Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №1**

З дисципліни «Основи паралельного програмування»

**Потоки в мові Ада**

ВИКОНАЛА:

Студентка ІІІ курсу ФІОТ

Групи ІВ-82

Лихацька Юлія

ПЕРЕВІРИВ:

Корочкін О. В.

Київ 2020 р.

**Варіант завдання:**

F1: MD = (B\*C)\*(MA\*ME)

F2: h = MAX(MF+MG\*(MH\*ML))

F3: s = MIN(MO\*MP+MS)

**Лістинг програми:**

*main.adb*

----------------------------------------------------------

-- Лабораторна робота з ОПП №1, варіант:

-- F1: MD = (B\*C)\*(MA\*ME)

-- F2: h = MAX(MF+MG\*(MH\*ML))

-- F3: s = MIN(MO\*MP+MS)

-- Виконала: Лихацька Юлія Дмитрівна, ІВ-82

-- 17.09.2020

----------------------------------------------------------

with Data, Text\_IO, Ada.Integer\_Text\_IO, System.Multiprocessors;

use Text\_IO, Ada.Integer\_Text\_IO, System.Multiprocessors;

procedure Main is

n: Integer := 6 ;

package data1 is new data (n);

use data1;

procedure tasks is

task T1 is

pragma Priority(1);

pragma Storage\_Size(100000);

pragma CPU(1);

end;

task body T1 is

B, C: Vector;

MA, ME: Matrix;

f1: Matrix;

begin

Put\_Line("T1 started");

VectorFilling1(B);

VectorFilling1(C);

MatrFilling1(MA);

MatrFilling1(ME);

f1:=Func1(B, C, MA, ME);

delay(0.5);

if n < 7 then

Put\_Line("---F1: MD = (B\*C)\*(MA\*ME)---");

MatrOut(f1);

New\_Line;

end if;

Put\_Line("T1 finished");

New\_Line;

New\_Line;

end T1;

task T2 is

pragma Priority(3);

pragma Storage\_Size(100000);

pragma CPU(2);

end;

task body T2 is

MF, MG, MH, ML: Matrix;

f2: Integer;

begin

Put\_Line("T2 started");

MatrFilling1(MF);

MatrFilling1(MG);

MatrFilling1(MH);

MatrFilling1(ML);

f2:=Func2(MF, MG, MH, ML);

delay(1.0);

if n < 7 then

Put\_Line("---F2: h = MAX(MF+MG\*(MH\*ML))---");

NumberOut(f2);

New\_Line;

end if;

Put\_Line("T2 finished");

New\_Line;

New\_Line;

end T2;

task T3 is

pragma Priority(5);

pragma Storage\_Size(100000);

pragma CPU(2);

end;

task body T3 is

MO, MP, MS : Matrix;

f3: Integer;

begin

Put\_Line("T3 started");

MatrFilling1(MO);

MatrFilling1(MP);

MatrFilling1(MS);

f3:=Func3(MO, MP, MS);

delay(1.5);

if n < 7 then

Put\_Line("---F3: s = MIN(MO\*MP+MS)---");

NumberOut(f3);

New\_Line;

end if;

Put\_Line("T3 finished");

New\_Line;

New\_Line;

end T3;

begin

null;

end tasks;

begin

tasks;

end Main;

*data.adb*

with Text\_IO, Ada.Integer\_Text\_IO;

use Text\_IO, Ada.Integer\_Text\_IO;

package body data is

--Write number to console

procedure NumberOut(a: in Integer) is

begin

Put(a);

Put(" ");

end NumberOut;

--Write matrix to console

procedure MatrOut (A: in Matrix) is

begin

for i in 1..n loop

for j in 1..n loop

Put(A(i)(j));

Put(" ");

end loop;

Put\_Line(" ");

end loop;

end MatrOut;

--multiply 2 matrices

function MultMatr(A, B: in Matrix) return Matrix is

P: Matrix;

S: Integer;

begin

for k in 1..n loop

for i in 1..n loop

s := 0;

for j in 1..n loop

S := S + A(k)(j)\*B(j)(i);

P(k)(i) := s;

end loop;

end loop;

end loop;

return P;

end MultMatr;

--multiply vector and matrix

function VectorMatrMult(A: in Vector; B: in Matrix) return Vector is

P: Vector;

s: Integer;

begin

for i in 1..n loop

s := 0;

for j in 1..n loop

S := s + A(i)\*B(j)(i);

end loop;

