**Техническое задание**

МА= MB \* (MC + α \* MZ \* MK)



**Этап 1. Построение параллельного алгоритма**

1. МAH = MB \* (MСH + α \* MZ \* MKH)

**Этап 2. Разработка алгоритмов процессов (задач)**

**№ Алгоритм задачи Т1**

1. Ввод МВ
2. Передать МВ задаче Т2
3. Принять MСН от задачи Т2
4. Принять α, MКН от задачи Т2
5. Принять МZ от задачи Т2
6. Cчёт МAH = MB \* (MСH + α \* MZ \* MKH)
7. Принять MA4Н от задачи Т3
8. Вывод МА

**№ Алгоритм задачи Т2**

1. Ввод МС
2. Принять МВ от задачи Т1
3. Передать MСН задаче Т1
4. Принять α, MК2Н от задачи Т3
5. Передать α, MКН задаче Т1
6. Передать MС3Н задаче Т3
7. Передать МВ задаче Т3
8. Принять МZ от задачи Т3
9. Передать МZ задаче Т1
10. Cчёт МAH = MB \* (MСH + α \* MZ \* MKH)
11. Принять MA3Н от задачи Т3
12. Передать MA4Н задаче Т1

**№ Алгоритм задачи Т3**

1. Ввод MZ
2. Принять α, MК3Н от задачи Т4
3. Передать α, MК2Н задаче Т2
4. Передать МZ задаче Т4
5. Принять MС3Н от задачи Т2
6. Принять МВ от задачи Т2
7. Передать МZ задаче Т2
8. Передать MС2Н задаче Т4
9. Передать МВ задаче Т4
10. Cчёт МAH = MB \* (MСH + α \* MZ \* MKH)
11. Принять MA2Н от задачи Т4
12. Передать MA3Н задаче Т2

**№ Алгоритм задачи Т4**

1. Ввод α, MК
2. Передать α, MКН задаче Т5
3. Передать α, MК3Н задаче Т3
4. Принять МZ от задачи Т3
5. Принять MС2Н от задачи Т3
6. Принять МВ от задачи Т3
7. Передать МZ задаче Т5
8. Передать MСН задаче Т5
9. Передать МВ задаче Т5
10. Cчёт МAH = MB \* (MСH + α \* MZ \* MKH)
11. Принять MAН от задачи Т5
12. Передать MA2Н задаче Т3

**№ Алгоритм задачи Т5**

1. Принять МКH, α от задачи Т4
2. Принять МZ от задачи Т4
3. Принять MСН от задачи Т4
4. Принять МВ от задачи Т4
5. Cчёт МAH = MB \* (MСH + α \* MZ \* MKH)
6. Передать MAН задаче Т4

**Этап 3. Разработка схемы взаимодействия задач**

**Этап 4. Разработка программы.**

**Листинг программы**

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

with Package1; use Package1;

with Ada.Text\_IO; use Ada.Text\_IO;

with Ada.Integer\_Text\_IO; use Ada.Integer\_Text\_IO;

with Ada.Calendar; use Ada.Calendar;

with ada.float\_text\_io; use ada.float\_text\_io;

procedure CoursePRO is

N: integer := 1000; --size of structures

P: integer := 5; --count of processors

H: integer := N/P; --size of piece

time\_all: duration;

time1,time2: time;

A: Integer;

--òèïû îïòèìàëüíûå äëÿ ðàññûëêè äàííûõ ìåæäó çàäà÷àìè.

type vector is array(integer range <>) of integer;

Subtype Vector\_N is Vector(1..N);

type Matrix is array(integer range <>) of Vector\_N;

Subtype Matrix\_N is Matrix(1..N);

Subtype Matrix\_4H is Matrix(1..4\*H);

Subtype Matrix\_3H is Matrix(1..3\*H);

Subtype Matrix\_2H is Matrix(1..2\*H);

