**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ»**

**Факультет Программной Инженерии и Компьютерной Техники**

**Дисциплина:**

**«*Вычислительная математика*»**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3  
*«Численное интегрирование»***

***Вариант 3***

**Выполнил:**

Студент гр. P32151 *Горинов Даниил Андреевич*

**Проверил:**

*Машина Екатерина Алексеевна*

Санкт-Петербург

2023г.

Цель лабораторной работы:

Найти приближенное значение определенного интеграла с требуемой точностью различными численными методами.

Порядок выполнения лабораторной работы:

**Программная реализация задачи:**

1. Реализовать в программе методы по выбору пользователя:

* Метод прямоугольников (3 модификации: левые, правые, средние)
* Метод трапеций
* Метод Симпсона

1. Методы должны быть оформлены в виде отдельной(ого) функции/класса.
2. Вычисление значений функции оформить в виде отдельной(ого) функции/класса.
3. Для оценки погрешности и завершения вычислительного процесса использовать правило Рунге.
4. Предусмотреть вывод результатов: значение интеграла, число разбиения интервала интегрирования для достижения требуемой точности.

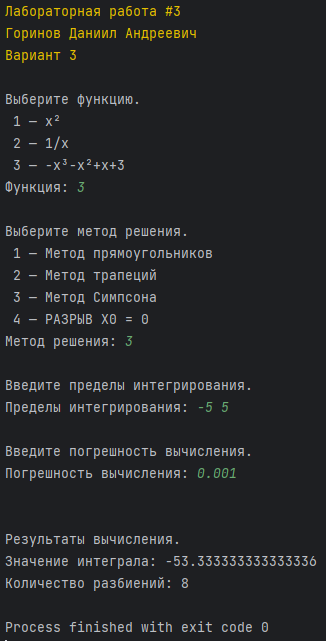
**Вычислительная реализация задачи:**

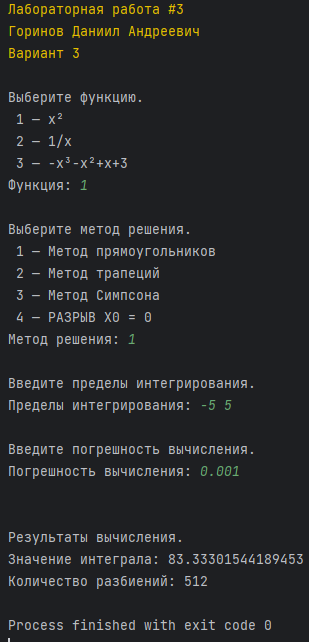
1. Вычислить интеграл, приведенный в таблице 1, точно.
2. Вычислить интеграл по формуле Ньютона – Котеса при n = 5.
3. Вычислить интеграл по формулам средних прямоугольников, трапеций и Симпсона при n = 10.
4. Сравнить результаты с точным значением интеграла.
5. Определить относительную погрешность вычислений для каждого метода.
6. В отчете отразить последовательные вычисления.

Рабочие формулы методов:

Листинг программы:

1. def rectangle\_method(f, a, b, e, min\_n=4, max\_itr=10):
2. n = min\_n
3. result = float('inf')
4. while n <= n \* (2 \*\* max\_itr):
5. last\_result = result
6. result = 0
7. h = (b - a) / n
8. x = a
9. for \_ in range(n):
10. result += f(x + h / 2)
11. x += h
12. result \*= h
13. if abs(result - last\_result) <= e:
14. break
15. else:
16. n \*= 2
17. return {'result': result, 'n': n}
18. def trapezoid\_method(f, a, b, e, min\_n=4, max\_itr=10):
19. n = min\_n
20. result = float('inf')
21. while n <= n \* (2 \*\* max\_itr):
22. last\_result = result
23. result = (f(a) + f(b)) / 2
24. h = (b - a) / n
25. x = a + h
26. for \_ in range(n - 1):
27. result += f(x)
28. x += h
29. result \*= h
30. if abs(result - last\_result) <= e:
31. break
32. else:
33. n \*= 2
34. return {'result': result, 'n': n}
35. def simpson\_method(f, a, b, e, min\_n=4, max\_itr=10):
36. if min\_n % 2 != 0:
37. return None
38. n = min\_n
39. result = float('inf')
40. while n <= n \* (2 \*\* max\_itr):
41. last\_result = result
42. result = f(a) + f(b)
43. h = (b - a) / n
44. x = a + h
45. for i in range(n - 1):
46. coeff = 4 if i % 2 == 0 else 2
47. result += coeff \* f(x)
48. x += h
49. result \*= h / 3
50. if abs(result - last\_result) <= e:
51. break
52. else:
53. n \*= 2
54. return {'result': result, 'n': n}
55. def death\_dot(f, a, b, e, x0):
56. answer1 = rectangle\_method(f, a, x0 - e, e)
57. answer2 = rectangle\_method(f, x0 + e, b, e)
58. data = {
59. 'result': answer1['result'] + answer2['result'],
60. 'n': answer1['n'] + answer2['n']
61. }
62. return data

 Результаты выполнения программы:



Вычисление заданного интеграла:

**Точное вычисление:**

**Вычисление по формуле Ньютона-Котеса при n=5:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **i** | **c** | **x** | **f(x)** | **f(x)\*c** |
| 0 | 0.132 | 0 | 3 | 0.396 |
| 1 | 0.521 | 0.4 | 3.176 | 1.654 |
| 2 | 0.347 | 0.8 | 2.648 | 0.919 |
| 3 | 0.347 | 1.2 | 1.032 | 0.358 |
| 4 | 0.521 | 1.6 | -2.056 | -1.071 |
| 5 | 0.132 | 2 | -7 | -0.924 |

Ответ:

Относительная погрешность: 0%.

**Вычисление по формуле средних прямоугольников:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **i** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| **x** | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 | 1.1 | 1.3 | 1.5 | 1.7 | 1.9 |
| **f(x)** | 3.089 | 3.183 | 3.125 | 2.867 | 2.361 | 1.559 | 0.413 | -1.125 | -3.103 | -5.569 |

Ответ:

Относительная погрешность: 2.7%.

**Вычисление по формуле трапеций:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **i** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| **x** | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 | 1.2 | 1.4 | 1.6 | 1.8 |
| **f(x)** | 3.152 | 3.176 | 3.024 | 2.648 | 2.0 | 1.032 | -0.304 | -2.056 | -4.272 |

Ответ:

Относительная погрешность: 5.33%.

**Вычисление по формуле Симпсона:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **i** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| **x** | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 | 1.2 | 1.4 | 1.6 | 1.8 | 2.0 |
| **f(x)** | 3 | 3.152 | 3.176 | 3.024 | 2.648 | 2.0 | 1.032 | -0.304 | -2.056 | -4.272 | -7 |

Ответ:

Относительная погрешность: 0%.

Вывод:

Вычисление определенного интеграла с использованием численных методов - задача, которая может быть достаточно простой, если необходимо учитывать несобственные интегралы. В основном, это сводится к созданию массива значений функции для заданного значения n и затем суммированию этих значений с использованием заранее известной формулы. Однако, при работе с определенными интегралами требуется учитывать критические точки и проверять сходимость интеграла в этих точках как справа, так и слева.