

Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

**Лабораторна робота №7**

“Ада. Рандеву”

з дисципліни “Програмування для Паралельних Компьютерних Систем”

Виконав:

студент 3 курсу групи ІО-52

Бояршин Ігор

Номер заліковки: 5207

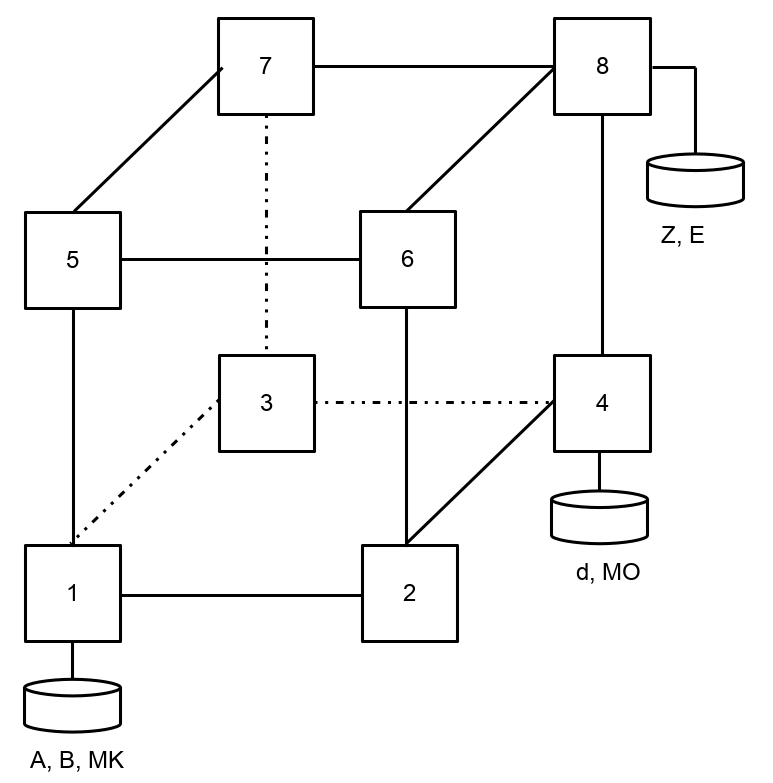
Перевірив:

Корочкін О.В.

Київ 2018 р.

**Техническое задание**

Структура паралельной компъютерной системы с общей памятью:



Выражение для подсчёта:

A = B\*d + min(Z) \* E \* (MO\*MK)

Язык программирования:

Ада

Средства взаимодействия задач:

Рандеву

**Выполнение работы**

**Этап 1:** Разработка параллельного математического алгоритма

1. xi = min(ZH) , i = (1..P), P – количество потоков, P = 8
2. x = min(x; xi) , i = (1..P)
3. AH = BH \* d + x \* E \* (MOH \* MK)

**Этап 2:** Разработка алгоритмов потоков

**Задача Т1**

1. Ввод B, MK
2. Передать в Т2: B4H, MK
3. Передать в Т3: B2H, MK
4. Принять из Т3: d, MO2H
5. Передать в Т5: BH, MK, d, MOH
6. Принять из Т5: ZH, E
7. Счёт1: x := min(ZH)
8. Принять из Т2: x\_other
9. Счёт: x := min(x, x\_other)
10. Принять из Т3: x\_other
11. Счёт: x := min(x, x\_other)
12. Принять из Т5: x\_other
13. Счёт: x := min(x, x\_other)
14. Передать в Т5: x
15. Передать в Т3: x
16. Передать в Т2: x
17. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
18. Принять из Т2: AH
19. Принять из Т3: A2H
20. Принять из Т5: A4H
21. Вывод А

