

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

Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА №6

Вариант 7

| | |
|----------------|-------------------------------|
| Студент: | Кулагин Вячеслав Дмитриевич |
| Преподаватель: | Наумова Надежда Александровна |
| Группа: | Р3209 |

Санкт-Петербург
2025

Содержание

| | | |
|----------|---|-----------|
| 1 | Цели работы | 2 |
| 2 | Программная реализация | 2 |
| 2.1 | Метод Эйлера | 3 |
| 2.2 | Модифицированный метод Эйлера | 5 |
| 2.3 | Метод Милна | 7 |
| 3 | Вывод | 10 |

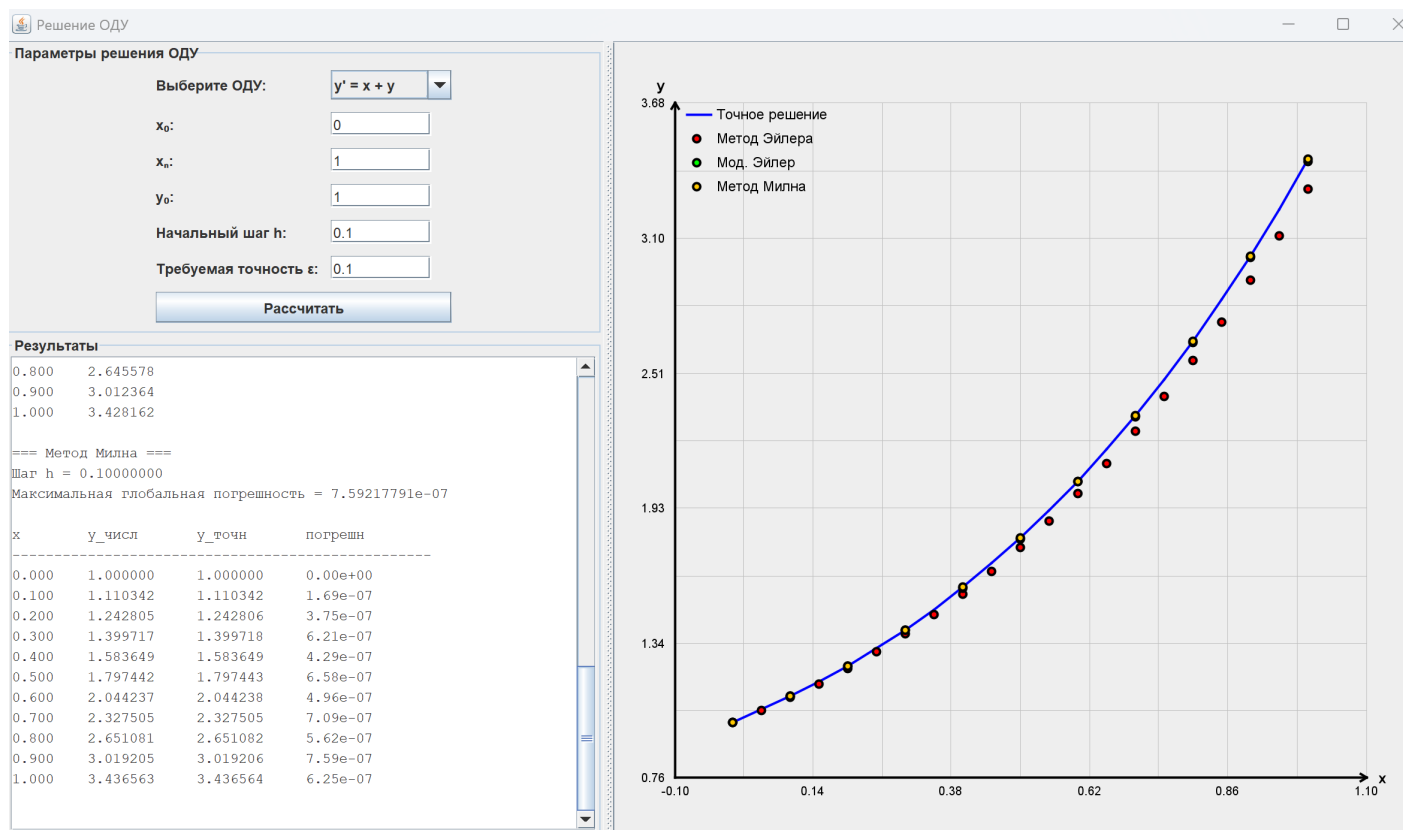
1 Цели работы

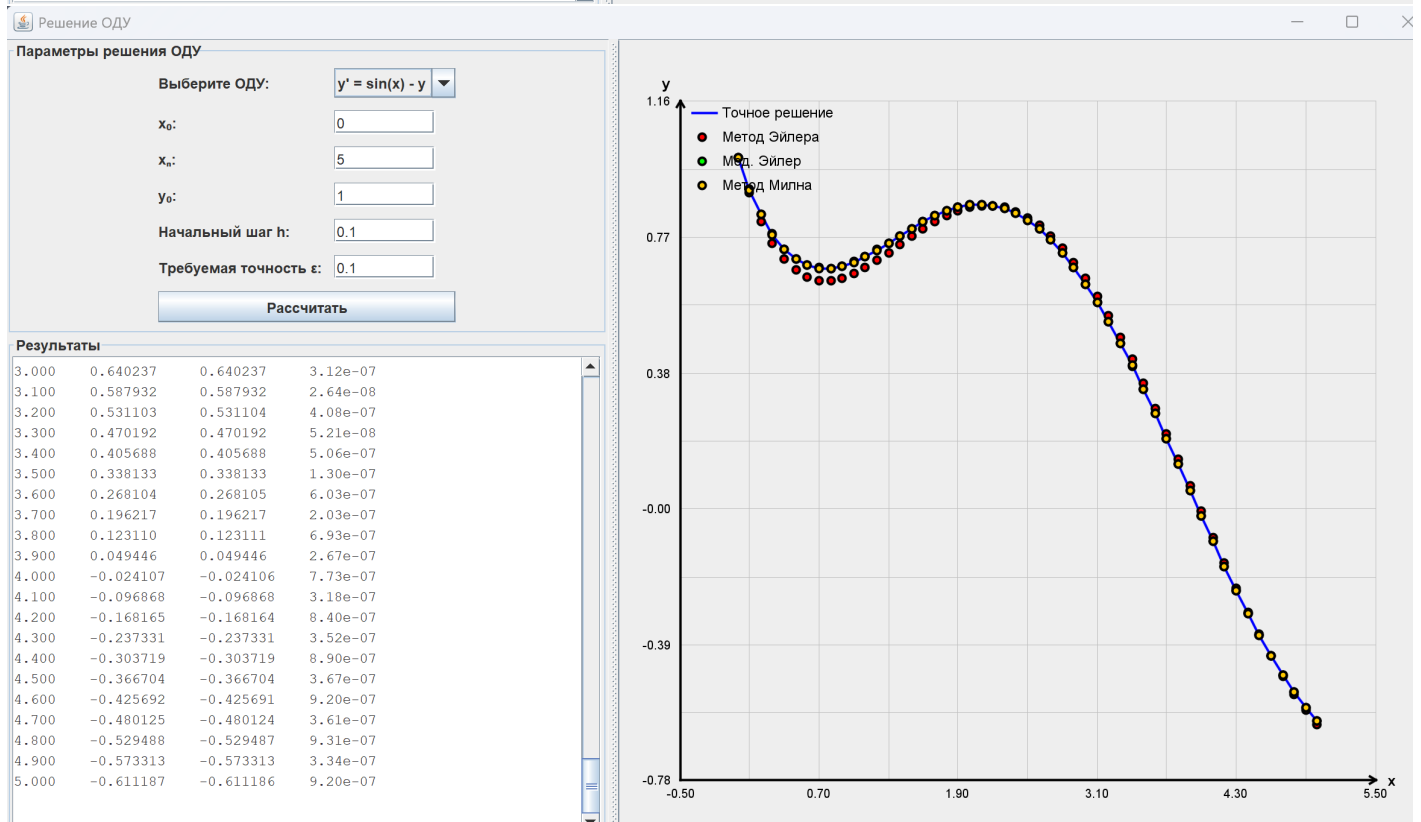
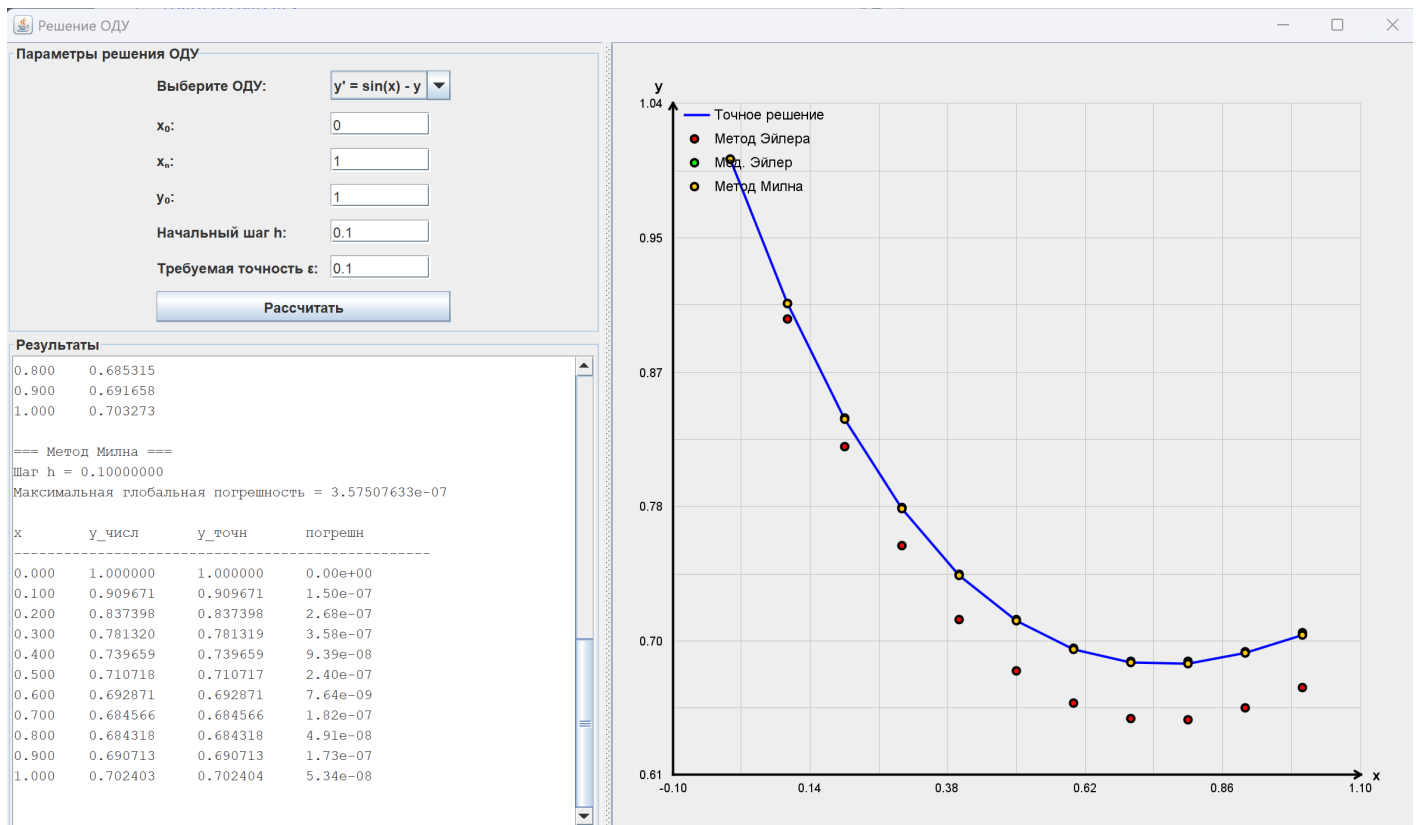
решить задачу Коши для обыкновенных дифференциальных уравнений численными методами.

