МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

**Приближенное решение обыкновенных дифференциальных**

**уравнений**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к лабораторной работе №6

по дисциплине

**Вычислительная Математика**

РУКОВОДИТЕЛЬ:

Суркова Анна Сергеевна

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

СТУДЕНТ:

Цветков Николай Максимович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

19-ИВТ-3

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2021

Оглавление

[Цель 3](#_Toc71720598)

[Постановка задачи 4](#_Toc71720599)

[Теоретические сведения 5](#_Toc71720600)

[Расчетные данные 16](#_Toc71720601)

[Листинг разработанной программы 18](#_Toc71720602)

[Результаты работы программы 20](#_Toc71720603)

[Вывод 21](#_Toc71720604)

# Цель

Закрепление знаний и умений по численному решению обыкновенных дифференциальных уравнений методом Эйлера, методом Эйлера с пересчетом, методом Рунге-Кутты и методом Адамса.

# Постановка задачи

Задание 1

Используя метод Эйлера и метод Эйлера с пересчетом, составить таблицу приближенных значений интеграла дифференциального уравнения , удовлетворяющих начальным условиям на отрезке ; шаг . Все вычисление вести с четырехзначными знаками. Проверить полученные значения, использую метод Рунге-Кутты 4 порядка.

Задание 2

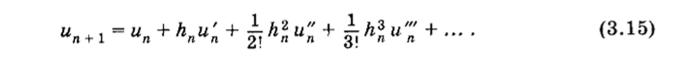
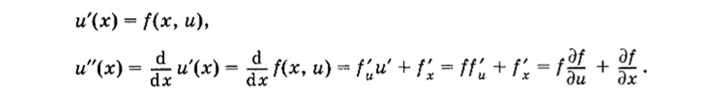
Используя метод Адамса с третьими разностями составить таблицу приближенных значений интеграла дифференциального уравнения , удовлетворяющих начальным условиям на отрезке ; шаг . Все вычисление вести с четырехзначными знаками. Начальный отрезок определить методом Рунге-Кутты. Проверить полученные значения, используя метод Эйлера с пересчетом.

# Теоретические сведения

**Метод Эйлера и метод Эйлера с пересчетом**

Обозначим границы отрезка а и Ь через х0 и xN соответственно и введем на отрезке [а, ?>] сетку (в общем случае неравномерную) значений аргумента х такую, чтобы выполнялось соотношение х0 < х1 < ... < xN. Выполним разложение решения и (л:) в окрестности узла сетки хп по формуле Тейлора. Обозначив шаг сетки через hn = хп + 1- хп и и(хп) через ип, получаем

Если функция f имеет непрерывную р-ю производную, то в соотношении (3.15) можно оставлять члены вплоть до 0(hP ' \*). Эти производные можно найти, дифференцируя правую часть уравнения (3.14) требуемое число раз. Например, для первой и второй производных имеем

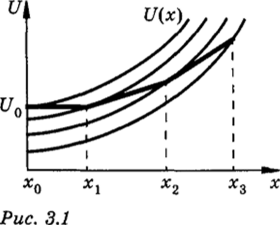


налогично можно получить производные более высоких порядков.

Однако использование формулы (3.15) с большим числом членов имеет ряд недостатков: во-первых, с ростом порядка производной выражение для нее может оказаться очень сложным; кроме того, если функция f известна лишь приближенно или задана таблично, ее производные находятся с большой ошибкой. В связи с этим в разложении (3.15) оставляют только два члена. При такой замене вместо точного решения и{хп т j) получается его приближенное значение и(хп + t), которое находится по формуле

https://studme.org/htm/img/33/2271/366.png

Так как значение п(дс0) = ь то, последовательно пользуясь женные решения uv и2, ..., uN,



0 известно из начального условия, формулой (3.16), находим прибли- Формула (3.16) записана для случая неравномерной сетки. Полагая шаг сетки hn = h постоянным, получим

https://studme.org/htm/img/33/2271/368.png

Формула (3.16а) является ОСНОВНОЙ ФОРМУЛОЙ МЕТОДА ЭЙЛЕРА или МЕТОДА ЛОМАНЫХ.

