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

Факультет программной инженерии и компьютерной техники

Отчет по лабораторной работе №2

«Численное решение нелинейных уравнений и систем»

По дисциплине «Вычислительная математика»

Вариант 8

Выполнила: Иванова Мария Максимовна

Группа: P3208

Преподаватель: Машина Екатерина Алексеевна

Санкт-Петербург

~ 2024 ~

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

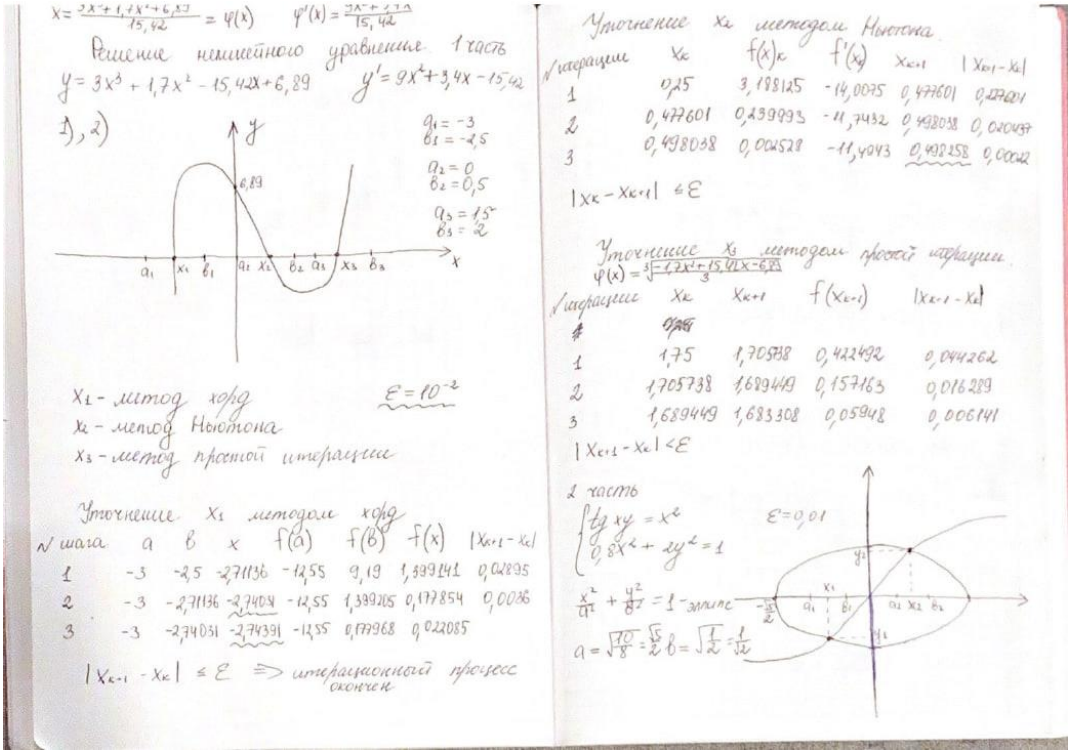
Рабочие формулы используемых методов:

- Метод половинного деления:
  $x_i = \frac{a_i + b_i}{2}$

- Метод Ньютона (касательных):
  $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$
  $x_i = x_{i-1} - \frac{f(x_{i-1})}{f'(x_{i-1})}$

- Метод простой итерации:
  $x_i = \varphi(x_i)$

1. Вычислительная реализация задачи:



Решение нелинейного уравнения 1 часть

$y = 3x^3 + 1{,}7x^2 - 15{,}42x + 6{,}89$ $\quad y' = 9x^2 + 3{,}4x - 15{,}42$

1), 2)

$a_1 = -3$, $b_1 = -2{,}5$; $a_2 = 0$, $b_2 = 0{,}5$; $a_3 = 1{,}5$, $b_3 = 2$

x_1 - метод хорд $\quad \varepsilon = 10^{-2}$

x_2 - метод Ньютона

x_3 - метод простой итерации

Уточнение x_1 методом хорд

№ шага	a	b	x	f(a)	f(b)	f(x)	$\lvert x_{k+1} - x_k \rvert$
1	-3	-2,5	-2,71136	-12,55	9,19	1,599141	0,02895
2	-3	-2,71136	-2,7404	-12,55	1,599205	0,177854	0,0086
3	-3	-2,74051	-2,74391	-12,55	0,177968	0,022085	

$\lvert x_{k+1} - x_k \rvert \le \varepsilon \Rightarrow$ итерационной процесс окончен

