Университет ИТМО

ФПИиКТ

Лабораторная работа №3  
по Вычислительной математике

Выполнил: Балтабаев Дамир  
Группа: P3210  
Вариант: 4

Преподаватель: Малышева Татьяна Алексеевна

Санкт-Петербург  
2022

**Цель лабораторной работы:**

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

**Порядок выполнение работы:**

**Программная реализация задачи:**

1. Реализовать в программе методы по выбору пользователя, исходя из варианта:

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

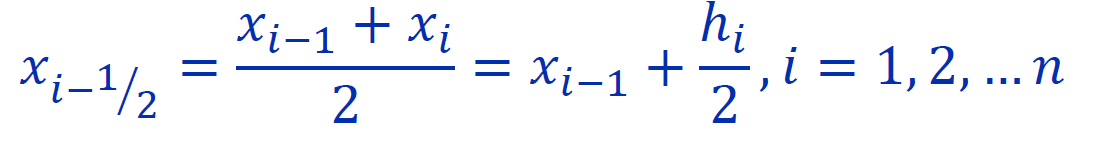
1. Методы должны быть оформлены в виде отдельной(ого) функции/класса.
2. Вычисление значений функции оформить в виде отдельной(ого) функции/класса.
3. Для оценки погрешности и завершения вычислительного процесса использовать правило Рунге.
4. Предусмотреть вывод результатов: значение интеграла, число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

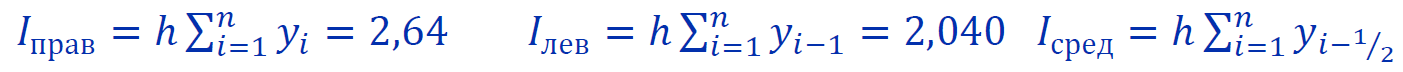
**Вычислительная реализация задачи:**

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

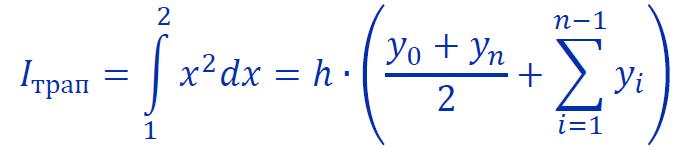
**Рабочие формулы используемых методов:**

Метод прямоугольников:

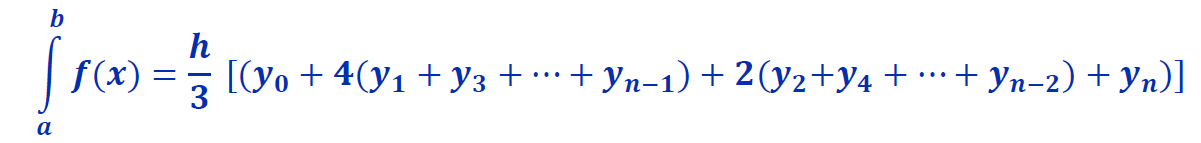




Метод трапеций:



Метод Симпсона:



Формула Ньютона - Котеса :

