

```

1  with Ada.Float_Text_IO;
2  use Ada.Float_Text_IO;
3  with Ada.Text_IO;
4  use Ada.Text_IO;
5  with Ada.Integer_Text_IO;
6  use Ada.Integer_Text_IO;
7
8  with Ada.Numerics.Elementary_Functions;
9  use Ada.Numerics.Elementary_Functions;
10
11 with Ada.Numerics.Discrete_Random;
12
13 procedure inverse is
14
15     type matrix is array(integer range <>, integer range <>) of float;
16
17     spaces : constant integer := 2;
18     symb : constant integer := 3;
19     proc : integer := 5;
20     dim : constant integer := 10;
21     A : matrix(1..dim, 1..dim);
22
23     procedure matr_init is
24         subtype value is Positive range 1..10;
25         package Rand is
26             new Ada.Numerics.Discrete_Random(value);
27         seed : Rand.Generator;
28     begin
29         Rand.Reset(seed);
30         for row in 1..dim loop
31             for col in 1..dim loop
32                 A(row, col) := float(Rand.Random(seed))/1000.0;
33                 if row < col then
34                     A(row, col) := 0.0;
35                 end if;
36             end loop;
37         end loop;
38     end matr_init;
39
40     procedure check(inverted: in matrix) is
41         eps : constant float := 0.5;
42         flag : boolean;
43         C : matrix(1..dim, 1..dim);
44     begin
45         C := (others => (others => 0.0));
46         for row in 1 .. dim loop
47             for col in 1 .. dim loop
48                 for pos in 1 .. dim loop
49                     C(row, col) := C(row, col) + A(row, pos) * inverted(pos, col);
50                 end loop;
51             end loop;
52         end loop;
53         for row in 1 .. dim loop
54             for col in 1 .. dim loop
55                 if ( (row = col) and abs(C(row, col) - 1.0) > eps )
56                 then
57                     put("Element");

```

```

58         put(row);
59         put(col);
60         put(" = ");
61         put(C(row, col), spaces, symb);
62         put(" failed");
63         new_line;
64         flag := true;
65     end if;
66     if(row /= col) and abs(C(row, col)) > eps
67     then
68         put("Element");
69         put(row);
70         put(col);
71         put(" = ");
72         put(C(row, col), spaces, symb);
73         put(" failed");
74         new_line;
75         flag := true;
76     end if;
77 end loop;
78 end loop;
79 if flag then
80     put("Some problems with accuracy : ");
81     put(eps, spaces, symb);
82     new_line;
83     put("Test failed");
84 else
85     put("OK");
86 end if;
87 end check;
88
89 function inv(A: in matrix; proc: in integer) return matrix is
90
91     h:integer;
92     inverted: matrix(1..dim, 1..dim);
93
94     task type part is
95         entry set(left, right:in integer);
96     end part;
97
98     parts: array(1..proc) of part;
99
100    task body part is
101        l, r: integer;
102        sum: float;
103    begin
104        accept set(left,right: in integer)
105        do
106            l := left;
107            r := right;
108        end set;
109        for col in l..r loop
110            for row in col + 1 .. dim loop
111                sum := 0.0;
112                for j in col .. row - 1 loop
113                    sum := sum + A(row, j)*inverted(j, col);
114                end loop;

```

```
115             inverted(row, row) := 1.0 / a(row, row);
116             inverted(row, col) := - sum * inverted(row, row);
117         end loop;
118     end loop;
119 end part;
120
121 begin
122     inverted := (others => (others => 0.0));
123     h := dim/proc;
124     for i in 1..dim loop
125         inverted(i, i) := 1.0/A(i, i);
126     end loop;
127     for i in 1..proc loop
128         parts(i).set((i - 1)*h + 1, i*h);
129     end loop;
130     return(inverted);
131 end inv;
132
133 begin
134     matr_init;
135     check(inv(A, proc));
136 end inverse;
137
138 --Минаков Александр K5-224
139 --Вывод программы:
140 --OK
141 --
142 --Или, например:
143 --Element      10      9 = 1.429E+00 failed
144 --Some problems with accuracy : 5.000E-01
145 --Test failed
146
147
148
```