Университет ИТМО

**Факультет: ПИиКТ**

**Дисциплина: Вычислительная математика**

**Лабораторная работа №5**

Вариант “2”

Выполнили: **Кудлаков Роман**

Группа: **P3231**

Преподаватель: **Перл Ольга Вячеславовна**

Санкт-Петербург, 2021

**Описание метода**

**Метод прогонки –** метод для решения дифференциальных уравнений второго порядка. С его помощью можно найти решение на отрезке [a, b] неоднородоного обыкновенного дифференциального уравнения второго порядка вида

Которое удовлетворяет граничным условиям

**Алгоритм решения**

1. Вводятся дифференциальное уравнение, которое надо решить, и граничные условия

2. Формируется система ДУ первого порядка для прямого хода и ДУ первого порядка для обратного хода по следующим правилам:

a) Если

Прямой ход:

Обратный ход:

b) Если

Прямой ход:

Обратный ход:

3. Решается задача Коши в прямом ходе

4. По найденным значениям решается задача Коши в обратном ходе

5. Строится график по результатам обратного хода

**Листинг кода**

**def** **advanced\_euler\_method** (func, x0, y0, xn, step, sign):

x = [x0]

y = [y0]

current\_x = x0

current\_y = y0

while current\_x < xn:

current\_res\_func = func.subs([('x', current\_x), ('y', current\_y)]).evalf()

delta\_y = step \* func.subs([('x', current\_x + step/2), ('y', current\_y + step/2 \* current\_res\_func)]).evalf()

current\_y += delta\_y \* sign

current\_x += step

x.append(current\_x)

y.append(current\_y)

return y

**def** **straight\_turn**(system, interval, start\_values, step):

z1\_points = [start\_values[0]]

z2\_points = [start\_values[1]]

current\_x = interval[0] + step

while current\_x < interval[1]:

new\_point1 = advanced\_euler\_method(system[0], current\_x, z1\_points[-1], current\_x + step, step, 1)

new\_point2 = advanced\_euler\_method(system[1].subs('z1', z1\_points[-1]), current\_x, z2\_points[-1],

current\_x + step, step, 1)

z1\_points.append(new\_point1[-1])

z2\_points.append(new\_point2[-1])

current\_x += step

return z1\_points, z2\_points

**def** **reverse\_turn**(func, interval, z1\_points, z2\_points, yn, step):

x = [interval[1]]

y = [yn]

current\_x = x[-1] - step

i = 2

while current\_x > interval[0]:

new\_y = advanced\_euler\_method(func.subs([('z1', z1\_points[-i]), ('z2', z2\_points[-i])]),

current\_x, y[-1], current\_x + step, step, -1)

x.append(current\_x)

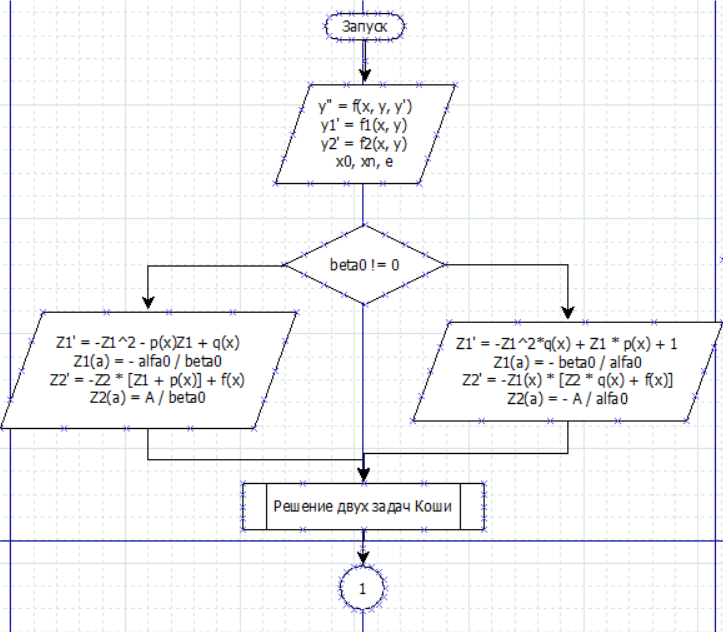
y.append(new\_y[-1])