Изображение выглядит как текст

Автоматически созданное описание

**Листинг программы**

**Метод прямоугольников:**

**@Override**

**public void mainMethod() {**

**Integer n = 4;**

**Double IAverage;**

**Double IRight;**

**Double ILeft;**

**Double IAveragePrevious = Double.MAX\_VALUE;**

**Double IRightPrevious = Double.MAX\_VALUE;**

**Double ILeftPrevious = Double.MAX\_VALUE;**

**while (true) {**

**IRight = 0.0;**

**IAverage = 0.0;**

**ILeft = 0.0;**

**Double h = (getB() - getA()) / n;**

**for (int i = 1; i <= n; i++) {**

**IRight = IRight + functionEvaluation(getA() + i \* h);**

**ILeft = ILeft + functionEvaluation(getA() + (i - 1) \* h);**

**IAverage = IAverage + functionEvaluation(((getA() + (i - 1) \* h) + (getA() + (i) \* h)) / 2);**

**}**

**IAverage = IAverage \* h;**

**IRight = IRight \* h;**

**ILeft = ILeft \* h;**

**if ((Math.abs(IAveragePrevious - IAverage) / 3 <= getAccuracy()**

**&& Math.abs(IRightPrevious - IRight) / 3 <= getAccuracy()**

**&& Math.abs(ILeftPrevious - ILeft) / 3 <= getAccuracy())) break;**

**if (n > 1000000){**

**System.out.println("Количество разбиений превысило отметку в 1.000.000!");**

**break;**

**}**

**IAveragePrevious = IAverage;**

**IRightPrevious = IRight;**

**ILeftPrevious = ILeft;**

**n = n \* 2;**

**}**

**System.out.println("Ответ полученный методом средних прямоугольников: " + IAverage);**

**System.out.println("Ответ полученный методом правых прямоугольников: " + IRight);**

**System.out.println("Ответ полученный методом левых прямоугольников: " + ILeft + "\n");**

**System.out.println("Оценка погрешности правилом Рунге: \n" +**

**"Оценка погрешности для метода средних прямоугольников: " + Math.abs((IAveragePrevious - IAverage) / 3) + "\n" +**

**"Оценка погрешности для метода правых прямоугольников: " + Math.abs((IRightPrevious - IRight) / 3) + "\n" +**

**"Оценка погрешности для метода левых прямоугольников: " + Math.abs((ILeftPrevious - ILeft) / 3) + "\n");**

**System.out.println("Число разбиения интервала: " + n + "\n");**

**System.out.println("Точное значение интегралла методом Ньютона Лейбница = " + functionNewtonLeibniz());**

**System.out.println("Погрешность в вычислении интеграла составляет:\n" +**

**"ΔIсред = I-Iсред = " + Math.abs(functionNewtonLeibniz() - IAverage) + "\n" +**

**"ΔIлев = I-Iлев = " + Math.abs(functionNewtonLeibniz() - ILeft) + "\n" +**

**"ΔIправ = I-Iправ = " + Math.abs(functionNewtonLeibniz() - IRight) + "\n");**

**}**

**Метод трапеций:**

**@Override**

**public void mainMethod() {**

**Integer n = 4;**

**Double answer;**

**Double answerPrevious = Double.MAX\_VALUE;**

**while (true) {**

**answer = 0.0;**

**Double h = (getB() - getA()) / n;**

**for (int i = 1; i <= n - 1; i++) {**

**answer += functionEvaluation(getA() + i \* h);**

**}**

**answer = h \* (answer + ((functionEvaluation(getB()) + functionEvaluation(getA())) / 2));**

**if ((Math.abs((answerPrevious - answer) / 3) <= getAccuracy())) break;**

**if (n > 1000000) {**

**System.out.println("Количество разбиений превысило отметку в 1.000.000!");**

**break;**

**}**

**answerPrevious = answer;**

**n = n \* 2;**

**}**

**System.out.println("Ответ полученный методом трапеций: " + answer + "\n");**

**System.out.println("Оценка погрешности правилом Рунге: \n" +**

**"Оценка погрешности правилом Рунге для метода трапеций: " + Math.abs((answerPrevious - answer) / 3) + "\n");**

**System.out.println("Число разбиения интервала: " + n + "\n");**

**System.out.println("Точное значение интегралла методом Ньютона Лейбница = " + functionNewtonLeibniz());**

**System.out.println("Погрешность в вычислении интеграла составляет:\n" +**

**"ΔI =I-Iтрап = " + Math.abs(functionNewtonLeibniz() - answer));**

**}**

**Результаты выполнения программы**

Метод прямоугольников:

