федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №2
Вариант 7

Студент:	Кулагин Вячеслав Дмитриевич
Преподаватель:	Наумова Надежда Александровна
Группа:	P3209

Санкт-Петербург
2025

Содержание

1 Цели работы

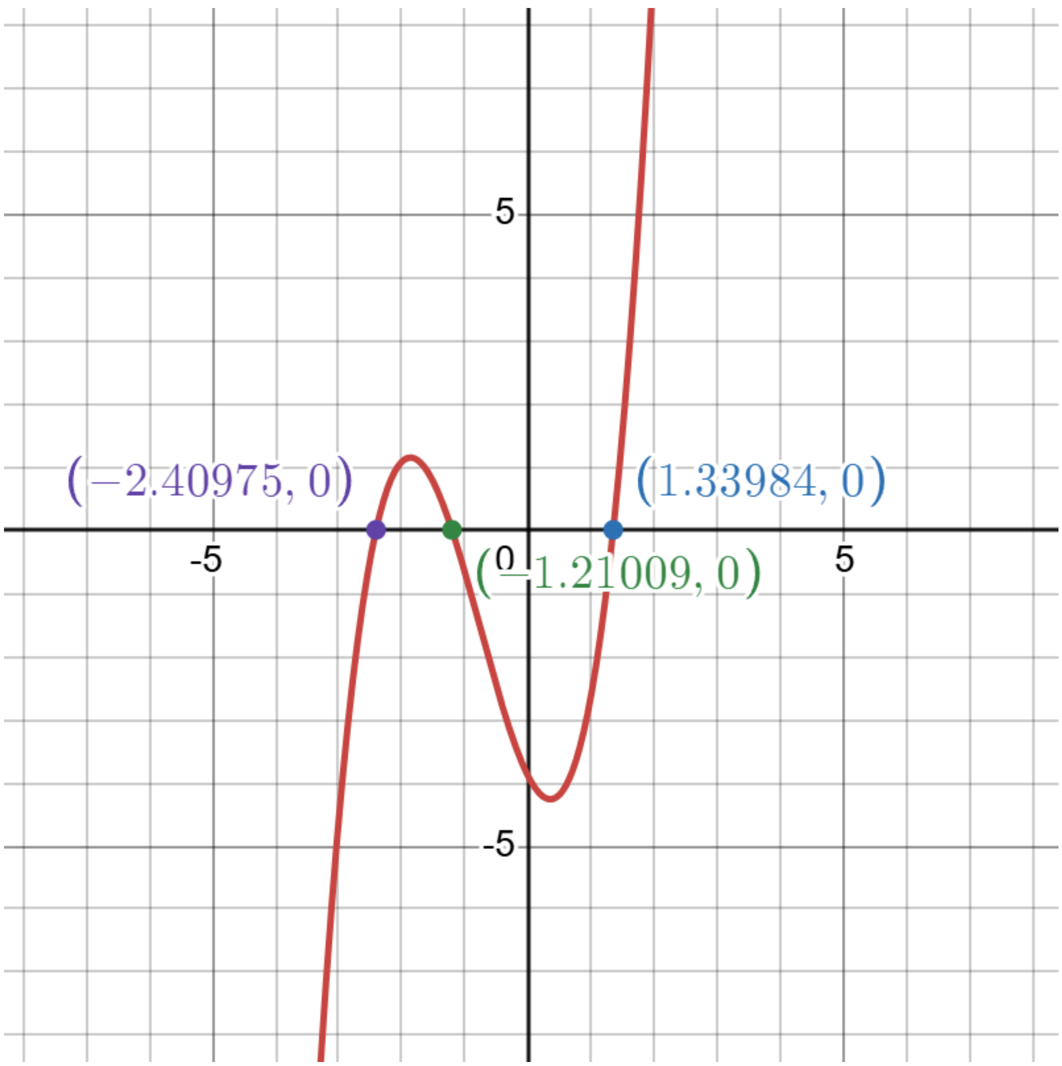
Изучить численные методы решения нелинейных уравнений и их систем, найти корни заданного нелинейного уравнения/системы нелинейных уравнений, выполнить программную реализацию методов.