Уточнение x_2 методом Ньютона

№ итерации	x_k	$f(x)_k$	$f'(x_k)$	x_{k+1}	$\lvert x_{k+1} - x_k \rvert$
1	0,25	3,188125	-14,0075	0,477601	0,227601
2	0,477601	0,239993	-11,7432	0,498038	0,020437
3	0,498038	0,004528	-11,4043	0,498258	0,00022

$\lvert x_k - x_{k+1} \rvert \le \varepsilon$

Уточнение x_3 методом простой итерации

$\varphi(x) = \sqrt[3]{\frac{-1{,}7x^2 + 15{,}42x - 6{,}89}{3}}$

№ итерации	x_k	x_{k+1}	$f(x_{k-1})$	$\lvert x_{k+1} - x_k \rvert$
1	1,75	1,705738	0,422492	0,044262
2	1,705738	1,689449	0,157183	0,016289
3	1,689449	1,683308	0,05948	0,006141

$\lvert x_{k+1} - x_k \rvert < \varepsilon$

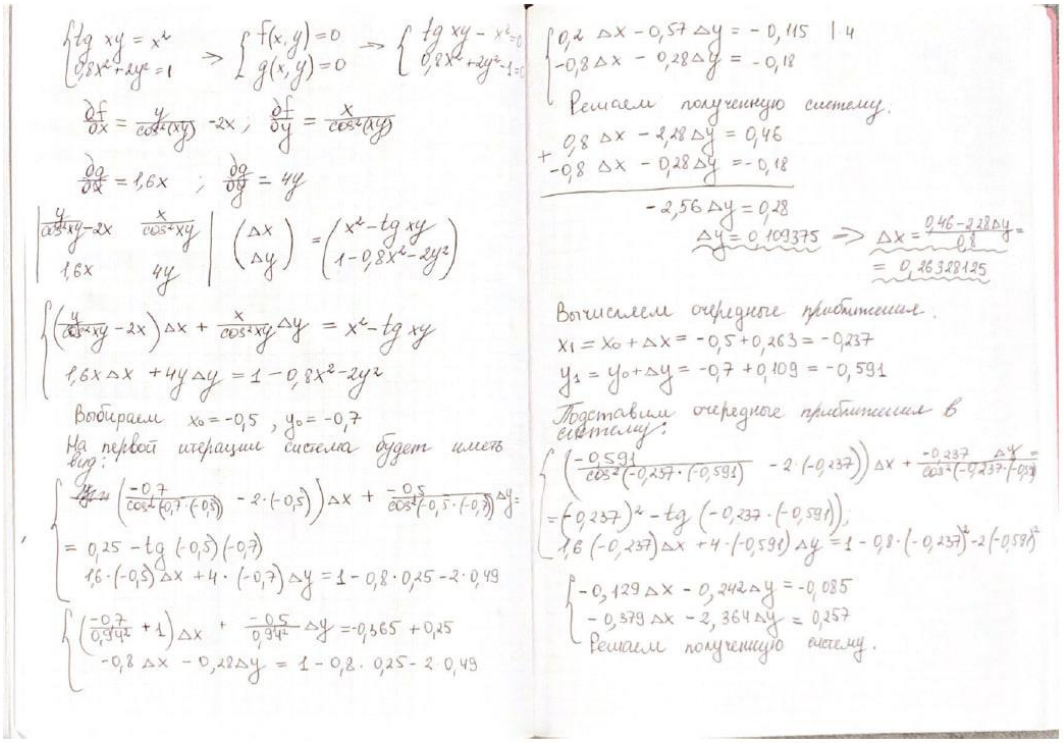
2 часть

$\varepsilon = 0{,}01$

$\begin{cases} tg\,xy = x^2 \\ 0{,}8x^2 + 2y^2 = 1 \end{cases}$

$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$ - эллипс

$a = \sqrt{\frac{10}{8}} = \frac{\sqrt{5}}{2}$ $\quad b = \sqrt{\frac{1}{2}} = \frac{1}{\sqrt{2}}$



$\begin{cases} tg\,xy = x^2 \\ 0{,}8x^2 + 2y^2 = 1 \end{cases} \Rightarrow \begin{cases} f(x,y) = 0 \\ g(x,y) = 0 \end{cases} \Rightarrow \begin{cases} tg\,xy - x^2 = 0 \\ 0{,}8x^2 + 2y^2 - 1 = 0 \end{cases}$

