Отчёт о выполнении 6 Лабораторной работы по Выч. Мату.

Выполнил: Пшонкин Вадим 932101г.

1. **Постановка задачи:**

В данной лабораторной работе мне требовалось изучить 2 алгоритма (Простых итераций и Халецкого) и написать алгоритм их решения.

1. **Теоретическая часть:**

а) Для метода простых итераций необходима выполнимость нескольких условий.

Первое условие это det Матрцы не равный нулю.

Второе условие это условие сходимости.

Если матрица подходит под оба условия, то мы вычисляем её по следующей системе:  
1)Делаем систему уравнений, путём выражение x-ов главной диагонали.  
2)Находим нулевое приближение по формуле bk/akk (где akk значение матрицы находящееся на главной диагонали)

1. Потом в систему, которую мы получили на 1 шаге, мы подставляем значение из нулевого приближения и получаем первые корни.
2. Далее по формуле находим приближение для каждого корня по формуле ((xk-xk-1)/xk), и если хотя-бы одно приближение больше, чем данное нам eps, то мы находим следующее приближение.

б) Для метода Халецкого необходима выполнимость нескольких условий.

Первое условие - это также det Матрцы не равный нулю.

Второе условие - это симметричность матрицы.

Третье условие - это положительная определённость матрицы.

Если матрица подходит под все условия, то мы вычисляем её по следующей системе:  
1)Находим треугольную матрицу L которая соответствует условию A = L\*LT

1. Далее решаем систему Ly = b и получаем из неё значение y
2. И в конце решаем систему LTx = y и получаем значение x-ов.
3. **Программный код:**

Код для метода простых итераций:

void SimpleOperations(double \*\*a, double \*b, const int n)

{

    double \*x = new double[n]{};

    double eps = 0.01;

    double norma = 0;

    double sum = 0;

    int k = 0;

    do

    {

        norma = 0;

        for (int i = 0; i < n; i++)

        {

            sum = 0;

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

            {

                if (i != j)

                {

                    sum += a[i][j] \* x[j];

                }

            }

            double temp = x[i];

            x[i] = (b[i] - sum) / a[i][i];

            norma = max(norma, abs(x[i] - temp));

        }

        k++;

    } while (norma > eps);

    cout << "Solution: " << endl;

    for (int i = 0; i < n; i++)

    {

        cout << "x[" << i << "] = " << x[i] << endl;

        ;

    }

    cout << endl

         << "Iterations: " << k << endl;

}

Код для метода Халецкого:

void solveCholesky(double \*\*a, double \*b, const int n)

{

    double \*\*l = new double \*[n];

    double \*y = new double[n]{};

    double \*x = new double[n]{};

    for (int i = 0; i < n; i++)

    {

        l[i] = new double[n]{};

    }

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j <= i; j++)

        {

            double sum = 0;

            if (j == i)

            {

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

                {

                    sum += pow(l[j][k], 2);

                }

                l[j][j] = sqrt(a[j][j] - sum);

            }

            else

            {

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

                {

                    sum += l[i][k] \* l[j][k];

                }

                l[i][j] = (a[i][j] - sum) / l[j][j];

            }

        }

    }

    for (int i = 0; i < n; i++)

    {

        double sum = 0;

        for (int j = 0; j < i; j++)

        {

            sum += l[i][j] \* y[j];

        }

        y[i] = (b[i] - sum) / l[i][i];

    }

    for (int i = n - 1; i >= 0; i--)

    {

        double sum = 0;

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

        {

            sum += l[j][i] \* x[j];

        }

        x[i] = (y[i] - sum) / l[i][i];

    }

    cout << "Solution: " << endl;

    for (int i = 0; i < n; i++)

    {

        cout << "x[" << i << "] = " << x[i] << endl;

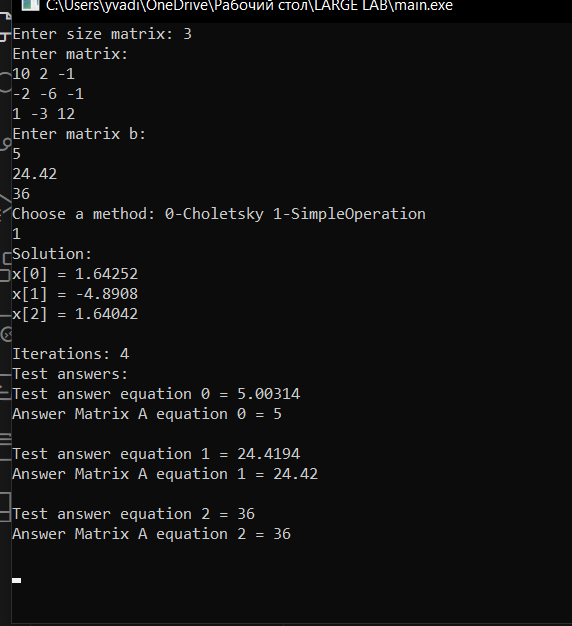
        ;

    }

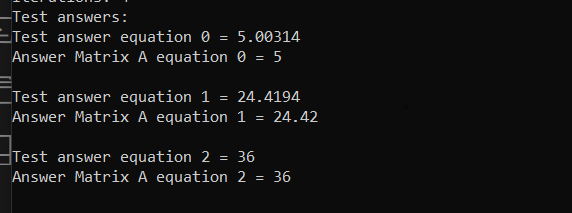
    cout << endl;

}

**4.Вывод результата работы программы:**



**5. Результат тестирования:**



1. **Вывод:**

В выводе хочется выделить сложность проделанной работы и её увлекательность. По сути что бы написать код недостаточно знать только формулы, но надо и понимать как работает сам метод. По ходу работы мне пришлось во всём разобраться и найти решение. Благодарю за представленную возможность.