2 Вычислительная реализация задачи

2.1 Решение нелинейного уравнения

2.1.1 Корни графически

Дано: $x^3 + 2,28x^2 - 1,934x - 3,907$



2.1.2 Интервалы изоляции корней

Для их определения можно воспользоваться методом интервалов знакопеременности. Для этого найдем значения знака на разных интервалов. Имеем приближенное значение корней: $x_1 \approx -2,4$; $x_2 \approx -1,2$; $x_3 \approx 1,3$. Разобьем ось на 4 интервала, и найдем какие на каждом из них знаки:

$(-\infty, -2.4); (-2.4, -1.2); (-1.2, 1.3); (1.3, +\infty)$. Если при переходе от одного корня к другому будет меняться знак, это подтвердит изоляцию корней. Выберем точки в каждом интервале для проверки: $-3, -1, 0, 2$. Получаем:

Интервал	$(-\infty, -2.4)$	$(-2.4, -1.2)$	$(-1.2, 1.3)$	$(1.3, +\infty)$
Число	-3	-1	0	2
Знак	$-$	$+$	$-$	$+$

Каждый интервал содержит ровно один корень, так как знак меняется при переходе через каждую границу. Получаем примерные интервалы изоляции корней: $(-3, -1.5); (-1.5, 0); (1, 2)$.

2.1.3 Уточнить корни с точностью 10^{-2}

Получаем: $x_1 \approx -2,41$; $x_2 \approx -1,21$; $x_3 \approx 1,34$.

2.1.4 Нахождение крайнего правого корня методом половинного деления

Вычислим крайний правый корень методом половинного деления, получаем:

№	a	b	x	$f(a)$	$f(b)$	$f(x)$	**|a-b|**
1	1	2	1.5	$-$	$+$	$+$	1
2	1	1.5	1.25	$-$	$+$	$-$	0.5
3	1.25	1.5	1.375	$-$	$+$	$+$	0.25
4	1.25	1.375	1.3125	$-$	$+$	$-$	0.125
5	1.3125	1.375	1.34375	$-$	$+$	$+$	0.0625
6	1.3125	1.34375	1.328125	$-$	$+$	$-$	0.03125
7	1.328125	1.34375	1.335938	$-$	$+$	$-$	0.015625
8	1.335938	1.34375	**1.339844**	$-$	$+$	$+$	0.007813

Окончательно понимаем, что $x_3 = 1.339844$

2.1.5 Нахождение крайнего левого корня методом простых итераций

$f(x) = x^3 + 2,28x^2 - 1,934x - 3,907$
$f'(x) = 3x^2 + 4,56x - 1,934$
$\lambda = -\frac{1}{3,667}$
$\varphi(x) = x - \frac{1}{3,667}(x^3 + 2,28x^2 - 1,934x - 3,907)$
Интервал у нас равняется: $(-3, -1.5)$. Берем начальное приближение $x_0 = -2$. Получаем вычисления:

№	x_i	x_{i+1}	$\varphi(x_{i+1})$	$f(x_{i+1})$	$\|x_{i+1} - x_i\|$
0	-2	-2.29479	-2.41839	0.453234	0.294791
1	-2.29479	-2.41839	-2.40769	-0.03922	0.123598
2	-2.41839	-2.40769	-2.41021	0.00924	0.010695
3	-2.40769	**-2.41021**	-2.40965	-0.00207	0.00252

Получаем, что $x_1 = -2.41021$

2.1.6 Нахождение центрального корня методом Ньютона

Возьмем $x_0 = -1$, интервал: $(-1.5, 0)$. Получаем:

№	x_i	$f(x_i)$	$f'(x_i)$	x_{i+1}	$\lvert x_{i+1} - x_i \rvert$
0	-1	-0.693	-4.494	-1.15421	0.154206
1	-1.15421	-0.17499	-4.5328	-1.19281	0.038606
2	-1.19281	-0.05325	-4.52762	-1.20457	0.011761
3	-1.20457	-0.01692	-4.52486	**-1.20831**	0.003738

