**ИТМО Кафедра Информатики и прикладной математики**

Отчет по лабораторной работе №2

«Интегрирование»   
Вариант : метод трапеций

**Выполнил: студент группы P3217**

**Плюхин Дмитрий**

**Преподаватель: Калёнова О. В.**

**2016 год**

1. **Описание метода**

Метод трапеций — метод численного интегрирования функции одной переменной, заключающийся в замене на каждом элементарном отрезке подынтегральной функции на многочлен первой степени, то есть линейную функцию. Площадь под графиком функции аппроксимируется прямоугольными трапециями.

Суть метода состоит в разбиении промежутка интегрирования на некоторое количество областей, подсчитывании значений функции на границах каждой области, нахождении и складывании площадей получаемых трапеций.

Если оценка погрешности производится методом Рунге, то изначально рассматривается только одна область разбиения, приближенное значение интеграла на которой рассчитывается по формуле (\*). Далее количество областей увеличивается вдвое и приближенное значение интеграла при новых областях разбиения рассчитывается по формуле (\*\*) до тех пор, пока погрешность, даваемая формулой (\*\*\*) не станет меньше требуемой.

Если необходимая точность так и не была достигнута, либо на каком-то этапе получено некорректное значение (функция дает слишком большое значение при некотором аргументе или не определена), то метод не работает в данном случае.

1. **Листинг основной части программы**

public static ArrayList<Double> CountIntegral(Integrable f, double lowbound, double highbound, double accuration){

boolean changeSign = false;

ArrayList<Double> rezult = new ArrayList<>();

double TmpValue = 0d;

double Value = 0d;

double realAccuration = 0d;

int numberOfSegments = 1;

if (lowbound > highbound){

double tmp = lowbound;

lowbound = highbound;

highbound = tmp;

changeSign = true;

}

double step = highbound - lowbound;

Value = (highbound - lowbound)\*(f.getValue(highbound) + f.getValue(lowbound))/2;

if (!Double.isFinite(Value) || Double.isNaN(Value)){

throw new IllegalArgumentException();

}

do{

if (numberOfSegments > 100\_000){

throw new IllegalArgumentException();

}

TmpValue = Value;

Value = 0;

numberOfSegments \*= 2;

step = (highbound - lowbound)/numberOfSegments;

for (int i = 0; i < numberOfSegments; i++){

Value += step\*(f.getValue(lowbound + i\*step) +

f.getValue(lowbound + (i+1)\*step))/2;

}

} while(Math.abs(Value - TmpValue)/3 > Math.abs(accuration));

realAccuration = Math.abs(Value - TmpValue)/3;

if (changeSign){

Value = -Value;

}

if (!Double.isFinite(Value) || Double.isNaN(Value)){

throw new IllegalArgumentException();

}

rezult.add(Value);

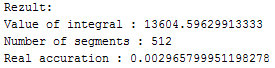
rezult.add((double)numberOfSegments);

rezult.add(realAccuration);

return rezult;

}

**3. Примеры и результаты работы программы**

Пример 1. Функция:

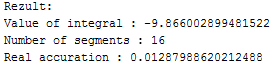
a(x) = 13.6x^2 + 63.1x + 61.13

Пределы интегрирования:

От 6.0 до 13.0

Погрешность:

0.01

Пример 2. Функция:

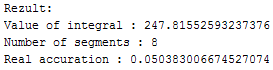
b(x) = -4.0sin(x) + 5.87cos(x) + 6.6tg(x) - 7.0sin(x)cos(x)

Пределы интегрирования:

От 1.0 до -1.0

Погрешность:

0.02



Пример 3. Функция:

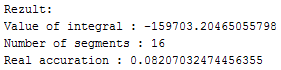
c(x) = 8.1ln(x) + 9.2lg(x) + 10.5

Пределы интегрирования:

От 6.6 до 13.13

Погрешность:

0.1



Пример 4. Функция:

d(x) = ( ( 15.8ln(x) + 3.14lg(x) + 2.718281828459045 ) \*

\* ( 15.8sin(x) + 3.14cos(x) + 2.718281828459045tg(x) +

+ 0.1sin(x)cos(x) ) ) -

- ( 15.8x^2 + 3.14x + 2.718281828459045 )

Пределы интегрирования:

От 100 до 101

Погрешность:

0.3

**4. Блок – схема численного метода**



1. **Вывод**

Так, при решении интегралов численными методами приходится сталкиваться с небольшими сложностями вычислений, однако, эти сложности позволяют организовать процесс вычислений таким образом, чтобы мгновенно находить значения интегралов наиболее трудноинтегрируемых вручную функций. Полученные навыки можно использовать в различных областях, начиная от создания карманных приложений по расчету интегралов до применения численных методов в производстве, а написанная программа может послужить основой для создания более крупной и совершенной системы расчета математических величин. Также с помощью написанной программы и полученных знаний можно значительно ускорять процесс вычислений при изучении новых тем в курсе высшей математики и решении соответствующих примеров.