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

Факультет программной инженерии и компьютерной техники
Направление подготовки: 09.03.04 — Системное и прикладное программное обеспечение
Дисциплина «Вычислительная математика»

Лабораторная работа №3

Вариант 6

Выполнил:

Капарулин Тимофей Иванович

Преподаватель:

Машина Екатерина Алексеевна

г.Санкт-Петербург 2025 г.

Цель работы

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

Вычисление заданного интеграла

$$\int_1^2 (3x^3 + 5x^2 + 3x - 6)dx$$

1. Точное решение

$$\int_1^2 \left(3x^3 + 5x^2 + 3x - 6\right)dx = \left(0.75x^4 + \frac{5}{3x^3} + 1.5x^2 - 6x\right)_1^2 = \frac{257}{12} \approx 21.416667$$

2. Формула Ньютона-Котеса при n=6

$$h = \frac{2-1}{6} \approx 0.16667$$

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**
x_i	1	1.16667	1.33333	1.5	1.66667	1.83333	2
y_i	5	9.06953	13.99989	19.875	26.77793	34.79149	44

n	C_n	c_n^0	c_n^1	c_n^2	c_n^3	c_n^4	c_n^5	c_n^6
6	840	41	216	27	272	27	216	41

$$\int_1^2 \left(3x^3 + 5x^2 + 3x - 6\right)dx = \frac{n * h}{C_n} * \sum_{i=1}^{n} c_n^i * f(x_i) =$$

$$= \frac{6 * 0.16667}{840} * (41 * 5 + 216 * 9.06953 + 27 * 13.99989 + 272 * 19.875 + 27 * 26.77793 + 216 * 34.79149 + 41 * 44)$$

$$= \frac{1}{840} * 17989.98146 = 21.41665$$

3. Формула средних прямоугольников при n=10

	0	1	2	3	4	5	6	7	8	9	10
x_i	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2
$f(\frac{x_{i-1}+x_i}{2})$	-	6.13538	8.62512	11.42188	14.54363	18.00838	21.83413	26.03887	30.64062	35.65738	41.10713

$$\int_1^2 \left(3x^3 + 5x^2 + 3x - 6\right)dx = \frac{(b-a)}{n} * \sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right) =$$

$$\frac{1}{10} * (6.13538 + 8.62512 + 11.42188 + 14.54363 + 18.00838 + 21.83413 + 26.03887 + 30.64062 + 35.65738 + 41.10713) = 21.40125$$

3. Формула трапеций при n=10

	0	1	2	3	4	5	6	7	8	9	10
x_i	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2
$f(x_i)$	5	7.343	9.984	12.941	16.232	19.875	23.888	28.289	33.096	38.327	44

$$\int_1^2 \left(3x^3 + 5x^2 + 3x - 6\right)dx = \frac{(b-a)}{n} * \left(\frac{f(x_0) + f(x_n)}{2} + \sum_{i=1}^{n-1} f(x_i)\right) =$$

$$\frac{1}{10} * (24.5 + 189.975) = 21.4475$$

4. Формула Симпсона при n=10

	0	1	2	3	4	5	6	7	8	9	10
x_i	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2
$f(x_i)$	5	7.343	9.984	12.941	16.232	19.875	23.888	28.289	33.096	38.327	44

$$\int_1^2 \left(3x^3 + 5x^2 + 3x - 6\right)dx =$$

$$= \frac{(b-a)}{3n} * \left(f(x_0) + 4 * \sum_{i=0}^{\lfloor n/2 \rfloor} f(x_{2i+1}) + 2 * \sum_{i=0}^{\lfloor n/3 \rfloor} f(x_{2i+2}) + f(x_n) \right)$$

$$\frac{1}{30} * (5 + 427.1 + 166.39999 + 44) = 21.416666$$

5. Результаты

I	Метод ***Ньютона – Котеса***	Метод ***прямоугольников***	Метод трапеций	Метод Симпсона
21.416667	21.416645	21.40125	21.4475	21.416666

	Метод ***Ньютона – Котеса***	Метод ***прямоугольников***	Метод трапеций	Метод Симпсона
Погрешность	0.000022	0.01542	0.03083	0.000001

