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

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа **№3**
«Численное интегрирование»

по дисциплине «Вычислительная математика»

Вариант: **2**

Преподаватель:
Машина Е. А.

Выполнил:
Вальц Мартин Эдуардович
Группа: P3210

Санкт-Петербург, 2024 г.

Цель работы: найти приближенное значение определенного интеграла с требуемой точностью различными численными методами.

1. Вычислительная реализация задачи

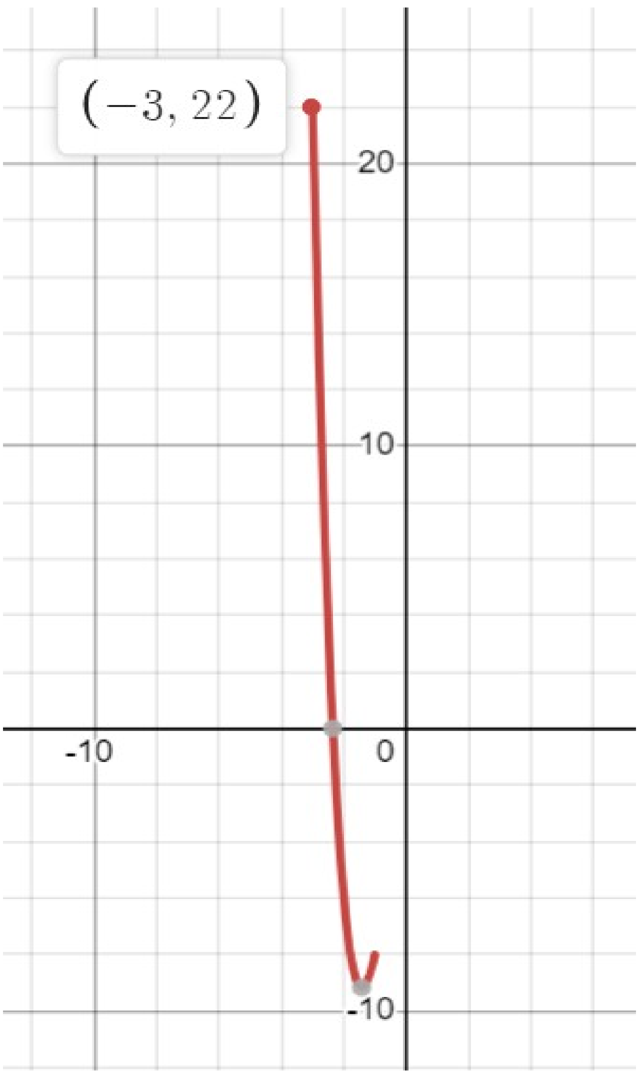
1. **Вычислить интеграл**, приведенный в таблице 1, **точно:**

$$\int_{-3}^{-1} \left(-3x^3-5x^2+4x-2\right) dx$$

$$F(x)=\frac{-3x^4}{4}-\frac{5x^3}{3}+2x^2-2x;\ F(-1)=\frac{59}{12};\ F(-3)=\frac{33}{4}$$

$$I_{точн}=F(x)=F(-1)-F(-3)=\frac{59}{12}-\frac{33}{4}=\frac{-10}{3}\approx -3.(3)$$

f(x)



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

$$h=\frac{b-a}{6}=\frac{(-1)-(-3)}{6}=\frac{1}{3}$$

$$\int_a^b f(x)\,dx\approx c_6^0 f(a)+c_6^1 f(a+h)+c_6^2 f(a+2h)+c_6^3 f(a+3h)+c_6^4 f(a+4h)+c_6^5 f(a+5h)+c_6^6 f(b)$$

$$I_{cotes}=((-1)-(-3))\times\left(\frac{41}{840}f(-3)+\frac{216}{840}f\left(\frac{-8}{3}\right)+\frac{27}{840}f\left(\frac{-7}{3}\right)+\frac{272}{840}f(-2)+\frac{27}{840}f\left(\frac{-5}{3}\right)+\frac{216}{840}f\left(\frac{-4}{3}\right)+\right.$$

[Решение на Wolfram Alpha]

