Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Основи паралельного програмування

Лабораторна робота №4

**«Потоки в бібліотеці Win32»**

Виконала:

студентка групи ІВ-71

Молчанова В.С.

Перевірив:

Долголенко О.М.

Київ

2019 р.

**Завдання:**

F1: A = B + C + D\*(MD\*ME)

F2: MF = TRANS(MG) + MK\*ML

F3: T = MO\*P + (MR\*MS)\*S

**Лістинг програми:**

**Lab4.cpp**

#include <iostream>

#define WIN32\_LEAN\_AND\_MEAN

#include <Windows.h>

#include <process.h>

#include "math.h"

// Molchanova Varvara, IV-71

//

// F1: A = B + C + D\*(MD\*ME)

// F2: MF = TRANS(MG) + MK\*ML

// F3: T = MO\*P + (MR\*MS)\*S

const int N = 3;

int flag = 0;

int\* f1Result;

int\*\* f2Result;

int\* f3Result;

int\* f1(int b[], int c[], int d[], int\* md[n], int\* me[n]) {

    // A = B + C + D\*(MD\*ME)

    return add(b, add(c, mul(d, mul(md, me))));

}

int\*\* f2(int\* mg[n], int\* mk[n], int\* ml[n]) {

    // MF = TRANS(MG) + MK\*ML

    return add(mg, mul(mk, ml));

}

int\* f3(int s[], int p[], int\* mo[n], int\* mr[n], int\* ms[n]) {

    // T = MO\*P + (MR\*MS)\*S

    return add(mul(p, mo), mul(s, mul(mr, ms)));

}

void F1(PVOID pvoid) {

    std::cout << "Task 1 start\n";

    int a[n];

    vector\_fill(a, 1);

    int b[n];

    vector\_fill(b, 1);

    int d[n];

    vector\_fill(d, 1);

    int\* ma[n];

    matrix\_fill(ma, 1);

    int\* me[n];

    matrix\_fill(me, 1);

    f1Result = f1(a, b, d, ma, me);

    flag += 1;

    std::cout << "Task 1 end\n";

    \_endthread();

}

void F2(PVOID pvoid) {

    std::cout << "Task 2 start\n";

    int\* mg[n];

    matrix\_fill(mg, 1);

    int\* mk[n];

    matrix\_fill(mk, 1);

    int\* ml[n];

    matrix\_fill(ml, 1);

    f2Result = f2(mg, mk, ml);

    flag += 2;

    std::cout << "Task 2 end\n";

    \_endthread();

}

void F3(PVOID pvoid) {

    std::cout << "Task 3 start\n";

    int s[n];

    vector\_fill(s, 1);

    int p[n];

    vector\_fill(p, 1);

    int\* mo[n];

    matrix\_fill(mo, 1);

    int\* mr[n];

    matrix\_fill(mr, 1);

    int\* ms[n];

    matrix\_fill(ms, 1);

    f3Result = f3(s, p, mo, mr, ms);

    flag += 3;

    std::cout << "Task 3 end\n";

    \_endthread();

}

int main() {

    std::cout << "Lab 4 start" << std::endl << std::endl;

    \_beginthread(F1, 0, NULL);

    \_beginthread(F2, 0, NULL);

    \_beginthread(F3, 0, NULL);

    while (flag != 6)

    {

    }

    cout << endl << "F1 = " << endl;

    vector\_output(f1Result);

    cout << endl << "F2 = " << endl;

    matrix\_output(f2Result);

    cout << endl << "F3 = " << endl;

    vector\_output(f3Result);

    std::cout << std::endl << "Lab 4 end" << std::endl;

    system("pause");

}

**math.h**

#include <iostream>

#include <algorithm>

#include <functional>

using namespace std;

const int n = 3;

void vector\_fill(int v[n], int value);

void matrix\_fill(int\* matrix[n], int value);

void vector\_output(int vector[n]);

void matrix\_output(int\* matrix[n]);

int\* add(int a[n], int b[n]);

int\*\* add(int\* ma[n], int\* mb[n]);

int scalar\_mul(int a[n], int b[n]);

int\*\* mul(int a, int\* ma[n]);

int\* mul(int a[n], int\* ma[n]);

int\*\* mul(int\* ma[n], int\* mb[n]);

void vector\_sort(int v[n]);

int matrix\_max(int\* ma[n]);

int matrix\_min(int\* ma[n]);