Листинг программы

Формула прямоугольников:

```
def _rectangle_rule(func, a, b, n, frac):
    """Обобщённое правило прямоугольников."""
    dx = (b - a) / n
    ans = 0.0
    xstart = a + frac * dx  # 0 <= frac <= 1 задаёт долю смещения точки,
                            # в которой вычисляется функция,
                            # от левого края отрезка dx
    for i in range(n):
        ans += func(xstart + i * dx)


    return ans*dx


def _rectangle_rule_with_e(func, a, b, n, e, frac):
    ans = _rectangle_rule(func, a, b, n, frac)
    err_est = max(1, abs(ans))


    while (err_est > e):
        old_ans = ans
        ans = _rectangle_rule(func, a, b, 2*n, frac)


        n *= 2
        err_est = abs(ans - old_ans)


    return ans, n


class RectangleRule:


    @classmethod
    def left_rectangle_rule(self, func, a, b, n = 4, e = 1e-4):
        """Правило левых прямоугольников"""
        return _rectangle_rule_with_e(func, a, b, n, e, 0.0)


    @classmethod
    def right_rectangle_rule(self, func, a, b, n = 4, e = 1e-4):
        """Правило правых прямоугольников"""
```

```
        return _rectangle_rule_with_e(func, a, b, n, e, 1.0)


    @classmethod
    def midpoint_rectangle_rule(self, func, a, b, n = 4, e = 1e-4):
        """Правило прямоугольников со средней точкой"""
        return _rectangle_rule_with_e(func, a, b, n, e, 0.5)
```

Метод трапеций:

```
def _rectangle_rule(func, a, b, n, frac):
    """Обобщённое правило прямоугольников."""
    dx = 1.0 * (b - a) / n
    sum = 0.0
    xstart = a + frac * dx # 0 <= frac <= 1 задаёт долю смещения точки,
                           # в которой вычисляется функция,
                           # от левого края отрезка dx
    for i in range(n):
        sum += func(xstart + i * dx)


    return sum * dx


def midpoint_rectangle_rule(func, a, b, n):
    """Правило прямоугольников со средней точкой"""
    return _rectangle_rule(func, a, b, n, 0.5)


class TrapezoidRule:
    @classmethod
    def trapezoid_rule(self, func, a, b, n = 4, e = 1e-4):
        """Правило трапеций
        e - желаемая относительная точность вычислений
        n0 - начальное число отрезков разбиения"""
        old_ans = 0.0
        dx = (b - a) / n
        ans = 0.5 * (func(a) + func(b))


        for i in range(1, n):
            ans += func(a + i * dx)


        ans *= dx
        err_est = max(1, abs(ans))
        while (err_est > e):
```

```
        old_ans = ans
        ans = 0.5 * (ans + midpoint_rectangle_rule(func, a, b, n)) # новые точки для
уточнения интеграла
                                                    # добавляются ровно в середины предыдущих
отрезков
        n *= 2
        err_est = abs(ans - old_ans)


    return ans, n
```

Метод Симпсона:

```
class SimpsonRule:


    @classmethod
    def simpson_rule(self, func, a, b, n = 4, e = 1e-4):
        """Интегрирование методом парабол с заданной точностью.
            e - относительная точность,
            n - число отрезков начального разбиения"""
        old_trapez_sum, _ = TrapezoidRule.trapezoid_rule(func, a, b, n, e=float('inf'))
        new_trapez_sum, _ = TrapezoidRule.trapezoid_rule(func, a, b, 2*n, e=float('inf'))
        ans = (4 * new_trapez_sum - old_trapez_sum) / 3


        err_est = max(1, abs(ans))


        while (err_est > e):
            n*=2
            old_ans = ans
            old_trapez_sum = new_trapez_sum
            new_trapez_sum, _ = TrapezoidRule.trapezoid_rule(func, a, b, 2*n, e=float('inf'))


            ans = (4 * new_trapez_sum - old_trapez_sum) / 3
            err_est = abs(old_ans - ans)


        return ans, n
```

Пример работы программы

- Пример 1

$$f(x) = sin(x) + cos(x)$$

Отрезок: [1, 2]
E: 0.001
Метод трапеций:1.0241922273158819

- Пример 2

$$f(x) = 3x^3 + 5x^2 + 3x - 6$$

Начально придлижение: [1, 2]
E: 0.001
Метод Симпсона: 21.416666

Выводы

В данной работе были реализованы методы численного интегрирования. Методы были протестированы на различных примерах. Результаты показали, что реализованные алгоритмы успешно справляется с поставленной задачей и находят решения в пределах допустимых погрешностей.