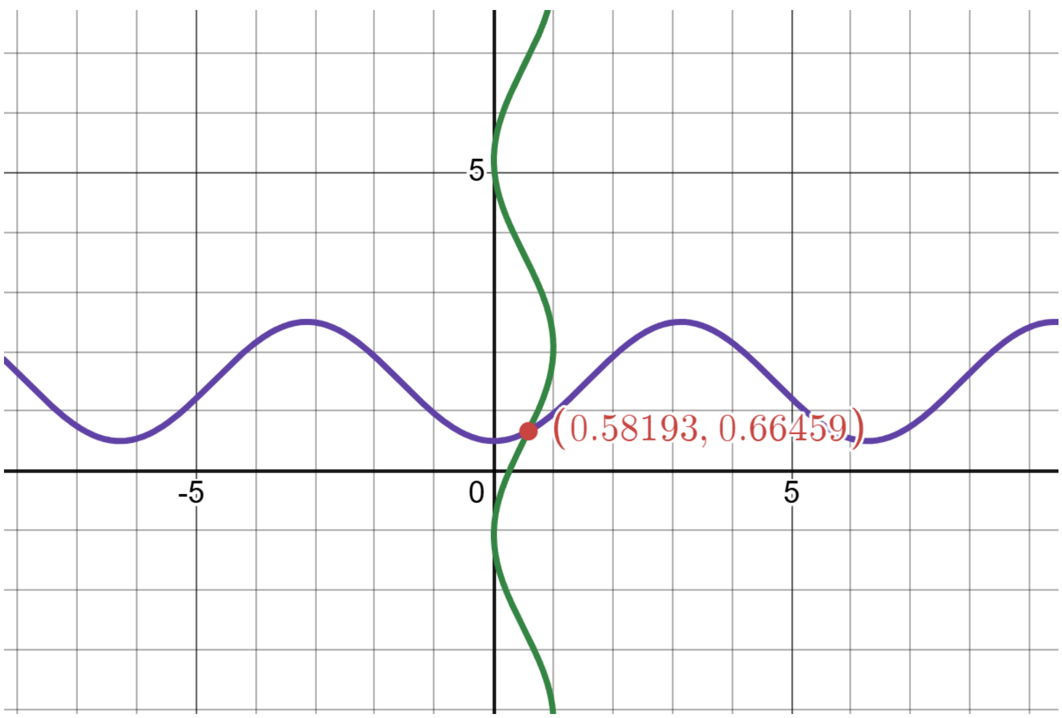
Окончательно понимаем, что $x_2 = -1,20831$

2.2 Решение системы нелинейных уравнений

Дано (метод простых итераций):

$$\begin{cases} 2x - \sin(y - 0,5) = 1 \\ y + \cos x = 1,5 \end{cases}$$

2.2.1 Корни графически



Понимаем, что искомых корень в пределах квадрата $0 < x < 1; 0 < y < 1$.

$$\begin{cases} 2x - \sin(y - 0,5) = 1 \\ y + \cos x = 1,5 \end{cases} \Rightarrow \begin{cases} x = \frac{1}{2} + \frac{1}{2}\sin(y - 0.5) \\ y = 1.5 - \cos(x) \end{cases}$$

2.2.2 Проверим сходимость

$$\frac{\partial \varphi_1}{\partial x} = 0$$

$$\frac{\partial \varphi_2}{\partial x} = \sin(x)$$

$$\frac{\partial \varphi_1}{\partial y} = \cos\left(y - \frac{1}{2}\right) \qquad \frac{\partial \varphi_2}{\partial y} = 0$$

$$\left|\frac{\partial \varphi_1}{\partial x}\right| + \left|\frac{\partial \varphi_1}{\partial y}\right| = 0 + \left|\cos\left(y - \frac{1}{2}\right)\right| < 1$$

$$\left|\frac{\partial \varphi_2}{\partial x}\right| + \left|\frac{\partial \varphi_2}{\partial y}\right| = 0 + |\sin(x)| < 0,1$$

Получаем, что $\max|\varphi'(x)| < 1$, следовательно, процесс сходящийся.

2.2.3 Итерационный процесс

Возьмем начальное приближение $x^{(0)} = 1, y^{(0)} = 1$. Подставляем каждый раз оба значения в систему уравнений, затем сравниваем точность.

