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

**Численное интегрирование функций**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к лабораторной работе №4

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](#_Toc68972440)

[Постановка задачи 4](#_Toc68972441)

[Теоретические сведения 5](#_Toc68972442)

[Метод средних(центральных) прямоугольников 5](#_Toc68972443)

[Метод трапеций 7](#_Toc68972445)

[Метод Симпсона 9](#_Toc68972446)

[Расчетные данные 12](#_Toc68972447)

[Листинг разработанной программы 13](#_Toc68972448)

[Результаты работы программы 15](#_Toc68972449)

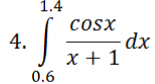
[Вывод 16](#_Toc68972450)

# Цель

Закрепление знаний и умений по численному интегрированию функций.

# Постановка задачи

Вычислить интеграл по формулам центральных (средних) прямоугольников, трапеций и формуле Симпсона, при n=8 и n=20; оценить погрешность результата.



# Теоретические сведения

# Метод средних(центральных) прямоугольников

Пусть функция *y = f(x)* непрерывна на отрезке *[a; b]*. Нам требуется вычислить определенный интеграл формула.

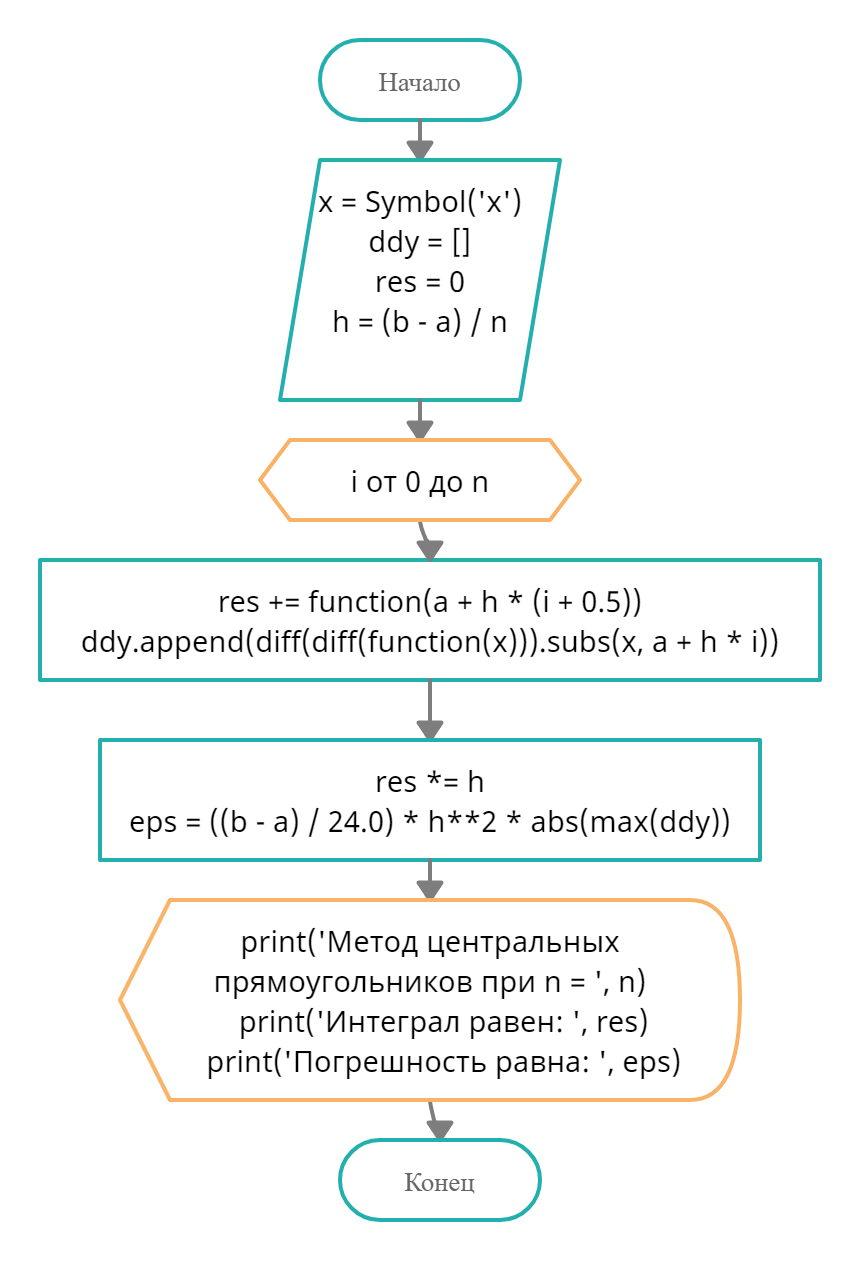
Обратимся к [понятию определенного интеграла](http://www.cleverstudents.ru/integral/definite_integral_definition.html). Разобьем отрезок *[a;b]* на *n* частей формула точками формула. Внутри каждого отрезка формула выберем точку формула. Так как по определению определенный интеграл есть предел интегральных сумм при бесконечном уменьшении длины элементарного отрезка разбиения формула, то любая из интегральных сумм является приближенным значением интеграла формула.

