Федеральное государственное автономное

образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
| Программная инженерия |
| кафедра |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

|  |
| --- |
| Основы функционального программирования в Scala |
| тема |

Вариант 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | К. В. Богданов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-17/1Б, 032156940 |  |  |  | Н. А. Самарин |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2024

**СОДЕРЖАНИЕ**

1 Задание............................................................................................................... 3

2 Исходный код разработанного алгоритма..................................................... 3

3 Результат........................................................................................................... 4

**1 Задание**

Написать скрипт, вычисляющий и выводящий в консоль в виде таблицы  
значения функции, заданной с помощью ряда Тейлора на интервале от Хнач до  
Хкон с шагом dx с точностью e. Таблицу снабдить заголовком.



Рисунок 1 – Функция и ей ряд Тейлора варианта 4

**2 Исходный код разработанного алгоритма**

Листинг 1 – Исходный код разработанного алгоритма

import scala.annotation.tailrec  
import scala.math.{log, abs}  
import scala.io.StdIn.readDouble  
object Main {  
 def main(args: Array[String]): Unit = {  
 /\*\*  
 \* Вычисление n-го элемента ряда Тейлора для ln(x + 1)  
 \* @param x x функции  
 \* @param n номнр элемента ряда Тейлора  
 \* @return значение элемента ряда Тейлора  
 \*/  
 def taylorElement(x: Double, n: Double): Double = {  
 Math.pow(-1, n) \* Math.pow(x, n + 1) / (n + 1)  
 }  
 /\*\*  
 \* Вычисление значения функции ln(x + 1)  
 \* @param x x функции  
 \* @return значение функции при заданном x  
 \*/  
 def realFunc(x: Double): Double = {  
 log(x + 1)  
 }  
 /\*\*  
 \* Вычисление ряда Тейлора вплоть до элемента ряда не большего e  
 \* @param x x функции  
 \* @param n номер текущего элемента ряда Тейлора  
 \* @param e точность  
 \* @return сумма элементов ряда Тейлора  
 \*/  
 def taylorSeries(x: Double, n: Int = 0, e: Double): (Double, Int) = {  
 val tElem = taylorElement(x, n)  
 if (abs(tElem) > (e / 10)) {  
 val taylorSeriesRec = taylorSeries(x, n + 1, e)  
 (tElem + taylorSeriesRec.\_1, taylorSeriesRec.\_2)  
 }  
 else (0, n)  
 }  
 /\*\*  
 \* Вычислить значения функции и значения ряда Тейлора для последовательности  
 x-ов  
 \* @param x текущий x

Окончание листинга 1

\* @param xEnd конечный x  
 \* @param dx шаг x  
 \* @param e точность  
 \* @param acc уже вычисленные значения  
 \* @return Список кортежей значений функций, значений ряда Тейлора и количес  
 тво элементов в вычисленных рядах Тейлора  
 \*/  
 @tailrec  
 def calculateTaylorSeriesForXSequence(x: Double, xEnd: Double, dx: Double, e  
 : Double, acc: List[(Double, Double, Double, Int)] = Nil): List[(Double, Dou  
 ble, Double, Int)] = {  
 if ((xEnd - x) / dx < dx)  
 acc.reverse  
 else {  
 val realResult = realFunc(x)  
 val taylorResult = taylorSeries(x, 0, e)  
 calculateTaylorSeriesForXSequence(x + dx, xEnd, dx, e, (x, realResult, t  
 aylorResult.\_1, taylorResult.\_2) :: acc)  
 }  
 }  
 if (args.length == 4) {

val xStart = args(0).toDouble // стартовое значение x

val xEnd = args(1).toDouble // конечное значение x

val dx = args(2).toDouble // шаг

val e = args(3).toDouble // точность

if (xStart <= -1 || xStart > 1 || xEnd <= -1 || xEnd > 1) {

println("Incorrect x bounds")

}

else if (e <= 0) {

println("Incorrect e")

}

else if ((xEnd - xStart) / dx < 0) {

println("Incorrect dx")

}

else {

val table = calculateTaylorSeriesForXSequence(xStart, xEnd, dx, e)

println("x\t\tf(x)\t\t\t\tTaylor(x)\t\t\t\tTI")

table.foreach { case (x, realResult, taylorResult, iterations) => println(f"$x%.3f $realResult%.20f $taylorResult%.20f $iterations%d") }

}

}

else (println(usage))  
 }  
}

**3 Результат**

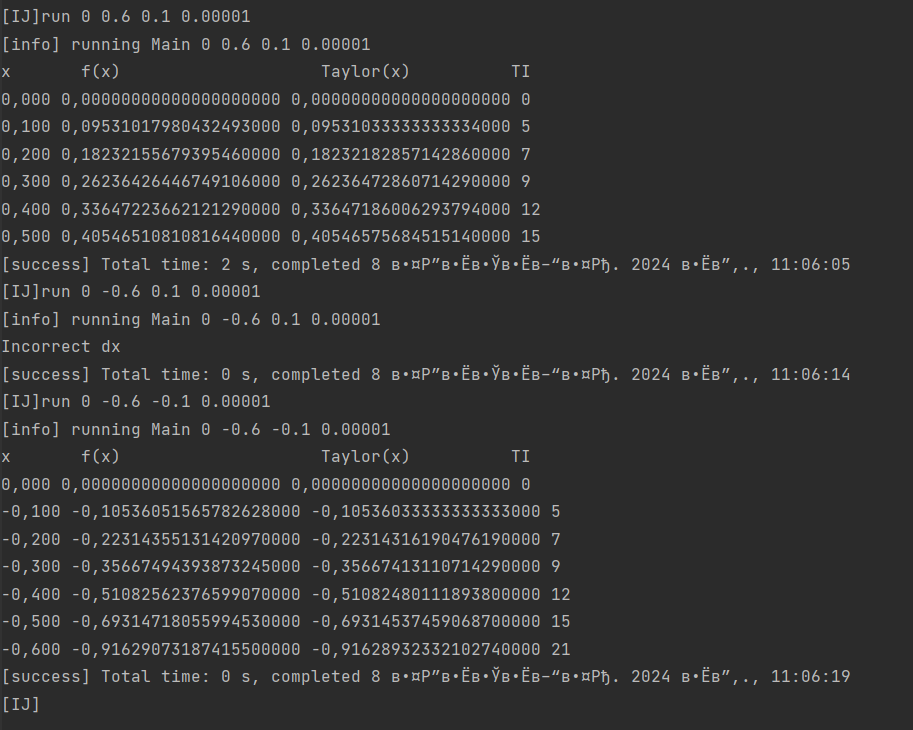


Рисунок 2 – Результат работы алгоритма