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

$$h=\frac{b-a}{n}=\frac{-1-(-3)}{10}=\frac{1}{5}$$

- **Метод средних прямоугольников**:

$$I_{ср.прям}=h\sum_{i=1}^{n}y_{i-\frac{1}{2}}=h\cdot\left(f\left(a+\frac{h}{2}\right)+f\left(a+\frac{3h}{2}\right)+f\left(a+\frac{5h}{2}\right)+f\left(a+\frac{7h}{2}\right)+f\left(a+\frac{9h}{2}\right)+f\left(a+\frac{11h}{2}\right)+f\left(a+\frac{13\ldots}{2}\right.\right.$$

$$0.2(f(-3+0.1)+f(-3+0.3)+f(-3+0.5)+f(-3+0.7)+f(-3+0.9)+f(-3+1.1)+f(-3+1.3)+f(-3+$$

[Решение на Wolfram Alpha]

- **Метод трапеций**:

$$I_{трапеция}=h\cdot\left(\frac{y_0+y_n}{2}+\sum_{i=1}^{n-1}y_i\right)$$

$$I_{трапеция}=0.2\left(\frac{f(-3)+f(-1)}{2}+f(-3+0.2)+f(-3+0.4)+f(-3+0.6)+f(-3+0.8)+f(-3+1)+f(-3+1.\right.$$

[Решение на Wolfram Alpha]

- **Метод Симпсона**:

$$I_{Симпсона}=\frac{h}{3}\cdot\left(y_0+4\sum_{i=1}^{n-1}y_{нечёт}+2\sum_{i=2}^{n-2}y_{чёт}+y_n\right)$$

$$I_{Симпсона}=\frac{0.2}{3}(f(-3)+4*(f(-3+0.2)+f(-3+0.6)+f(-3+1)+f(-3+1.4)+f(-3+1.8))+2*(f(-3$$

[Решение на Wolfram Alpha]

4. Сравнить результаты с точным значением интеграла:

Точное значение интеграла на интервале вычислено как $\frac{-10}{3} = -3.(3)$

1. Для метода **Ньютона–Котеса** при $n=6$: $I_{точн} = I_{cotes} = -3.(3)$, **значения совпадают**.
   $R = |I_{точн} - I_{cotes}| = \left|\frac{-10}{3} - (-3.33333\ldots)\right| = 0$
2. Для метода **средних прямоугольников** при $n=10$: $I_{ср.прям} = -3.42$.
   $R = |I_{точн} - I_{ср.прям}| = \left|\frac{-10}{3} - (-3.42)\right| = 0.08(6)$
3. Для метода **трапеций** при $n=10$: $I_{трапеция} = -3.16$.
   $R = |I_{точн} - I_{трапеция}| = \left|\frac{-10}{3} - (-3.16)\right| = 0.17(3)$
4. Для метода **Симпсона** при $n=10$: $I_{точн} = I_{Симпсона} = -3.(3)$, **значения совпадают**.
   $R = |I_{точн} - I_{cotes}| = \left|\frac{-10}{3} - (-3.33333\ldots)\right| = 0$

5. Определить относительную погрешность вычислений для каждого метода.

1. Для метода **Ньютона–Котеса**: $R=0 \rightarrow$ **погрешности нет.**
2. Для метода **средних прямоугольников**:
   $\Delta = -3.(3) - (-3.42) \vee \frac{\square}{-3.(3) \vee} \approx 2.6\%$
3. Для метода **трапеций**: $\Delta = -3.(3) - (-3.16) \vee \frac{\square}{-3.(3) \vee} \approx 5.2\%$
4. Для метода **Симпсона**: $R=0 \rightarrow$ **погрешности нет.**

Как видно из результатов, все методы дали относительно малую погрешность, особенно при использовании формулы Ньютона–Котеса и Симпсона. Наилучший результат был получен при использовании формулы Ньютона–Котеса с $n=6$ и формулы Симпсона с $n=10$, при которых значения интеграла полностью совпали.

