```
------------------------------------------------------------------------------
--  *  Prog name exo01.adb
--  *  Project name td_01_03
--  *
--  *  Version 1.0
--  *  Last update 11/09/08
--  *
--  *  Created by Pierre-FranAM-^Mois LAM-^Non on 11/09/08.
--  *  Copyright (c) 2008 __MyCompanyName__.
--  *  All rights reserved.
--  *    or (keep only one line or write your own)
--  *  GNAT modified GNU General Public License
--  *
------------------------------------------------------------------------------
with Ada.Text_IO, ada.integer_text_io;
with Ada.Numerics.Discrete_Random;
use  Ada.Text_IO, ada.integer_text_io;

procedure Exo01 is
   N : constant Positive := 5;
   M : constant Positive := 4;
   type Matrix is array (1..N, 1..M) of Float;

   -- comparaisons : N*M*4
   function IsYoungMatrix_m1(M : in Matrix) return Boolean is
   begin
      for I in M'Range(1) loop
         for J in M'Range(2) loop
            if I < M'Last(1) then
               if M(I, J) > M(I + 1, J) then
                  return False;
               end if;
            end if;
            if J < M'Last(2) then
               if M(I, J) > M(I, J + 1) then
                  return False;
               end if;
            end if;
         end loop;
      end loop;

      return True;
   end IsYoungMatrix_m1;

   -- comparaisons : (n - 1) * (m - 1) * 2 + m + n - 2
   -- = 2 * n * m - 2n - 2m + 2 + m + n - 2
   -- = 2 * n * m - n - m
   function IsYoungMatrix_m2(M : in Matrix) return Boolean is
   begin
      for I in M'First(1)..M'Last(1) - 1 loop
         for J in M'First(2)..M'Last(2) - 1 loop
            if M(I, J) > M(I + 1, J) or M(I, J) > M(I, J + 1) then
               return False;
            end if;
         end loop;
      end loop;

      for I in M'First(1)..M'Last(1) - 1 loop
         if M(I, M'Last(2)) > M(I + 1, M'Last(2)) then
            return False;
         end if;
      end loop;

      for J in M'First(2)..M'Last(2) - 1 loop
         if M(M'Last(1), J) > M(M'Last(1), J + 1) then
            return False;
         end if;
      end loop;
```



```
      return True;
   end IsYoungMatrix_m2;
begin
   null;
end Exo01;
```

```
-------------------------------------------------------------------------------
--  *  Prog name exo01.adb
--  *  Project name td_01_03
--  *
--  *  Version 1.0
--  *  Last update 11/09/08
--  *
--  *  Created by Pierre-FranAM-^Mois LAM-^Non on 11/09/08.
--  *  Copyright (c) 2008 __MyCompanyName__.
--  *  All rights reserved.
--  *    or (keep only one line or write your own)
--  *  GNAT modified GNU General Public License
--  *
-------------------------------------------------------------------------------
with Ada.Text_IO, ada.integer_text_io;
with Ada.Numerics.Discrete_Random;
use  Ada.Text_IO, ada.integer_text_io;

procedure Exo02 is
   N : constant Positive := 10;
   type Tableau is array (1..N) of Integer;

   type Tableau_Comprime is record
      Data : Tableau := (others => 0);
      Size : Integer := 0;
      Occ  : Tableau := (others => 0);
   end record;

   procedure Put(T : in Tableau; N : in Integer := Tableau'Last) is
   begin
      Put("Tableau(");
      Put(T(T'First), 1);
      for I in (T'First + 1)..N loop
         Put(",");
         Put(T(I), 1);
      end loop;
      Put(')');
   end Put;

   procedure Put(T : in Tableau_Comprime) is
   begin
      Put("Tableau_comprime(");
      Put(T.Data, T.Size);
      Put(",");
      Put(T.Size, 1);
      Put(",");
      Put(T.Occ, T.Size);
      Put(')');
   end Put;

   procedure Comprime_Q1(T : in out Tableau) is
      Current : Integer := T'First;
   begin
      for I in T'Range loop
         if T(I) /= T(Current) then
            Current := Current + 1;
            T(Current) := T(I);
         end if;
      end loop;
   end Comprime_Q1;

   function Comprime(T : in Tableau) return Tableau_Comprime is