2 Программная реализация

Код на GitHub: <https://github.com/Felijy/ITMO/tree/main/calc-math/lab6>

Примеры работы:





2.1 Метод Эйлера

```

public static List<Double> solve(ODESystem system, double x0, double xn, double y0,
    List<Double> ys = new ArrayList<>();
    double x = x0;

```

```

double y = y0;
ys.add(y);

while (x < xn - 1e-12) {
    double hCur = Math.min(h, xn - x);
    y += hCur * system.derivative(x, y);
    x += hCur;
    ys.add(y);
}
return ys;
}

public static RungeResult solveWithRunge(ODESystem system,
                                         double x0, double xn,
                                         double y0, double eps,
                                         double h) {

    double hCur = h;

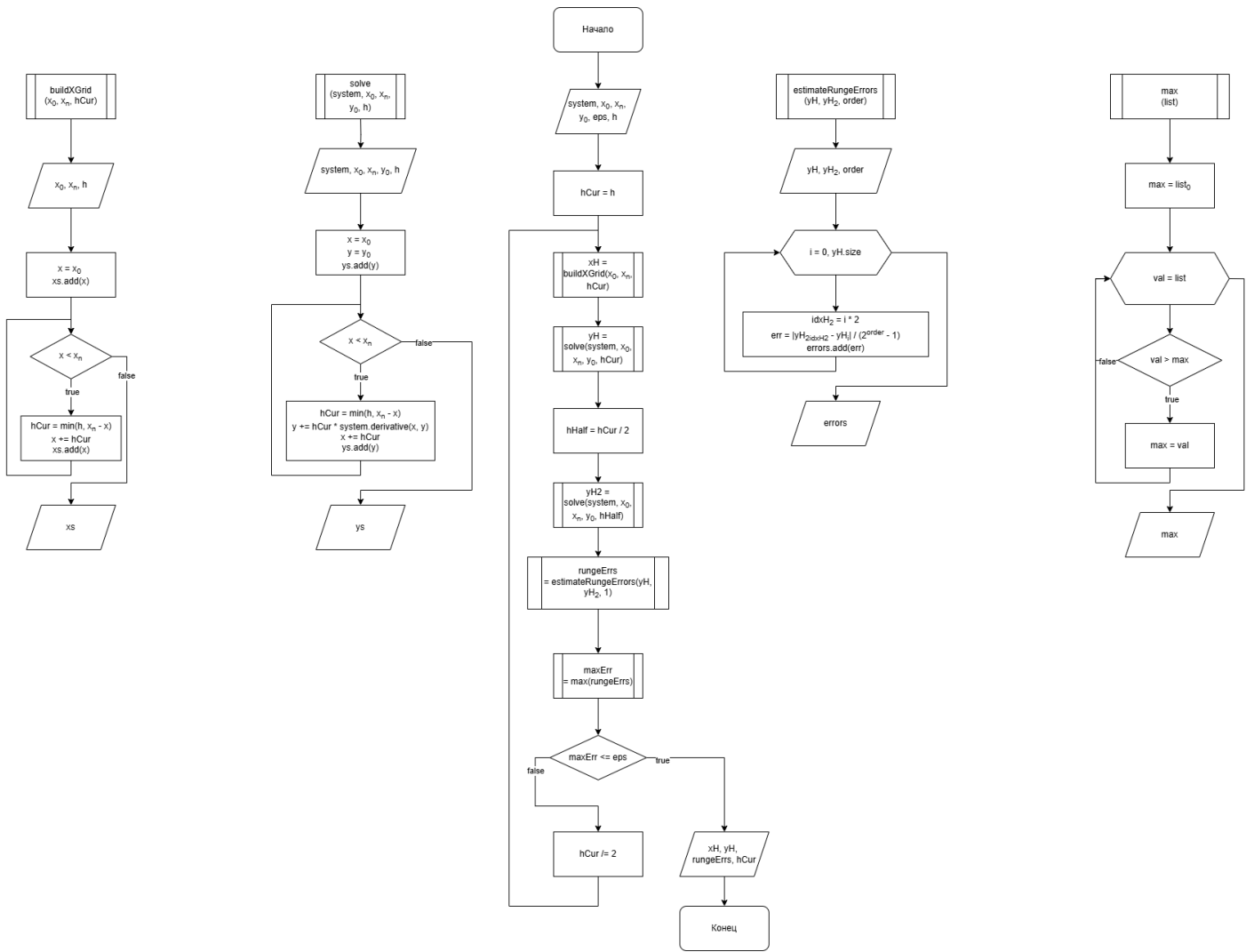
    while (true) {
        List<Double> xH = RungeHelper.buildXGrid(x0, xn, hCur);
        List<Double> yH = solve(system, x0, xn, y0, hCur);

        double hHalf = hCur / 2.0;
        List<Double> yH2 = solve(system, x0, xn, y0, hHalf);

        List<Double> rungeErrs = RungeHelper.estimateRungeErrors(yH, yH2, 1);
        double maxErr = RungeHelper.max(rungeErrs);

        if (maxErr <= eps) {
            return new RungeResult(xH, yH, rungeErrs, hCur);
        } else {
            hCur /= 2.0;
        }
    }
}

```



2.2 Модифицированный метод Эйлера

```

public static List<Double> solve(ODESystem system, double x0, double xn, double y0,
    List<Double> ys = new ArrayList<>();
    double x = x0;
    double y = y0;
    ys.add(y);

    while (x < xn - 1e-12) {
        double hCur = Math.min(h, xn - x);

        double f1 = system.derivative(x, y);
        double yPred = y + hCur * f1;
        double f2 = system.derivative(x + hCur, yPred);
        y = y + (hCur / 2.0) * (f1 + f2);

        x += hCur;
        ys.add(y);
    }
    return ys;