2. Программная реализация задачи

RectangleMethod.java

```
package methods;

import storage.FunctionStorage;
import util.Printer;

public class RectangleMethod {
   private static int typeOfRectangleMethod;

   public static void setTypeOfRectangleMethod(int typeOfRectangleMethod) {
      RectangleMethod.typeOfRectangleMethod = typeOfRectangleMethod;
   }

   public static int getTypeOfRectangleMethod() {
      return typeOfRectangleMethod;
   }

   public static double execute(double a, double b, double n) {
      switch (typeOfRectangleMethod){
         case 1 -> {
            return executeRight(a, b, n);
         }
         case 2 -> {
            return executeLeft(a, b, n);
         }
         default-> {
            return executeMedium(a, b, n);
         }
      }

   }

   private static double executeRight(double a, double b, double n){
      double sum = 0;
      double h = (b-a)/n;
      a += h;
      for(int i = 1; i < n+1; i++){
         sum += (FunctionStorage.getFunction(a));
         a += h;
      }
      return h * sum;
   }

   private static double executeLeft(double a, double b, double n){
      double sum = 0;
      double h = (b-a)/n;
      for(int i = 0; i < n; i++){
         sum += (FunctionStorage.getFunction(a));
         a += h;
```

```
        }
        return h * sum;
    }

    private static double executeMedium(double a, double b, double n){
        double sum = 0;
        double h = (b-a)/n;
        for(int i = 0; i < n; i++){
            sum += (FunctionStorage.getFunction(a + h/2));
            a += h;
        }
        return h * sum;
    }


}
```

SimpsonMethod.java

```
package methods;

import storage.FunctionStorage;
import util.Printer;

public class RectangleMethod {
    private static int typeOfRectangleMethod;

    public static void setTypeOfRectangleMethod(int typeOfRectangleMethod) {
        RectangleMethod.typeOfRectangleMethod = typeOfRectangleMethod;
    }

    public static int getTypeOfRectangleMethod() {
        return typeOfRectangleMethod;
    }

    public static double execute(double a, double b, double n) {
        switch (typeOfRectangleMethod){
            case 1 -> {
                return executeRight(a, b, n);
            }
            case 2 -> {
                return executeLeft(a, b, n);
            }
            default-> {
                return executeMedium(a, b, n);
            }
        }

    }

    private static double executeRight(double a, double b, double n){
        double sum = 0;
        double h = (b-a)/n;
```

```
        a += h;
        for(int i = 1; i < n+1; i++){
            sum += (FunctionStorage.getFunction(a));
            a += h;
        }
        return h * sum;
    }

    private static double executeLeft(double a, double b, double n){
        double sum = 0;
        double h = (b-a)/n;
        for(int i = 0; i < n; i++){
            sum += (FunctionStorage.getFunction(a));
            a += h;
        }
        return h * sum;
    }

    private static double executeMedium(double a, double b, double n){
        double sum = 0;
        double h = (b-a)/n;
        for(int i = 0; i < n; i++){
            sum += (FunctionStorage.getFunction(a + h/2));
            a += h;
        }
        return h * sum;
    }


}
```

TrapezoidMethod.java

```
package methods;

import storage.FunctionStorage;

public class TrapezoidMethod {
    public static double execute(double a, double b, double n) {
        double sum = 0;
        double h = (b-a)/n;
        a += h;
        for(int i = 1; i < n; i++){
            sum += (FunctionStorage.getFunction(a));
            a += h;
        }
        return h * (sum+(FunctionStorage.getFunction(a)+FunctionStorage.getFunction(b))/2);
    }
}
```

Вывод

В ходе выполнения лабораторной работы были изучены численные методы интегрирования с использованием Java. В результате работы были рассмотрены различные численные методы вычисления определенных интегралов: метод прямоугольников (левых, правых, средних), метод трапеций, метод Ньютона-Котеса и метод Симпсона.

Была реализована программа, позволяющая выбрать одну из предложенных функций, задать пределы интегрирования, точность и начальное значение числа разбиения интервала интегрирования. Написав реализации всех трех методов решения интегралов, можно сделать вывод, что самым точным и быстрым является метод Симпсона.

В ходе вычислительной реализации задачи были рассчитаны интегралы различными методами и проведено сравнение результатов с точными значениями интегралов.

Также была выполнена дополнительная задача по установлению сходимости рассматриваемых несобственных интегралов 2 рода и их вычислению заданными численными методами в случаях, когда подынтегральная функция терпит бесконечный разрыв в точке a, в точке b или на отрезке интегрирования.