НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
КАФЕДРА
Автоматизованих Систем Обробки інформації та Управління

Лабораторна робота №2

з дісципліни «Паралельне програмуваня»

Виконав:
студент 3 курсу
ФІОТ гр. ІП-31
Степанов Александр

Перевірив:
Корочкін О. В.

Київ 2015 р.

Завдання:

Релізувати дані три функції в мові Ада за визначеним викладачем варіантом:

F1: 1.9: MC = MIN(A) *(MD*MB)
F2: 2.4: MN = MAX(ML) *(MK*MO)
F3: 3.4: Z = SORT(R)*SORT(MW*MV)

Разработать программу, содержащую п а р а л л е л ь н ы е задачи,
каждая из которых реализует функцию F1, F2, F3 из лабораторной работы номер 1.
Обязательно использовать пакет Data из лабораторной работы 1 !
Задачи независимы, общих данных не имеют!
При создании задач необходимо:
- указать имя задачи T1,T2, T3
- задать приоритет каждой задачи
- задать размер стека для задачи
- выбрать и задать номер процессора (ядра) для выполнения каждой задачи.
Первый и последний операторы тела задачи выводят на экран информацию о старте и завершении соответствующей задачи (“Task T1 started”, “ Task T2 finished”).
В теле задачи показать использование оператора задержки delay, поставив его перед выполнением функции F1, F2, F3.