Существует несколько модификаций метода Эйлера. Остановимся на одной из них — методе Эйлера с пересчетом, который называют также МЕТОДОМ ЭЙЛЕРА — КОШИ. В этом методе значение ип \_ j находится по формуле https://studme.org/htm/img/33/2271/369.png

т. е. вместо тангенса угла наклона касательной к интегральной кривой в точке (хп, ип), который имеет место в формуле (3.16а), выражающей метод Эйлера, используется полусумма значений тангенсов углов наклона касательных в известной (хп, ип) и искомой (хп + j, ип + t) точках. Поскольку, однако, значение ип + г неизвестно, то (3.17) есть в общем случае нелинейное уравнение относительно ип + 1, которое можно решить различными методами, изложенными в главе 1.

В рассматриваемом случае логично использовать метод простой итерации, представленный формулой (1.10), поскольку нелинейное уравнение уже разрешено относительно ип + v Тогда, если номер итерации обозначить верхним индексом, итерационный процесс запишется в виде

https://studme.org/htm/img/33/2271/370.png

В качестве значения и(л0)+1 можно принять либо ип, либо и^+1 = ~ ип + hf(xn, ип), т. е. использовать значение, вычисленное по формуле Эйлера (3.16а). В этом случае в первой итерации имеем

https://studme.org/htm/img/33/2271/371.png

Формула (3.19) и есть ОСНОВНАЯ ФОРМУЛА МЕТОДА ЭЙЛЕРА С ПЕРЕСЧЕТОМ. Подобные схемы часто называют схемами типа ПРОГНОЗ-КОРРЕКТОР (ПРЕДИКТОР-КОРРЕКТОР). Сначала по формуле (3.16) определяется прогнозируемое приближение решения, а затем по формуле (3.19) это решение уточняется.

Метод Эйлера с пересчетом обладает третьим порядком точности на шаге. Действительно, из соотношения (3.15) следует

https://studme.org/htm/img/33/2271/372.png

Разлагая в ряд второй член в квадратных скобках (3.19), имеем

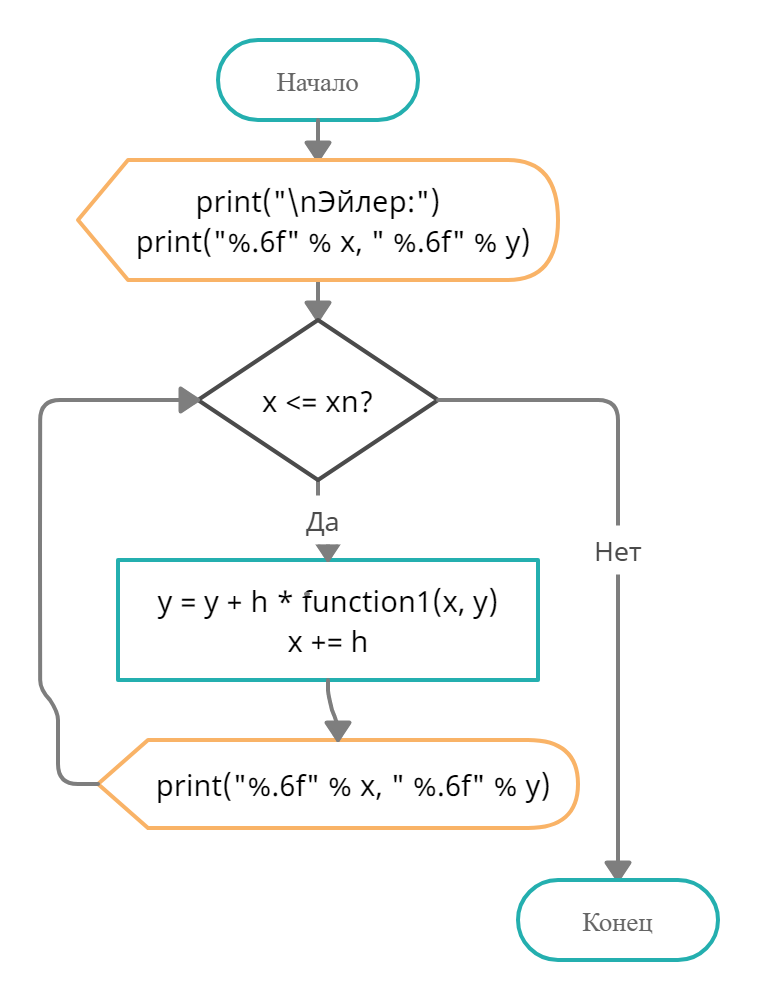
https://studme.org/htm/img/33/2271/373.png