P(i) := S;

end loop;

return P;

end VectorMatrMult;

--multiply matrix and integer

function MatrNumMult(A: in Matrix; k: in Integer) return Matrix is

P: Matrix;

begin

for i in 1..n loop

for j in 1..n loop

P(i)(j):=A(i)(j)\*k;

end loop;

end loop;

return P;

end MatrNumMult;

--Find max element in Matrix

function MatrMax(A : in Matrix) return Integer is

CurrentMax : Integer := A(1)(1);

begin

for i in 1..n loop

for j in 1..n loop

if A(i)(j) > CurrentMax then

CurrentMax := A(i)(j);

end if;

end loop;

end loop;

return CurrentMax;

end MatrMax;

--Find min element in Matrix

function MatrMin(A : in Matrix) return Integer is

CurrentMin : Integer := A(1)(1);

begin

for i in 1..n loop

for j in 1..n loop

if A(i)(j) < CurrentMin then

CurrentMin := A(i)(j);

end if;

end loop;

end loop;

return CurrentMin;

end MatrMin;

--add 2 matrices

function MatrAdd (MA,MB: in Matrix) return Matrix is

M: Matrix;

begin

for i in 1..n loop

for j in 1..n loop

M(i)(j):=MA(i)(j)+MB(i)(j);

end loop;

end loop;

return M;

end MatrAdd;

-- multiply 2 vectors

function VectMult(A,B: in Vector) return Integer is

s: Integer := 0;

begin

for i in 1..n loop

s:=s+A(i)\*B(i);

end loop;

return s;

end VectMult;

-- fill matrix with ones

procedure MatrFilling1(A: out Matrix) is

begin

for i in 1..n loop

for j in 1..n loop

A(i)(j) := 1;

end loop;

end loop;

end MatrFilling1;

--Filling vector with ones

procedure VectorFilling1 (A: out vector) is

begin

for i in 1..n loop

A(i) := 1;

end loop;

end VectorFilling1;

function Func1 (B, C: in Vector; MA, ME : in Matrix) return Matrix is

a: Integer;

MD: Matrix;

begin

a := VectMult(B, C);

MD := MultMatr(MA, ME);

MD := MatrNumMult(MD, a);

return MD;

end Func1;

function Func2 (MF, MG, MH, ML: in Matrix) return Integer is

max: Integer;

MA: Matrix;

MB: Matrix;

MC: Matrix;

begin

MA := MultMatr(MH, ML);

MB := MultMatr(MG, MA);

MC := MatrAdd(MF, MB);

max := MatrMax(MC);

return max;

end Func2;

function Func3 (MO, MP, MS : in Matrix) return Integer is

MA:Matrix;

min: Integer;

begin

MA := MultMatr(MO, MP);

MA := MatrAdd(MA, MS);

min := MatrMin(MA);

return min;

end Func3;

end data;

*data.ads*

generic

n: Integer;

package data is

---Declaration of private types

type Vector is private;

type Matrix is private;

--Write vector on screen

procedure NumberOut(A: in Integer);

--Write matrix on screen

procedure MatrOut (A: in Matrix);

--Calculation function 1

function Func1 (B, C: in Vector; MA, ME : in Matrix) return Matrix;

--Calculation function 2

function Func2 (MF, MG, MH, ML: in Matrix) return Integer;

--Calculation function 3

function Func3 (MO, MP, MS : in Matrix) return Integer;

--Filling matrix with ones

procedure MatrFilling1(A: out Matrix);

--Filling vector with ones

procedure VectorFilling1 (A: out vector);