$\frac{\partial f}{\partial x} = \frac{y}{\cos^2(xy)} - 2x$; $\frac{\partial f}{\partial y} = \frac{x}{\cos^2(xy)}$

$\frac{\partial g}{\partial x} = 1{,}6x$; $\frac{\partial g}{\partial y} = 4y$

$\begin{vmatrix} \frac{y}{\cos^2 xy} - 2x & \frac{x}{\cos^2 xy} \\ 1{,}6x & 4y \end{vmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} x^2 - tg\,xy \\ 1 - 0{,}8x^2 - 2y^2 \end{pmatrix}$

$\begin{cases} \left(\frac{y}{\cos^2 xy} - 2x\right)\Delta x + \frac{x}{\cos^2 xy}\Delta y = x^2 - tg\,xy \\ 1{,}6x\,\Delta x + 4y\,\Delta y = 1 - 0{,}8x^2 - 2y^2 \end{cases}$

Выбираем $x_0 = -0{,}5$, $y_0 = -0{,}7$

На первой итерации система будет иметь вид:

$\begin{cases} \left(\frac{-0{,}7}{\cos^2(-0{,}7 \cdot (-0{,}5))} - 2 \cdot (-0{,}5)\right)\Delta x + \frac{-0{,}5}{\cos^2(-0{,}5 \cdot (-0{,}7))}\Delta y = \\ = 0{,}25 - tg\,(-0{,}5)(-0{,}7) \\ 1{,}6 \cdot (-0{,}5)\Delta x + 4 \cdot (-0{,}7)\Delta y = 1 - 0{,}8 \cdot 0{,}25 - 2 \cdot 0{,}49 \end{cases}$

$\begin{cases} \left(\frac{-0{,}7}{0{,}9412} + 1\right)\Delta x + \frac{-0{,}5}{0{,}9412}\Delta y = -0{,}565 + 0{,}25 \\ -0{,}8\,\Delta x - 0{,}28\,\Delta y = 1 - 0{,}8 \cdot 0{,}25 - 2 \cdot 0{,}49 \end{cases}$

$\begin{cases} 0{,}2\,\Delta x - 0{,}57\,\Delta y = -0{,}115 \quad | \cdot 4 \\ -0{,}8\,\Delta x - 0{,}28\,\Delta y = -0{,}18 \end{cases}$

Решаем полученную систему.

$+ \begin{cases} 0{,}8\,\Delta x - 2{,}28\,\Delta y = 0{,}46 \\ -0{,}8\,\Delta x - 0{,}28\,\Delta y = -0{,}18 \end{cases}$

$-2{,}56\,\Delta y = 0{,}28$

$\Delta y = 0{,}109375 \Rightarrow \Delta x = \frac{0{,}46 - 2{,}28\Delta y}{0{,}8} = 0{,}263281125$

Вычислим очередные приближения.

$x_1 = x_0 + \Delta x = -0{,}5 + 0{,}263 = -0{,}237$

$y_1 = y_0 + \Delta y = -0{,}7 + 0{,}109 = -0{,}591$

Подставим очередное приближение в систему:

$\begin{cases} \left(\frac{-0{,}591}{\cos^2(-0{,}237 \cdot (-0{,}591))} - 2 \cdot (-0{,}237)\right)\Delta x + \frac{-0{,}237}{\cos^2(-0{,}237 \cdot (-0{,}591))}\Delta y = \\ = (-0{,}237)^2 - tg\,(-0{,}237 \cdot (-0{,}591)) \\ 1{,}6\,(-0{,}237)\Delta x + 4 \cdot (-0{,}591)\Delta y = 1 - 0{,}8 \cdot (-0{,}237)^2 - 2\,(-0{,}591)^2 \end{cases}$

$\begin{cases} -0{,}129\,\Delta x - 0{,}242\,\Delta y = -0{,}085 \\ -0{,}379\,\Delta x - 2{,}364\,\Delta y = 0{,}257 \end{cases}$

Решаем полученную систему.

