sol.Diag_45_Libre (Integer(I-J)) := True;
                      Sol.Diag_135_Libre(Integer(I+J)) := True;
                   end if;
                end loop;
      end if;
   end Explore;
   ------------------------------------------------------

   begin -- début du corps de n-reines
      Explore (0); -- au début il y a 0 reines de placées
      return Nbre_Sol;
   end Calcul_N_Reine;

   -- ============================================================================
begin
   -- corps du programme principal avec interaction avec l'utilisateur

   Put_Line("calcul du nombre de solutions pour le placement de n reines sur un échiquier nxn");
   New_Line;
   loop
      Put_Line(" donnez une valeur de n (0 pour arrêter)");
      Get(Taille);
      exit when Taille = 0;
      Put_Line ("dans ce cas le nombre de solutions est :" & Integer'Image(Calcul_N_Reine(Taille)) );
      New_Line;
   end loop;

   Put_Line("au revoir");
end N_Reines;
```