Изображение выглядит как текст

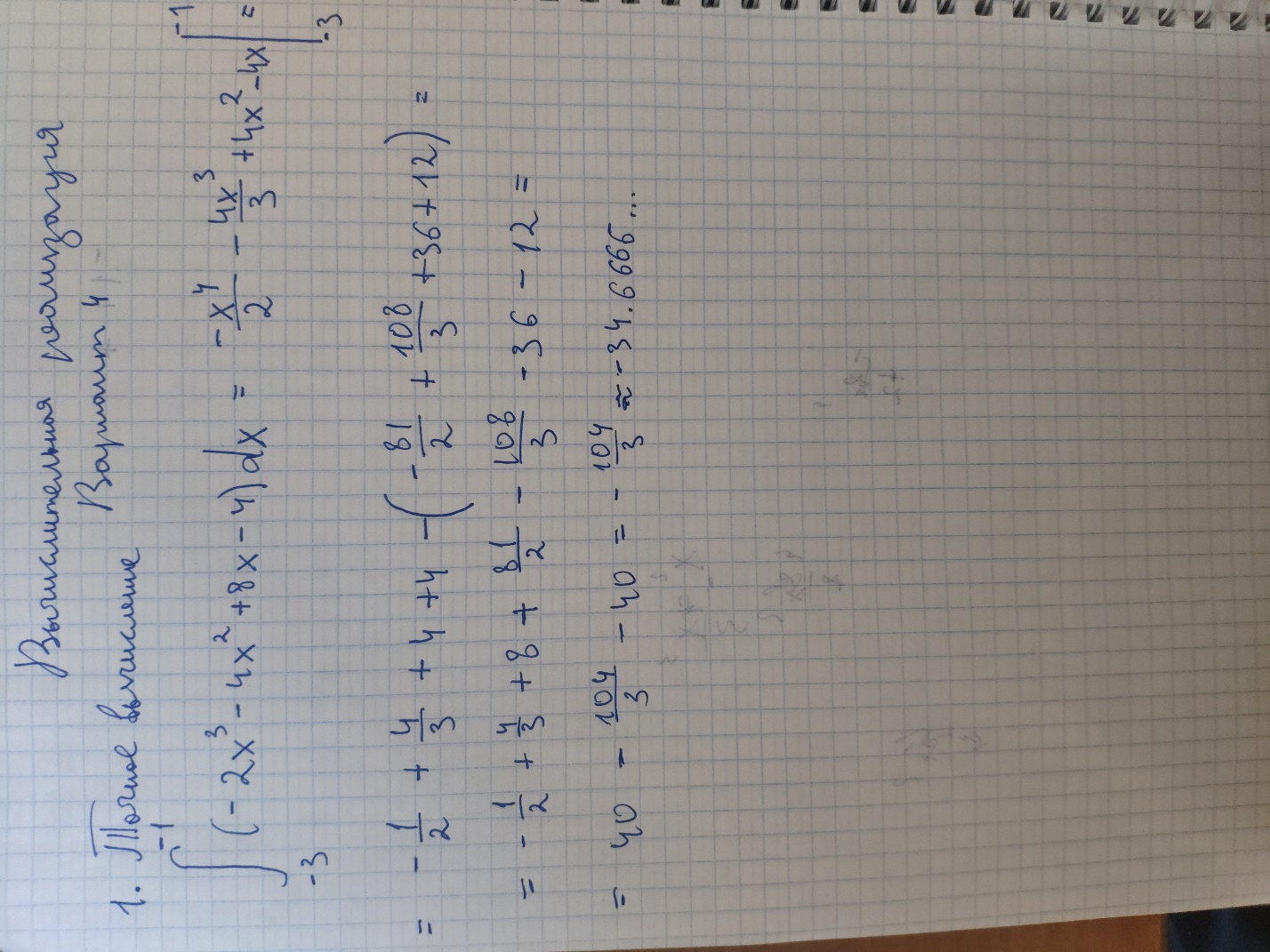