**Суть метода прямоугольников** заключается в том, что в качестве приближенного значения определенного интеграла берут интегральную сумму

### Формула метода средних прямоугольников.

Если отрезок интегрирования *[a;b]* разбить на РАВНЫЕ части длины *h* точками формула (то есть формула) и в качестве точек формула выбрать СЕРЕДИНЫ элементарных отрезков формула (то есть формула), то приближенное равенство формула можно записать в виде формула. Это и есть **формула метода прямоугольников**. Ее еще называют формулой средних прямоугольников из-за способа выбора точек формула.

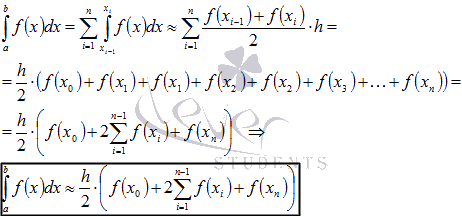
формула называют **шагом разбиения отрезка** *[a;b]*.

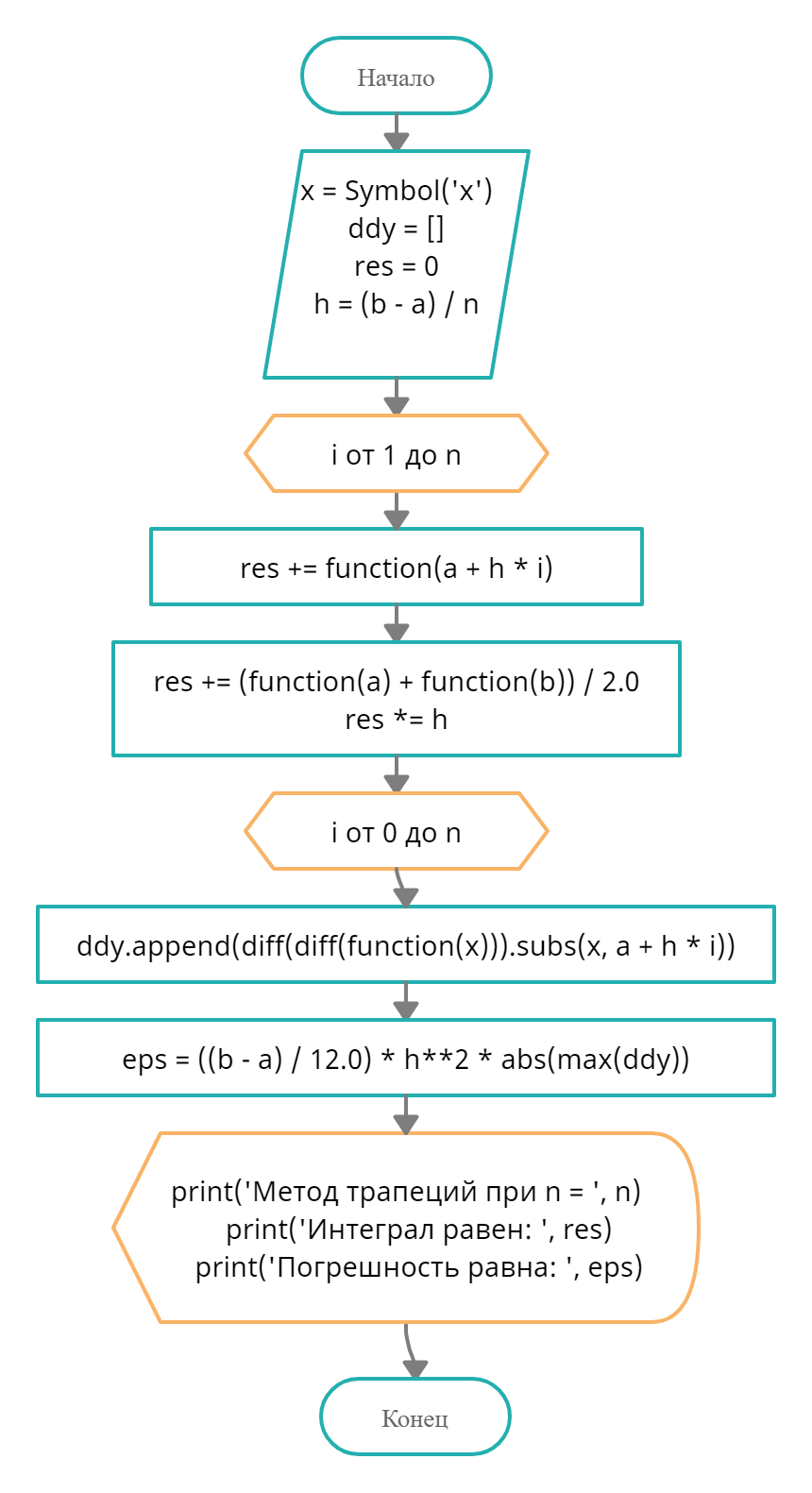


# Метод трапеций

Метод трапеций состоит в представлении определенного формула интеграла в виде суммы интегралов вида формула на каждом элементарном отрезке и в последующей приближенной заменеформула.

В силу пятого [свойства определенного интеграла](http://www.cleverstudents.ru/integral/definite_integral_properties.html) формула.

Если вместо интегралов формула подставить их приближенные значения, то получится **формула метода трапеций**:  