Subtype Matrix\_H is Matrix(1..H);

procedure START is

--Ñïåöèôèêàöèÿ çàäà÷

task T1 is

entry MCH(MC1: in Matrix\_H);

entry MKHAlfa(MK1: in Matrix\_H; Alfa1: in Integer);

entry MZZ(MZ1: in Matrix\_N);

entry REZ(MA1: in Matrix\_4H); --êóñî÷êè äàííûõ

end T1;

task T2 is

entry MBB(MB1: in Matrix\_N); --êóñî÷êè äàííûõ

entry MKHAlfa(MK1: in Matrix\_2H; Alfa1: in Integer);

entry MZZ(MZ1: in Matrix\_N);

entry REZ(MA1: in Matrix\_3H); --êóñî÷êè äàííûõ

end T2;

task T3 is

entry MKHAlfa(MK1: in Matrix\_3H; Alfa1: in Integer);

entry MCH(MC1: in Matrix\_3H);

entry MBB(MB1: in Matrix\_N);

entry REZ(MA1: in Matrix\_2H); --êóñî÷êè äàííûõ

end T3;

task T4 is

entry MZZ(MZ1: in Matrix\_N);

entry MCH(MC1: in Matrix\_2H);

entry MBB(MB1: in Matrix\_N);

entry REZ(MA1: in Matrix\_H); --êóñî÷êè äàííûõ

end T4;

task T5 is

entry MKHAlfa(MK1: in Matrix\_H; Alfa1: in Integer);

entry MZZ(MZ1: in Matrix\_N);

entry MCH(MC1: in Matrix\_H);

entry MBB(MB1: in Matrix\_N);

end T5;

--Ðåàëèçàöèÿ çàäà÷

task body T1 is

Sum1: Integer := 0;

Alfa: Integer;

MB, MZ, MA: Matrix\_N;

MC, MK: Matrix\_H;

MTx: Matrix\_H;

begin

put\_Line("T1 started!");

--1. Ââîä ÌÂ

for i in 1 .. N loop

for j in 1 .. N loop

MB(i)(j) := 1;

end loop;

end loop;

--2. Ïåðåäàòü ÌÂ çàäà÷å Ò2

T2.MBB(MB);

--3. Ïðèíÿòü MÑÍ îò çàäà÷è Ò2

accept MCH(MC1: in Matrix\_H) do

MC := MC1;

end MCH;

--4. Ïðèíÿòü ?, MÊÍ îò çàäà÷è Ò2

accept MKHAlfa(MK1: in Matrix\_H; Alfa1: in Integer) do

MK := MK1;

Alfa := Alfa1;

end MKHAlfa;

--5. Ïðèíÿòü ÌZ îò çàäà÷è Ò2

accept MZZ(MZ1: in Matrix\_N) do

MZ := MZ1;

end MZZ;

--6. C÷¸ò ÌAH = MB \* (MÑH + ? \* MZ \* MKH)

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MK(J)(Z) \* MZ(Z)(I);

end loop;

MTx(J)(I) := MC(J)(I) + Alfa \* Sum1;

end loop;

end loop;

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MTx(J)(Z) \* MB(Z)(I);

end loop;

MA(J)(I) := Sum1;

end loop;

end loop;

--7. Ïðèíÿòü MA4Í îò çàäà÷è Ò3

accept REZ(MA1: in Matrix\_4H) do

MA(H+1 .. N) := MA1;

end REZ;

--8. Âûâîä ÌÀ

if N < 30 then

for i in 1 .. N loop

for j in 1 .. N loop

Put(MA(i)(j), 4);

end loop;

put\_Line(" ");

end loop;

end if;

time2:=clock;

time\_all:=time2-time1;

Put(Integer(time\_all));

end T1;

task body T2 is

Sum1: Integer := 0;

Alfa: Integer;

MC, MB, MZ: Matrix\_N;

MK: Matrix\_2H;

MA: Matrix\_4H;

MTx: Matrix\_H;

begin

put\_Line("T2 started!");

--1. Ââîä ÌÑ

for i in 1 .. N loop

for j in 1 .. N loop

MC(i)(j) := 1;

end loop;

end loop;

--2. Ïðèíÿòü ÌÂ îò çàäà÷è Ò1

accept MBB(MB1: in Matrix\_N) do

MB := MB1;

end MBB;

--3. Ïåðåäàòü MÑÍ çàäà÷å Ò1

T1.MCH(MC(H+1 .. 2\*H));

