

**Міністерство освіти, науки, молоді та спорту України**

**Національний технічний університет України**

**«Київський політехнічний інститут»**

**Фізико-технічний інститут**

**Лабораторна робота №2**

**Методи обчислень**

**«РОЗВ’ЯЗАННЯ СИСТЕМ**

**ЛІНІЙНИХ АЛГЕБРАЇЧНИХ РІВНЯНЬ (СЛАР) ПРЯМИМИ МЕТОДАМ»**

**Варіант 4**

Підготував:

студент 3 курсу

групи ФІ-84

Коломієць Андрій Юрійович

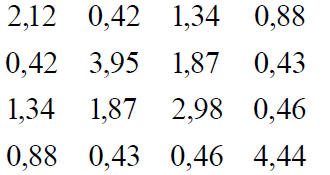
Викладач:

Стьопочкіна Ірина Валеріївна

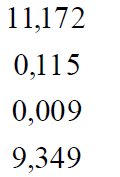
**Київ – 2021**

**Вхідна система**

Матриця А

****

Вектор В



**Результати роботи програми**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Work of program which solve matrix**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Matrix left side**

**2.12 0.42 1.34 0.88**

**0.42 3.95 1.87 0.43**

**1.34 1.87 2.98 0.46**

**0.88 0.43 0.46 4.44**

**Matrix right side**

**11.172**

**0.115**

**0.009**

**9.349**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Process of finding solving matrix...**

**Direct course factorization:**

**-step with index i=0 j=0:**

**1.45602 0 0 0**

**0 0 0 0**

**0 0 0 0**

**0 0 0 0**

**-step with index i=1 j=0:**

**1.45602 0 0 0**

**0.288457 0 0 0**

**0 0 0 0**

**0 0 0 0**

**-step with index i=2 j=0:**

**1.45602 0 0 0**

**0.288457 0 0 0**

**0.920316 0 0 0**

**0 0 0 0**

**-step with index i=3 j=0:**

**1.45602 0 0 0**

**0.288457 0 0 0**

**0.920316 0 0 0**

**0.604386 0 0 0**

**-step with index i=1 j=1:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0 0 0**

**0.604386 0 0 0**

**-step with index i=2 j=1:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 0 0**

**0.604386 0 0 0**

**-step with index i=2 j=2:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0 0 0**

**-step with index i=3 j=1:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 0 0**

**-step with index i=3 j=2:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 0**

**-step with index i=3 j=3:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=0 j=1:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=0 j=2:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=0 j=3:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=1 j=2:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=1 j=3:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**-step with index i=2 j=3:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**Temp solving step:**

**-step with index i=0:**

**7.67296**

**0**

**0**

**0**

**-step with index i=1:**

**7.67296**

**-1.06708**

**0**

**0**

**-step with index i=2:**

**7.67296**

**-1.06708**

**-5.10353**

**0**

**-step with index i=3:**

**7.67296**

**-1.06708**

**-5.10353**

**1.99151**

**Matrix of factorization:**

**1.45602 0 0 0**

**0.288457 1.96642 0 0**

**0.920316 0.815966 1.21129 0**

**0.604386 0.130013 -0.167023 2.00747**

**Transponate matrix of factorization:**

**1.45602 0.288457 0.920316 0.604386**

**0 1.96642 0.815966 0.130013**

**0 0 1.21129 -0.167023**

**0 0 0 2.00747**

**Solving step:**

**-step with index i=4:**

**0**

**0**

**0**

**0.992054**

**-step with index i=2:**

**0**

**0**

**-4.07652**

**0.992054**

**-step with index i=1:**

**0**

**1.08331**

**-4.07652**

**0.992054**

**-step with index i=0:**

**7.22006**

**1.08331**

**-4.07652**

**0.992054**

**Inconspicuous vector:**

**1.77636e-15**

**2.63678e-16**

**1.75207e-16**

**0**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Код**

#include <iostream>

#include <string>

#include <cmath>

#include <stdio.h>

#include <windows.h>

#include <conio.h>

using namespace std;

#define \_type\_ long double

#define line cout<<endl<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

\_type\_\*\* find\_lower\_matrix(\_type\_\*\* matrix\_left, \_type\_\*\* matrix\_lower, int size\_matrix);

\_type\_\* temp\_solving(\_type\_\*\* matrix\_lower, \_type\_\* temp\_matrix, \_type\_\* matrix\_right, int size\_matrix);

\_type\_\* finding\_solving(\_type\_\*\* matrix\_lower, \_type\_\* temp\_matrix, \_type\_\* solving, int size\_matrix);

\_type\_\* inconspicuous(\_type\_\*\* matrix\_left, \_type\_\* matrix\_right, \_type\_\* solving, \_type\_\* temp\_matrix, int size\_matrix);

void output\_vector(\_type\_\* matrix\_right, int size\_matrix);

void output\_double\_matrix(\_type\_\*\* matrix\_left, int size\_matrix);

int main()