2. Программная реализация задачи:

Класс, реализующий метод половинного деления:

```
package org.example;

public class Method1 {

    Functions functions = new Functions();
    private int iterations = 0;

    private double x = 0;
    private int equation = 0;

    public double solve(double a, double b, int functionChoice, double
epsilon) {

        equation = functionChoice;

        if (functions.getFunction(a, functionChoice) *
functions.getFunction(b, functionChoice) >= 0) {
            throw new IllegalArgumentException("Условие теоремы о
промежуточном значении не выполнено.");

        } else {
            x = a;
            while ((b - a) >= epsilon) {
                x = (a + b) / 2;

                // Найден корень
                if (functions.getFunction(x, functionChoice) == 0.0) {
                    break;
                } else if (functions.getFunction(x, functionChoice) *
functions.getFunction(a, functionChoice) < 0) {
                    b = x;
                } else {
                    a = x;
                }
                iterations++;
            }
            return x;
        }


    }
    public double getFunctionValue(){
        return functions.getFunction(x, equation);
    }

    public int getIterations(){
        return iterations;
    }
}
```

Класс, реализующий метод Ньютона:

```
package org.example;

public class Method3 {
    private int iterations = 0;
    private double x;

    private int equation;
    Functions functions = new Functions();

    public double solve( int functionChoice, double epsilon, double a, double
b{
        x = (a+b)/2;
        equation = functionChoice;
        while (Math.abs(functions.getFunction(x,functionChoice)) > epsilon) {
            x = x - functions.getFunction(x,functionChoice)/
functions.getDerivativeFunction(x, functionChoice);
            iterations++;
        }
        return x;
    }

    public int getIterations(){
        return iterations;
    }

    public double getFunctionValue(){
        return functions.getFunction(x, equation);
    }
}
```

Класс, реализующий метод простой итерации:

```
package org.example;

public class Method5 {
    Functions functions = new Functions();

    private int equation;
    private int iterations;

    private double x;

    public double solve(int functionChoice, double epsilon, double a, double
b) {
        equation = functionChoice;
        double prev = (a+b)/2;
        x = functions.getNextApproximation(prev, functionChoice);
        iterations = 0;

        while (Math.abs(prev - x) > epsilon) {
            prev = x;
            x = functions.getNextApproximation(x, functionChoice);
            iterations++;
        }

        return x;
    }

    public int getIterations(){
        return iterations;
    }

    public double getFunctionValue(){
```

```
        return functions.getFunction(x, equation);
    }

    public boolean checkShodimost(double a, double b, int number){
        if (functions.getFiDerivative(Math.max(Math.abs(a),Math.abs(b)),
number)>1){
            System.out.println("Условие сходимости не выполняется на данном
интервале.");
            return false;
        }else {
            System.out.println("Условие сходимости выполняется.");
            return true;
        }
    }

}
```

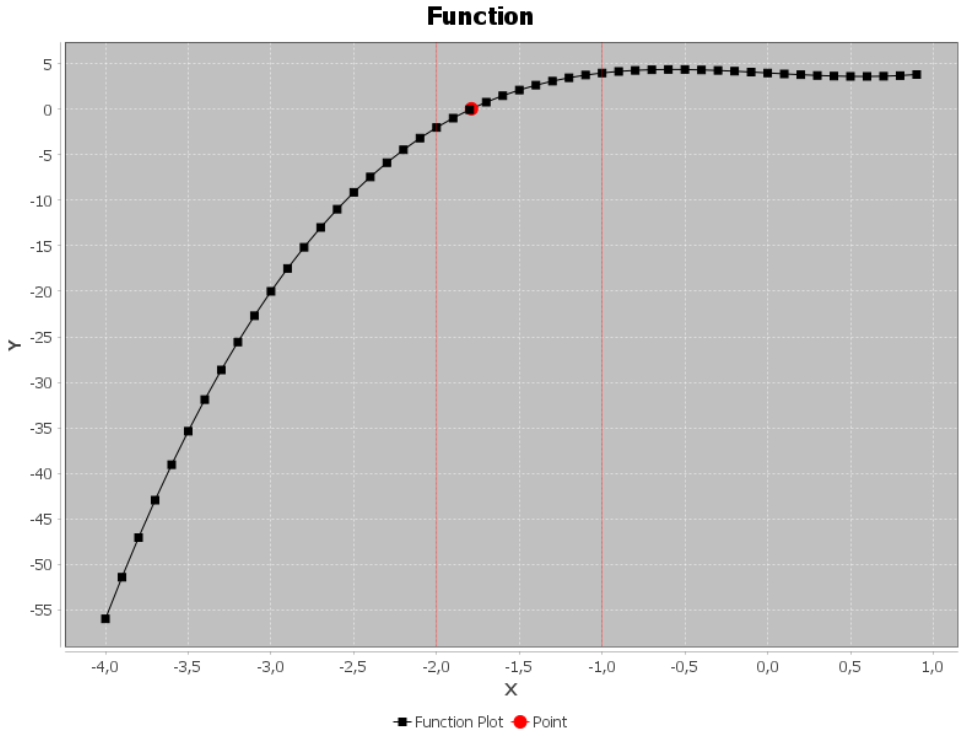
Класс, реализующий метод простой итерации для систем:

```
package org.example;

public class Method7 {
    Functions functions = new Functions();

    int choice;
    private int iterations;


    private static final int MAX_ITERATIONS = 1000; // Максимальное
количество итераций

    public  double[] solve(double initialGuessX, double initialGuessY, int
systemChoice, double epsilon) {
        double x = initialGuessX;
        double y = initialGuessY;
        double [] solution = new double[2];
        iterations = 0;
        int choice = systemChoice;

        while (iterations < MAX_ITERATIONS) {
            double newX = functions.getNextApproximationForX(x, y,
systemChoice);
            double newY = functions.getNextApproximationForY(x, y,
systemChoice);

            // Проверяем условие останова
            if (Math.max(Math.abs(newX - x),Math.abs(newY - y)) < epsilon) {
                System.out.println("Количество итераций: " + iterations);
                break;
            }

            x = newX;
            y = newY;
            iterations++;
        }

        solution[0] = x;
        solution[1] = y;
        return solution;
    }

    public boolean checkShodimost(int number, double initialGuessX, double
initialGuessY){
```