**math.cpp**

#include <iostream>

#include <algorithm>

#include <functional>

#include "math.h"

using namespace std;

void vector\_fill(int v[n], int value) {

    for (auto i = 0; i < n; i++)

    {

        v[i] = value;

    }

}

void matrix\_fill(int\* matrix[n], int value) {

    for (auto i = 0; i < n; i++)

    {

        matrix[i] = new int[n];

        for (auto j = 0; j < n; j++) {

            matrix[i][j] = value;

        }

    }

}

void vector\_output(int vector[n])

{

    if (n <= 10)

    {

        for (auto i = 0; i < n; i++)

        {

            cout << vector[i] << " ";

        }

        cout << endl;

    }

}

void matrix\_output(int\* matrix[n])

{

    if (n <= 10)

    {

        for (auto i = 0; i < n; i++)

        {

            for (auto j = 0; j < n; j++)

            {

                cout << matrix[i][j] << " ";

            }

            cout << endl;

        }

    }

}

void vector\_sort(int v[])

{

    sort(v, v + n);

}

int matrix\_min(int\* ma[n]) {

    auto n = ma[0][0];

    for (auto i = 0; i < n; i++) {

        for (auto j = 0; j < n; j++) {

            n = (n < ma[i][j]) ? n : ma[i][j];

        }

    }

    return n;

}

int matrix\_max(int\* ma[n]) {

    auto n = ma[0][0];

    for (auto i = 0; i < n; i++) {

        for (auto j = 0; j < n; j++) {

            n = (n > ma[i][j]) ? n : ma[i][j];

        }

    }

    return n;

}

int\* add(int a[], int b[]) {

    int\* result = new int[n];

    for (auto i = 0; i < n; i++)

    {

        result[i] = a[i] + b[i];

    }

    return result;

}

int\*\* add(int\* ma[n], int\* mb[n]) {

    int\*\* result = new int\*[n];

    for (auto i = 0; i < n; i++) {

        result[i] = new int[n];

        for (auto j = 0; j < n; j++) {

            result[i][j] = ma[i][j] + mb[i][j];

        }

    }

    return result;

}

int scalar\_mul(int a[], int b[])

{

    auto c = 0;

    for (auto i = 0; i < n; i++) {

        c += a[i] \* b[i];

    }

    return c;

}

int\*\* mul(int a, int\* ma[n])

{

    int\*\* result = new int\*[n];

    for (auto i = 0; i < n; i++) {

        result[i] = new int[n];

        for (auto j = 0; j < n; j++) {

            result[i][j] = ma[i][j] \* a;

        }

    }

    return result;

}

int\* mul(int a[], int\* ma[n]) {

    int\* result = new int[n];

    for (auto j = 0; j < n; j++)

    {

        result[j] = 0;

        for (auto k = 0; k < n; k++)

        {

            result[j] += a[k] \* ma[k][j];

        }

    }

    return result;

}

int\*\* mul(int\* ma[n], int\* mb[n])

{

    int\*\* result = new int\*[n];

    for (auto i = 0; i < n; i++) {

        result[i] = new int[n];

        for (auto j = 0; j < n; j++) {

            result[i][j] = 0;

            for (auto k = 0; k < n; k++)

            {

                result[i][j] += ma[i][k] \* mb[k][j];

            }

        }

    }

    return result;

}

**Результат роботи:**

Lab 4 start

Task 1 start

Task 1 end

Task 3 start

Task 2 start

Task 2 end

Task 3 end

F1 =

11 11 11

F2 =

4 4 4

4 4 4

4 4 4

F3 =

12 12 12

Lab 4 end

Для продолжения нажмите любую клавишу . . .