**Задача Т2**

1. Принять из Т1: B4H, MK
2. Принять из Т4: d, MO2H
3. Передать в Т4: B2H, MK
4. Принять из Т6: ZH, E
5. Передать в Т6: BH, MK, d, MOH
6. Счёт1: x := min(ZH)
7. Передать в Т1: x
8. Принять из Т1: x\_new
9. Счёт: x:=x\_new
10. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
11. Передать в Т1: AH

**Задача Т3**

1. Принять из Т4: d, MO4H
2. Принять из Т1: B2H, MK
3. Передать в Т1: d, MO2H
4. Принять из Т7: ZH, E
5. Передать в Т7: BH, MK, d, MOH
6. Счёт1: x := min(ZH)
7. Принять из Т4: x\_other
8. Счёт: x := min(x, x\_other)
9. Передать в Т1: х
10. Принять из Т1: x\_new
11. Счёт: x:=x\_new
12. Передать в Т4: x
13. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
14. Принять из Т4: AH
15. Передать в Т1: A2H

**Задача Т4**

1. Ввод d, MO
2. Передать в Т3: d, MO4H
3. Передать в Т2: d, MO2H
4. Принять из Т2: B2H, MK
5. Принять из Т8: ZH, E
6. Передать в Т8: BH, MK, d, MOH
7. Счёт1: x := min(ZH)
8. Передать в Т3: x
9. Принять из Т3: x\_new
10. Счёт: x:=x\_new
11. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
12. Передать в Т3: AH

**Задача Т5**

1. Принять из Т7: Z2H, E
2. Принять из Т1: BH, MK, d, MOH
3. Передать в Т1: ZH, E
4. Счёт1: x := min(ZH)
5. Принять из Т6: x\_other
6. Счёт: x := min(x, x\_other)
7. Принять из Т7: x\_other
8. Счёт: x := min(x, x\_other)
9. Передать в Т1: x
10. Принять из Т1: x\_new
11. Счёт: x:=x\_new
12. Передать в Т7: x
13. Передать в Т6: x
14. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
15. Принять из Т6: AH
16. Принять из Т7: A2H
17. Передать в Т1: A4H

**Задача Т6**

1. Принять из Т8: Z2H, E
2. Передать в Т2: ZH, E
3. Принять из Т2: BH, MK, d, MOH
4. Счёт1: x := min(ZH)
5. Передать в Т5: х
6. Принять из Т5: x\_new
7. Счёт: x:=x\_new
8. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
9. Передать в Т5: AH