--4. Ïðèíÿòü ?, MÊ2Í îò çàäà÷è Ò3

accept MKHAlfa(MK1: in Matrix\_2H; Alfa1: in Integer) do

MK := MK1;

Alfa := Alfa1;

end MKHAlfa;

--5. Ïåðåäàòü ?, MÊÍ çàäà÷å Ò1

T1.MKHAlfa(MK(H+1..2\*H), Alfa);

--6. Ïåðåäàòü MÑ3Í çàäà÷å Ò3

T3.MCH(MC(2\*H+1 .. N));

--7. Ïåðåäàòü ÌÂ çàäà÷å Ò3

T3.MBB(MB);

--8. Ïðèíÿòü ÌZ îò çàäà÷è Ò3

accept MZZ(MZ1: in Matrix\_N) do

MZ := MZ1;

end MZZ;

--9. Ïåðåäàòü ÌZ çàäà÷å Ò1

T1.MZZ(MZ);

--10. C÷¸ò ÌAH = MB \* (MÑH + ? \* MZ \* MKH)

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MK(J)(Z) \* MZ(Z)(I);

end loop;

MTx(J)(I) := MC(J)(I) + Alfa \* Sum1;

end loop;

end loop;

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MTx(J)(Z) \* MB(Z)(I);

end loop;

MA(J)(I) := Sum1;

end loop;

end loop;

--11. Ïðèíÿòü MA3Í îò çàäà÷è Ò3

accept REZ(MA1: in Matrix\_3H) do

MA(H+1 .. 4\*H) := MA1;

end REZ;

--12. Ïåðåäàòü MA4Í çàäà÷å Ò1

T1.REZ(MA);

Put\_Line("T2 ended");

end T2;

task body T3 is

Sum1: Integer := 0;

Alfa: Integer;

MB, MZ: Matrix\_N;

MC, MA, MK: Matrix\_3H;

MTx: Matrix\_H;

begin

put\_Line("T3 started!");

--1. Ââîä MZ

for i in 1 .. N loop

for j in 1 .. N loop

MZ(i)(j) := 1;

end loop;

end loop;

--2. Ïðèíÿòü ?, MÊ3Í îò çàäà÷è Ò4

accept MKHAlfa(MK1: in Matrix\_3H; Alfa1: in Integer) do

MK := MK1;

Alfa := Alfa1;

end MKHAlfa;

--3. Ïåðåäàòü ?, MÊ2Í çàäà÷å Ò2

T2.MKHAlfa(MK(H+1 .. 3\*H), Alfa);

--4. Ïåðåäàòü ÌZ çàäà÷å Ò4

T4.MZZ(MZ);

--5. Ïðèíÿòü MÑ3Í îò çàäà÷è Ò2

accept MCH(MC1: in Matrix\_3H) do

MC := MC1;

end MCH;

--6. Ïðèíÿòü ÌÂ îò çàäà÷è Ò2

accept MBB(MB1: in Matrix\_N) do

MB := MB1;

end MBB;

--7. Ïåðåäàòü ÌZ çàäà÷å Ò2

T2.MZZ(MZ);

--8. Ïåðåäàòü MÑ2Í çàäà÷å Ò4

T4.MCH(MC(H+1..3\*H));

--9. Ïåðåäàòü ÌÂ çàäà÷å Ò4

T4.MBB(MB);

--10. C÷¸ò ÌAH = MB \* (MÑH + ? \* MZ \* MKH)

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MK(J)(Z) \* MZ(Z)(I);

end loop;

MTx(J)(I) := MC(J)(I) + Alfa \* Sum1;

end loop;

end loop;

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MTx(J)(Z) \* MB(Z)(I);

end loop;

MA(J)(I) := Sum1;

end loop;

end loop;

--11. Ïðèíÿòü MA2Í îò çàäà÷è Ò4

accept REZ(MA1: in Matrix\_2H) do

MA(H+1 .. 3\*H) := MA1;

end REZ;

--12. Ïåðåäàòü MA3Í çàäà÷å Ò2

T2.REZ(MA);

Put\_Line("T3 ended");

end T3;

task body T4 is

Sum1: Integer := 0;