Подставляя правую часть соотношения (3.21) в последний член выражения (3.19), получаем

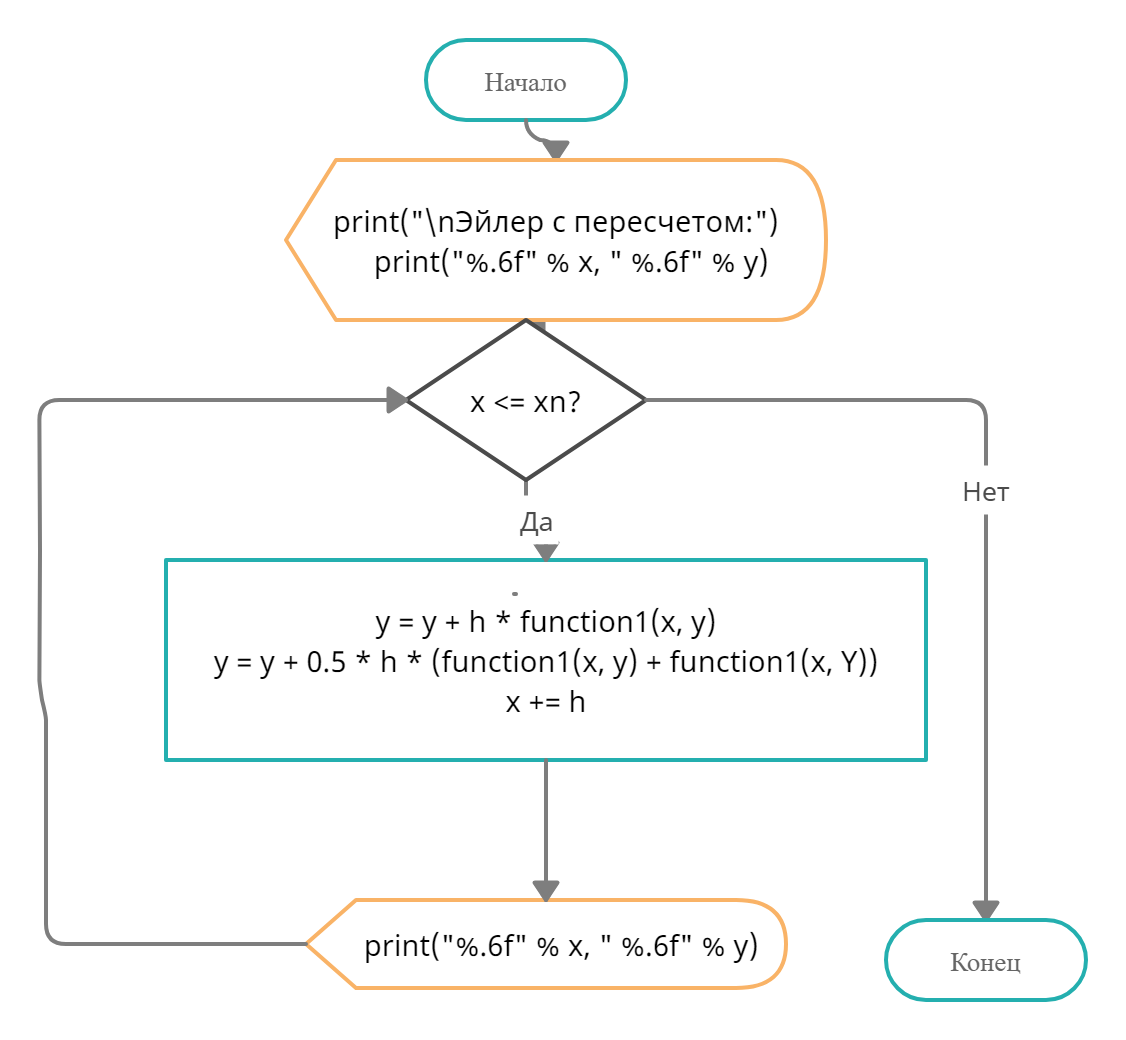
https://studme.org/htm/img/33/2271/374.png

