ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Дисциплина:

«Вычислительная математика»

Лабораторная работа №5

«Решение ОДУ и задачи Коши»

Выполнил:

Студент группы P3231

Колмаков Дмитрий Владимирович

Преподаватель:

Перл О.В.

Санкт-Петербург

2024 г.

Оглавление

Задание

Title: **Метод Адамса**

Description: Реализуйте метод Адамса для решения обыкновенных дифференциальных уравнений по начальному значению (задача Коши) в интервале от a до b [a,b].
f
epsilon
a
y(a)
b
f **-** номер уравнения, где уравнение в виде y'=f(x,y). Вы должны получить функцию по номеру из входных данных в методе get_function.
Вы должны определить и пересчитать шаг h самостоятельно.
Вы должны вычислить и вернуть y(b) с разницей, не превышающей epsilon.

Подсказка: Чтобы использовать метод Адамса для решения задачи Коши, вам нужно больше 1 начальной точки. Поэтому вам также необходимо реализовать еще один метод для вычисления 3 дополнительных разгоночных точек. Вы можете выбрать этот метод самостоятельно, но если вы неправильно рассчитаете начальный набор точек, то это может повлиять на все решение.

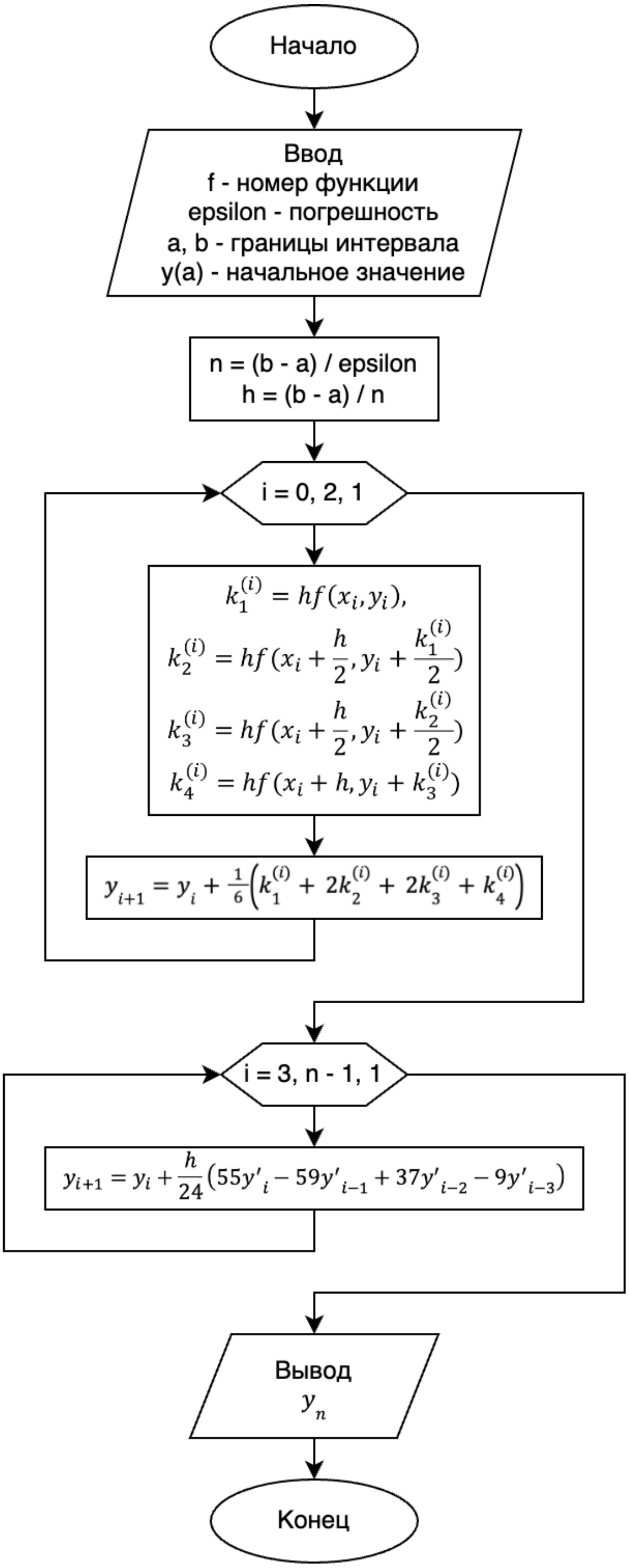
Теория

Метод Адамса (Адамса-Башфорта) - многошаговый метод для решения задачи Коши. Относится к явным методам, то есть расчет следующей точки основывается только на предыдущих точках и значениях. Чаще всего используется метод Адамса-Башфорта 4-го порядка, он требует предварительного вычисления решения в 4 начальных точках. Зная только одну начальную точку, можно рассчитать 3 дополнительных начальных точки с помощью одношагового метода, например, метода Рунге-Кутты. Метод Рунге-Кутты имеет порядок точности $O(h^5)$, что выше, чем $O(h^4)$ у Адамса-Башфорта, поэтому его используют чаще всего. Метод Эйлера с порядком точности $O(n^2)$ использовать не следует, так как порядок точности меньше, и ошибка может повлиять на все решение.

