Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Вычислительная математика

Лабораторная работа №3

Вариант 3

Группа: P3269

Выполнили:

Грибкова В.Е

Долганова О.А

Проверил:

Машина Е. А.

Г. Санкт-Петербург

1. Цель работы: найти приближенное значение определенного интеграла с требуемой точностью различными численными методами.
2. Рабочие формулы используемых методов.

Вводим коэффициенты Котеса: $c_n^i = \int_a^b L_n^i(x)dx$

Формула Ньютона-Котеса порядка n:

$$\int_a^b f(x)dx \approx \int_a^b L_n(x)dx = \sum_{i=0}^{n} f(x_i)c_n^i$$

$c_6^0 = c_6^6 = \frac{41(b-a)}{840}$	$c_6^1 = c_6^5 = \frac{216(b-a)}{840}$	$c_6^2 = c_6^4 = \frac{27(b-a)}{840}$	$c_6^3 = \frac{272(b-a)}{840}$

При $h_i = h = \frac{b-a}{n} = const$ формула трапеций:

$$\int_a^b f(x)dx = h \cdot \left(\frac{y_0 + y_n}{2} + \sum_{i=1}^{n-1} y_i\right)$$

При $h_i = h = \frac{b-a}{n} = const$:

$$\int_a^b f(x)dx = h\sum_{i=1}^{n} f(x_{i-1/2})$$

Формула Симпсона

$$\int_a^b f(x) = \frac{h}{3}\left[(y_0 + 4(y_1 + y_3 + \dots + y_{n-1}) + 2(y_2 + y_4 + \dots + y_{n-2}) + y_n)\right]$$

3. Вычисление заданного интеграла

$$\int_0^2 (-x^3 - x^2 + x + 3)dx$$

1. Вычислим интеграл точно.

$$\int_0^2 (-x^3 - x^2 + x + 3)dx$$

$$= -\frac{x(3x^3 + 4x^2 - 6x - 36)}{12} + C = \frac{4}{3} = 1.3333$$

2. Вычислить интеграл по формуле **Ньютона – Котеса** при $n = 6$.

$$\int_0^2 (-x^3 - x^2 + x + 3)dx$$

h=(b-a)n=⅓

i	0	1	2	3	4	5	6
xi	0	1/3	2/3	1	4/3	5/3	2
yi	3	3,185	2,926	2	0,185	-2,741	-7
c_6^i	0,098	0,514	0,064	0,648	0,064	0,514	0,098

$I = \sum_{i=0}^{n} c_n^i * f(x_i)$=1.331

1,333-1,331=0,002

Погрешность вычислений - 0.15%

3. Вычислить интеграл по формулам средних прямоугольников, трапеций и Симпсона при n = 10 .

Формула **средних прямоугольников**:

h=(b-a)/n=0.2

i	0	1	2	3	4	5	6	7	8	9	10
xi	0	0.2	0,4	0,6	0,8	1,0	1,2	1,4	1,6	1,8	2
$(x_i + x_{i-1})/2$		0.1	0.3	0.5	0.7	0.9	1.1	1.3	1.5	1.7	1.9
$f((x_i + x_{i-1})/2)$		3.09	3.08	3.13	2.87	2.36	1.56	0.41	-1.13	-3.1	-5.57

$I = h * \sum_{i=1}^{n} f(\frac{((x_i + x_{i-1})}{2}) = 1.360$

1,333-1,360=0,027

Погрешность вычислений - 2,25%

Формула **трапеций**:

h=(b-a)/n=0.2

i	0	1	2	3	4	5	6	7	8	9	10
xi	0	0.2	0,4	0,6	0,8	1,0	1,2	1,4	1,6	1,8	2
yi	3	3.15	3.18	3.02	2.65	2	1.03	-0.3	-2.06	-4.27	-7

$I = h * (\frac{y0+yn}{2} + \sum_{i=1}^{n-1} yi) = 1.280$

1.333-1.280=0.053

Погрешность вычислений - 3,75%

Формула **Симпсона**:

h=(b-a)/n=0.2

i	0	1	2	3	4	5	6	7	8	9	10

xi	0	0.2	0,4	0,6	0,8	1,0	1,2	1,4	1,6	1,8	2
yi	3	3.15	3.18	3.02	2.65	2	1.03	-0.3	-2.06	-4.27	-7