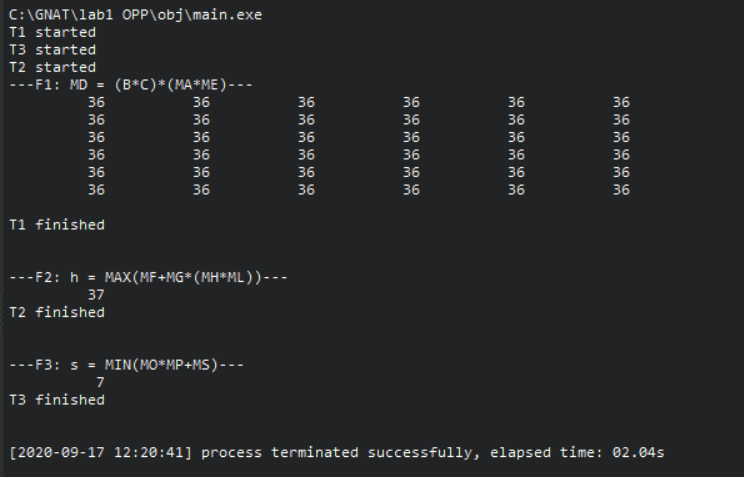
--Determination private types

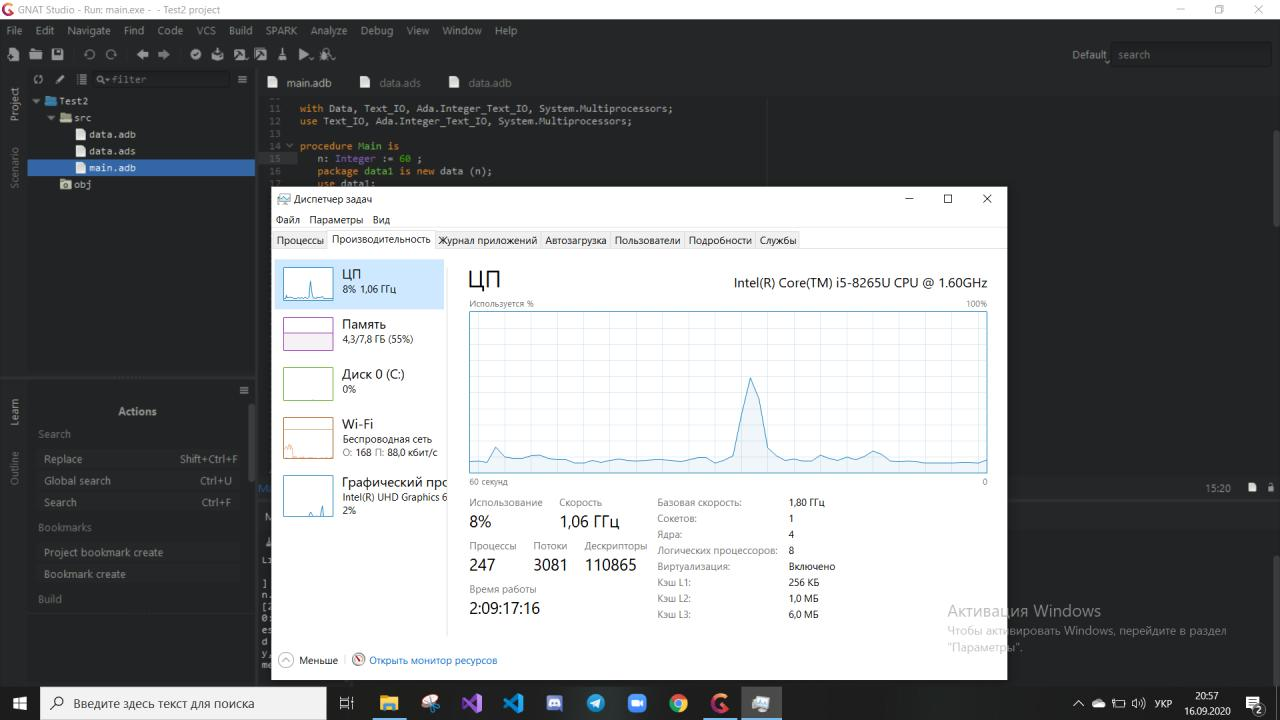
private

type Vector is array (1..n) of Integer;

type Matrix is array (1..n) of Vector;

end data;

**Результат роботи програми:**

ЦП для n=60:

**Висновки:**

Під час виконання завдання виникали труднощі з введенням даних у матриці та вектори з клавіатури, через що було прийнято заповнювати матриці та вектори одиницями.

Також для коректності відображення даних у консолі було зроблено обмеження розмірності матриць та векторів до 7. Якщо розмірність матриці або вектора більша, на екран нічого не виводиться, а тільки навантажується процесор.

Для виведення результатів послідовно та у правильному порядку, була використана команду delay() з різними параметрами: 0.5s, 1s, 1.5s.