Сравнивая соотношения (3.20) и (3.22), видим, что метод Эйлера с пересчетом обладает третьим порядком точности на шаге и, соответственно, вторым на интервале. Метод Эйлера с пересчетом дает двустороннее приближение к решению.

**Эйлер**

****

**Эйлер с пересчетом**

****

**Метод Рунге-Кутты**

Метод Рунге-Кутты используют для расчета стандартных моделей достаточно часто, так как при небольшом объеме вычислений он обладает точностью метода Ο4(h).

Для построения разностной схемы интегрирования воспользуемся разложением функции

[ Формула 01 ]

в ряд Тейлора:

[ Формула 02 ]

Заменим вторую производную в этом разложении выражением

[ Формула 03 ]

где

[ Формула 04 ]

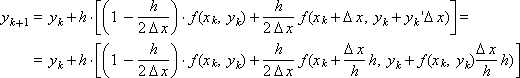
Причем Δx подбирается из условия достижения наибольшей точности записанного выражения. Для дальнейших выкладок произведем замену величины «y с тильдой» разложением в ряд Тейлора:

[ Формула 05 ]

Для исходного уравнения (1) построим вычислительную схему:

[ Формула 06 ]

которую преобразуем к виду:



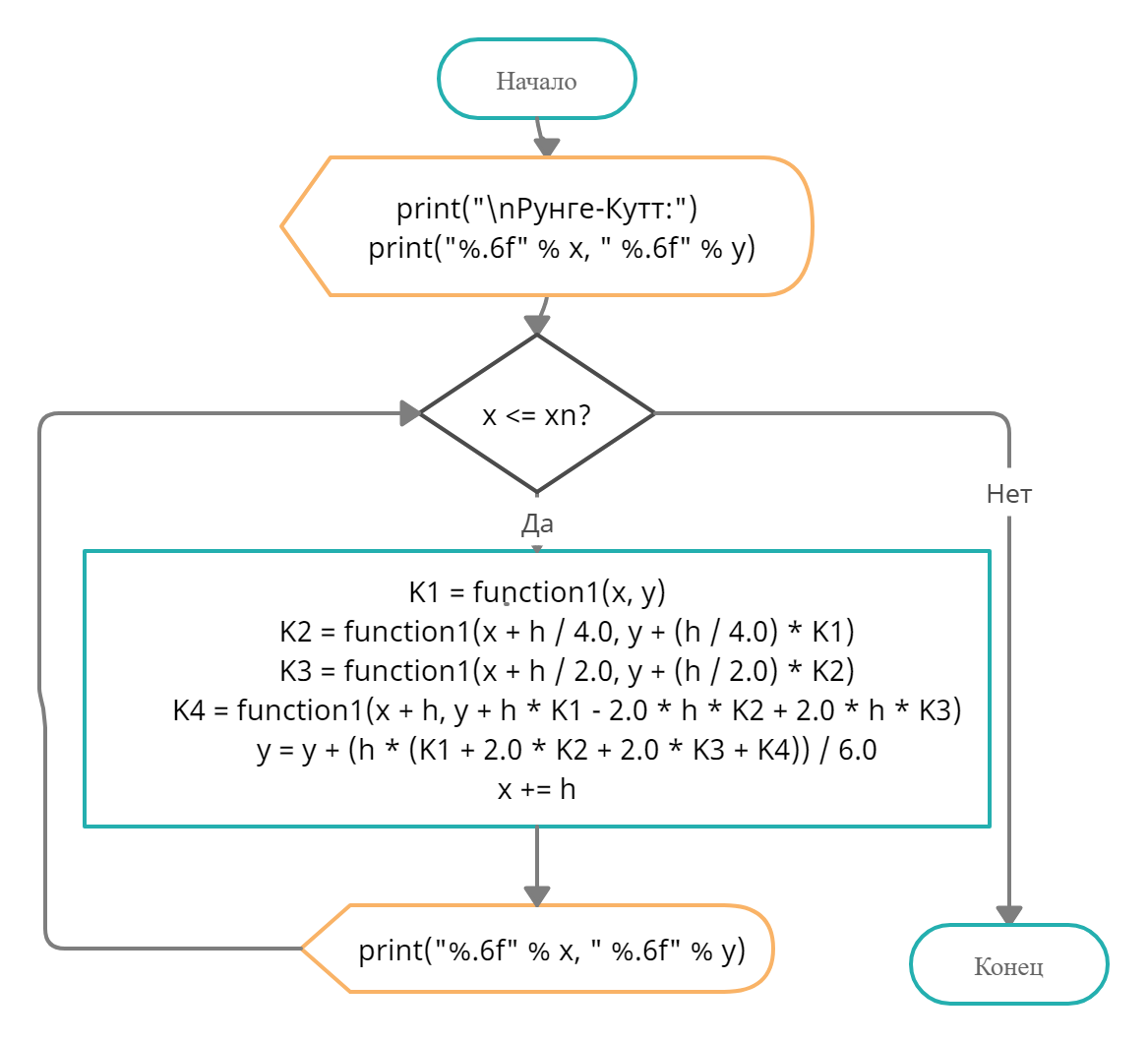
Введем следующие обозначения:

[ Формула 08 ]

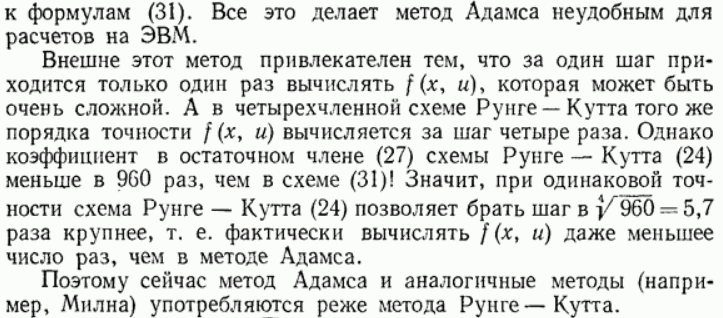
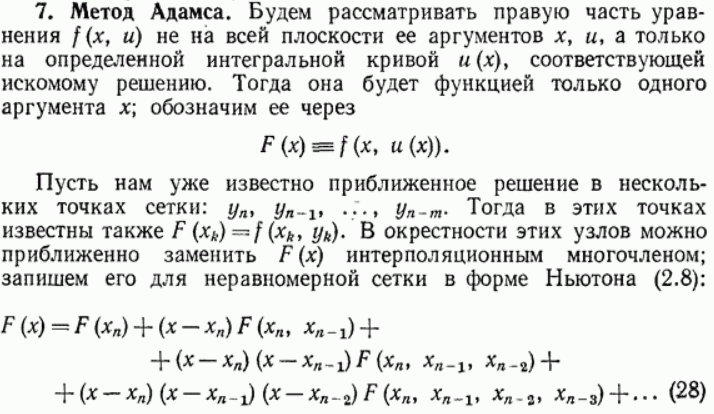
Эти обозначения позволяют записать предыдущее выражение в форме:

[ Формула 09 ]