$I = h/3 * (y0 + 4 * (y1 + y3 + ... + yn - 1) + 2 * (y2 + y4 + ... + yn - 2) + yn = 1.333$

1.333-1.280=0.053

Погрешность вычислений - 0%

4. Листинг программы.

```
import math

class IntegralOfFunction:
    def __init__(self):
        # Инициализация параметров и выбор метода ввода данных
        self.func_num = None
        self.method_num = None
        self.a = None
        self.b = None
        self.epsilon = None
        self.nmin = None
        self.k = None

        # Запрос способа ввода данных
        choice = self.input_method()
        if choice == '1':
            self.get_data_from_file()
        else:
            self.get_data_input()

    def func1(self, x):
        # Первая функция: 5x³ - 2x² + x - 14
        return 5 * x**3 - 2 * x**2 + x - 14

    def func2(self, x):
        # Вторая функция: 0.3x⁴ + 0.7x² - 0.4
        return 0.3 * x**4 + 0.7 * x**2 - 0.4

    def func3(self, x):
        # Третья функция: x³
        return x**3

    def func4(self, x):
        # Четвертая функция: 3x² + 8x - 7
        return 3 * x**2 + 8 * x - 7

    def get_function(self):
        # Запрос выбора функции у пользователя
        print("Выберите функцию")
        print(" 1) 5x³ - 2x² + x - 14")
        print(" 2) 0.3x⁴ + 0.7x² - 0.4")
```

```
        print(" 3) x³")
        print(" 4) 3x² + 8x -7")
        print("Выберите функцию из списка")
        self.func_num = int(input())
        while self.func_num < 1 or self.func_num > 4:
            print("Выберите функцию из списка")
            self.func_num = int(input())

    def get_method(self):
        # Запрос выбора метода численного интегрирования у 
пользователя
        print("Выберите метод")
        print(" 1) Метод левых прямоугольников")
        print(" 2) Метод средних прямоугольников")
        print(" 3) Метод правых прямоугольников")
        print(" 4) Метод трапеций")
        print(" 5) Метод Симпсона")
        print("Выберите метод из списка")
        self.method_num = int(input())
        while self.method_num < 1 or self.method_num > 5:
            print("Выберите метод из списка")
            self.method_num = int(input())

    def get_data_input(self):
        # Получение входных данных от пользователя
        self.get_function()
        self.get_method()
        print("Введите пределы интегрирования:")
        self.a = float(input())
        self.b = float(input())
        print("Введите точность вычисления: ")
        self.epsilon = float(input())
        print("Введите начальное значение числа разбиения:")
        self.nmin = int(input())

    def get_data_from_file(self):
        # Чтение входных данных из файла
        try:
            with open('data.txt', 'r') as file:
                data = file.read().split()
                self.func_num = int(data[0])
                self.method_num = int(data[1])
                self.a = float(data[2])
                self.b = float(data[3])
                self.epsilon = float(data[4])
                self.nmin = int(data[5])
        except FileNotFoundError:
            raise RuntimeError("Файл не доступен")
```

```
    def input_method(self):
        # Запрос способа ввода данных: из файла или вручную
        print("Взять исходные данные из файла (1) или ввести с клавиатуры (2)?")
        print("Режим ввода: ")
        choice = input().strip()
        while choice != '1' and choice != '2':
            print("Введите (1) или (2) для выбора способа ввода.")
            print("Режим ввода: ")
            choice = input().strip()
        return choice

    def apply(self, x):
        # Применение выбранной функции к x
        if self.func_num == 1:
            return self.func1(x)
        elif self.func_num == 2:
            return self.func2(x)
        elif self.func_num == 3:
            return self.func3(x)
        else:
            return self.func4(x)

    def left_rectangle_method(self, a, b, h):
        # Метод левых прямоугольников
        res = 0
        i = a
        while i < b - h/2:
            res += h * self.apply(i)
            i += h
        return res

    def right_rectangle_method(self, a, b, h):
        # Метод правых прямоугольников