Лістинг:

```
GNAT 4.8.1
Copyright 1992-2013, Free Software Foundation, Inc.

Compiling: lab2.adb (source file time stamp: 2015-09-29 16:19:30)

     1. ------------------------------------------------------------------------
     2. --                                                                    --
     3. --                      Parallel Programming                          --
     4. --         Laboratory work #2. Process in Ada                         --
     5. --                                                                    --
     6. --  File: lab2.ada                                                   --
     7. --  Task: F1: MC = MIN(A) * (MD * MB)                                --
     8. --        F2: MN = MAX(ML) * (MK * MO)                               --
     9. --        F3: Z = SORT(R) * SORT(MW * MV)                            --
    10. --                                                                    --
    11. --  Author: Stepanov Alexander, group IP - 31                         --
    12. --  Date: 29.09.2015                                                 --
    13. --                                                                    --
    14. ------------------------------------------------------------------------
    15.
    16. -- with Data, Ada.Text_IO, Ada.Integer_Text_IO, System.Multiprocessors;
    17. -- use Ada.Text_IO, Ada.Integer_Text_IO, System.Multiprocessors;
    18. with Data, Ada.Text_IO, Ada.Integer_Text_IO;
    19. use Ada.Text_IO, Ada.Integer_Text_IO;
    20.
    21. procedure Lab2 is
    22.     Size : Natural;
    23.     Value : Integer;
    24. begin
    25.
    26.     Put ("Input size: ");
    27.     Get (Item => Size);
    28.     Put ("Input value: ");
    29.     Get (Item => Value);
    30.
    31.     declare
    32.         package mainData is new Data(Size, Value);
    33.         use mainData;
    34.         --Processor numbers:
    35.         -- CPU_1 :  CPU_Range := 0;
    36.         -- CPU_2 :  CPU_Range := 1;
    37.         -- CPU_3 :  CPU_Range := 2;
    38.     begin
    39.
    40.         declare
    41.             ----------------------------------
    42.             --            Task 1            --
    43.             --  F1: MC = MIN(A) * (MD * MB)  --
    44.             ----------------------------------
    45.             task T1 is
    46.                 pragma Priority(15);
    47.                 pragma Task_Name ("T1");
    48.                 pragma Storage_Size(100000000);
    49.                 -- pragma CPU (CPU_3);
```

```
50.             end T1;
51.
52.             task body T1 is
53.                 A : Vector;
54.                 MD, MB, MC : Matrix;
55.             begin
56.                 delay 0.0;
57.                 Put_line("Task T1 started");
58.                 Input (A);
59.                 Input (MD);
60.                 Input (MB);
61.                 F1 (A, MD, MB, MC);
62.                 Output (MC);
63.                 Put_line ("Task T1 finished");
64.             end T1;
65.
66.             ----------------------------------
67.             --            Task 2             --
68.             --  F2: MN = MAX(ML) * (MK * MO) --
69.             ----------------------------------
70.             task T2 is
71.                 pragma Priority(15);
72.                 pragma Task_Name ("T2");
73.                 pragma Storage_Size(100000000);
74.                 -- pragma CPU (CPU_1);
75.             end T2;
76.
77.             task body T2 is
78.                 ML, MK, MO, MN : Matrix;
79.             begin
80.                 delay 0.1;
81.                 Put_line ("Task T2 started");
82.                 Input (ML);
83.                 Input (MK);
84.                 Input (MO);
85.                 F2 (ML, MK, MO, MN);
86.                 Output (MN);
87.                 Put_line ("Task T2 finished");
88.             end T2;
89.
90.             --------------------------------------
91.             --            Task 3                --
92.             --  F3: Z = SORT(R) * SORT(MW * MV) --
93.             --------------------------------------
94.             task T3 is
95.                 pragma Priority(15);
96.                 pragma Task_Name ("T3");
97.                 pragma Storage_Size(100000000);
98.                 -- pragma CPU (CPU_1);
99.             end T3;
100.
101.            task body T3 is
102.                R, Z : Vector;
103.                MW, MV : Matrix;
104.            begin
105.                delay 0.2;
106.                Put_line ("Task T3 started");
107.                Input (R);
```

```
   108.                 Input (MW);
   109.                 Input (MV);
   110.                 F3 (R, MW, MV, Z);
   111.                 Output (Z);
   112.                 Put_line ("Task T3 finished");
   113.             end T3;
   114.         begin
   115.             null;
   116.         end;
   117.
   118.         null;
   119.     end;
   120.
   121. end Lab2;

 121 lines: No errors

GNAT 4.8.1
Copyright 1992-2013, Free Software Foundation, Inc.

Compiling: data.adb (source file time stamp: 2015-09-29 10:46:30)

     1. ------------------------------------------------------------------------
     2. --                                                                    --
     3. --                      Parallel Programming                          --
     4. --         Laboratory work #2. Process in Ada                         --
     5. --                                                                    --
     6. --  File: lab1.ada                                                    --
     7. --  Task: F1: MC = MIN(A) * (MD * MB)                                 --
     8. --        F2: MN = MAX(ML) * (MK * MO)                                --
     9. --        F3: Z = SORT(R) * SORT(MW * MV)                             --
    10. --                                                                    --
    11. --  Author: Stepanov Alexander, group IP - 31                         --
    12. --  Date: 29.09.2015                                                  --
    13. --                                                                    --
    14. ------------------------------------------------------------------------
    15.
    16. with Ada.Text_IO, Ada.Integer_Text_IO, Ada.Strings, Ada.Strings.Fixed;
    17. use Ada.Text_IO, Ada.Integer_Text_IO;
    18.
    19. package body Data is
    20.
    21.     ------------------------------------------------------------------
    22.     --        additional functions for calculation (prototypes)       --
    23.     ------------------------------------------------------------------
    24.     -- Matrix * Matrix
    25.     function "*" (Left  : Matrix; Right : Matrix) return Matrix;
    26.     -- Vector * Matrix
    27.     function "*" (Left  : Vector; Right : Matrix) return Vector;
    28.     -- Integer * Matrix
    29.     function "*" (Left  : Integer; Right : Matrix) return Matrix;
    30.     -- maximum value of the matrix
    31.     function Max (A : Matrix) return Integer;
    32.     -- minimum value of the vector
    33.     function Min (A : Vector) return Integer;
    34.     -- vector sorting
    35.     function Sort (A : Vector) return Vector;
    36.     -- matrix sorting
```

```
37.     function Sort (MA : Matrix) return Matrix;
38.
39.     ------------------------------------------------------------------