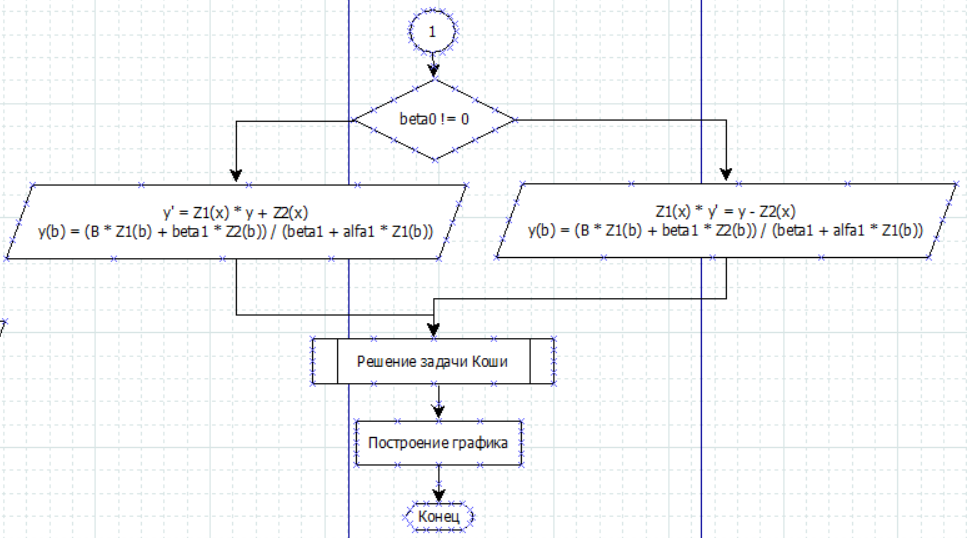
current\_x -= step

i += 1

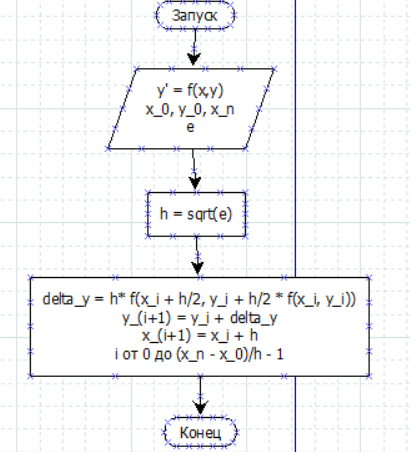
return x, y

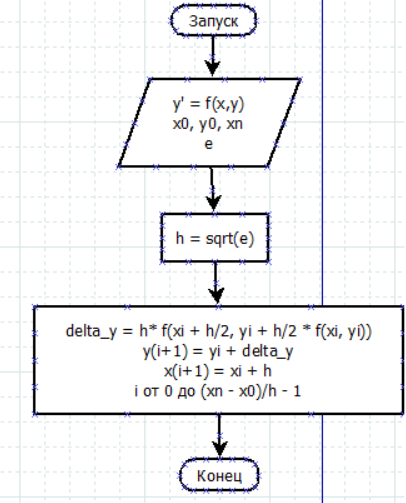
**Блок-схема**

****

****

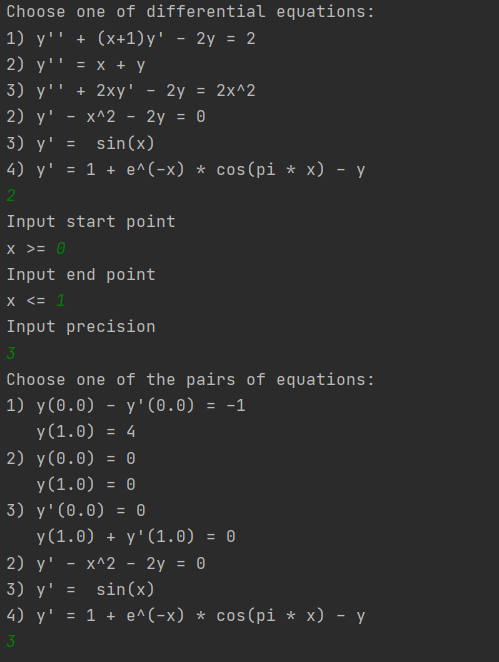
**Решение задачи Коши**

****

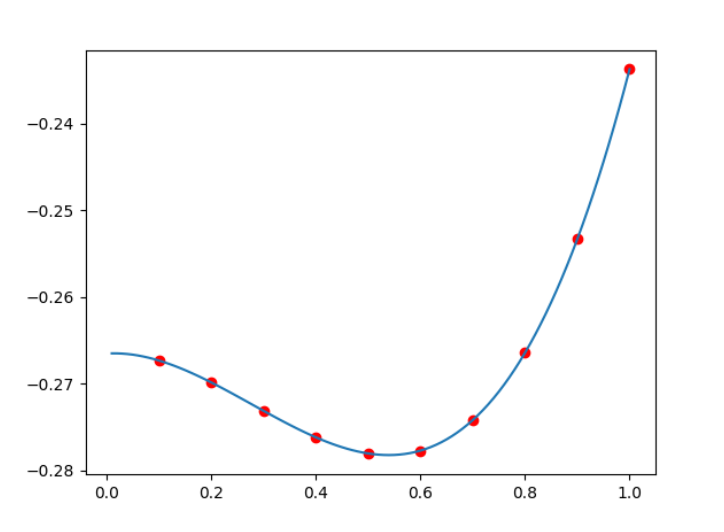
****

**Тесты и результаты**

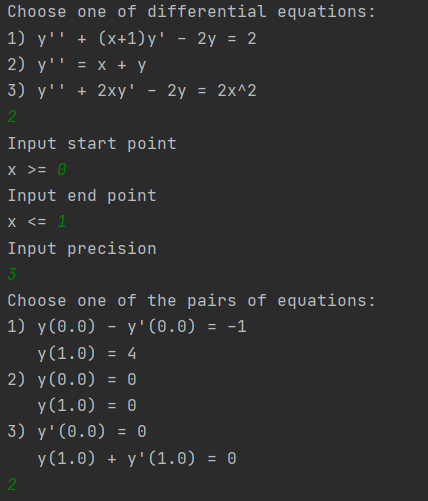
**Тест 1.**



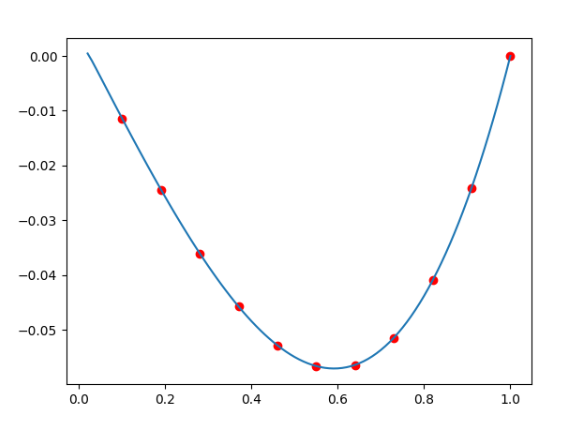
Точное решение:



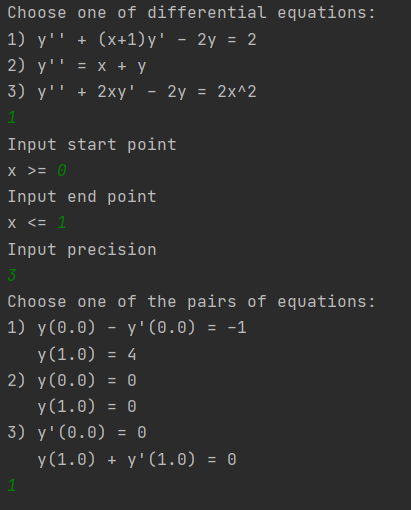
**Тест 2.**

****

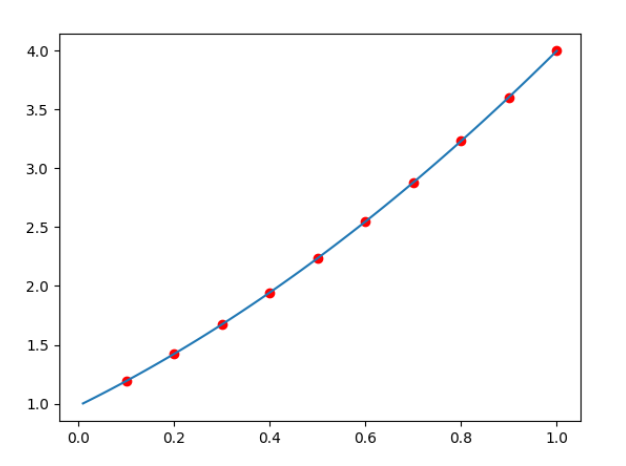
Точное решение:

****

**Тест 3**

****

Точное решение:



**Вывод.**

Дифференциальное уравнение (дальше ДУ) – уравнение, в которое входят производные функции и также могут входить сама функция, независимая переменная x и параметры. Найти решение ДУ – значит найти множество всех n раз дифференцируемых функций, которые удовлетворяют заданному уравнению. Обычно такое множество имеет вид , где С – произвольная постоянная. Также такое множество называют общим решением ДУ.

Краевая задача – задача о нахождении решения заданного дифференциального уравнения или их системы, удовлетворяющего краевым условиям в концах интервала или на границе области. Решить задачу означает найти функцию или множество функций или доказать, что задача решений не имеет.

Суть метода прогонки состоит в последовательном исключении неизвестных из трехдиагональной матрицы. Его отличием от других методов решения краевых задач является наличие двух ходов: прямого и обратного во время решения ДУ. По сравнению с методом стрельбы в методе прогонки изначально нужно разбить отрезок на n равных по длине частей, в то время как в методе стрельбы используются значения только в тех точках, которые были получены при помощи метода решения нелинейного уравнения. Также метод прогонки на более большей длине отрезка сделает меньшее количество итераций, нежели метод стрельбы, так как количество итераций для метода прогонки – 3\*n, а для метода стрельбы k\*n, где k – количество попыток пристрелок. Однако метод прогонки может быть использован только для линейных ДУ, а метод стрельбы как для линейных, так и для нелинейных.