Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЁТ

По лабораторной работе №1

По курсу: «Моделирование»

Тема: «ОДУ. Задача Коши. Приближённый аналитический метод Пикара и численный метод Эйлера»

Студент: Керимов А. Ш.

Группа: ИУ7-64Б

Преподаватель: Градов В. М.

Оценка: ________________

Москва

2020

Рассмотрим задачу с начальным условием для дифференциального уравнения (задачу Коши):

$$\begin{cases} u'(x) = f(x, u), \\ u(\xi) = \eta \end{cases} \tag{1}$$

Решение можно найти приближённым аналитическим методом Пикара:

$$\begin{aligned} y^{(0)}(x) &= \eta \\ y^{(n+1)}(x) &= \eta + \int\limits_{\xi}^{x} f(t, y^{(n)}(t))\,\mathrm{d}t \end{aligned} \tag{2}$$

На примере $u'(x) = x^2 + y^2$ при $u(0) = 0$:

$$\begin{aligned} &y^{(0)}(0) = 0, \\ &y^{(1)}(x) = 0 + \int_0^x t^2\,\mathrm{d}t = \frac{x^3}{3}, \\ &y^{(2)}(x) = 0 + \int_0^x \left[t^2 + \left(\frac{t^3}{3}\right)^2\right]\mathrm{d}t = \frac{x^3}{3} + \frac{x^7}{63}, \\ &y^{(3)}(x) = 0 + \int_0^x \left[t^2 + \left(\frac{t^3}{3} + \frac{t^7}{63}\right)^2\right]\mathrm{d}t = \frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^{11}}{2079} + \frac{x^{15}}{59535}, \\ &\quad \ldots \end{aligned} \tag{3}$$

Кроме того, эту задачу можно решить численными методами. Следующая формула для явного способа:

$$y_{n+1} = y_n + h \cdot f(x_n, y_n) \tag{4}$$

Похожим образом выглядит неявный метод:

$$y_{n+1} = y_n + h \cdot f(x_{n+1}, y_{n+1}) \tag{5}$$

Стоит заметить, что для всех рассмотренных методов результат будет тем лучше, чем ближе значение x к ξ.

Реализованная программа производит расчет для $f(x, y) = x^2 + y^2$. Исходя из этого, можно упростить нахождения решения в неявном виде:

получаем квадратное уравнение и в качестве решения берем меньший корень.

Листинг 1: Реализация аналитического метода Пикара

```
std::vector<Polynomial> picard_polynomials(double xi, double eta, size_t
    iters_count) {
        const Polynomial x2({0, 0, 1});
        std::vector<Polynomial> ys{Polynomial({eta})};
        for (size_t i = 0; i < iters_count; ++i) {
                ys.push_back(eta + (x2 + (ys[i]^2)).integrate_from(xi));
        }
        return ys;
}

std::vector<std::vector<double>> picard_iterative_method(const Range& xs,
    double xi, double eta, size_t iters_count) {
        const auto ys = picard_polynomials(xi, eta, iters_count);

        std::vector<std::vector<double>> result(ys.size(),
            std::vector<double>(xs.size()));

        for (size_t i = ys.size() - 3; i < ys.size(); ++i) {
                for (size_t j = 0; j < xs.size(); ++j) {
                        result[i][j] = ys[i](xs[j]);
                }
        }

        return result;
}
```

Листинг 2: Реализация явного численного метода

```
std::vector<double> euler_explicit_method(const Range& xs, double xi,
    double eta) {
        const double h = xs.delta();

        double yn = eta;
        double xn = xi;

        std::vector<double> result(xs.size());
        for (size_t i = 0; i < xs.size(); ++i, xn += h) {
                yn = yn + h * f(xn, yn);
                result[i] = yn;
        }

        return result;
}
```

Листинг 3: Реализация неявного численного метода

```
double get_y(double x, double y, double h, bool& ok) {
        double root = y;
        const double c = h * x * x + y;
        const double d = 1 - 4 * h * c;
        ok = d >= 0;
        if (ok) {
                root = (1 - std::sqrt(d)) / (2 * h);
        }
        return root;
}

std::vector<double> euler_implicit_method(const Range& xs, double xi,
   double eta) {
        const double h = xs.delta();

        double yn = eta;
        double xn = xi;

        std::vector<double> result(xs.size());
        bool ok = true;
        size_t i = 0;
        for (; i < xs.size() && ok; ++i, xn += h) {
                yn = get_y(xn, yn, h, ok);
                result[i] = yn;
        }
        for (; i < xs.size(); ++i) {
                result[i] = yn;
        }

        return result;
}
```