40.     --                        main procedures                       --
41.     ------------------------------------------------------------------
42.     --  F1: MC = MIN(A) * (MD * MB)
43.     procedure F1 (A : in Vector; MD, MB: in Matrix; MC: out Matrix) is
44.     begin
45.         MC := MIN(A) * (MD * MB);
46.     end F1;
47.
48.     -- F2: MN = MAX(ML) * (MK * MO)
49.     procedure F2 (ML, MK, MO: in Matrix; MN: out Matrix) is
50.     begin
51.         MN := MAX(ML) * (MK * MO);
52.     end F2;
53.
54.     -- F3: Z = SORT(R) * SORT(MW * MV)
55.     procedure F3 (R: in Vector; MW, MV: in Matrix; Z: out Vector) is
56.     begin
57.         Z := SORT(R) * SORT(MW * MV);
58.     end F3;
59.
60.     ------------------------------------------------------------------
61.     --        additional functions for calculation (realization)      --
62.     ------------------------------------------------------------------
63.     -- Matrix * Matrix
64.     function "*" (Left  :  Matrix; Right :  Matrix) return Matrix is
65.         Result : Matrix;
66.     begin
67.         for i in Index loop
68.             for J in Index loop
69.                 Result(I)(J) := 0;
70.                     for K in Index loop
71.                         Result(I)(J) := Result(I)(J) + Left(I)(K) * Right(K)(J);
72.                     end loop;
73.                 end loop;
74.         end loop;
75.         return Result;
76.     end  "*";
77.
78.     -- Vector * Matrix
79.     function "*" (Left : Vector; Right : Matrix) return Vector is
80.         R : Vector;
81.     begin
82.         for J in Index loop
83.             R(j) := 0;
84.             begin
85.                 for K in Index loop
86.                     R(J) := R(J) + Left(K) * Right(K)(J);
87.                 end loop;
88.             end;
89.         end loop;
90.         return R;
91.     end "*";
92.
93.     -- Integer * Matrix
94.     function "*" (Left : Integer; Right : Matrix) return Matrix is
```

```
        Result : Matrix;
    begin
        for i in Index loop
            for j in Index loop
                Result(i)(j) := Left * Right(i)(j);
            end loop;
        end loop;
        return Result;
    end "*";

    -- Maximum value of the matrix
    function Max (A : Matrix) return Integer is
        Var : Integer := A(1)(1);
    begin
        for i in Index loop
            for j in Index loop
                if A(i)(j) > Var then
                    Var := A(i)(j);
                end if;
            end loop;
        end loop;
        return Var;
    end Max;

    -- Minimum value of the vector
    function Min (A : Vector) return Integer is
        Var : Integer := A(1);
    begin
        for i in Index loop
            if A(i) < Var then
                Var := A(i);
            end if;
        end loop;
        return Var;
    end Min;

    -- vector sorting
    function Sort (A : Vector) return Vector is
        M : Vector := A;
        Buf : Integer;
        K : Integer;
    begin
        for i in 1..(Size - 1) loop
            K := i;
            for j in (i + 1)..Size loop
                if M(k) > M(j) then
                    K := j;
                end if;
            end loop;
            Buf := M(k);
            M(k) := M(i);
            M(i) := buf;
        end loop;
        return M;
    end Sort;

    -- matrix sorting
    function Sort (MA : Matrix) return Matrix is
```

```
153.         MT : Matrix;
154.     begin
155.         for i in Index loop
156.             MT(i) := Sort(MA(i));
157.         end loop;
158.         return MT;
159.     end Sort;
160.
161.     ------------------------------------------------------------------------
162.     --                      input\output functions:                       --
163.     ------------------------------------------------------------------------
164.     -- input values for vectors
165.     procedure Input (V : out Vector) is
166.     begin
167.         for I in Index loop
168.             V(I) := Value;
169.         end loop;
170.     end Input;
171.
172.     -- input values for matrices
173.     procedure Input (MA : out Matrix) is
174.     begin
175.         for I in Index loop
176.             for J in Index loop
177.                 MA(I)(J) := Value;
178.             end loop;
179.         end loop;
180.     end Input;
181.
182.     -- output values of vectors
183.     procedure Output (V : in Vector) is
184.     begin
185.         if (Size < 10) then
186.             New_Line;
187.             for I in Index loop
188.                 Put(Item => V(I), Width => 5);
189.             end loop;
190.             New_Line;
191.         end if;
192.     end Output;
193.
194.     -- output values for matrices
195.     procedure Output (MA : in Matrix) is
196.     begin
197.         if (Size < 10) then
198.             New_Line;
199.             for I in Index loop
200.                 for J in Index loop
201.                     Put(Item => MA(i)(j), Width => 5);
202.                 end loop;
203.                 New_line;
204.             end loop;
205.         end if;
206.     end Output;
207.
208. end Data;

Compiling: data.ads (source file time stamp: 2015-09-29 10:14:02)
```

```
  1. ------------------------------------------------------------------------
  2. --                                                                    --
  3. --                      Parallel Programming                          --
  4. --         Laboratory work #2. Process in Ada                         --
  5. --                                                                    --
  6. --  File: lab1.ada                                                    --
  7. --  Task: F1: MC = MIN(A) * (MD * MB)                                 --
  8. --        F2: MN = MAX(ML) * (MK * MO)                                --
  9. --        F3: Z = SORT(R) * SORT(MW * MV)                             --
 10. --                                                                    --
 11. --  Author: Stepanov Alexander, group IP - 31                         --
 12. --  Date: 29.09.2015                                                  --
 13. --                                                                    --
 14. ------------------------------------------------------------------------
 15.
 16. generic
 17.
 18.     -- matrix and vector size
 19.     Size : in Natural;
 20.     -- matrix and vector filling values
 21.     Value : in Integer;
 22.
 23. package Data is
 24.
 25.     type Vector is private;
 26.     type Matrix is private;
 27.
 28.     -- F1: MC = MIN(A) * (MD * MB)
 29.     procedure F1 (A : in Vector; MD, MB: in Matrix; MC: out Matrix);
 30.     -- F2: MN = MAX(ML) * (MK * MO)
 31.     procedure F2 (ML, MK, MO: in Matrix; MN: out Matrix);
 32.     -- F3: Z = SORT(R) * SORT(MW * MV)
 33.     procedure F3 (R: in Vector; MW, MV: in Matrix; Z: out Vector);
 34.
 35.     -- input values for vectors
 36.     procedure Input (V : out Vector);
 37.     -- input values for matrices
 38.     procedure Input (MA : out Matrix);
 39.     -- output values of vectors
 40.     procedure Output (V : in Vector);
 41.     -- output values of matricescr
 42.     procedure Output (MA : in Matrix);
 43.
 44.     private
 45.         subtype Index is Integer range 1..Size;
 46.         type Vector is array (Index) of Integer;
 47.         type Matrix is array (Index) of Vector;
 48.
 49. end Data;

208 lines: No errors
```