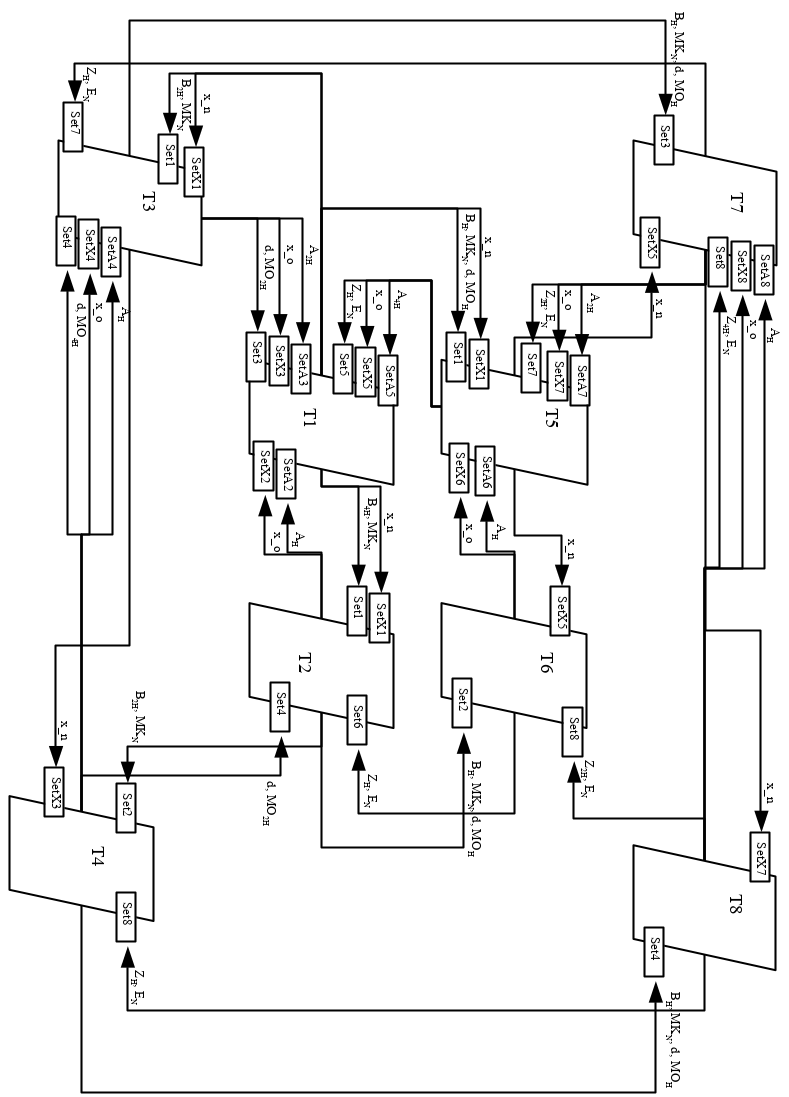
**Задача Т7**

1. Принять из Т8: Z4H, E
2. Передать в Т5: Z2H, E
3. Передать в Т3: ZH, E
4. Принять из Т3: BH, MK, d, MOH
5. Счёт1: x := min(ZH)
6. Принять из Т8: x\_other
7. Счёт: x := min(x, x\_other)
8. Передать в Т5: x
9. Принять из Т5: x\_new
10. Счёт: x:=x\_new
11. Передать в Т8: x
12. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
13. Принять из Т8: AH
14. Передать в Т5: A2H

**Задача Т8**

1. Ввод Z, E
2. Передать в T7: Z4H, E
3. Передать в Т6: Z2H, E
4. Передать в Т4: ZH, E
5. Принять из Т4: BH, MK, d, MOH
6. Счёт1: x := min(ZH)
7. Передать в Т7: x
8. Принять из Т7: x\_new
9. Счёт: x:=x\_new
10. Счёт2: AH = BH \* d + x \* E \* (MOH \* MK)
11. Передать в Т7: AH

**Этап 3:** Разработка схемы взаимодействия потоков



**Этап 4:** Разработка программы

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

-- Lab7: Ada. Randezvous

-- Task: A = B\*d + min(Z) \* E \* (MO\*MK)

-- Author: Igor Boyarshin

-- Date: 01.05.2018

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

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

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

procedure Main

is

N: constant := 3200;

P: constant := 8;

H: constant := N / P;

Memory: constant := 5000000;

Output\_Threshold: constant := 16;

package My\_Data is new Data(N, H);

use My\_Data;

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

procedure Tasks\_Start is

-- Tasks specification

task T1 is

pragma Task\_Name("Task 1");

pragma Storage\_Size(Memory);

entry Set3(d\_in: in Integer; MO\_in: in Matrix\_2H);

entry Set5(Z\_in: in Vector\_H; E\_in: in Vector\_N);

entry SetX2(x\_other: in Integer);

entry SetX3(x\_other: in Integer);

entry SetX5(x\_other: in Integer);

entry SetA2(A\_in: in Vector\_H);

entry SetA3(A\_in: in Vector\_2H);

entry SetA5(A\_in: in Vector\_4H);

end T1;

task T2 is

pragma Task\_Name("Task 2");

pragma Storage\_Size(Memory);

entry Set1(B\_in: in Vector\_4H; MK\_in: in Matrix\_N);

entry Set4(d\_in: in Integer; MO\_in: in Matrix\_2H);

entry Set6(Z\_in: in Vector\_H; E\_in: in Vector\_N);