        res = 0
        i = a + h
        while i < b + h/2:
            res += h * self.apply(i)
            i += h
        return res

    def middle_rectangle_method(self, a, b, h):
        # Метод средних прямоугольников
        res = 0
        i = a + h/2
        while i < b:
            res += h * self.apply(i)
```

```
            i += h
        return res

    def trapezoid_method(self, a, b, h):
        # Метод трапеций
        sum_val = 0
        i = a + h
        while i < b - h/2:
            sum_val += self.apply(i)
            i += h
        return h * ((self.apply(a) + self.apply(b)) / 2 + sum_val)

    def simpson_method(self, a, b, h):
        # Метод Симпсона
        sum1 = 0
        sum2 = 0
        i = a + h
        while i < b - h/2:
            sum1 += self.apply(i)
            i += 2 * h
        i = a + 2 * h
        while i < b - h/2:
            sum2 += self.apply(i)
            i += 2 * h
        return h / 3 * (self.apply(a) + 4 * sum1 + 2 * sum2 +
self.apply(b))

    def sup_method(self, a, b, h):
        # Выбор метода численного интегрирования в зависимости от
выбора пользователя
        if self.method_num == 1:
            self.k = 2
            return self.left_rectangle_method(a, b, h)
        elif self.method_num == 2:
            self.k = 2
            return self.right_rectangle_method(a, b, h)
        elif self.method_num == 3:
            self.k = 2
            return self.middle_rectangle_method(a, b, h)
        elif self.method_num == 4:
            self.k = 2
            return self.trapezoid_method(a, b, h)
        else:
            self.k = 4
            return self.simpson_method(a, b, h)

    def runge_rule(self, res, res2):
        # Правило Рунге для оценки погрешности
```

```
        return abs(res2 - res) / (2**self.k - 1)

    def sub_result(self, a, b):
        # Подвычисления для интеграла
        result = {'res': None, 'res2': None, 'n': self.nmin}
        while True:
            h = abs(b - a) / result['n']
            result['res'] = self.sup_method(a, b, h)
            result['n'] *= 2
            result['res2'] = self.sup_method(a, b, h)
            if abs(result['res2'] - result['res']) <= self.epsilon:
                break
        return result

    def result_integral(self):
        # Вычисление и вывод результата интегрирования
        result = self.sub_result(self.a, self.b)
        print(f"Значение интеграла: {result['res']}")
        print(f"Число разбиения: {result['n'] // 2}")
        print(f"Погрешность: {self.runge_rule(result['res'],
result['res2'])}")

def main():
    # Основная функция, запрашивающая повторное использование
других методов
    integral = IntegralOfFunction()
    integral.result_integral()
    while True:
        print("Попробуйте с другими методами (+/-) ?")
        print("Режим ввода: ")
        choice = input().strip()
        while choice != '1' and choice != '2':
            print("Введите (1) или (2) для выбора способа ввода.")
            print("Режим ввода: ")
            choice = input().strip()
        if choice == '1':
            integral.get_method()
            integral.result_integral()
        else:
            break

if __name__ == "__main__":
    main()
```

5. Результаты выполнения программы

```
Взять исходные данные из файла (1) или ввести с клавиатуры (2)?
Режим ввода:
2
Выберите функцию
1) 5x³ - 2x² + x - 14
2) 0.3x⁴ + 0.7x² - 0.4
3) x³
4) 3x² + 8x -7
Выберите функцию из списка
3
Выберите метод
1) Метод левых прямоугольников
2) Метод средних прямоугольников
3) Метод правых прямоугольников
4) Метод трапеций
5) Метод Симпсона
Выберите метод из списка
5
Введите пределы интегрирования:
1
7
Введите точность вычисления:
3
Введите начальное значение числа разбиения:
1
Значение интеграла: 688.0
Число разбиения: 1
Погрешность: 0.0
Попробуйте с другими методами (+/-) ?
```

6. Вывод: В ходе лабораторной работы изучили численные методы решения определённых интегралов: метод Ньютона-Котеса(метод прямоугольников), метод трапеций, метод Симпсона. Нашли приближенное значение определенного интеграла с требуемой точностью всеми этими численными методами.