1 шаг

$x^{(1)} = 0,7397$
$y^{(1)} = 0,9596$
$|x^{(1)} - x^{(0)}| = 0,2603 > \varepsilon$
$|y^{(1)} - y^{(0)}| = 0,041 > \varepsilon$

2 шаг

$x^{(2)} = 0,7218$
$y^{(2)} = 0,7613$
$|x^{(2)} - x^{(1)}| = 0,0179 > \varepsilon$
$|y^{(2)} - y^{(1)}| = 0,1983 > \varepsilon$

3 шаг

$x^{(3)} = 0,6291$
$y^{(3)} = 0,7494$
$|x^{(3)} - x^{(2)}| = 0,0927 > \varepsilon$
$|y^{(3)} - y^{(2)}| = 0,0119 > \varepsilon$

4 шаг

$x^{(4)} = 0,6234$
$y^{(4)} = 0,6914$
$|x^{(4)} - x^{(3)}| = 0,0057 < \varepsilon$
$|y^{(4)} - y^{(3)}| = 0,058 > \varepsilon$

5 шаг

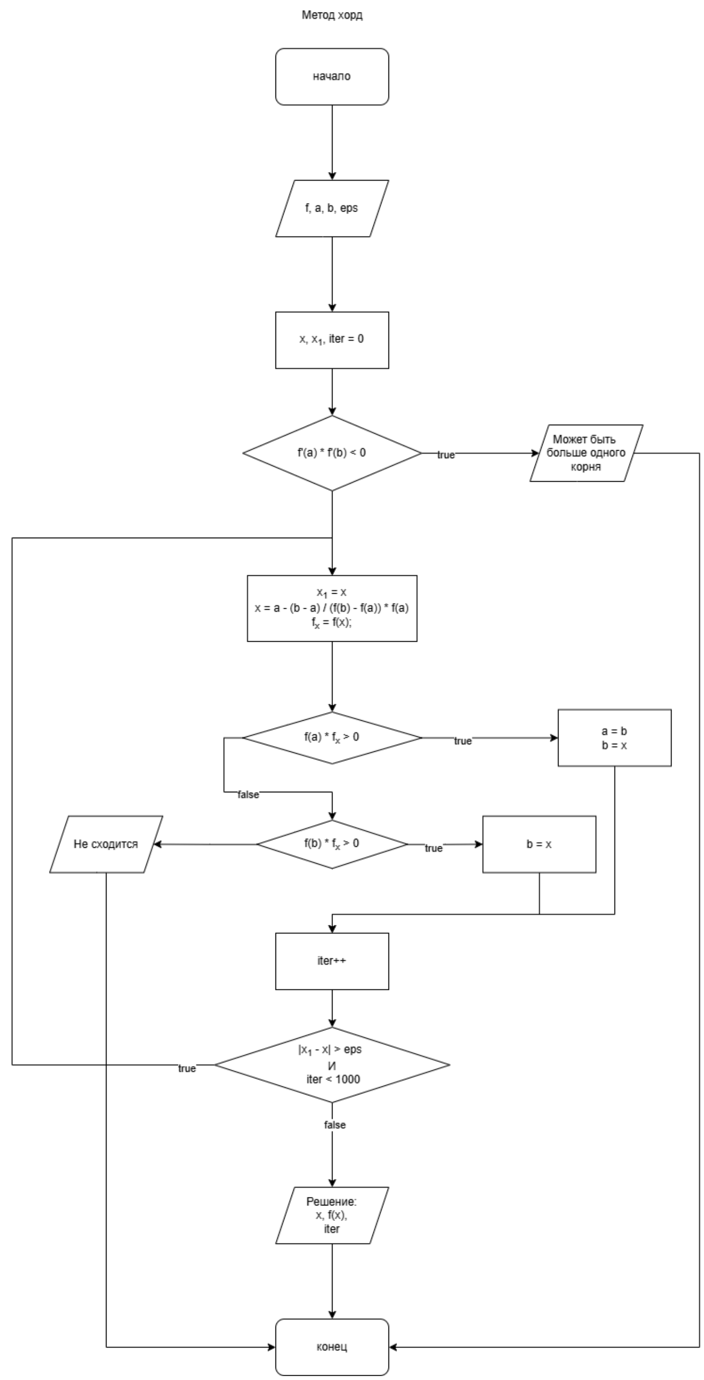
$x^{(5)} = 0,5951$
$y^{(5)} = 0,6881$
$|x^{(5)} - x^{(4)}| = 0,0057 < \varepsilon$
$|y^{(5)} - y^{(4)}| = 0,003 > \varepsilon$