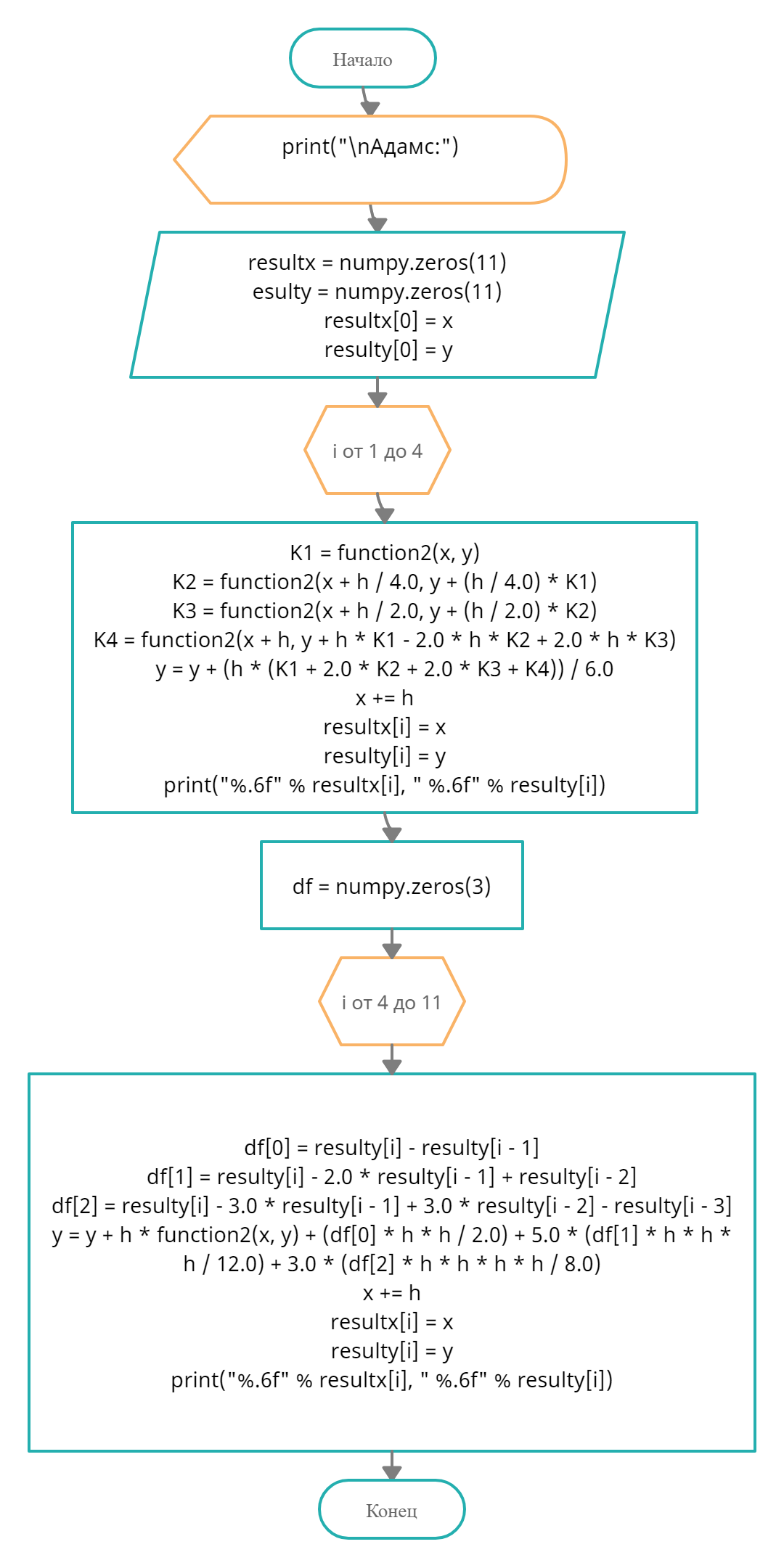
**Рунге-Кутт**

****

**Метод Адамса**



**Адамс**

****

# Расчетные данные

**Задание 1**

Метод Эйлера

|  |  |
| --- | --- |
| X | Y |
| 0.500000 | 0.600000 |
| 0.600000 | 0.747440 |
| 0.700000 | 0.903476 |
| 0.800000 | 1.067702 |
| 0.900000 | 1.239669 |
| 1.000000 | 1.418891 |
| 1.100000 | 1.604852 |
| 1.200000 | 1.797012 |
| 1.300000 | 1.994820 |
| 1.400000 | 2.197717 |
| 1.500000 | 2.405156 |

