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

Факультет Программной Инженерии и Компьютерной Техники

**Лабораторная работа по дисциплине «Вычислительная математика» №6**

Вариант: 4

Преподаватель:   
Рыбаков Степан Дмитриевич

Выполнил: Васильченко Роман

Группа: Р32081

Санкт-Петербург, 2023г

# Цель работы

# Решить задачу Коши для обыкновенных

# дифференциальных уравнений численными методами.

# Листинг программы

# import numpy as np

# import matplotlib.pyplot as plt

# from tabulate import tabulate

# def exact\_solution\_f1(x):

# return 2 \* np.exp(x) - x - 1

# def exact\_solution\_f2(x):

# return 1.49\*np.exp(-x) + np.sin(x) / 2 - np.cos(x) / 2

# def euler(f, y0, x0, xn, h):

# n = int((xn - x0) / h)

# x = np.linspace(x0, xn, n+1)

# y = np.zeros(n+1)

# y[0] = y0

# for i in range(n):

# y[i+1] = y[i] + h \* f(x[i], y[i])

# return x, y

# def improved\_euler(f, y0, x0, xn, h):

# n = int((xn - x0) / h)

# x = np.linspace(x0, xn, n+1)

# y = np.zeros(n+1)

# y[0] = y0

# for i in range(n):

# k1 = f(x[i], y[i])

# k2 = f(x[i] + h, y[i] + h \* k1)

# y[i+1] = y[i] + h \* (k1 + k2) / 2

# return x, y

# def adams(f, y0, x0, xn, h, m):

# n = int((xn - x0) / h)

# x = np.linspace(x0, xn, n+1)

# y = np.zeros(n+1)

# y[0] = y0

# for i in range(m):

# y[i+1] = y[i] + h \* f(x[i], y[i])

# for i in range(m, n):

# y[i+1] = y[i] + h \* (55\*f(x[i], y[i]) - 59\*f(x[i-1], y[i-1]) + 37\*f(x[i-2], y[i-2]) - 9\*f(x[i-3], y[i-3])) / 24

# return x, y

# def runge\_rule(f, y0, x0, xn, h1, h2, p):

# n1 = int((xn - x0) / h1)

# n2 = int((xn - x0) / h2)

# y\_h1 = np.zeros(n1+1)

# y\_h2 = np.zeros(n2+1)

# \_, y\_h1 = euler(f, y0, x0, xn, h1)

# \_, y\_h2 = euler(f, y0, x0, xn, h2)

# return np.abs(y\_h1[-1] - y\_h2[-1]) / (2\*\*p - 1)

# def main():

# f1 = lambda x, y: x + y

# f2 = lambda x, y: np.sin(x) - y

# f3 = lambda x, y: np.exp(-x) - y\*\*2

# print("f1 = x + y")

# print("f2 = sin(x) - y")

# print("f3 = e^(-x) - y^2")

# func\_input = input("Выберите функцию f1/f2/f3: ")

# func = f1

# if func\_input == "f1":

# func = f1

# exact\_solution = exact\_solution\_f1

# elif func\_input == "f2":

# func = f2

# exact\_solution = exact\_solution\_f2

# elif func\_input == "f3":

# func = f3

# exact\_solution = None

# else:

# main()

# return

# epsilon = runge\_rule(func, 1, 0, 1, 0.1, 0.05, 1)

# print(f"Точность метода Эйлера по правилу Рунге: {epsilon}")

# x0, xn, h = 0, 1, 0.1

# x\_exact = np.arange(x0, xn + h, h)

# if exact\_solution is not None:

# y\_exact = exact\_solution(x\_exact)

# plt.plot(x\_exact, y\_exact, label='Exact solution')

# x\_euler, y\_euler = euler(func, 1, 0, 1, 0.1)

# plt.plot(x\_euler, y\_euler, label='Euler')

# x\_improved\_euler, y\_improved\_euler = improved\_euler(func, 1, 0, 1, 0.1)

# plt.plot(x\_improved\_euler, y\_improved\_euler, label='Improved Euler')

# x\_adams, y\_adams = adams(func, 1, 0, 1, 0.1, 4)

# plt.plot(x\_adams, y\_adams, label='Adams')

# # Подготовка данных для таблицы

# table\_data = list(zip(x\_euler, y\_euler, y\_improved\_euler, y\_adams))

# # Вывод таблицы с использованием tabulate

# print(tabulate(table\_data, headers=["x", "Euler", "Improved Euler", "Adams"]))

# epsilon = runge\_rule(func, 1, 0, 1, 0.1, 0.05, 1)

# print(f"\nТочность метода Эйлера по правилу Рунге: {epsilon}\n")

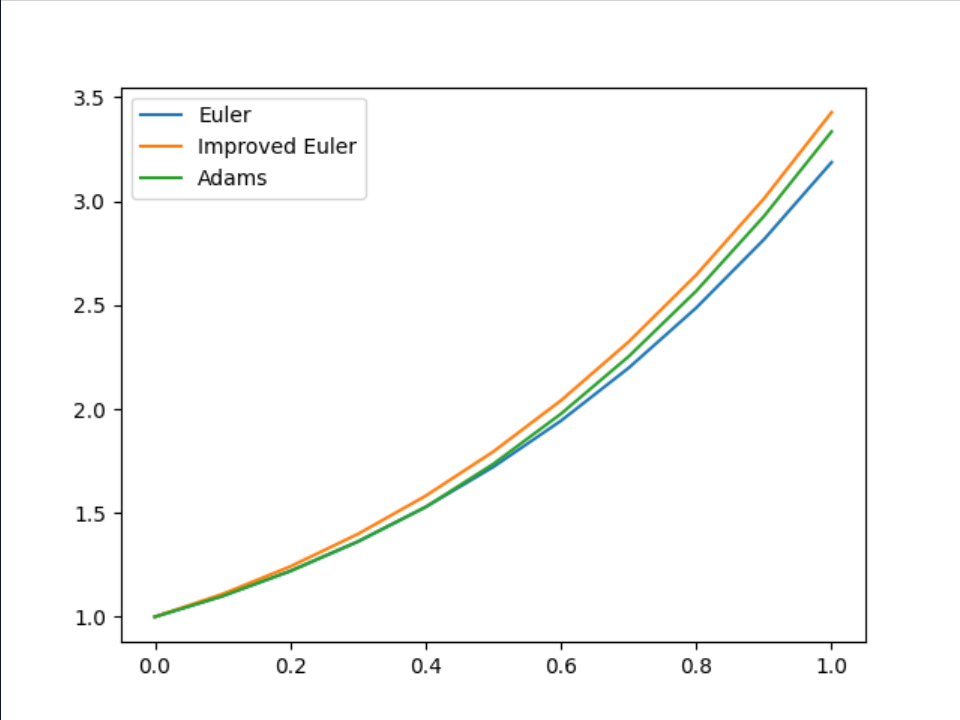
# plt.legend()