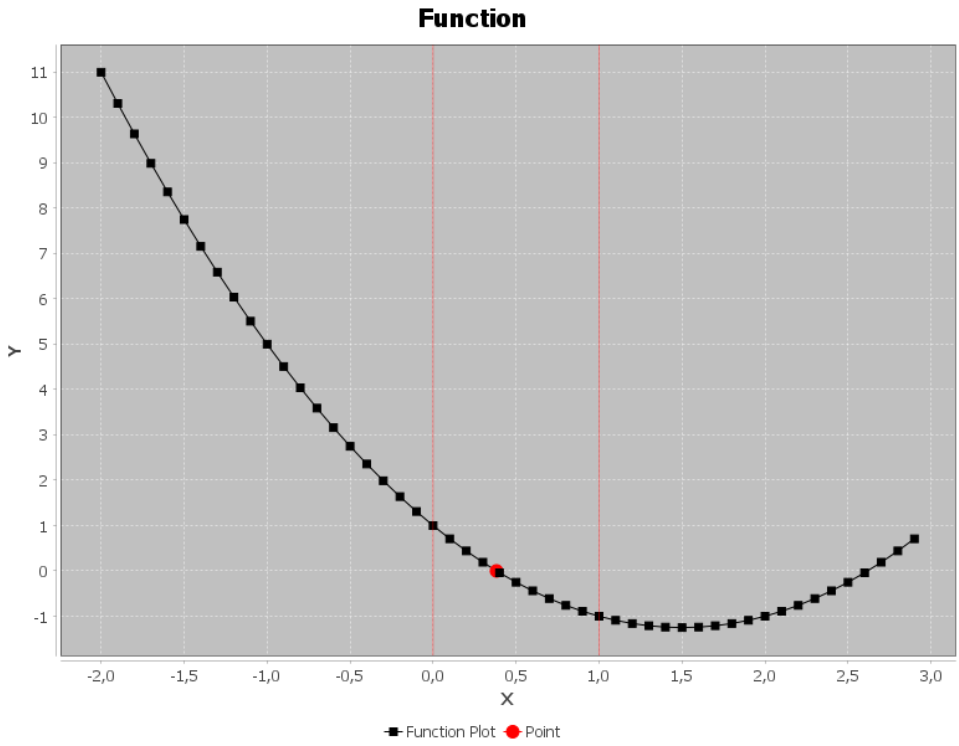
```
        if (Math.max(functions.getSystemFiDerivative1(initialGuessX,
initialGuessY, number),(functions.getSystemFiDerivative2(initialGuessX,
initialGuessY, number)))>1){
            System.out.println("Условие сходимости не выполняется на данном
интервале.");
            return false;
        }else {
            System.out.println("Условие сходимости выполняется.");
            return true;
        }
    }

    public int getIterations(){
        return iterations;
    }


}
```

Результаты работы программы:

