БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Лабораторная работа
по вычислительным методам алгебры на тему:

Решение систем линейных алгебраических уравнений с помощью метода квадратного корня

Выполнил:
Архангельский И.А.

Проверил:
Кондратюк А.П.

Минск, 2012

Входные и выходные данные.

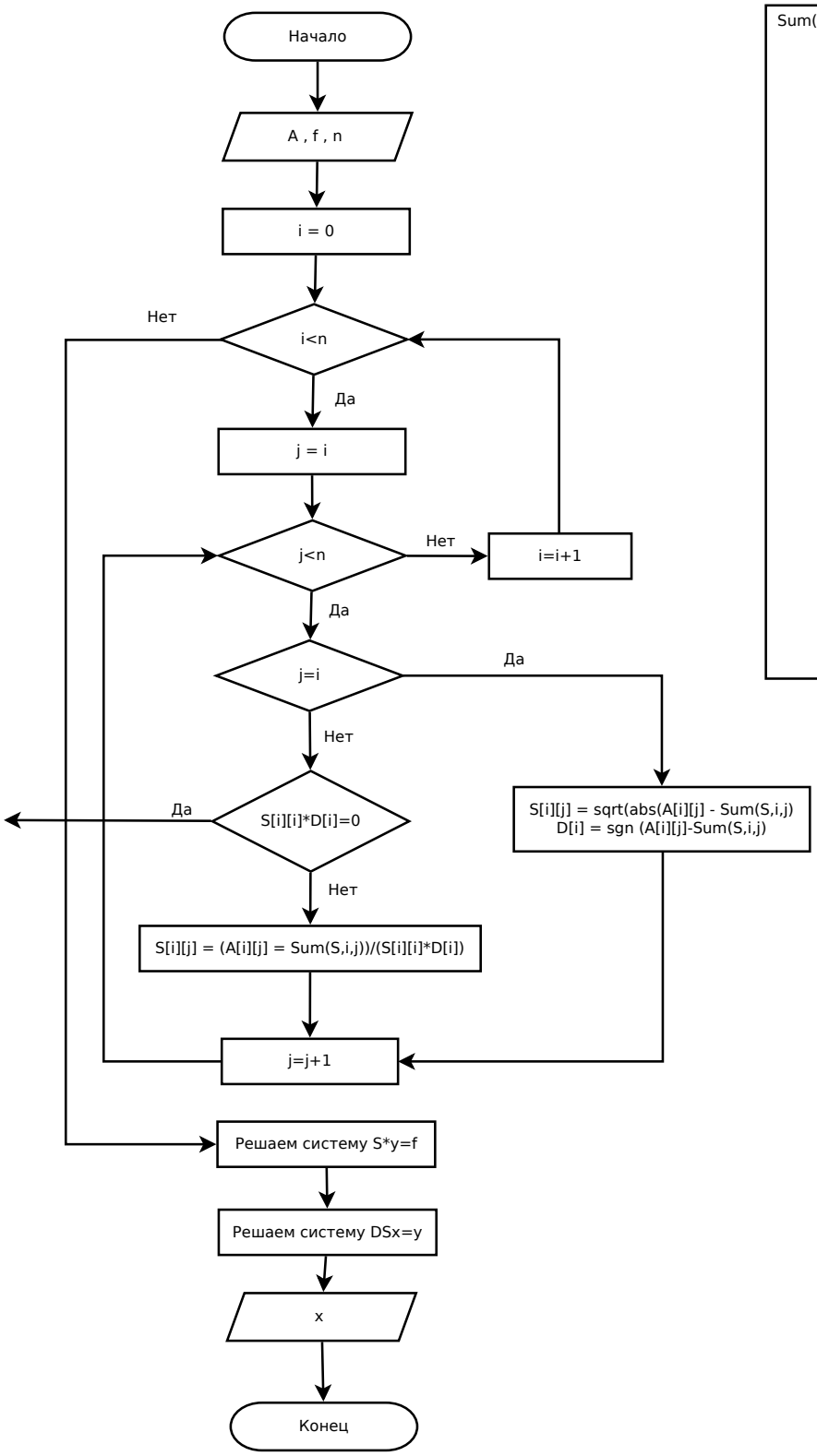
Входные данные

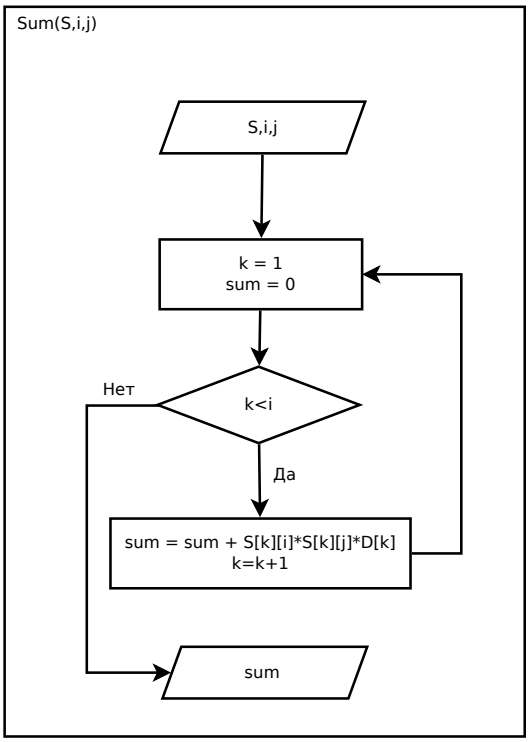
Входной файл в первой строке содержит число n - размерность матрицы коэффициентов, следующие n строк содержат матрицу $(A|f)$, где A - квадратная матрица коэффициентов СЛУ, f - вектор-столбец свободных членов.

