**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе № 6**

**по дисциплине «Вычислительная математика»**

**Тема: Численное интегрирование**

| Студент гр. 1303 |  | Чубан Д.В. |
| --- | --- | --- |
| Преподаватель |  | Лисс А.Р. |

Санкт-Петербург

2022

**I. Цель работы:**

Используя квадратурные формулы прямоугольников, трапеций и Симпсона, вычислить значения интеграла



и, применив правило Рунге, найти наименьшее значение (наибольшее значение шага ), при котором каждая из указанных формул дает приближенное значение интеграла с погрешностью , не превышающей заданную.



**II. Общие сведения:**

Повышения точности численного интегрирования добиваются путем применения составных формул. Для этого при нахождении определенного интеграла отрезок разбивают на четное число отрезков длины и на каждом из отрезков длины применяют соответствующую формулу. Таким образом получают составные формулы прямоугольников, трапеций и Симпсона.



На сетке , , , составные формулы имеют следующий вид:



формула прямоугольников

;



формула трапеций

;



формула Симпсона

,



где - остаточные члены. При приближенные значения интегралов для всех трех формул (в предположении отсутствия погрешностей округления) стремятся к точному значению интеграла.



Для практической оценки погрешности квадратурной можно использовать правило Рунге. Для этого проводят вычисления на сетках с шагом и , получают приближенные значения интеграла и и за окончательные значения интеграла принимают величины:



- для формулы прямоугольников;



- для формулы трапеций;



- для формулы Симпсона.



За погрешность приближенного значения интеграла для формул прямоугольников и трапеций тогда принимают величину , а для формулы Симпсона .



**III. Порядок выполнения работы:**

1. Составить программы-функции для вычисления интегралов по формулам прямоугольников, трапеций и Симпсона.
2. Составить программу-функцию для вычисления подынтегральной функции.
3. Составить головную программу, содержащую оценку по Рунге погрешности каждой из перечисленных выше квадратурных формул, удваивающих до тех пор, пока погрешность не станет меньше , и осуществляющих печать результатов: значения интеграла и значения для каждой формулы.



1. Провести вычисления по программе, добиваясь, чтобы результат удовлетворял требуемой точности.
2. Результаты работы оформить в виде краткого отчета, содержащего сравнительную оценку применяемых для вычисления формул.

**IV. Выполнение работы:**

1)Составим программы-функции для вычисления интегралов по формулам прямоугольников, трапеций и Симпсона.

def rectangle(n) -> float:  
 h = 1/n  
 summ = 0  
  
 for i in range(n):  
 x = i\*h  
 summ+=func(x+(h/2))  
 return h\*summ  
  
  
def trapeziod(n) -> float:  
 summ = 0  
 h = 1/n  
  
 for i in range(n):  
 x = i\*h  
 x\_1 = (i+1)\*h  
 summ += (func(x) + func(x\_1))  
 return h\*summ/2  
  
def simpson(n) -> float:  
 summ = 0  
 h = 1/n  
 m = n >> 1  
 for i in range(m):  
 x = 2\*i\*h  
 x\_1 = (2\*i+1)\*h  
 x\_2 = (2\*i+2)\*h  
 summ += (func(x) + 4\*func(x\_1) + func(x\_2))  
 return h\*summ/3

2) Составим программу-функцию для вычисления подынтегральной функции.

def func(x):

return np.exp(np.sin(x))

3) Составим головную программу, содержащую оценку по Рунге погрешности каждой из перечисленных выше квадратурных формул, удваивающих до тех пор, пока погрешность не станет меньше , и осуществляющих печать результатов: значения интеграла и значения для каждой формулы.



def main():

eps = [0.000001, 0.0001, 0.01]

print("I = exp(sin x), [0,1]")

print()

for elem in eps:

n = [2, 2, 2]

cur\_1 = abs(rectangle(n[0]<<1) - rectangle(n[0]))/3

while cur\_1 >= elem:

n[0]<<=1

cur\_1 = abs(rectangle(n[0] << 1) - rectangle(n[0])) / 3

Ir = rectangle(n[0]<<1) - cur\_1

cur\_2 = abs(trapeziod(n[1] << 1) - trapeziod(n[1])) / 3

while cur\_2 >= elem:

n[1] <<= 1

cur\_2 = abs(trapeziod(n[1] << 1) - trapeziod(n[1])) / 3

It = trapeziod(n[1] << 1) - cur\_2

cur\_3 = abs(simpson(n[2]<<1) - simpson(n[2]))/15

while cur\_3 >= elem:

n[2] <<= 1

cur\_3 = abs(simpson(n[2]<<1) - simpson(n[2]))/15

Is = simpson(n[2]<<1) - cur\_3

print(f"Ir = {Ir}, n = {n[0]}, eps = {elem}")

print(f"It = {It}, n = {n[1]}, eps = {elem}")

print(f"Is = {Is}, n = {n[2]}, eps = {elem}")

print()

4) Проведем вычисления по программе, добиваясь, чтобы результат удовлетворял требуемой точности.

| **Eps** | **Integral** | **n** |
| --- | --- | --- |
| 0,000001 | 1.6318683195467638 | 64 |
| 1.6318696084128104 | 128 |
| 1.6318689149266108 | 8 |
| 0,0001 | 1.6317866755792110 | 8 |
| 1.6318692637090697 | 8 |
| 1.6318581811273714 | 4 |
| 0,01 | 1.6304287606971740 | 2 |
| 1.6317762692999538 | 2 |
| 1.6316618736853366 | 2 |

Зависимость количества отрезков n от требуемой точности:

5) Из полученных результатов видно, что, чем более высокая точность результата нам необходима, тем больше нам необходимо сделать итераций. Кроме того, из таблицы видно, что меньше всего итераций для достижения требуемой точности необходимо при вычислении интеграла по формуле Симпсона, на 2-м месте формула прямоугольников и больше всего итераций требуется для формулы трапеций.

**V. Вывод:**

Проанализировав результаты работы программы, мы можем сделать вывод, что число итераций, необходимых для вычисления интеграла по составным формулам прямоугольника, трапеции и Симпсона возрастает с ростом требуемой точности результата. Исходя из результатов тестирования программы, мы можем сделать вывод, что меньше всего итераций для получения необходимой точности результата требуется при вычислении интеграла по формуле Симпсона, далее в порядке увеличения числа итераций идет формула прямоугольников, а затем формула трапеций.