Метод Эйлера с пересчетом

|  |  |
| --- | --- |
| X | Y |
| 0.500000 | 0.600000 |
| 0.600000 | 0.746738 |
| 0.700000 | 0.901877 |
| 0.800000 | 1.064995 |
| 0.900000 | 1.235633 |
| 1.000000 | 1.413297 |
| 1.100000 | 1.597471 |
| 1.200000 | 1.787619 |
| 1.300000 | 1.983202 |
| 1.400000 | 2.183677 |
| 1.500000 | 2.388520 |

Метод Рунге-Кутта

|  |  |
| --- | --- |
| X | Y |
| 0.500000 | 0.600000 |
| 0.600000 | 0.751032 |
| 0.700000 | 0.910451 |
| 0.800000 | 1.077820 |
| 0.900000 | 1.252664 |
| 1.000000 | 1.434474 |
| 1.100000 | 1.622717 |
| 1.200000 | 1.816845 |
| 1.300000 | 2.016302 |
| 1.400000 | 2.220536 |
| 1.500000 | 2.429010 |

**Задание 2**

Метод Адамса

|  |  |
| --- | --- |
| X | Y |
| 0.100000 | 0.102123 |
| 0.200000 | 0.207621 |
| 0.300000 | 0.313531 |
| 0.400000 | 0.416727 |
| 0.500000 | 0.515522 |
| 0.600000 | 0.608032 |
| 0.700000 | 0.693177 |
| 0.800000 | 0.770682 |
| 0.900000 | 0.840974 |
| 1.000000 | 0.905013 |

Метод Эйлера с пересчетом

|  |  |
| --- | --- |
| X | Y |
| 0.100000 | 0.102500 |
| 0.200000 | 0.208632 |
| 0.300000 | 0.315352 |
| 0.400000 | 0.419767 |
| 0.500000 | 0.519484 |
| 0.600000 | 0.612846 |
| 0.700000 | 0.699013 |
| 0.800000 | 0.777913 |
| 0.900000 | 0.850111 |
| 1.000000 | 0.916652 |

# Листинг разработанной программы

**Main.py**

from solutionMethods import \*

print("\nЗадание №1")

eulerMethod(x = 0.5, xn = 1.5, y = 0.6, h = 0.1)

eulerMethodWithRecalculation1(x = 0.5, xn = 1.5, y = 0.6, h = 0.1)

rungeKuttMethod(x = 0.5, xn = 1.5, y = 0.6, h = 0.1)

print("\nЗадание №2")

adamsMethod(x = 0.0, xn = 1.0, y = 0.0, h = 0.1)

eulerMethodWithRecalculation2(x = 0.0, xn = 1.0, y = 0.0, h = 0.1)

**solutionMethods.py**

import math

import numpy

def function1(x, y) -> float: # Функция для первого задания

    return x + math.cos(y / math.sqrt(7))

def function2(x, y) -> float: # Функция для второго задания

    return (1 - y\*\*2) \* math.cos(x) + 0.6 \* y

def eulerMethod(x, xn, y, h):

    print("\nЭйлер:")

    print("%.6f" % x, " %.6f" % y) # Показываем первые значения

    while x <= xn: # Вычисляем все остальные точки по формуле

        y = y + h \* function1(x, y)

        x += h

        print("%.6f" % x, " %.6f" % y)

def eulerMethodWithRecalculation1(x, xn, y, h):

    print("\nЭйлер c пересчётом:")

    print("%.6f" % x, " %.6f" % y) # Показываем первые значения

    while x <= xn: # Вычисляем все остальные точки по формуле

        Y = y + h \* function1(x, y)

        y = y + 0.5 \* h \* (function1(x, y) + function1(x, Y))

        x += h

        print("%.6f" % x, " %.6f" % y)

def rungeKuttMethod(x, xn, y, h):

    print("\nРунге-Кутт:")

    print("%.6f" % x, " %.6f" % y) # Показываем первые значения

    while x <= xn: # Вычисляем все остальные точки по формуле

        K1 = function1(x, y)