# Метод Симпсона

Этот метод более точный по сравнению с методами прямоугольников и трапеций.

В основе формулы Симпсона лежит квадратичная интерполяция подынтегральной функции на отрезке *[а,* 6] по трем равноотстоящим узлам.

Разобьем интервал интегрирования *[а, b]* на четное число *п* равных отрезков с шагом *h.*

Примем: https://studme.org/htm/img/15/2505/335.png

Значения функций в точках обозначим соответственно:

https://studme.org/htm/img/15/2505/336.png

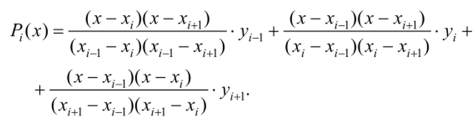
На каждом отрезке [х0, х2], [х2, х4], ..., [x,\_i, х,+ |] подынтегральную функцию Дх) заменим интерполяционным многочленом второй степени:

https://studme.org/htm/img/15/2505/337.png

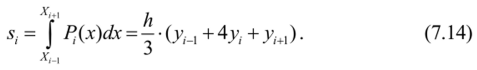
где https://studme.org/htm/img/15/2505/338.png

В качестве Р,(х) можно принять интерполяционный многочлен Лагранжа второй степени, проходящий через концы каждых трех ординат: https://studme.org/htm/img/15/2505/339.png

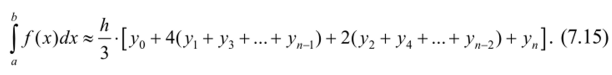
Формула Лагранжа для интервала [x,\_i, ,t,+i] имеет вид:

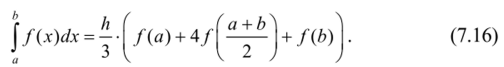


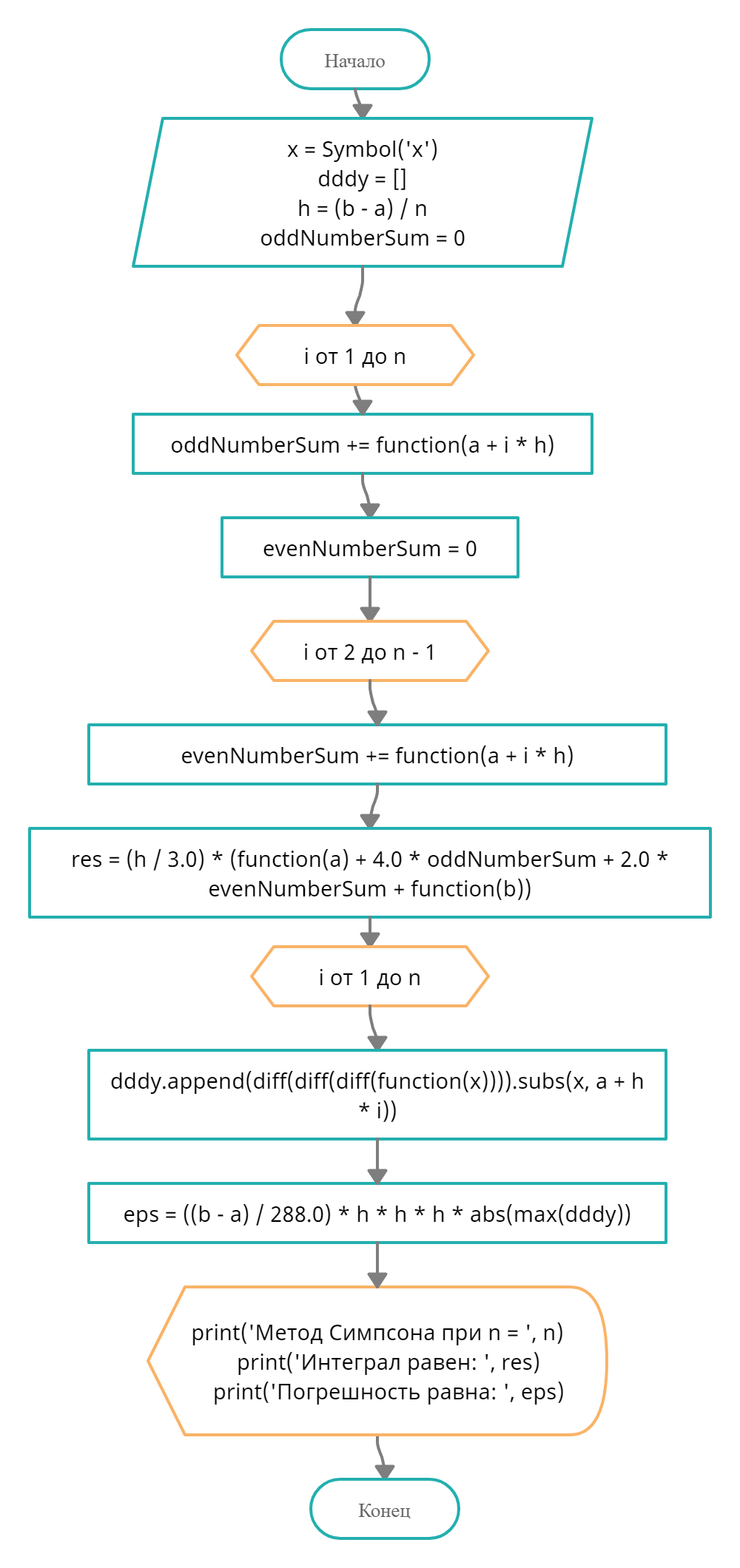
Элементарная площадь s, (см. рис. 7.7) может быть вычислена с помощью определенного интеграла. Учитывая, что х, - х,\_i=x,+i - х, = Л, проведем вычисления и получим для каждого элементарного участка:



После суммирования интегралов по всем отрезкам, получим составную формулу Симпсона:

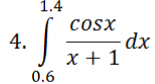


Часто пользуются простой формулой Симпсона 



# Расчетные данные

Исходная функция:



**n = 8**

|  |  |  |
| --- | --- | --- |
|  | Значение интеграла | Значение погрешности |
| Метод средних прямоугольников | 0.222167897115371 | 0.000109429359579646 |
| Метод трапеций | 0.222461800899640 | 0.000218858719159292 |
| Метод Симпсона | 0.222266020718306 | 1.05885814696600e-7 |

**n = 20**

|  |  |  |
| --- | --- | --- |
|  | Значение интеграла | Значение погрешности |
| Метод средних прямоугольников | 0.222250176423056 | 1.75086975327434e-5 |
| Метод трапеций | 0.222297211608253 | 3.50173950654868e-5 |
| Метод Симпсона | 0.222265858852776 | 7.14903395037209e-9 |

# Листинг разработанной программы

**Main.py**

from solutionMethods import \*

a = 0.6 # Наши границы интегрирования

b = 1.4

n1 = 8 # Наши границы разбиения

n2 = 20

centerRectangleMethod(a, b, n1)

trapeziumMethod(a, b, n1)

simpsonMethod(a, b, n1)

centerRectangleMethod(a, b, n2)

trapeziumMethod(a, b, n2)

simpsonMethod(a, b, n2)