entry SetX1(x\_new: in Integer);

end T2;

task T3 is

pragma Task\_Name("Task 3");

pragma Storage\_Size(Memory);

entry Set4(d\_in: in Integer; MO\_in: in Matrix\_4H);

entry Set1(B\_in: in Vector\_2H; MK\_in: in Matrix\_N);

entry Set7(Z\_in: in Vector\_H; E\_in: in Vector\_N);

entry SetX4(x\_other: in Integer);

entry SetX1(x\_new: in Integer);

entry SetA4(A\_in: in Vector\_H);

end T3;

task T4 is

pragma Task\_Name("Task 4");

pragma Storage\_Size(Memory);

entry Set2(B\_in: in Vector\_2H; MK\_in: in Matrix\_N);

entry Set8(Z\_in: in Vector\_H; E\_in: in Vector\_N);

entry SetX3(x\_new: in Integer);

end T4;

task T5 is

pragma Task\_Name("Task 5");

pragma Storage\_Size(Memory);

entry Set7(Z\_in: in Vector\_2H; E\_in: in Vector\_N);

entry Set1(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H);

entry SetX6(x\_other: in Integer);

entry SetX7(x\_other: in Integer);

entry SetX1(x\_new: in Integer);

entry SetA6(A\_in: in Vector\_H);

entry SetA7(A\_in: in Vector\_2H);

end T5;

task T6 is

pragma Task\_Name("Task 6");

pragma Storage\_Size(Memory);

entry Set8(Z\_in: in Vector\_2H; E\_in: in Vector\_N);

entry Set2(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H);

entry SetX5(x\_new: in Integer);

end T6;

task T7 is

pragma Task\_Name("Task 7");

pragma Storage\_Size(Memory);

entry Set8(Z\_in: in Vector\_4H; E\_in: in Vector\_N);

entry Set3(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H);

entry SetX8(x\_other: in Integer);

entry SetX5(x\_new: in Integer);

entry SetA8(A\_in: in Vector\_H);

end T7;

task T8 is

pragma Task\_Name("Task 8");

pragma Storage\_Size(Memory);

entry Set4(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H);

entry SetX7(x\_new: in Integer);

end T8;

-- Tasks implementation

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

task body T1

is

A, B: access Vector\_N := new Vector\_N;

MK: access Matrix\_N := new Matrix\_N;

d, x: Integer;

MO: access Matrix\_2H := new Matrix\_2H;

Z: access Vector\_H := new Vector\_H;

E: access Vector\_N := new Vector\_N;

begin

Put\_Line(":> Starting Task 1...");

Fill\_Vector\_Ones(B);

Fill\_Matrix\_Ones(MK);

T2.Set1(B.all(4\*H+1..8\*H), MK.all);

T3.Set1(B.all(2\*H+1..4\*H), MK.all);

accept Set3(d\_in: in Integer; MO\_in: in Matrix\_2H) do

d:= d\_in;

MO.all := MO\_in;

end Set3;

T5.Set1(B.all(H+1..2\*H), MK.all, d, MO.all(H+1..2\*H));

accept Set5(Z\_in: in Vector\_H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set5;

x := Find\_Min(Z.all);

accept SetX2(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX2;

accept SetX3(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX3;

accept SetX5(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX5;

T5.SetX1(x);

T3.SetX1(x);

T2.SetX1(x);

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

accept SetA2(A\_in: in Vector\_H) do

A.all(H+1..2\*H) := A\_in;

end SetA2;

accept SetA3(A\_in: in Vector\_2H) do

A.all(2\*H+1..4\*H) := A\_in;

end SetA3;

accept SetA5(A\_in: in Vector\_4H) do

A.all(4\*H+1..8\*H) := A\_in;

end SetA5;

if (N <= Output\_Threshold) then

Output\_Vector(A);

end if;

Put\_Line(":> Finished Task 1");

end T1;

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

task body T2

is

A: access Vector\_H := new Vector\_H;