Автоматически созданное описание

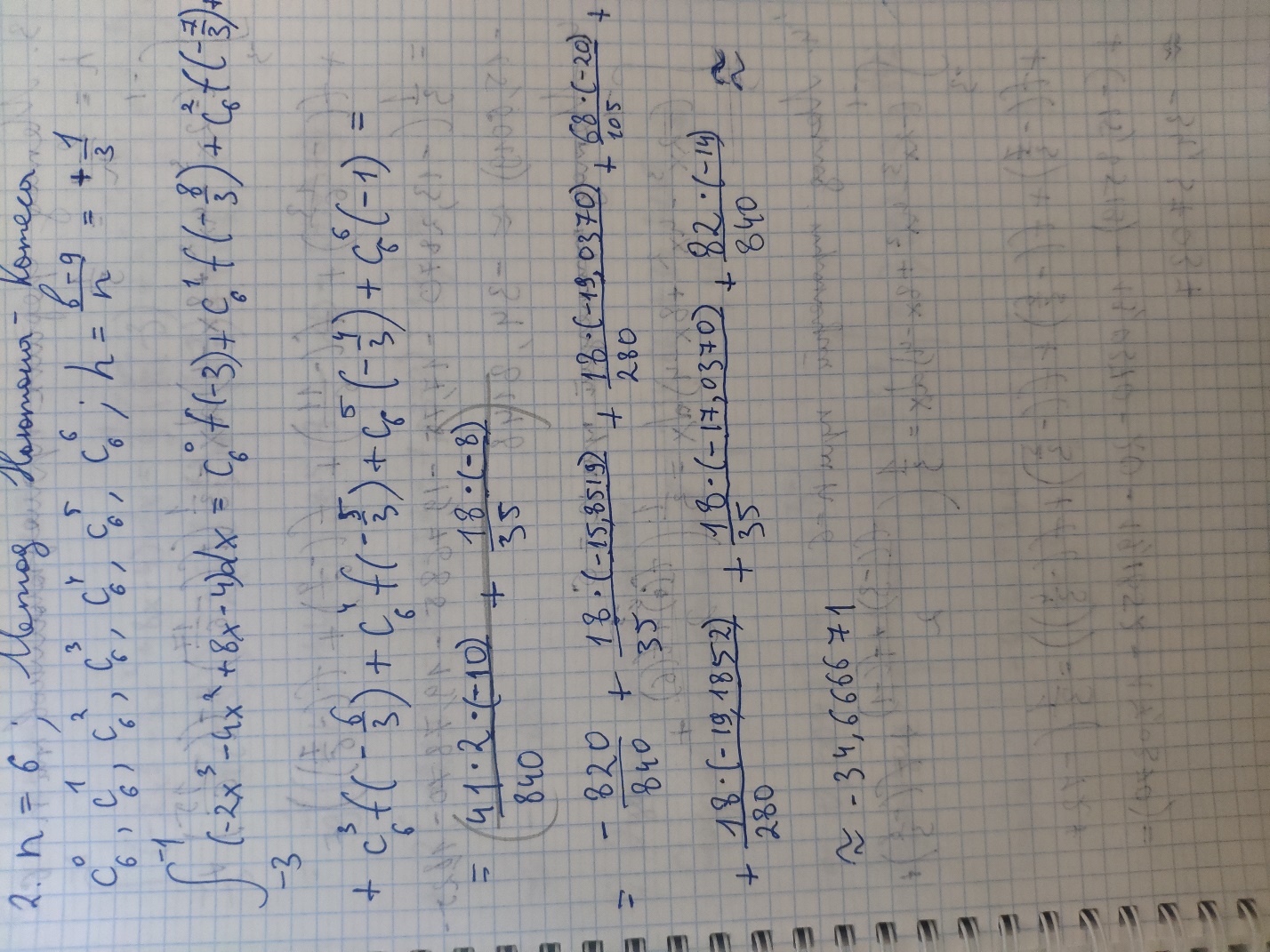
Метод трапеций:

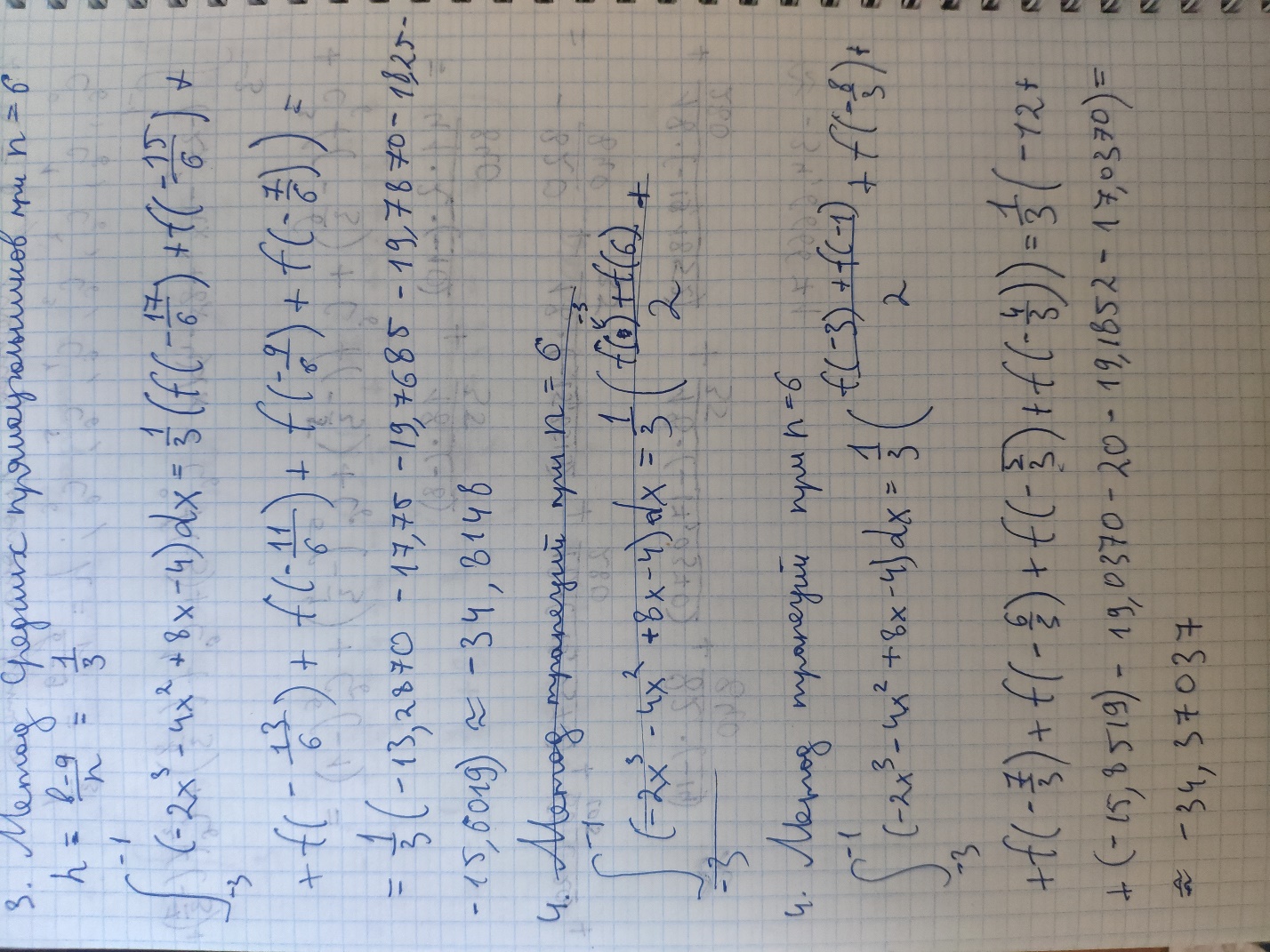
Изображение выглядит как текст

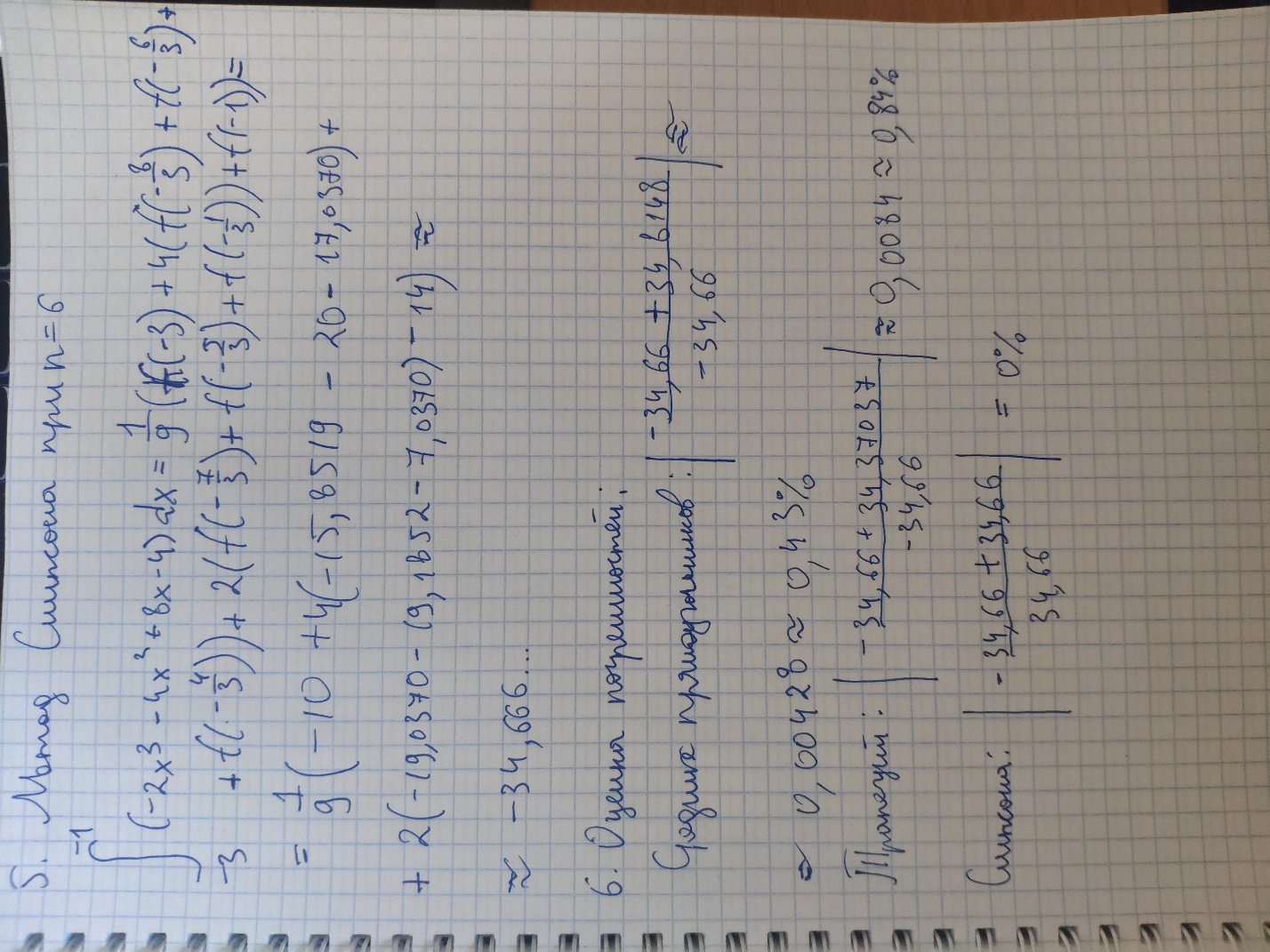
Автоматически созданное описание

**Вычисление заданного интеграла**

****

****

****

****

**Выводы**

В ходе выполнения данной лабораторной работы я научился находить приближенное значение интеграла разными методами. Поработал с методами: Ньютона-Котеса, прямоугольников, трапеций, Симпсона. Узнал о правиле Рунге, для завершения вычислительного процесса и оценки погрешности.