Таким образом, зная 4 начальные точки, можем найти все остальные по формуле:

$$y_{i+1} = y_i + \frac{h}{24}\left(55y'(x_{i-1}, y_{i-1}) - 59y'(x_{i-2}, y_{i-2}) + 37y'(x_{i-3}, y_{i-3}) - 9y'(x_{i-4}, y_{i-4})\right)$$

Блок-схема



Начало
Ввод
f - номер функции
epsilon - погрешность
a, b - границы интервала
y(a) - начальное значение
n = (b - a) / epsilon
h = (b - a) / n
i = 0, 2, 1
$k_1^{(i)} = hf(x_i, y_i),$
$k_2^{(i)} = hf(x_i + \frac{h}{2}, y_i + \frac{k_1^{(i)}}{2})$
$k_3^{(i)} = hf(x_i + \frac{h}{2}, y_i + \frac{k_2^{(i)}}{2})$
$k_4^{(i)} = hf(x_i + h, y_i + k_3^{(i)})$
$y_{i+1} = y_i + \frac{1}{6}\left(k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)} + k_4^{(i)}\right)$
i = 3, n - 1, 1
$y_{i+1} = y_i + \frac{h}{24}(55y'_i - 59y'_{i-1} + 37y'_{i-2} - 9y'_{i-3})$
Вывод
y_n
Конец

Реализация программы

```python
import math


class Result:
   @staticmethod
   def first_function(x: float, y: float):
       return math.sin(x)

   @staticmethod
   def second_function(x: float, y: float):
       return (x * y) / 2

   @staticmethod
   def third_function(x: float, y: float):
       return y - (2 * x) / y

   @staticmethod
   def fourth_function(x: float, y: float):
       return x + y

   @staticmethod
   def default_function(x: float, y: float):
       return 0.0

   @staticmethod
   def get_function(n: int):
       if n == 1:
           return Result.first_function
       elif n == 2:
           return Result.second_function
       elif n == 3:
           return Result.third_function
       elif n == 4:
           return Result.fourth_function
       else:
           return Result.default_function

   @staticmethod
   def solveByAdams(f_num, epsilon, a, y_a, b):
       func = Result.get_function(f_num)
       n = math.ceil((b - a) / epsilon)
       h = (b - a) / n
       y = y_a
       values = [y_a, None, None, None]

       for i in range(3):
           x = a + i * h
           k1 = h * func(x, y)
           k2 = h * func(x + h / 2, y + k1 / 2)
           k3 = h * func(x + h / 2, y + k2 / 2)
           k4 = h * func(x + h, y + k3)
           dy = (k1 + 2 * k2 + 2 * k3 + k4) / 6
           y = y + dy
           values[i + 1] = y
```

```
        for i in range(3, n):
            x = a + i * h
             y = y + (55 * func(x, values[-1]) - 59 * func(x - h, values[-2]) + 37 *
func(x - 2 * h, values[-3]) - 9 * func(x - 3 * h, values[-4])) * h / 24
            values.pop(0)
            values.append(y)

        return y


if __name__ == '__main__':
    f = int(input().strip())
    epsilon = float(input().strip())
    a = float(input().strip())
    y_a = float(input().strip())
    b = float(input().strip())
    result = Result.solveByAdams(f, epsilon, a, y_a, b)
    print(str(result) + '\n')
```

Тесты

Тест 1

Первая функция ($f(x) = sin(x)$)

Входные данные:
1
0.0001
0
1
1

Выходные данные:
1.4596976941318742

Тест 2

Вторая функция ($f(x) = \frac{xy}{2}$)

Входные данные:
2
0.0001
0
1
1

Выходные данные:
1.2840254166877396

Тест 3

Третья функция ($f(x) = y - \frac{2x}{y}$)

Входные данные:
3
0.0001
0
1
1

Выходные данные:
1.7320508075688743

Тест 4

Четвертая функция ($f(x) = x + y$), низкая точность

Входные данные:
4
0.1
0
1
1

Выходные данные:
3.4364488783644997

Тест 5

Четвертая функция ($f(x) = x + y$), высокая точность

Входные данные:
4
0.0000001
0
1
1

Выходные данные:
3.4365636569184246

Тест 6

Четвертая функция ($f(x) = x + y$), большой интервал

Входные данные:
4
0.0001
0
1
100

Выходные данные:
5.376234283632442e+43

Вывод

В ходе проделанной работы я повторил общие сведения о дифференциальных уравнениях, решении ОДУ и задачи Коши. Изучил различные численные методы - одношаговые (метод Эйлера (обычный, улучшенный и усовершенствованный), метод Рунге-Кутты) и многошаговые (метод Милна, метод Адамса-Башфорта, метод Адамса-Мультона). Подробно изучил методы Адамса-Башфорта и Рунге-Кутты. Работая над ними, выяснил, что:

- так как метод Адамса - многошаговый, для него требуется знать не одну, а несколько начальных точек. Их можно найти с помощью одношагового метода. Для такого метода порядок точности должен быть выше или совпадать, поэтому часто используют метод Рунге-Кутты;
- алгоритмическая сложность как метода Адамса, так и всех методов - $O(n)$, где n - количество элементарных отрезков;
- однако у методов отличается порядок точности. Так, у метода Адамса 4-го порядка он составляет $O(h^4)$, у метода Рунге-Кутты 4-го порядка - $O(h^5)$, у методов Эйлера же он самый низкий - $O(h^2)$;
- в сравнение с методом Адамса:
  - метод Эйлера проще в реализации, но обладает большей погрешностью;
  - метод Рунге-Кутты обладает схожей точностью, но требует вычисления промежуточных значений функции на каждом шаге, что увеличивает количество вычислений;
  - метод Адамса-Мультона обладает такой же точностью, но требует решения нелинейного уравнения на каждом шаге интегрирования, что увеличивает количество вычислений.

Также я реализовал численный метод, реализующий метод Адамса (с применением метода Рунге-Кутты для вычисления начальных точек). На тестовых данных моя реализация работает корректно, находит правильное решение с заданной точностью.