Выберете способ ввода: 1 - через консоль, 2 - через файл.
1
Введите номер метода (1 - метод половинного деления, 3 - метод Ньютона, 5 - метод простой итерации, 7 - метод простой итерации для систем уравнений): 1
Введите желаемую погрешность epsilon: 0,01
Выберете способ вывода: 1 - в консоль, 2 - в файл:
1
Введите границы интервала [a, b]:
a = -2
b = -1
Введите номер уравнения: 1
Корень уравнения: -1.7890625
Число итераций: 7
Значение функции в корне: 0.06273031234741211

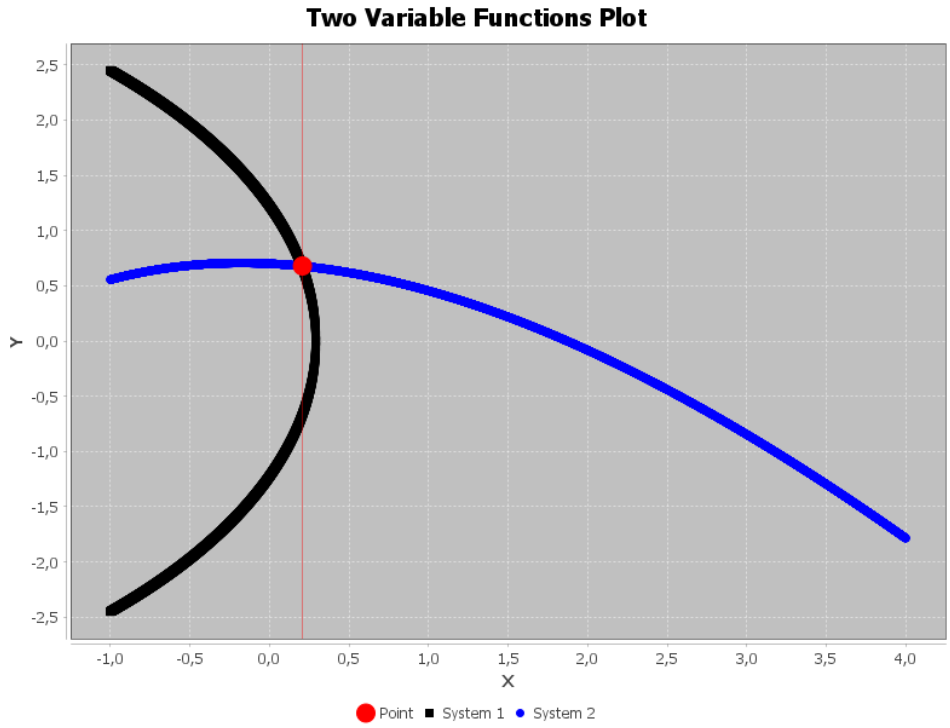


Выберете способ ввода: 1 - через консоль, 2 - через файл.
1
Введите номер метода (1 - метод половинного деления, 3 - метод Ньютона, 5 - метод простой итерации, 7 - метод простой итерации для систем уравнений): 3
Введите желаемую погрешность epsilon: 0,01
Выберете способ вывода: 1 - в консоль, 2 - в файл:
1
Введите границы интервала [a, b]:
a = 0
b = 1
Введите номер уравнения: 3
Корень уравнения: 0.3819444444444444
Число итераций: 2
Значение функции в корне: 4.822530864201369E-5



Выберете способ ввода: 1 - через консоль, 2 - через файл.
1
Введите номер метода (1 - метод половинного деления, 3 - метод Ньютона, 5 - метод простой итерации, 7 - метод простой итерации для систем уравнений): 7
Введите желаемую погрешность epsilon: 0,01
Выберете способ вывода: 1 - в консоль, 2 - в файл:
1
Введите номер системы: 1
Введите приближения для X и Y:
1 1
Условие сходимости выполняется.
Количество итераций: 4
Решение системы:
x = 0.20726010379801596

y = 0.679417317949696
Количество итераций: 4



Вывод:

Во время выполнения работы мне удалось изучить и реализовать метод половинного деления, метод Ньютона, метод простой итерации для уравнений и систем нелинейных уравнений. Метод половинного деления оказался самым простым для восприятия, также он не привязывает нас к свойствам функции и всегда сходится. Однако этот метод оказался затратным по времени. Метод Ньютона показался мне более сложным, так как функция должна быть дифференцируема, и нужно было вычислять производную на каждой итерации, что делает метод довольно затратным. Метод простой итерации было довольно просто реализовать (принцип понятен), но необходимость выбора начального приближения в окрестности корня – существенный недостаток метода.

Таким образом, метод половинного деления показался мне самым надежным.