        K2 = function1(x + h / 4.0, y + (h / 4.0) \* K1)

        K3 = function1(x + h / 2.0, y + (h / 2.0) \* K2)

        K4 = function1(x + h, y + h \* K1 - 2.0 \* h \* K2 + 2.0 \* h \* K3)

        y = y + (h \* (K1 + 2.0 \* K2 + 2.0 \* K3 + K4)) / 6.0

        x += h

        print("%.6f" % x, " %.6f" % y)

def adamsMethod(x, xn, y, h):

    print("\nАдамс:")

    resultx = numpy.zeros(11) # Здесь мы храним результаты

    resulty = numpy.zeros(11)

    resultx[0] = x # Добавляем в массив начальные значения

    resulty[0] = y

    for i in range(1, 4): # Считаем начальный отрезок методом Рунге-Кутта

        K1 = function2(x, y)

        K2 = function2(x + h / 4.0, y + (h / 4.0) \* K1)

        K3 = function2(x + h / 2.0, y + (h / 2.0) \* K2)

        K4 = function2(x + h, y + h \* K1 - 2.0 \* h \* K2 + 2.0 \* h \* K3)

        y = y + (h \* (K1 + 2.0 \* K2 + 2.0 \* K3 + K4)) / 6.0

        x += h

        resultx[i] = x

        resulty[i] = y

        print("%.6f" % resultx[i], " %.6f" % resulty[i])

    df = numpy.zeros(3) # Считаем все остальные значения при помощи метода Адамса

    for i in range(4, 11):

        df[0] = resulty[i] - resulty[i - 1]

        df[1] = resulty[i] - 2.0 \* resulty[i - 1] + resulty[i - 2]

        df[2] = resulty[i] - 3.0 \* resulty[i - 1] + 3.0 \* resulty[i - 2] - resulty[i - 3]

        y = y + h \* function2(x, y) + (df[0] \* h \* h / 2.0) + 5.0 \* (df[1] \* h \* h \* h / 12.0) + 3.0 \* (df[2] \* h \* h \* h \* h / 8.0)

        x += h

        resultx[i] = x

        resulty[i] = y

        print("%.6f" % resultx[i], " %.6f" % resulty[i])

def eulerMethodWithRecalculation2(x, xn, y, h):

    print("\nЭйлер c пересчётом:")

    while x < xn-h: # Вычисляем точки по формуле

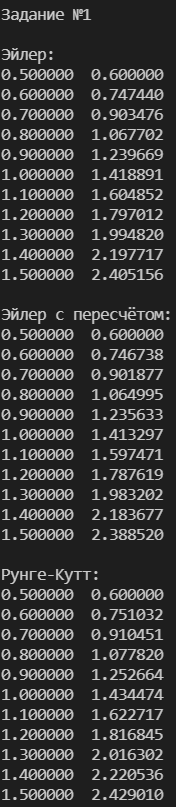
        Y = y + h \* function2(x, y)

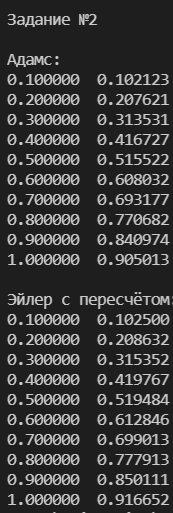
        y = y + 0.5 \* h \* (function2(x, y) + function2(x, Y))

        x += h

        print("%.6f" % x, " %.6f" % y)

# Результаты работы программы





# Вывод

В ходе данной работы были закреплены знания и умения по вычислению приближенных значений интеграла дифференциального уравнения при помощи методом Эйлера, Эйлера с пересчетом, Рунге-Кутты и Адамса.