/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Laba 6

\*\* Gurban Miroslav

\*\* IO - 91

\*\* 1.5 C = SORT(A) - SORT(B)

\*\* 2.18 c = MIN(MA \* MB)

\*\* 3.11 D = SORT(A - M) \* TRANS(MC)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "stdafx.h"

#include <iostream>

#include <conio.h>

#include <windows.h>

#include <stdio.h>

#include <omp.h>

using std::cout;

using std::endl;

const int n = 6000;

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

void VectorInput(int V[n])

{

for (int i = 0; i < n; i++)

{

V[i] = 1;

}

}

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

void MatrixInput(int M[n][n])

{

for (int i = 0; i < n; i++)

{

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

{

M[i][j] = 1;

}

}

}

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

void VectorOutput(int V[n])

{

if (n <= 10)

{

for (int i = 0; i < n; i++)

{

cout<<V[i]<<" ";

}

cout<<endl;

}

}

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

void MatrixOutput(int M[n][n])

{

if (n < 10)

{

for (int i = 0; i < n; i++)

{

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

{

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

}

cout<<endl;

}

cout<<endl;

}

}

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

void SortVector(int V[])

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n - 1; j++)

{

if (V[j] > V[j + 1])

{

int z = V[j];

V[j] = V[j + 1];

V[j + 1] = z;

}

}

}

}

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

//1.5 C = SORT(A) - SORT(B)

void GoFunc1()

{

cout<<"Func1 started"<<endl;

int A[n];

int B[n];

int C[n];

VectorInput(A);

VectorInput(B);

SortVector(A);

SortVector(B);

for (int i = 0; i < n; i++)

{

C[i] = A[i] - B[i];

}

VectorOutput(C);

cout<<"Func1 finished."<<endl;

}//GoFunc1

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

//2.18 c = MIN(MA \* MB)

void GoFunc2()

{

cout<<"Func2 started"<<endl;

int AM[n][n];

int BM[n][n];

int CM[n][n];

MatrixInput(AM);

MatrixInput(BM);

for (int i = 0; i < n; i++)

{

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

{

CM[i][j] = 0;

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

{

CM[i][j] += AM[i][k] \* BM[k][j];

}

}

}

int z = CM[0][0];

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n - 1; j++)

{

if (z > CM[i][j])

{

z = CM[i][j];

}

}

}

cout<<z<<endl;

cout<<"Func2 finished."<<endl;

}//GoFunc2

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

//3.11 D = SORT(A - M) \* TRANS(MC)

void GoFunc3()

{

cout<<"Func3 started"<<endl;

int A[n];

int M[n];

int C[n];

int D[n];

VectorInput(A);

VectorInput(M);

int CM[n][n];

MatrixInput(CM);

for (int i = 0; i < n; i++)

{

C[i] = A[i] - M[i];

}

SortVector(C);

for (int i = 1; i < n; i++)

{

for (int j = 0; j < i; j++)

{

int b = CM[i][j];

CM[i][j] = CM[j][i];

CM[j][i] = b;

}

}

for (int i = 0; i < n; i++)

{

D[i] = 0;

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

{

D[i] += C[j] \* CM[j][i];

}

}

VectorOutput(D);

cout<<"Func3 finished."<<endl;

}//GoFunc3

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

int \_tmain(int argc, \_TCHAR\* argv[]) {

cout<<"\* - Laba6 started"<<endl;

#pragma comment(linker, "/STACK:500000000")

#pragma omp parallel sections num\_threads(3)

{

#pragma omp sections

{

#pragma omp section

{

GoFunc1();

}

#pragma omp section

{

GoFunc2();

}

#pragma omp section

{

GoFunc3();

}

}

}

cout<<"\* - Laba6 finished."<<endl;

getchar();

}