```

```

}

public static RungeResult solveWithRunge(ODESystem system,
                                         double x0, double xn,
                                         double y0, double eps,
                                         double h) {
    double hCur = h;

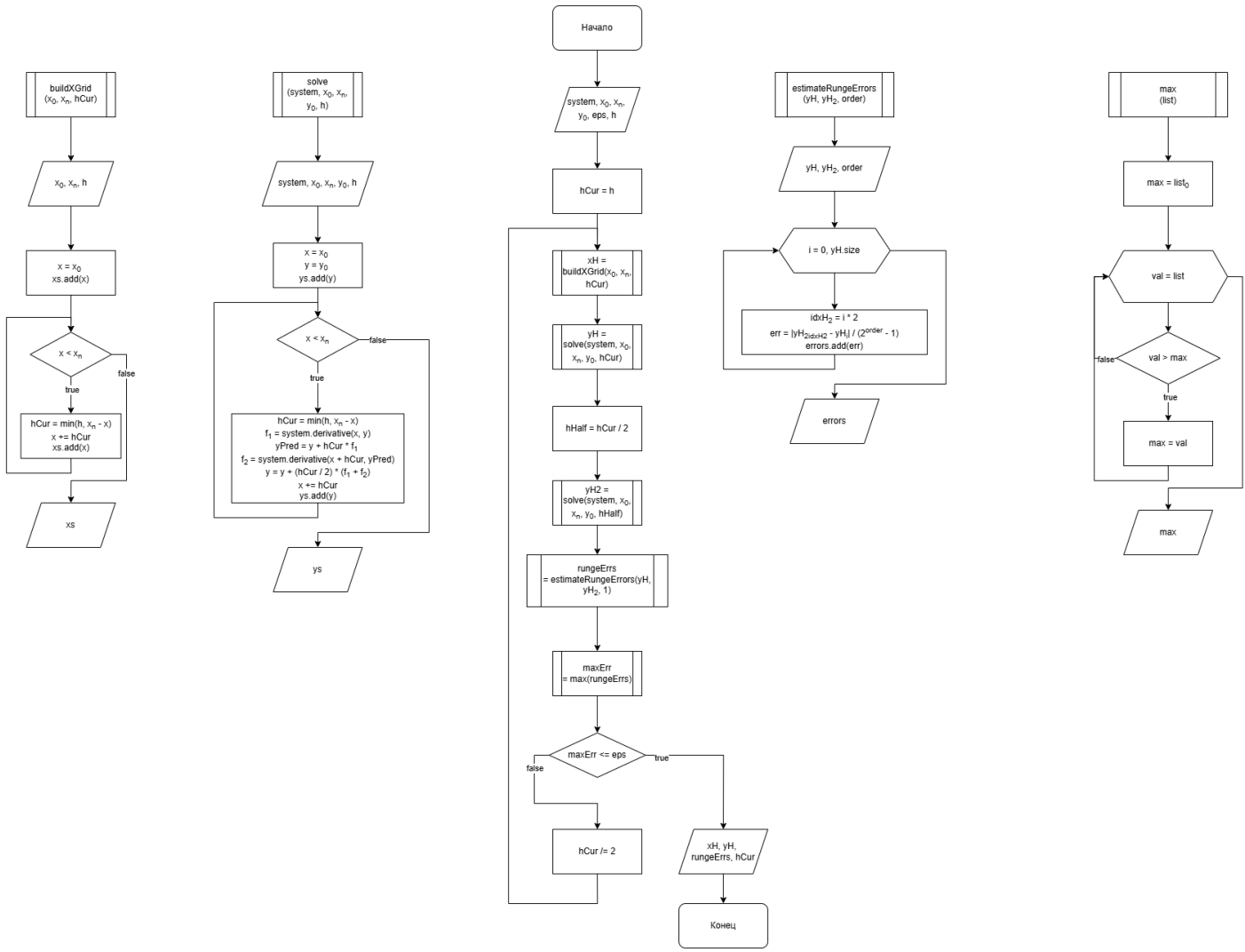
    while (true) {
        List<Double> xH = RungeHelper.buildXGrid(x0, xn, hCur);
        List<Double> yH = solve(system, x0, xn, y0, hCur);

        double hHalf = hCur / 2.0;
        List<Double> yH2 = solve(system, x0, xn, y0, hHalf);

        List<Double> rungeErrs = RungeHelper.estimateRungeErrors(yH, yH2, 2);
        double maxErr = RungeHelper.max(rungeErrs);

        if (maxErr <= eps) {
            return new RungeResult(xH, yH, rungeErrs, hCur);
        } else {
            hCur /= 2.0;
        }
    }
}