Получаем итоговые результаты: $x = 0,5951$, $y = 0,6881$

3 Программная реализация

3.1 Метод хорд

3.1.1 Блок схема



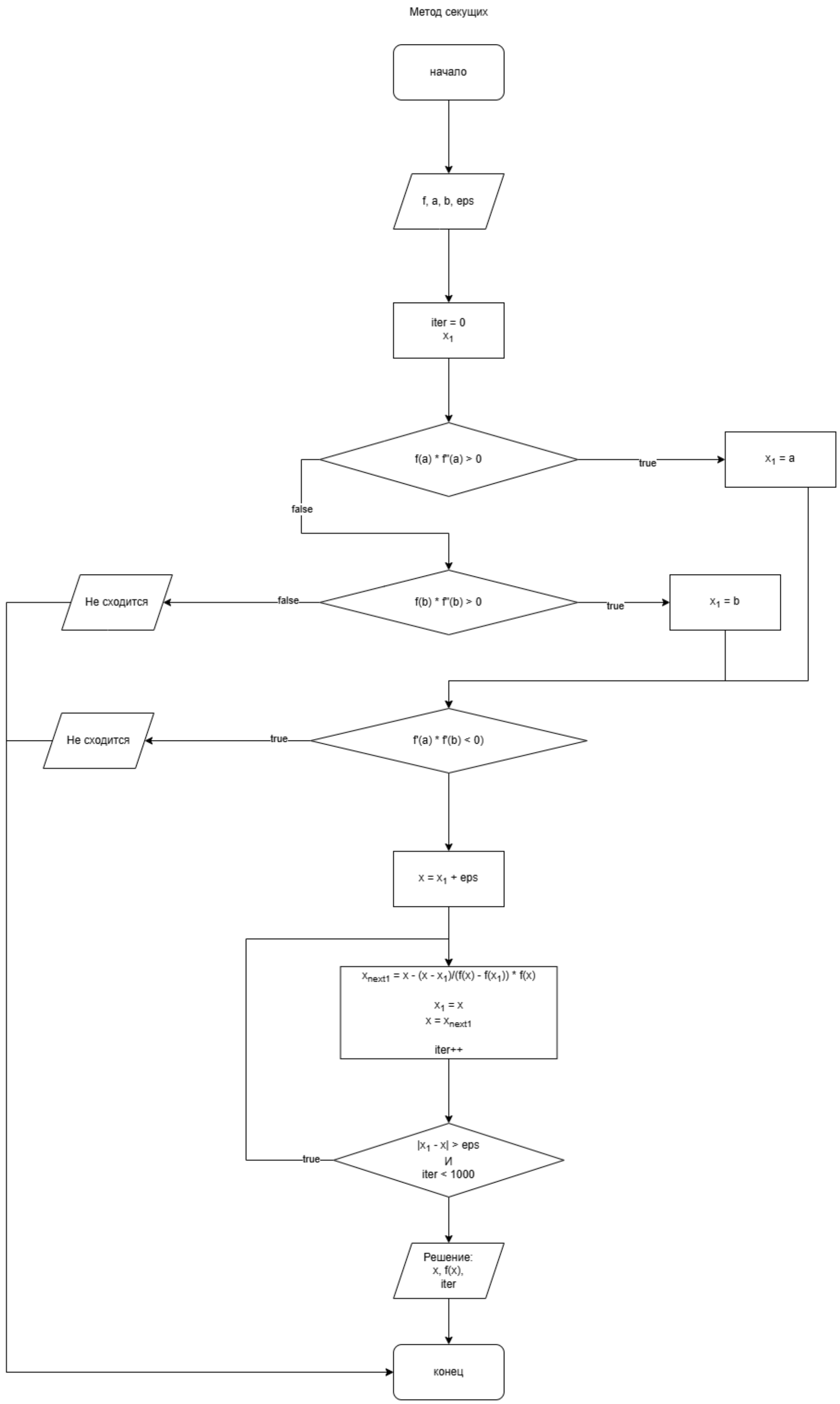
3.1.2 Реализация

```
package calc.math.logic.solvers;

import calc.math.logic.equations.Equation;

public class ChordMethod implements RootSolver {
```

```
    public String solve(Equation f, double a, double b, double eps) {
        int iter = 0;
        double x = 0;
        double x1;

        if (f.firstDerivative(a) * f.firstDerivative(b) < 0) {
            return "More then one root";
        }

        do {
            x1 = x;
            x = a - (b - a) / (f.evaluate(b) - f.evaluate(a)) * f.evaluate(a);
            double f_x = f.evaluate(x);
            if (f.evaluate(a) * f_x > 0) {
                a = b;
                b = x;
            } else if (f.evaluate(b) * f_x > 0) {
                b = x;
            } else {
                return "No solution";
            }
            iter++;
        } while (Math.abs(x1 - x) > eps && iter < 1000);

        return "Done!\n" +
                "Root: " + x + "\n" +
                "f(x): " + f.evaluate(x) + "\n" +
                "Iterations: " + iter;
    }
}
```

