

```

1   with Text_Io, Matrice_Pack;
2   use Text_Io, Matrice_Pack;
3
4   procedure Test8 is
5     $
6     $ i N : constant := 10;
7     $
8     $ i A : Matrice_Ptr :=
9     $   new Matrice ( 1 .. 10, 1 .. 10); -- bornes seulement
10    $ i B : Matrice_Ptr :=
11    $   new Matrice' ( 1 .. 10 =>( 1 .. 10 => 1.0));
12    $-- initialisation explicite
13    $ i C : Matrice_Ptr :=
14    $   new Matrice' ( 1 .. 10 =>( 1 .. 10 => 2.0));
15    $-- initialisation explicite
16    $ i Aa : Matrice (1 .. N, 1 .. N);
17    $ i Bb : Matrice (1 .. N, 1 .. N);
18    $ i Cc : Matrice (1 .. N, 1 .. N);
19    $
20    $ procedure Put (
21    $ $   A : in Matrice ) renames Matrice_Pack.Imprimer_Matrice;
22    $
23    $ procedure Put (
24    $ $   A : in Matrice_Ptr ) renames Matrice_Pack.Imprimer_Matri
25    $
26    $ begin
27    $   C(3, 4) := 8.0;
28    $   Cc := C.all;
29    $   Bb := B.all;
30    $   Bb(5, 3) := 14.0;
31    $   Aa := Cc * Bb;
32    $   A := B * C;
33    $   Aa := A.all;
34    $   Put_Line (" Matrice AA ");
35    $   Put(Aa);
36    $   Put_Line (" Matrice A ");
37    $   Put(A);
38    $ end Test8;
39
40    $
41    $ i package Matrice_Pack is
42    $   type REEL is digits 6;
43    $   type Matrice is array ( Integer range <>,
44    $     Integer range <>) of REEL;
45    $   type Matrice_Ptr is access Matrice;
46    $
47    $ function "*" ( A, B : in Matrice) return Matrice;

```

```

48  $
49  $
50  $ _111111111
50  $function "*" ( A, B : in Matrice_Ptr) return
51  $ $ Matrice_Ptr;
51  $ a111111111
52  $
53  $
54  $
54  $ _111111111
55  $procedure Imprimer_Matrice ( A : in Matrice );
55  $ a111111111
56  $
57  $
57  $ _111111111
58  $procedure Imprimer_Matrice ( A : in Matrice_Ptr );
58  $ a111111111
59  $
60  $
61  $end Matrice_Pack;
62  $ with TEXT_IO;
63  $ use TEXT_IO;
64  $
65  $
66  $i $$$$$$$$$$
66  $package body Matrice_Pack is
66  $E$|
67  $
68  $
69  $
69  $ _111111111
70  $function "*" ( A, B : in Matrice) return Matrice is
70  $ aE111111111
71  $ $
72  $ $ i C : Matrice (A' Range(1), B' Range(2));
73  $ $ i SUM : REEL;
74  $ $begin
75  $ $ 11±for I in C' Range(1) loop
76  $ $ 711±for J in C' Range(2) loop
77  $ $ 5 711 SUM := 0.0;
78  $ $ 5 711±for K in A' Range(2) loop
79  $ $ 5 5 711 SUM := SUM + A(I, K) * B(K, J);
80  $ $ 5 5 °end loop;
81  $ $ 5 711 C(I, J) := SUM;
82  $ $ 5 °end loop;
83  $ $ °end loop;
84  $ $A1A11 return C;
85  $ $ °end "*";
86  $
87  $
87  $ _111111111
88  $function "*" ( A, B : in Matrice_Ptr) return Matrice_Ptr is
88  $ aE111111111
89  $ $
90  $ $ i C : Matrice_Ptr := new Matrice (A' Range(1),
91  $ $ B' Range(2));
92  $ $begin

```

```

93   §   C.all := A.all * B.all;
94   §   return C;
95   §   end "*";
96   §
97   §   package REEL_IO is new Text_IO.Float_IO(REEL);
98   §   E|||||||||
99   §   use REEL_IO;
100  §
101  §
102  §   procedure Imprimer_Matrice ( A : in Matrice ) is
103  §   §
104  §   §begin
105  §   §   New_Line(2);
106  §   §   for I in A'Range(1) loop
107  §   §   §   for J in A'Range(2) loop
108  §   §   §   §   Put (A(I, J), 3, 1, 0);
109  §   §   §   §   end loop;
110  §   §   §   New_Line;
111  §   §   §   end loop;
112  §   §   end Imprimer_Matrice;
113  §
114  §
115  §   procedure Imprimer_Matrice ( A : in Matrice_Ptr ) is
116  §   §
117  §   §begin
118  §   §   New_Line(2);
119  §   §   for I in A'Range(1) loop
120  §   §   §   for J in A'Range(2) loop
121  §   §   §   §   Put (A(I, J), 3, 1, 0);
122  §   §   §   §   end loop;
123  §   §   §   New_Line;
124  §   §   §   end loop;
125  §   §   end Imprimer_Matrice;
126  §
127  §
128  §   end Matrice_Pack;
129
130

```