{

int size\_matrix=4;

\_type\_\*\* matrix\_left = new \_type\_\*[size\_matrix];

for (int i = 0; i < size\_matrix; i++)

{

matrix\_left[i] = new \_type\_[size\_matrix];

}

for (int i = 0; i < size\_matrix; i++)

{

for (int j = 0; j < size\_matrix; j++)

{

matrix\_left[i][j] = 0;

}

}

\_type\_\* matrix\_right = new \_type\_[size\_matrix];

for (int i = 0; i < size\_matrix; i++)

{

matrix\_right[i] = 0;

}

\_type\_\*\* matrix\_lower = new \_type\_\*[size\_matrix];

for (int i = 0; i < size\_matrix; i++)

{

matrix\_lower[i] = new \_type\_[size\_matrix];

}

for (int i = 0; i < size\_matrix; i++)

{

for (int j = 0; j < size\_matrix; j++)

{

matrix\_lower[i][j] = 0;

}

}

\_type\_\* temp\_matrix = new \_type\_[size\_matrix];

for (int i = 0; i < size\_matrix; i++)

{

temp\_matrix[i] = 0;

}

\_type\_\* solving = new \_type\_[size\_matrix];

for (int i = 0; i < size\_matrix; i++)

{

solving[i] = 0;

}

//////////////////////////////Progam start//////////////////////////////

line

cout << endl << "Work of program which solve matrix" << endl;

line

cout << endl << "Matrix left side" << endl << endl;

matrix\_left[0][0] = 2.12; matrix\_left[0][1] = 0.42; matrix\_left[0][2] = 1.34; matrix\_left[0][3] = 0.88; matrix\_right[0] =11.172;

matrix\_left[1][0] = 0.42; matrix\_left[1][1] = 3.95; matrix\_left[1][2] = 1.87; matrix\_left[1][3] = 0.43; matrix\_right[1] =0.115;

matrix\_left[2][0] = 1.34; matrix\_left[2][1] = 1.87; matrix\_left[2][2] = 2.98; matrix\_left[2][3] = 0.46; matrix\_right[2] =0.009;

matrix\_left[3][0] = 0.88; matrix\_left[3][1] = 0.43; matrix\_left[3][2] = 0.46; matrix\_left[3][3] = 4.44; matrix\_right[3] =9.349;

output\_double\_matrix(matrix\_left,size\_matrix);

cout << endl << "Matrix right side" << endl << endl;

output\_vector(matrix\_right, size\_matrix);

line

cout << endl << "Process of finding solving matrix..." << endl << endl;

find\_lower\_matrix(matrix\_left, matrix\_lower, size\_matrix);

temp\_solving(matrix\_lower, temp\_matrix, matrix\_right, size\_matrix);

finding\_solving(matrix\_lower,temp\_matrix,solving,size\_matrix);

inconspicuous(matrix\_left, matrix\_right, solving,temp\_matrix, size\_matrix);

line

//////////////////////////////Progam end//////////////////////////////

delete[] solving;

delete[] temp\_matrix;

for (int i = 0; i < size\_matrix; i++)

{

delete matrix\_lower[i];

}

delete[] matrix\_lower;

delete[] matrix\_right;

for (int i = 0; i < size\_matrix; i++)

{

delete matrix\_left[i];

}

delete[] matrix\_left;

return 0;

}

\_type\_\*\* find\_lower\_matrix(\_type\_\*\* matrix\_left, \_type\_\*\* matrix\_lower, int size\_matrix)

{

cout << endl << "Direct course factorization:" << endl;

matrix\_lower[0][0] = sqrt(matrix\_left[0][0]);

cout << endl << "\t -step with index i=" << 0 << " j=" << 0 <<":"<< endl;

output\_double\_matrix( matrix\_lower,size\_matrix);

for (int i = 1; i < size\_matrix; i++)

{

matrix\_lower[i][0] = (matrix\_left[i][0]) / matrix\_lower[0][0];

cout << endl << "\t -step with index i=" << i << " j=" << 0 << ":" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

}

for (int i = 1; i < size\_matrix; i++)

{

for (int j = 1; j < size\_matrix; j++)

{

if (i > j)

{

\_type\_ temp = 0;

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

{

temp = temp + matrix\_lower[i][k] \* matrix\_lower[j][k];

}

temp = matrix\_left[i][j] - temp;

matrix\_lower[i][j] = temp / matrix\_lower[j][j];

cout << endl << "\t -step with index i=" << i << " j=" << j << ":" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

}

}

\_type\_ temp = 0;

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

{

temp = temp + pow(matrix\_lower[i][j], 2);

}

temp = matrix\_left[i][i] - temp;

matrix\_lower[i][i] = sqrt(temp);

cout << endl << "\t -step with index i=" << i << " j=" << i << ":" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