3.2 Метод секущих

3.2.1 Блок схема



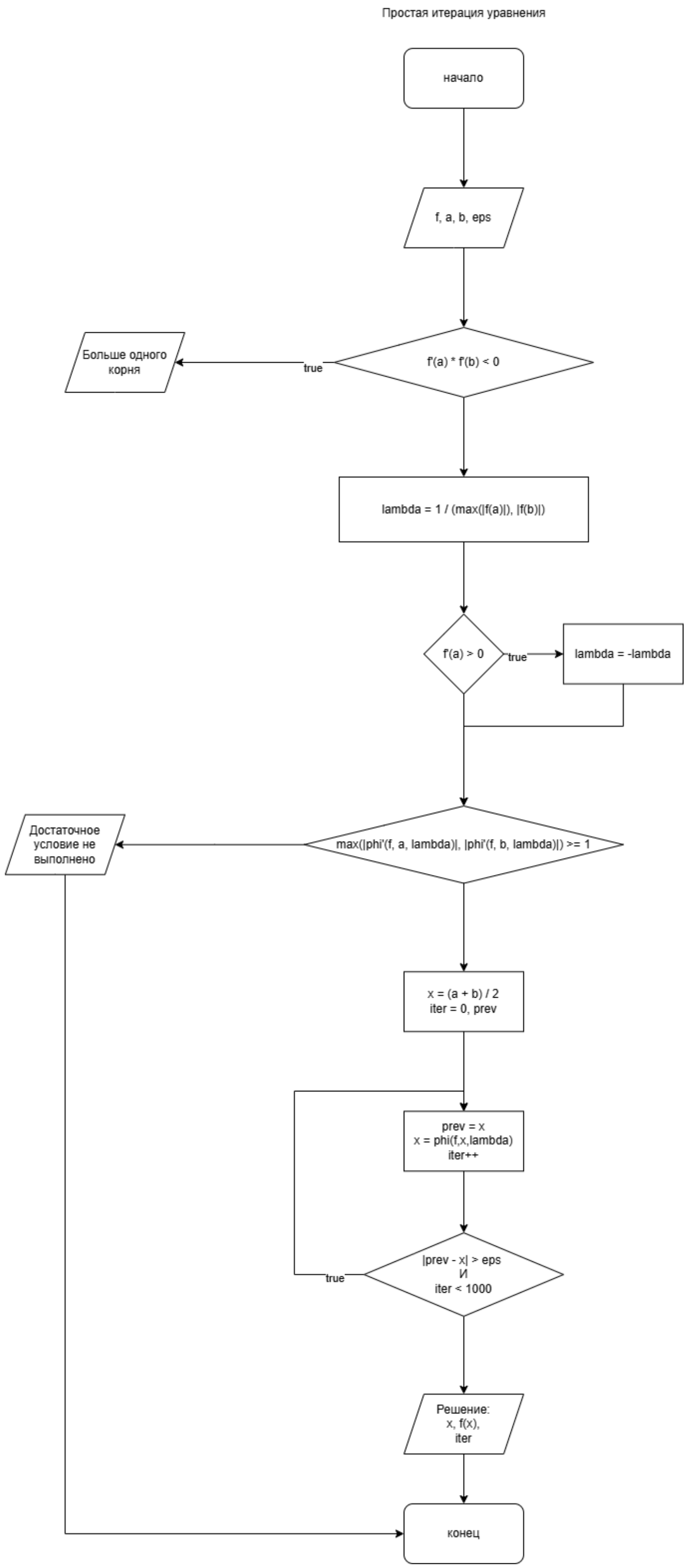
3.2.2 Реализация

```
package calc.math.logic.solvers;

import calc.math.logic.equations.Equation;
```

```java
public class SecantMethod implements RootSolver {
    public String solve(Equation f, double a, double b, double eps) {
        int iter = 0;
        double x1; // x i-1

        if (f.evaluate(a) * f.secondDerivative(a) > 0) {
            x1 = a;
        } else if (f.evaluate(b) * f.secondDerivative(b) > 0) {
            x1 = b;
        } else {
            return "No solution";
        }

        if (f.firstDerivative(a) * f.firstDerivative(b) < 0) {
            return "More then one root";
        }

        double x = x1 + eps; // x i
        do {
            double x_next_1 = x - (x - x1)/(f.evaluate(x) -
                    f.evaluate(x1)) * f.evaluate(x);

            x1 = x;
            x = x_next_1;

            iter++;
        } while (Math.abs(x1 - x) > eps && iter < 1000);

        return  "Done!\n" +
                "Root: " + x + "\n" +
                "f(x): " + f.evaluate(x) + "\n" +
                "Iterations: " + iter;
    }
}
```