**solutionMethod.py**

import math

from sympy import \*

def function(x): # Наша исходная функция

    return cos(x) / (x + 1)

def centerRectangleMethod(a, b, n):

    x = Symbol('x')

    ddy = [] # Здесь мы храним значения второй производной в точках от a до b

    res = 0 # Переменная, в которой будет храниться наш результат

    h = (b - a) / n  # Шаг h

    for i in range(n): # Считаем значения второй производной и сумму значений функции в промежутке

        res += function(a + h \* (i + 0.5)) # Начиная с точки a, прибавляем в результату значение y(xi - 0.5), которое вычисляется как h \* (i + 0.5)

        ddy.append(diff(diff(function(x))).subs(x, a + h \* i)) # Заносим наши значения второй производной от a до b с шагом h в список

    res \*= h # Результат домножаем на h, так как шаг является раномерным

    eps = ((b - a) / 24.0) \* h\*\*2 \* abs(max(ddy)) # Считаем погрешность по формуле Rn = (b - a) / 24 \* h^2 \* max(y''(x))

    print('Метод центральных прямоугольников при n = ', n)

    print('Интеграл равен: ', res)

    print('Погрешность равна: ', eps)

def trapeziumMethod(a, b, n):

    x = Symbol('x')

    ddy = [] # Здесь мы храним значения второй производной в точках от a до b

    res = 0 # Переменная, в которой будет храниться наш результат

    h = (b - a) / n  # Шаг h

    for i in range(1, n): # Считаем значения второй производной и сумму значений функции в промежутке

        res += function(a + h \* i) # Начиная с точки a, прибавляем в результату значение y(xi), которое вычисляется как h \* i

    res += (function(a) + function(b)) / 2.0 # К результату добавим (y0 + yn) / 2

    res \*= h # Результат домножаем на h, так как шаг является раномерным

    for i in range(n):

        ddy.append(diff(diff(function(x))).subs(x, a + h \* i)) # Заносим наши значения второй производной от a до b с шагом h в список

    eps = ((b - a) / 12.0) \* h\*\*2 \* abs(max(ddy)) # Считаем погрешность по формуле Rn = (b - a) / 12 \* h^2 \* max(y''(x))

    print('Метод трапеций при n = ', n)

    print('Интеграл равен: ', res)

    print('Погрешность равна: ', eps)

def simpsonMethod(a, b, n):

    x = Symbol('x')

    dddy = [] # Здесь мы храним значения третьей производной в точках от a до b

    h = (b - a) / n  # Шаг h

    oddNumberSum = 0 # Здесь мы храним сумму нечетных значений

    for i in range(1, n, 2): # Считаем сумму y1 + y3 + ... + yn - 1

        oddNumberSum += function(a + i \* h)

    evenNumberSum = 0 # Здесь мы храним сумму четных значений

    for i in range(2, n - 1, 2): # Считаем сумму y2 + y4 + ... + yn - 2

        evenNumberSum += function(a + i \* h)

    res = (h / 3.0) \* (function(a) + 4.0 \* oddNumberSum + 2.0 \* evenNumberSum + function(b)) # Вычисляем интеграл = h / 3(y0 + 4 \* (y1 + y3 + ... + yn-1) + 2 \* (y2 + y4 + ... + yn - 2) + yn)

    for i in range(n):

        dddy.append(diff(diff(diff(function(x)))).subs(x, a + h \* i)) # Заносим наши значения третьей производной от a до b с шагом h в список

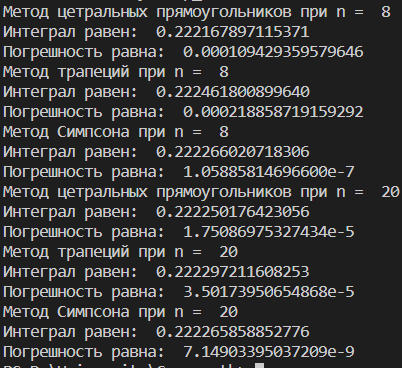
    eps = ((b - a) / 288.0) \* h \* h \* h \* abs(max(dddy)) # Считаем погрешность по формуле Rn = (b - a) / 288 \* h^3 \* max(y'''(x))

    print('Метод Симпсона при n = ', n)

    print('Интеграл равен: ', res)

    print('Погрешность равна: ', eps)

# Результаты работы программы



# Вывод

В ходе данной работы были закреплены знания и умения по вычислению интеграла при помощи методов средних прямоугольников, трапеция и методу Симпсона.

Самым точным является метод Симпсона, а самым не точным метод трапеций. При увеличении кол-во промежутков разбиения n точность результатов возрастает.