B: access Vector\_4H := new Vector\_4H;

d: Integer;

Z: access Vector\_H := new Vector\_H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_2H := new Matrix\_2H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 2...");

accept Set1(B\_in: in Vector\_4H; MK\_in: in Matrix\_N) do

B.all := B\_in;

MK.all := MK\_in;

end Set1;

accept Set4(d\_in: in Integer; MO\_in: in Matrix\_2H) do

d := d\_in;

MO.all := MO\_in;

end Set4;

T4.Set2(B.all(2\*H+1..4\*H), MK.all);

accept Set6(Z\_in: in Vector\_H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set6;

T6.Set2(B.all(H+1..2\*H), MK.all, d, MO.all(H+1..2\*H));

x := Find\_Min(Z.all);

T1.SetX2(x);

accept SetX1(x\_new: in Integer) do

x := x\_new;

end SetX1;

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

T1.SetA2(A.all);

Put\_Line(":> Finished Task 2");

end T2;

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

task body T3

is

A: access Vector\_2H := new Vector\_2H;

B: access Vector\_2H := new Vector\_2H;

d: Integer;

Z: access Vector\_H := new Vector\_H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_4H := new Matrix\_4H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 3...");

accept Set4(d\_in: in Integer; MO\_in: in Matrix\_4H) do

d := d\_in;

MO.all := MO\_in;

end Set4;

accept Set1(B\_in: in Vector\_2H; MK\_in: in Matrix\_N) do

B.all := B\_in;

MK.all := MK\_in;

end Set1;

T1.Set3(d, MO.all(2\*H+1..4\*H));

accept Set7(Z\_in: in Vector\_H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set7;

T7.Set3(B.all(H+1..2\*H), MK.all, d, MO.all(H+1..2\*H));

x := Find\_Min(Z.all);

accept SetX4(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX4;

T1.SetX3(x);

accept SetX1(x\_new: in Integer) do

x := x\_new;

end SetX1;

T4.SetX3(x);

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

accept SetA4(A\_in: in Vector\_H) do

A.all(H+1..2\*H) := A\_in;

end SetA4;

T1.SetA3(A.all);

Put\_Line(":> Finished Task 3");

end T3;

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

task body T4

is

A: access Vector\_H := new Vector\_H;

B: access Vector\_2H := new Vector\_2H;

d: Integer;

Z: access Vector\_H := new Vector\_H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_N := new Matrix\_N;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 4...");

d := 1;

Fill\_Matrix\_Ones(MO);

T3.Set4(d, MO.all(4\*H+1..8\*H));

T2.Set4(d, MO.all(2\*H+1..4\*H));

accept Set2(B\_in: in Vector\_2H; MK\_in: in Matrix\_N) do

B.all := B\_in;

MK.all := MK\_in;

end Set2;

accept Set8(Z\_in: in Vector\_H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set8;

T8.Set4(B.all(H+1..2\*H), MK.all, d, MO.all(H+1..2\*H));

x := Find\_Min(Z.all);

T3.SetX4(x);

accept SetX3(x\_new: in Integer) do

x := x\_new;

end SetX3;

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

T3.SetA4(A.all);

Put\_Line(":> Finished Task 4");

end T4;

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

task body T5

is

A: access Vector\_4H := new Vector\_4H;

B: access Vector\_H := new Vector\_H;

d: Integer;

Z: access Vector\_2H := new Vector\_2H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_H := new Matrix\_H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 5...");

accept Set7(Z\_in: in Vector\_2H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set7;

accept Set1(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H) do

B.all := B\_in;

MK.all := MK\_in;

d := d\_in;

MO.all := MO\_in;

end Set1;

T1.Set5(Z.all(H+1..2\*H), E.all);

x := Find\_Min(Z.all(1..H));

accept SetX6(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX6;

accept SetX7(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX7;