3.3 Метод простой итерации для уравнений

3.3.1 Блок схема

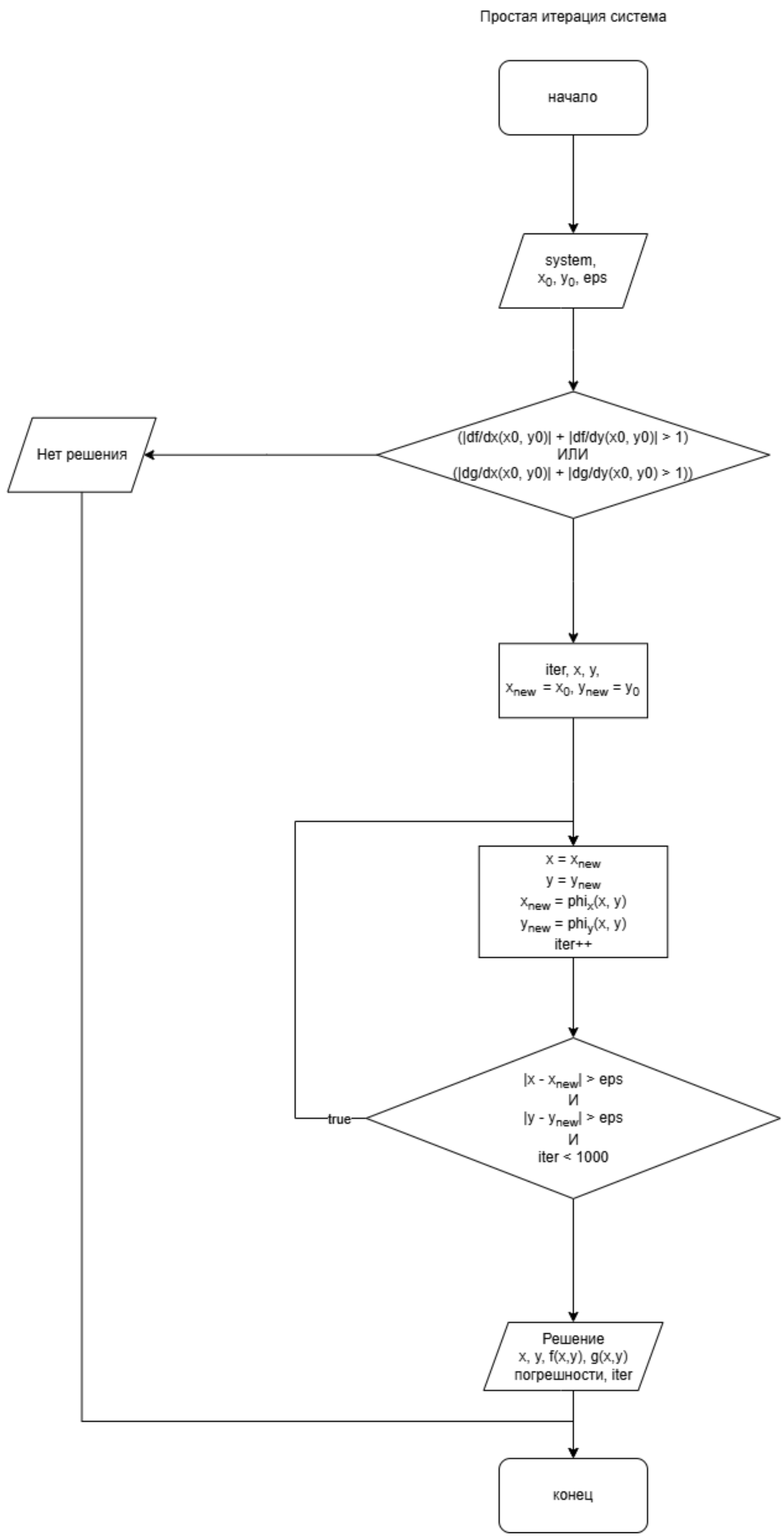


3.3.2 Реализация

```
package calc.math.logic.solvers;

import calc.math.logic.equations.Equation;

public class SimpleIterationMethod implements RootSolver {
    public String solve(Equation f, double a, double b, double eps) {
        if (f.firstDerivative(a) * f.firstDerivative(b) < 0) {
            return "More then one root";
        }

        double lambda = 1 / (Math.max(Math.abs(f.firstDerivative(a)),
        Math.abs(f.firstDerivative(b))));

        if (f.firstDerivative(a) > 0) {
            lambda = -lambda;
        }

        if (Math.max(Math.abs(firstDerivativePhi(f, a, lambda)),
        Math.abs(firstDerivativePhi(f, b, lambda))) >= 1) {
            return "No solution";
        }

        double x = (a + b) / 2;

        int iter = 0;
        double prev;
        do {
            prev = x;
            x = phi(f, x, lambda);
            iter++;
        } while (Math.abs(prev - x) > eps && iter < 1000);

        return "Root: " + x + "\n" +
                "f(x): " + f.evaluate(x) + "\n" +
                "Iterations: " + iter;
    }

    private double phi(Equation f, double x, double lambda) {
        return x + lambda * f.evaluate(x);
    }

    private double firstDerivativePhi(Equation f, double x, double lambda) {
        return 1 + lambda * f.firstDerivative(x);
    }
}
```

3.4 Метод простой итерации для систем

3.4.1 Блок схема



3.4.2 Реализация

```