Alfa: Integer;

MB, MZ, MK: Matrix\_N;

MC, MA: Matrix\_2H;

MTx: Matrix\_H;

begin

put\_Line("T4 started!");

--1. Ââîä ?, MÊ

Alfa := 1;

for i in 1 .. N loop

for j in 1 .. N loop

MK(i)(j) := 1;

end loop;

end loop;

--2. Ïåðåäàòü ?, MÊÍ çàäà÷å Ò5

T5.MKHAlfa(MK(4\*H+1..N), Alfa);

--3. Ïåðåäàòü ?, MÊ3Í çàäà÷å Ò3

T3.MKHAlfa(MK(H+1..4\*H), Alfa);

--4. Ïðèíÿòü ÌZ îò çàäà÷è Ò3

accept MZZ(MZ1: in Matrix\_N) do

MZ := MZ1;

end MZZ;

--5. Ïðèíÿòü MÑ2Í îò çàäà÷è Ò3

accept MCH(MC1: in Matrix\_2H) do

MC := MC1;

end MCH;

--6. Ïðèíÿòü ÌÂ îò çàäà÷è Ò3

accept MBB(MB1: in Matrix\_N) do

MB := MB1;

end MBB;

--7. Ïåðåäàòü ÌZ çàäà÷å Ò5

T5.MZZ(MZ);

--8. Ïåðåäàòü MÑÍ çàäà÷å Ò5

T5.MCH(MC(H+1..2\*H));

--9. Ïåðåäàòü ÌÂ çàäà÷å Ò5

T5.MBB(MB);

--10. C÷¸ò ÌAH = MB \* (MÑH + ? \* MZ \* MKH)

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MK(J)(Z) \* MZ(Z)(I);

end loop;

MTx(J)(I) := MC(J)(I) + Alfa \* Sum1;

end loop;

end loop;

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MTx(J)(Z) \* MB(Z)(I);

end loop;

MA(J)(I) := Sum1;

end loop;

end loop;

--11. Ïðèíÿòü MAÍ îò çàäà÷è Ò5

accept REZ(MA1: in Matrix\_H) do

MA(H+1 .. 2\*H) := MA1;

end REZ;

--12. Ïåðåäàòü MA2Í çàäà÷å Ò3

T3.REZ(MA);

Put\_Line("T4 ended");

end T4;

task body T5 is

Sum1: Integer := 0;

Alfa: Integer;

MB, MZ: Matrix\_N;

MC, MA, MK: Matrix\_H;

MTx: Matrix\_H;

begin

put\_Line("T5 started!");

--1. Ïðèíÿòü MKH, ? îò çàäà÷è Ò4

accept MKHAlfa(MK1: in Matrix\_H; Alfa1: in Integer) do

MK := MK1;

Alfa := Alfa1;

end MKHAlfa;

--2. Ïðèíÿòü ÌZ îò çàäà÷è Ò4

accept MZZ(MZ1: in Matrix\_N) do

MZ := MZ1;

end MZZ;

--3. Ïðèíÿòü MÑÍ îò çàäà÷è Ò4

accept MCH(MC1: in Matrix\_H) do

MC := MC1;

end MCH;

--4. Ïðèíÿòü ÌÂ îò çàäà÷è Ò4

accept MBB(MB1: in Matrix\_N) do

MB := MB1;

end MBB;

--5. C÷¸ò ÌAH = MB \* (MÑH + ? \* MZ \* MKH)

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MK(J)(Z) \* MZ(Z)(I);

end loop;

MTx(J)(I) := MC(J)(I) + Alfa \* Sum1;

end loop;

end loop;

for i in 1.. N loop

for j in 1 .. H loop

sum1 := 0;

for z in 1 .. N loop

Sum1 := sum1 + MTx(J)(Z) \* MB(Z)(I);

end loop;

MA(J)(I) := Sum1;

end loop;

end loop;

--6. Ïåðåäàòü MAÍ çàäà÷å Ò4

T4.REZ(MA);

Put\_Line("T5 ended");

end T5;

begin

null;

end START;

begin

Put\_Line("N == ");

Put(N);

--Get(A);

time1:=clock;

START;

end CoursePRO;