      Res : Tableau_Comprime;
      Current : Integer := T'First;
   begin
      Res.Size          := 1;
      Res.Data(Current) := T(Current);
      Res.Occ(Current)  := 1;
```



```
      for I in (T'First + 1)..T'Last loop
         if T(I) /= Res.Data(Current) then
            Current := Current + 1;
            Res.Data(Current) := T(I);
            Res.Size          := Res.Size + 1;
            Res.Occ(Current)  := 1;
         else
            Res.Occ(Current)  := Res.Occ(Current) + 1;
         end if;
      end loop;

      return Res;
   end Comprime;

   function Decomprime(T : in Tableau_Comprime) return Tableau is
      Res : Tableau;
      Current : Integer := Tableau'First;
   begin
      for I in T.Data'First..(T.Data'First + T.Size - 1) loop
         for J in 1..T.Occ(I) loop
            Res(Current) := T.Data(I);
            Current := Current + 1;
         end loop;
      end loop;

      return Res;
   end Decomprime;

   Tab1, Tab2 : Tableau := (2,3,3,4,4,4,4,2,7,7);
begin
   Put(Tab1); New_Line;
   Comprime_Q1(Tab1);
   Put(Tab1); New_Line;
   Put(Comprime(Tab2)); New_Line;
   Put(Decomprime(Comprime(Tab2))); New_Line;
end Exo02;
--  Tableau(2, 3, 3, 4, 4, 4, 4, 2, 7, 7)
--  Tableau(2, 3, 4, 2, 7, 4, 4, 2, 7, 7)
--  Tableau_comprime(Tableau(2, 3, 4, 2, 7), 5, Tableau(1, 2, 4, 1, 2))
--  Tableau(2, 3, 3, 4, 4, 4, 4, 2, 7, 7)
```

```
------------------------------------------------------------------------------
--  *  Prog name exo04.adb
--  *  Project name td_01_03
--  *
--  *  Version 1.0
--  *  Last update 11/09/08
--  *
--  *  Created by Pierre-FranAM-^Mois LAM-^Non on 11/09/08.
--  *  Copyright (c) 2008 __MyCompanyName__.
--  *  All rights reserved.
--  *    or (keep only one line or write your own)
--  *  GNAT modified GNU General Public License
--  *
------------------------------------------------------------------------------
with Ada.Text_IO, ada.integer_text_io;
with Ada.Numerics.Discrete_Random;
use  Ada.Text_IO, ada.integer_text_io;

procedure Exo04 is
   procedure Put(B : Boolean) is
   begin
      Put(Boolean'Image(B));
   end Put;

   function Contient_Pos_I(Grande, Petite : in String; Pos : in Integer) return Bo
olean is
   begin
      for I in Petite'Range loop
         if Petite(I) /= Grande(Pos + I - 1) then
            return False;
         end if;
      end loop;

      return True;
   end Contient_Pos_I;

   function Contient(Grande, Petite : String) return Natural is
   begin
      for I in Grande'First..(Grande'Last - Petite'Length + 1) loop
         if Contient_Pos_I(Grande, Petite, I) then
            return I;
         end if;
      end loop;

      return 0;
   end Contient;

begin
   Put("Contient_Pos_I(""abaabbab"", ""abb"", 1) = "); Put(Contient_Pos_I("abaabbab", "abb", 1)
); New_Line;
   Put("Contient_Pos_I(""abaabbab"", ""abb"", 2) = "); Put(Contient_Pos_I("abaabbab", "abb", 2)
); New_Line;
   Put("Contient_Pos_I(""abaabbab"", ""abb"", 3) = "); Put(Contient_Pos_I("abaabbab", "abb", 3)
); New_Line;
   Put("Contient_Pos_I(""abaabbab"", ""abb"", 4) = "); Put(Contient_Pos_I("abaabbab", "abb", 4)
); New_Line;
   Put("Contient_Pos_I(""abaabbab"", ""abb"", 5) = "); Put(Contient_Pos_I("abaabbab", "abb", 5)
); New_Line;

   Put("Contient(""abaabbab"", ""abb"") = "); Put(Contient("abaabbab", "abb"), 1); New_Line;
   Put("Contient(""abaabbab"", ""abc"") = "); Put(Contient("abaabbab", "abc"), 1); New_Line;

   Put("Contient(""abaabbabc"", ""abc"") = "); Put(Contient("abaabbabc", "abc"), 1); New_Line
;
end Exo04;
--  Contient_Pos_I("abaabbab", "abb", 1) = FALSE
--  Contient_Pos_I("abaabbab", "abb", 2) = FALSE
--  Contient_Pos_I("abaabbab", "abb", 3) = FALSE
--  Contient_Pos_I("abaabbab", "abb", 4) = TRUE
```



```
--  Contient_Pos_I("abaabbab", "abb", 5) = FALSE
--  Contient("abaabbab", "abb")  = 4
--  Contient("abaabbab", "abc")  = 0
--  Contient("abaabbabc", "abc") = 7
```