package calc.math.logic.solvers;

import calc.math.logic.systems.SystemEquation;

public class SystemSimpleIterationMethod {
    public String solve(SystemEquation system,
    double x0, double y0, double eps) {
        if ((Math.abs(system.dfdx(x0, y0)) + Math.abs(system.dfdy(x0, y0)) > 1)
        ||
                ((Math.abs(system.dgdx(x0, y0)) +
                Math.abs(system.dgdy(x0, y0)) > 1))) {
            return "No solution:\n" +
                    "dfdx:" + system.dfdx(x0, y0) + "\n" +
                    "dfdy:" + system.dfdy(x0, y0) + "\n" +
                    "dgdx:" + system.dgdx(x0, y0) + "\n" +
                    "dgdy:" + system.dgdy(x0, y0) + "\n";
        }

        String logs = "";

        int iter = 0;
        double x, y;
        double xNew = x0, yNew = y0;
        do {
            x = xNew;
            y = yNew;
            xNew = system.phiX(x, y);
            yNew = system.phiY(x, y);
            iter++;
        } while (Math.abs(x - xNew) > eps &&
        Math.abs(y - yNew) > eps && iter < 1000);

        return logs + "Solution:\n" +
                "x = " + x + "\n" +
                "y = " + y + "\n" +
                "f(x,y)=" + system.f(x,y) + ", g(x,y)=" + system.g(x,y) + "\n" +
                "Errors: [" + Math.abs(x - xNew) + ", " +
                Math.abs(y - yNew) + "]\n" +
                "Iterations: " + iter;
    }
}
```

4 Вывод

В ходе выполнения лабораторной работы были изучены численные методы решения нелинейных уравнений и систем нелинейных уравнений.