Выходные данные

На выход в stdout подается решение системы если оно существует, в случае, если систему решить невозможно выводится DivByZero.

Блок-схема


Начало
A , f , n
i = 0
i<n
Нет
Да
j = i
j<n
Нет
i=i+1
Да
j=i
Да
Нет
S[i][i]*D[i]=0
Да
Нет
S[i][j] = (A[i][j] = Sum(S,i,j))/(S[i][i]*D[i])
S[i][j] = sqrt(abs(A[i][j] - Sum(S,i,j)
D[i] = sgn (A[i][j]-Sum(S,i,j)
j=j+1
Решаем систему S*y=f
Решаем систему DSx=y
x
Конец


Sum(S,i,j)
S,i,j
k = 1
sum = 0
k<i
Нет
Да
sum = sum + S[k][i]*S[k][j]*D[k]
k=k+1
sum

Реализация

square.h

```
#include <cmath>
#include <stdio.h>

struct DivByZeroException {};

class Square
{
    double** matrix_;
    int sz;
    double* f;
    double sgn (double x);
public:
    Square(char *filename);
    double* solve ();
    int getSize () {return sz;}
    ~Square ();
};
```

square.cpp

```
#include "square.h"

Square::Square(char* filename)
{
    FILE* fin = fopen(filename,"r");
    fscanf (fin,"%d",&sz);
    matrix_ = new double* [sz];
    f = new double [sz];
    for (int i=0;i<sz;i++)
    {
        matrix_[i] = new double [sz];
        for (int j=0;j<sz;j++)
        {
            fscanf(fin,"%lf",&matrix_[i][j]);
        }
        fscanf (fin,"%lf", &f[i]);
    }
    fclose (fin);
}

Square::~Square()
{
    delete [] f;
    for (int i=0;i<sz;i++)
        delete [] matrix_[i];
    delete [] matrix_;
}

double Square::sgn(double x)
{
    if (x==0) return 0;
    return x>0?1:-1;
}

double* Square::solve()
{
    double** S = new double* [sz];
    double* D = new double[sz];
    for (int i=0;i<sz;i++)
    {
         S[i] = new double [sz];
    }
    for (int i=0;i<sz;i++)
    {

        for (int j=i;j<sz;j++)
        {
            if (i==j)
            {
                double summ = 0;
                for (int k = 0; k<i;k++)
                {
                    summ+=S[k][i]*S[k][i]*D[k];
                }
```

```
                D[i] = sgn (matrix_[i][j]-summ);
                S[i][j]=pow(fabs(summ-matrix_[i][j]),0.5);

            }
            if (i<j)
            {
                double summ = 0;
                for (int k=0;k<i;k++)
                {
                    summ+=S[k][i]*D[k]*S[k][j];
                }
                if (S[i][i]*D[i]==0) throw DivByZeroException ();
                S[i][j] = (matrix_[i][j] - summ)/(S[i][i]*D[i]);
                S[j][i] = S[i][j];
            }
            if (i>j)
            {
                S[i][j] = 0;
            }
        }
    }
    double* y = new double [sz];
    for (int i =0; i<sz;i++)
    {
        double summ = 0;
        for (int k=0;k<i;k++)
        {
            summ+=S[i][k]*y[k];
        }
        y[i] = (f[i] - summ)/S[i][i];
    }
    double* x = new double [sz];
    for (int i=sz-1;i>=0;i--)
    {
        double summ = 0;
        for (int k = i+1; k<sz;k++)
        {
            summ += S[i][k]*D[i]*x[k];
        }
        x[i] = (y[i]-summ)/S[i][i]*D[i];
    }
    delete [] y;
    for (int i=0;i<sz;i++)
    {
        delete [] S[i];
    }
    delete [] D;
    delete [] S;
    return x;
}
```

main.cpp

```
#include <iostream>
#include <square.h>

int main(int argc, char *argv[])
{
    for (int i=1;i<argc;i++)
    {
        Square sq =  Square(argv[i]);
        double* res;
        try
        {
            res  = sq.solve();
            for (int i=0;i<sq.getSize();i++)
            {
                printf ("%3.3lf  ",res[i]);
            }
            printf ("\n");
            delete [] res;
        }
        catch (DivByZeroException e)
        {
            printf ("DivByZero\n");
        }
    }
    return 0;
}
```

Тестовые данные

test01.in	test01.out
3 3.2 1 1 4 1 3.7 1 4.5 1 1 4.2 4	0.811 0.846 0.558

test02.in	test02.out
4 1 1 1 1 0 1 2 3 4 0 1 3 6 10 0 1 4 10 20 0	0.000 0.000 0.000 0.000

test03.in	test03.out
4 −1 1 1 1 5 1 2 1 1 2 1 1 −3 1 3 1 1 1 4 4	−2.474 1.947 −0.737 1.316

test04.in	test04.out
4 1 1 1 1 4 1 1 1 1 3 1 1 1 2 4 1 1 2 1 5	DivByZero

test05.in	test05.out
3 1 1 1 3 1 2 3 6 1 3 1 5	1.000 1.000 1.000