```



2.3 Метод Милна

```
public static List<Double> solve(ODESystem system, double x0, double xn, double y0,
    List<Double> xs = new ArrayList<>();
    List<Double> ys = new ArrayList<>();
```

```
    double x = x0;
    while (x <= xn + 1e-12) {
        xs.add(x);
        x += h;
        if (x > xn + 1e-12) {
            xs.add(xn);
            break;
        }
    }
}
```

```
ys.add(y0);
for (int i = 1; i < Math.min(4, xs.size()); i++) {
    double xi = xs.get(i - 1);
    double yi = ys.get(i - 1);
```



```

double hi = xs.get(i) - xi;

double k1 = hi * system.derivative(xi, yi);
double k2 = hi * system.derivative(xi + hi / 2, yi + k1 / 2);
double k3 = hi * system.derivative(xi + hi / 2, yi + k2 / 2);
double k4 = hi * system.derivative(xi + hi, yi + k3);

double yNext = yi + (k1 + 2 * k2 + 2 * k3 + k4) / 6;
ys.add(yNext);
}

for (int i = 4; i < xs.size(); i++) {
double xi = xs.get(i);
double hi = xs.get(i) - xs.get(i - 1);

double f1 = system.derivative(xs.get(i - 3), ys.get(i - 3));
double f2 = system.derivative(xs.get(i - 2), ys.get(i - 2));
double f3 = system.derivative(xs.get(i - 1), ys.get(i - 1));

double yPredictor = ys.get(i - 4) + 4 * hi * (2 * f1 - f2 + 2 * f3) / 3;

double yCorrector = yPredictor;
int maxIterations = 100;
int iteration = 0;

while (iteration < maxIterations) {
double f0 = system.derivative(xs.get(i - 2), ys.get(i - 2));
double f1_corr = system.derivative(xs.get(i - 1), ys.get(i - 1));
double f2_corr = system.derivative(xi, yCorrector);

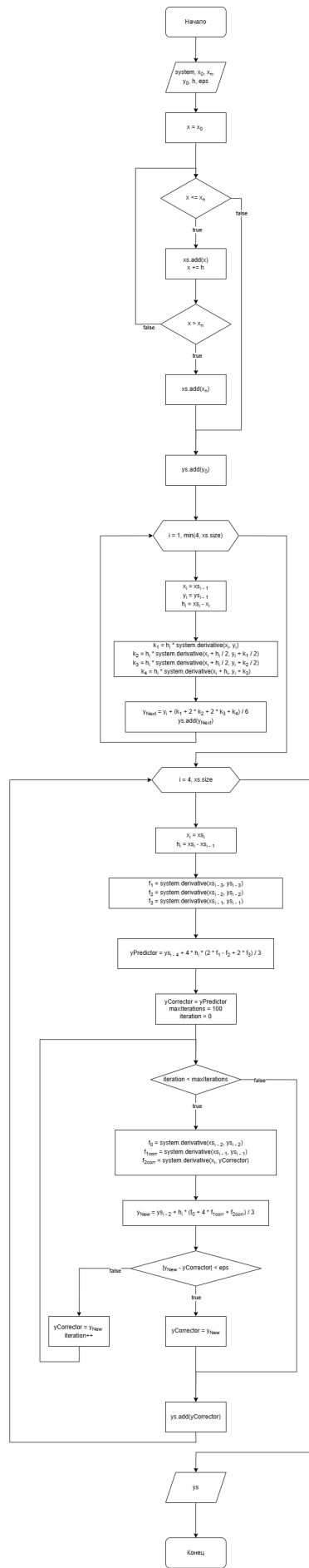
double yNew = ys.get(i - 2) + hi * (f0 + 4 * f1_corr + f2_corr) / 3;

if (Math.abs(yNew - yCorrector) < eps) {
yCorrector = yNew;
break;
}
yCorrector = yNew;
iteration++;
}

ys.add(yCorrector);
}

return ys;
}

```



3 Вывод

Проведя эту работу, я реализовал решение задач численными методами (методом Эйлера, модифицированным методом Эйлера, методом Милна). Также удалось их реализовать и построить графики для сравнения.