# plt.show()

# if \_\_name\_\_ == "\_\_main\_\_":

# main()

# Результаты выполнения программы



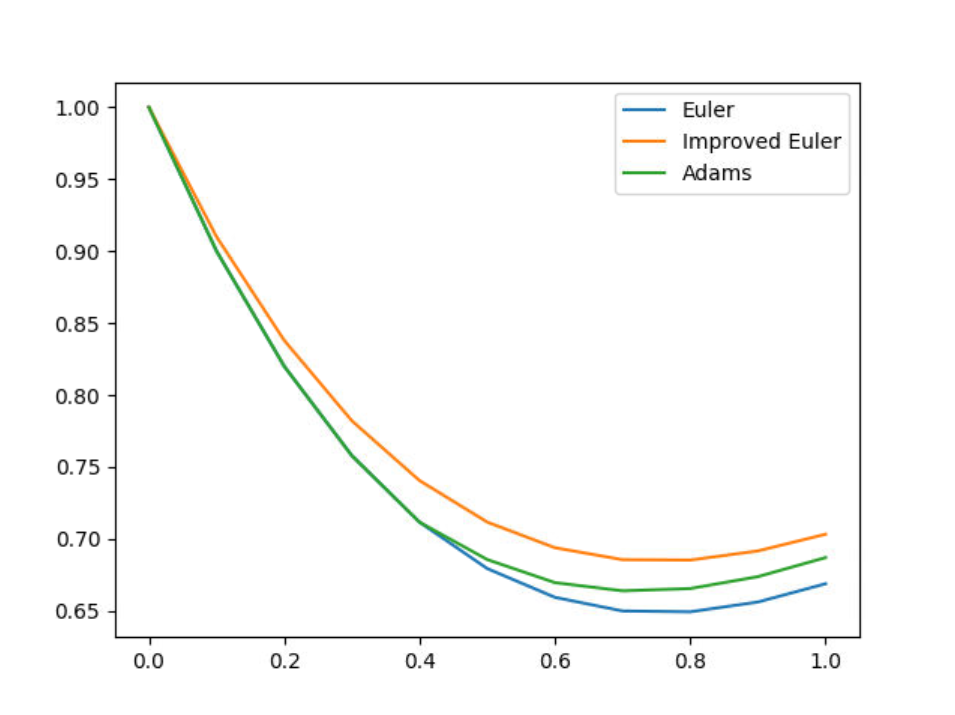
# f1 = x + y

# f2 = sin(x) - y

# f3 = e^(-x) - y^2

# Выберите функцию f1/f2/f3 f1

# Точность метода Эйлера по правилу Рунге: 0.11911049008884023



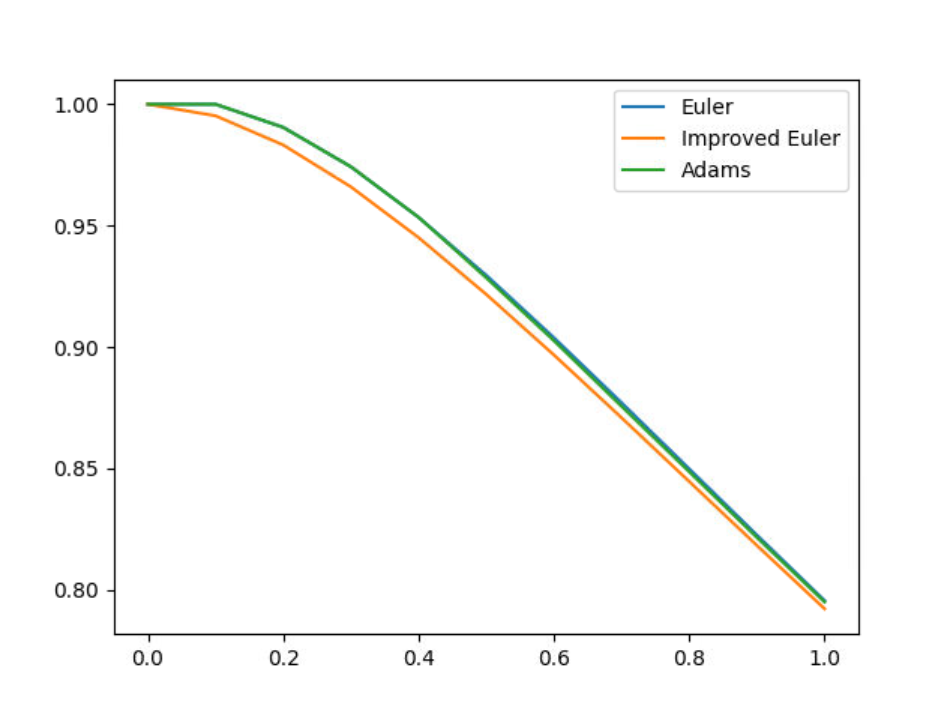
# f1 = x + y

# f2 = sin(x) - y

# f3 = e^(-x) - y^2

# Выберите функцию f1/f2/f3 f2

# Точность метода Эйлера по правилу Рунге: 0.017184509533635484



# f1 = x + y

# f2 = sin(x) - y

# f3 = e^(-x) - y^2

# Выберите функцию f1/f2/f3 f3

# Точность метода Эйлера по правилу Рунге: 0.0016874540522936465

# Выводы

В ходе выполнения данной лабораторной работы я рассмотрел и реализовал численные методы решения обыкновенных дифференциальных уравнений: метод Эйлера, усовершенствованный метод Эйлера и метод Адамса.

Реализация этих методов была произведена на языке Python. Я также реализовал правило Рунге для оценки точности одношагового метода (метода Эйлера). Визуализация результатов позволила продемонстрировать эффективность каждого из методов. Во время работы я поработал с численными методами в решении обыкновенных дифференциальных уравнений.