**Министерство науки и высшего образования Российской Федерации**

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

Факультет программной инженерии и компьютерной техники

**Дисциплина:** Вычислительная математика

**Лабораторная работа №6**

**«Численное дифференцирование»**

**Вариант №10**

Выполнил:

Мокров Семён Андреевич

Группа:

P3215

Проверила:

Малышева Татьяна Алексеевна

Санкт-Петербург

2022

# Цель работы

Решить задачу Коши численными методами.

Для исследования использовать:

* Одношаговые методы;
* Многошаговые методы.

# Задание лабораторной работы

***Программная реализация задачи:***

* Исходные данные: ОДУ вида , начальные условия , интервал дифференцирования [*a, b*], шаг *h*, точность .
* Составить таблицу приближенных значений интеграла дифференциального уравнения, удовлетворяющего начальным условиям. Для оценки точности использовать правило Рунге.
* Построить графики точного решения и полученного численного решения (разными цветами).

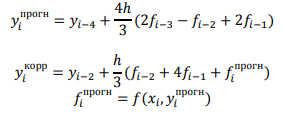
# Рабочие формулы

**Метод Эйлера:**

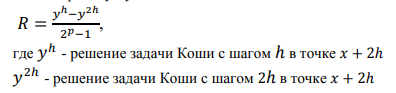
****

****

**Метод Милна:**

****

**Правило Рунге:**

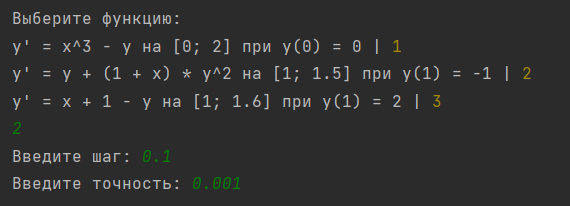
****

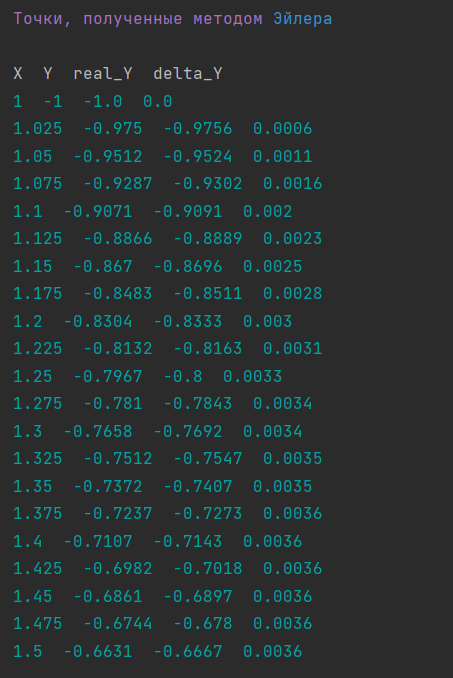
# Листинг программы

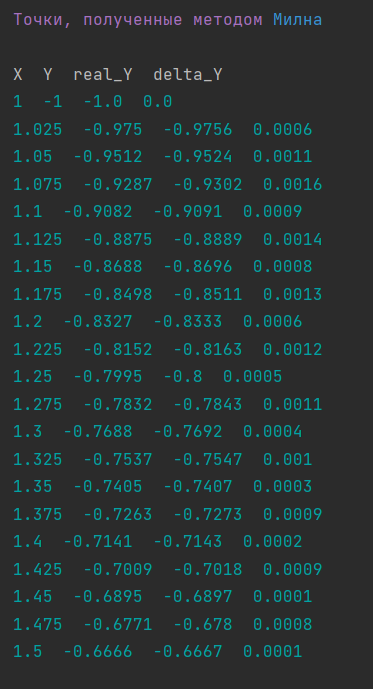
Весь исходный код программы располагается на GitHub (адрес: https://github.com/semwett0301/lab6\_math)

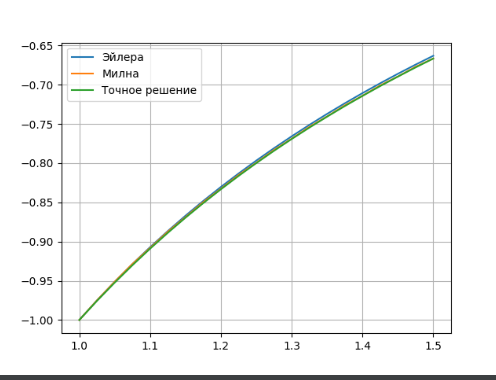
def euler\_method(source\_function, a, b, y0, h, e):  
 x = a  
 y = y0  
 points = [[x, y]]  
 while x <= b:  
 new\_y = y + h \* source\_function(x, y)  
 new\_y\_h\_2 = y + h \* source\_function(x, y) / 2  
 new\_y\_h\_2 += h \* source\_function(x + h / 2, new\_y\_h\_2) / 2  
  
 if abs(new\_y - new\_y\_h\_2) > e:  
 h /= 2  
 continue  
  
 x += h  
 y = new\_y  
 points.append([x, new\_y])  
  
 points.pop(len(points) - 1)  
  
 return points  
  
  
def miln\_method(source\_function, a, b, y0, h, e):  
 def calculate\_predicted(points):  
 return points[num - 4][1] + 4 \* h \* \  
 (2 \* source\_function(points[num - 3][0], points[num - 3][1]) - source\_function(points[num - 2][0], points[num - 2][1])  
 + 2 \* source\_function(points[num - 1][0], points[num - 1][1])) / 3  
  
 def calculate\_correct(points, y\_predicted):  
 return points[num - 2][1] + h \* (source\_function(points[num - 2][0], points[num - 2][1])  
 + 4 \* source\_function(points[num - 1][0], points[num - 1][1])  
 + source\_function(points[num - 1][0] + h, y\_predicted)) / 3  
  
 x = a  
 y = y0  
 points = [[x, y]]  
 count = 1  
 while x <= b:  
 new\_y = y + h \* source\_function(x, y)  
 new\_y\_h\_2 = y + h \* source\_function(x, y) / 2  
 new\_y\_h\_2 += h \* source\_function(x + h / 2, new\_y\_h\_2) / 2  
  
 if abs(new\_y - new\_y\_h\_2) > e:  
 h /= 2  
 continue  
  
 x += h  
 y = new\_y  
 points.append([x, new\_y])  
 count += 1  
 if count == 4:  
 break  
  
  
 for num in range(4, int((b - a) / h) + 1):  
 y\_predicted = calculate\_predicted(points)  
  
 y\_correct = calculate\_correct(points, y\_predicted)  
 while abs(y\_predicted - y\_correct) > e:  
 y\_predicted = y\_correct  
 y\_correct = calculate\_correct(points, y\_predicted)  
 points.append((points[num - 1][0] + h, y\_correct))  
 return points  
  
  
methods = [  
 [euler\_method, "Эйлера"],  
 [miln\_method, "Милна"]  
]

# Примеры и результаты работы программы

****

****

****

****

# Выводы

В данной лабораторной работе я:

* Познакомился с принципами решения задачи Коши при помощи вычислительных методов
* Ознакомился с методами решения задачи Коши - Эйлера и Милна;
* Реализовал их на языке Python;
* Так как методы имеют разную степень точности, то на различных значениях они выдают разные значения (метод Милна более точен и выдает значения, более приближенные к точному решению, тем больше, чем меньше h).