}

for (int i = 0; i < size\_matrix; i++)

{

for (int j = 0; j < size\_matrix; j++)

{

if (j > i)

{

matrix\_left[i][j] = 0;

cout << endl << "\t -step with index i=" << i << " j=" << j << ":" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

}

}

}

return matrix\_lower;

}

\_type\_\* temp\_solving(\_type\_\*\* matrix\_lower, \_type\_\* temp\_matrix, \_type\_\* matrix\_right, int size\_matrix)

{

cout << endl << "Temp solving step:" << endl;

temp\_matrix[0] = matrix\_right[0] / matrix\_lower[0][0];

cout << endl << "\t -step with index i=" << 0 << ":" << endl;

output\_vector(temp\_matrix, size\_matrix);

for (int i = 1; i < size\_matrix; i++)

{

\_type\_ temp = 0;

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

{

temp = temp + matrix\_lower[i][j] \* temp\_matrix[j];

}

temp = - temp+ matrix\_right[i];

temp\_matrix[i] = temp / matrix\_lower[i][i];

cout << endl << "\t -step with index i=" << i << ":" << endl;

output\_vector(temp\_matrix, size\_matrix);

}

return temp\_matrix;

}

\_type\_\* finding\_solving(\_type\_\*\* matrix\_lower, \_type\_\* temp\_matrix, \_type\_\* solving, int size\_matrix)

{

cout << endl << "Matrix of factorization:" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

for (int i = 0; i < size\_matrix; i++)

{

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

{

\_type\_ temp;

temp = matrix\_lower[i][j];

matrix\_lower[i][j] = matrix\_lower[j][i];

matrix\_lower[j][i] = temp;

}

}

cout << endl << "Transponate matrix of factorization:" << endl;

output\_double\_matrix(matrix\_lower, size\_matrix);

cout << endl << "Solving step:" << endl;

solving[size\_matrix - 1] = temp\_matrix[size\_matrix - 1] / matrix\_lower[size\_matrix - 1][size\_matrix - 1];

cout << endl << "\t -step with index i=" << size\_matrix << ":" << endl;

output\_vector(solving, size\_matrix);

for (int i = size\_matrix - 2; i >=0; i--)

{

\_type\_ temp = 0;

for (int j = i; j <size\_matrix; j++)

{

temp = temp + matrix\_lower[i][j+1] \* solving[j+1];

}

temp = -temp + temp\_matrix[i];

solving[i] = temp / matrix\_lower[i][i];

cout << endl << "\t -step with index i=" << i << ":" << endl;

output\_vector(solving, size\_matrix);

}

return solving;

}

\_type\_\* inconspicuous(\_type\_\*\* matrix\_left, \_type\_\* matrix\_right, \_type\_\* solving, \_type\_\* temp\_matrix, int size\_matrix)

{

matrix\_left[0][0] = 2.12; matrix\_left[0][1] = 0.42; matrix\_left[0][2] = 1.34; matrix\_left[0][3] = 0.88;

matrix\_left[1][0] = 0.42; matrix\_left[1][1] = 3.95; matrix\_left[1][2] = 1.87; matrix\_left[1][3] = 0.43;

matrix\_left[2][0] = 1.34; matrix\_left[2][1] = 1.87; matrix\_left[2][2] = 2.98; matrix\_left[2][3] = 0.46;

matrix\_left[3][0] = 0.88; matrix\_left[3][1] = 0.43; matrix\_left[3][2] = 0.46; matrix\_left[3][3] = 4.44;

for (int i = 0; i < size\_matrix; i++)

{

temp\_matrix[i] = 0;

}

for (int i = 0; i < size\_matrix; i++)

{

\_type\_ temp = 0;

for (int j = 0; j < size\_matrix; j++)

{

temp = temp + matrix\_left[i][j] \* solving[j];

}

temp\_matrix[i] = abs(temp-matrix\_right[i]);

}

cout << endl << "Inconspicuous vector:" << endl;

output\_vector(temp\_matrix, size\_matrix);

return temp\_matrix;

}

void output\_double\_matrix(\_type\_\*\* matrix\_left, int size\_matrix)

{

cout << endl;

for (int i = 0; i < size\_matrix; i++)

{

for (int j = 0; j < size\_matrix; j++)

{

if (matrix\_left[i][j]==0)

{

cout << "\t\t" << matrix\_left[i][j];

}

else

{

cout << "\t" << matrix\_left[i][j];

}

}

cout << endl;

}

cout << endl;

}

void output\_vector(\_type\_\* matrix\_right, int size\_matrix)

{

cout << endl;

for (int i = 0; i < size\_matrix; i++)

{

cout << "\t\t" << matrix\_right[i];

cout << endl << endl;

}

cout << endl;

}