T1.SetX5(x);

accept SetX1(x\_new: in Integer) do

x := x\_new;

end SetX1;

T7.SetX5(x);

T6.SetX5(x);

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

accept SetA6(A\_in: in Vector\_H) do

A.all(H+1..2\*H) := A\_in;

end SetA6;

accept SetA7(A\_in: in Vector\_2H) do

A.all(2\*H+1..4\*H) := A\_in;

end SetA7;

T1.SetA5(A.all);

Put\_Line(":> Finished Task 5");

end T5;

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

task body T6

is

A: access Vector\_H := new Vector\_H;

B: access Vector\_H := new Vector\_H;

d: Integer;

Z: access Vector\_2H := new Vector\_2H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_H := new Matrix\_H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 6...");

accept Set8(Z\_in: in Vector\_2H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set8;

T2.Set6(Z.all(H+1..2\*H), E.all);

accept Set2(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H) do

B.all := B\_in;

MK.all := MK\_in;

d := d\_in;

MO.all := MO\_in;

end Set2;

x := Find\_Min(Z.all(1..H));

T5.SetX6(x);

accept SetX5(x\_new: in Integer) do

x := x\_new;

end SetX5;

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

T5.SetA6(A.all);

Put\_Line(":> Finished Task 6");

end T6;

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

task body T7

is

A: access Vector\_2H := new Vector\_2H;

B: access Vector\_H := new Vector\_H;

d: Integer;

Z: access Vector\_4H := new Vector\_4H;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_H := new Matrix\_H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 7...");

accept Set8(Z\_in: in Vector\_4H; E\_in: in Vector\_N) do

Z.all := Z\_in;

E.all := E\_in;

end Set8;

T5.Set7(Z.all(2\*H+1..4\*H), E.all);

T3.Set7(Z.all(H+1..2\*H), E.all);

accept Set3(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H) do

B.all := B\_in;

MK.all := MK\_in;

d := d\_in;

MO.all := MO\_in;

end Set3;

x := Find\_Min(Z.all(1..H));

accept SetX8(x\_other: in Integer) do

if (x\_other < x) then

x := x\_other;

end if;

end SetX8;

T5.SetX7(x);

accept SetX5(x\_new: in Integer) do

x := x\_new;

end SetX5;

T8.SetX7(x);

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

accept SetA8(A\_in: in Vector\_H) do

A.all(H+1..2\*H) := A\_in;

end SetA8;

T5.SetA7(A.all);

Put\_Line(":> Finished Task 7");

end T7;

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

task body T8

is

A: access Vector\_H := new Vector\_H;

B: access Vector\_H := new Vector\_H;

d: Integer;

Z: access Vector\_N := new Vector\_N;

E: access Vector\_N := new Vector\_N;

MO: access Matrix\_H := new Matrix\_H;

MK: access Matrix\_N := new Matrix\_N;

x: Integer;

begin

Put\_Line(":> Starting Task 8...");

Fill\_Vector\_Ones(Z);

Fill\_Vector\_Ones(E);

T7.Set8(Z.all(4\*H+1..8\*H), E.all);

T6.Set8(Z.all(2\*H+1..4\*H), E.all);

T4.Set8(Z.all(H+1..2\*H), E.all);

accept Set4(B\_in: in Vector\_H; MK\_in: in Matrix\_N; d\_in: Integer; MO\_in: in Matrix\_H) do

B.all := B\_in;

MK.all := MK\_in;

d := d\_in;

MO.all := MO\_in;

end Set4;

x := Find\_Min(Z.all(1..H));

T7.SetX8(x);

accept SetX7(x\_new: in Integer) do

x := x\_new;

end SetX7;

Calc(A.all(1..H), B.all(1..H), d, x, E.all, MO.all(1..H), MK.all);

T7.SetA8(A.all);

Put\_Line(":> Finished Task 8");

end T8;

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

begin

null;

end Tasks\_Start;

begin

Tasks\_Start;

end Main;