Федеральное государственное автономное

образовательное учреждение высшего образования

“Национальный исследовательский университет ИТМО”

Факультет программной инженерии и компьютерной техники

Лабораторная работа №3

по вычислительной математике

Вариант №2

Выполнил:

Голиков Андрей

Группа: P32092

Принял: Рыбаков

Степан Дмитриевич

???

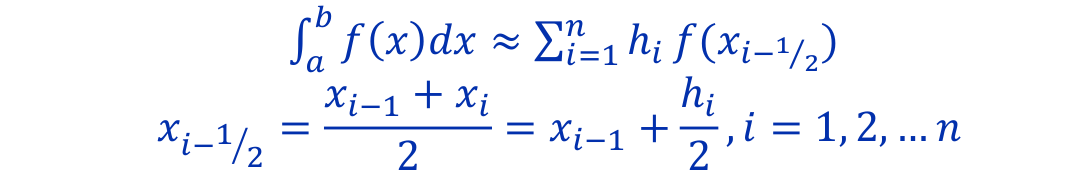
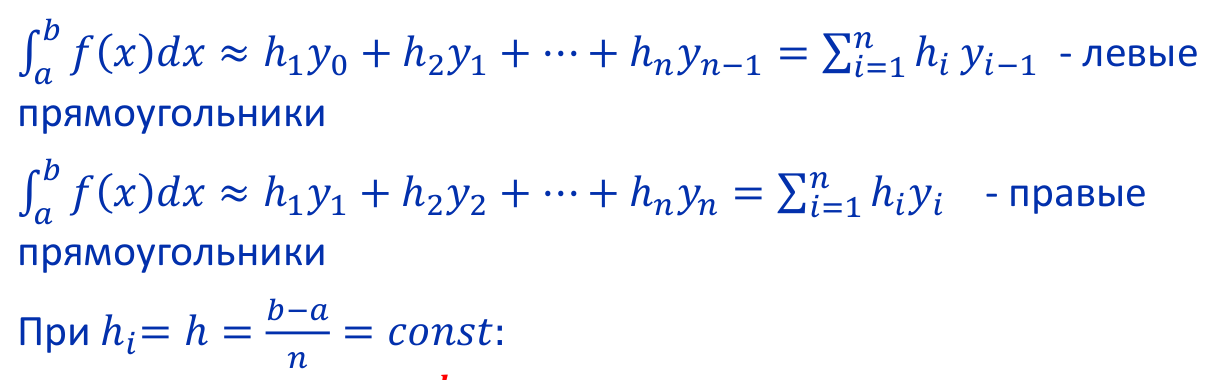
?

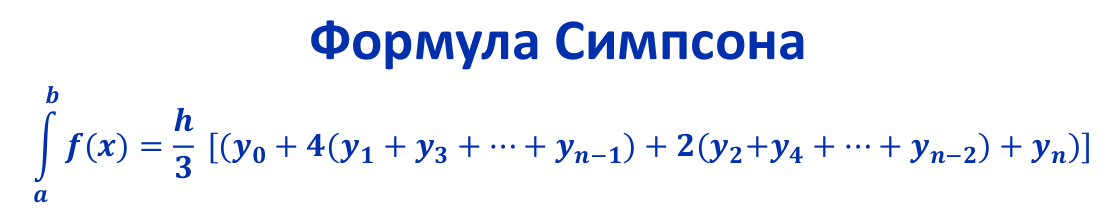
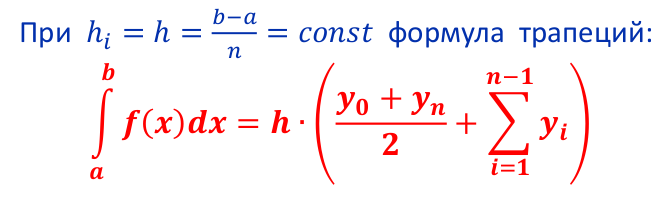


Санкт-Петербург, 2023

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

Рабочие формулы методов:





Листинг программы:

module Lib

( eval, parseFunction, parseMethod, Func, Method

) where

func0 :: Func

func0 x = -3\*x^(3 :: Integer) -5\*x^(2 :: Integer) + 4\*x - 2

rect :: Double -> Double -> Method

rect l r n f (a, b) = diff \* (sum $ map ((\x -> f (a + x\*diff))) [l..r]) where

diff = (b - a) / n

left\_rect :: Method

left\_rect n = rect 0.0 (n - 1) n

right\_rect :: Method

right\_rect n = rect 1.0 n n

middle\_rect :: Method

middle\_rect n = rect 0.5 (n - 0.5) n

trapeze :: Method

trapeze n f (a, b) = (left\_rect n f (a, b)) + ((f b) - (f a))/ 2 \* diff where

diff = (b - a)/n

simpson :: Method

simpson n f (a, b) = diff/3 \* (f a + f b + 2 \* (sum $ map (\x -> f (a + x\*diff)) [2,4..n-2]) + 4 \* (sum $ map (\x -> f (a + x\*diff)) [1,3..n-1])) where

diff = (b - a)/n

runge :: (Double, Double) -> Integer -> Accuracy -> Bool

runge (i1, i2) k eps = ((i1 - i2) / (2^k - 1)) < eps

eval :: Func -> Method -> Integer -> Points -> Accuracy -> (Double, Integer)

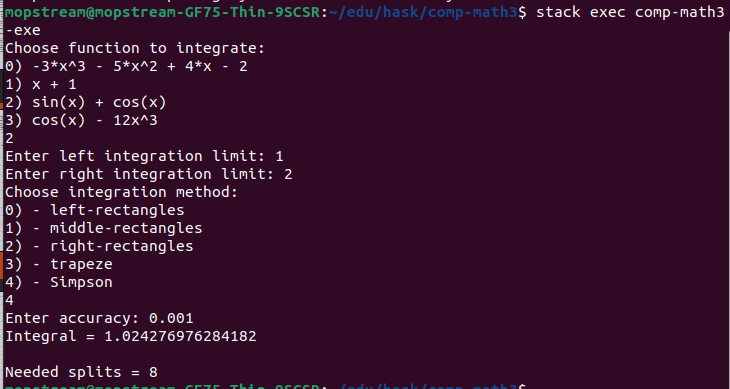
eval f method k points = eval' 4 (method 4 f points) f method k points

eval' :: Double -> Double -> Func -> Method -> Integer -> Points -> Accuracy -> (Double, Integer)

eval' n i1 f method k points eps = if (runge (i1, i2) k eps) then (i2, (round (n\*2))) else (eval' (n\*2) i2 f method k points eps) where

i2 = method (n\*2) f points

Результаты работы программы:



Вычислительная реализация:

Вычисление интеграла №2:

Ньютон-Котес

Погрешность - 0.141

Средние прямоугольники

Погрешность - 0.87

Трапеции

Погрешность 0.173

Симпсон

Погрешность - 0

Вывод: реализовал на языке Haskell приблизительное вычисление определенных интералов методами прямоугольников (левые, средние и правые), трапеций и Симпсона. Оценивал погрешности в соответствии с правилом Рунга. Метод Симпсона дает большую точность при меньшем разбиении исходного интеграла.