|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет социально-экономических и компьютерных наук* |
|  |
| Ханжин Александр Евгеньевич  **ЛАБОРАТОРНАЯ РАБОТА №3**  студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *09.03.04 Программная инженерия*   |  |  | | --- | --- | |  | Руководитель к.т.н., доцент кафедры ИТБ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  О. Л. Викентьева |   Пермь, 2023 |

# Задача 1

## Постановка задачи

Для х изменяющегося от a до b с шагом (b-a)/k, где (k=10), вычислить функцию f(x) (рис. 1.1.1), используя ее разложение в степенной ряд в двух случаях:

а) для заданного n;

б) для заданной точности e (e=0.0001).

Для сравнения найти точное значение функции.

## Анализ

Исходные данные:

* константа a = 0.1;
* константа b = 0.8;
* константа k = 10;
* константа n = 30;
* константа e = 0.0001;

С учетом эти значений нужно вычислить Sn (рис. 1.2.1) и Se (рис. 1.2.2).

Обе функции имеют числитель, который можно вычислить рекуррентно. В ходе преобразований числителя его рекуррентное соотношение будет равно x4.

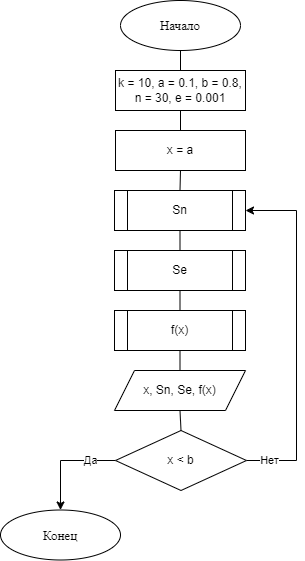
Исходные и выходные данные каждой функции представлены в таблице 1.

***Табл. 1. Анализ классов исходных данных.***

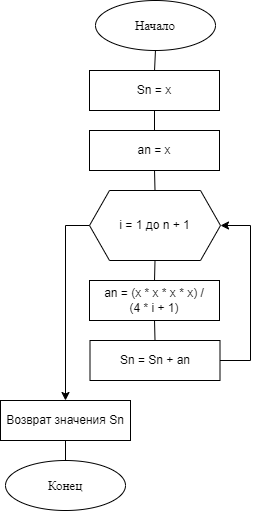
|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Описание функции** | **Исходные данные** | **Выходные данные** |
| 1 | Вычисление функции f(x) | x | Вещественное число |
| 2 | Вычисление Sn | n, x | Вещественное число |
| 3 | Вычисление Se | e, x | Вещественное число |

## Алгоритм (блок-схема)

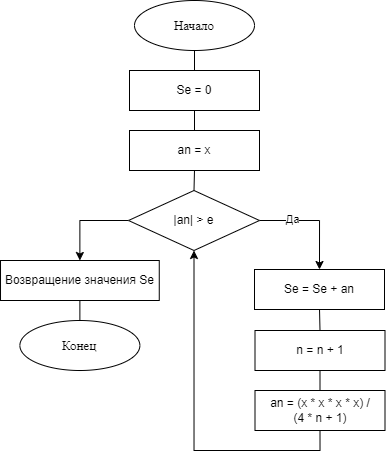
Для дальнейшей реализации проектирования необходимо составить блок схему. Она состоит из четырех частей: основная часть программы (рис. 1.3.1), функции вычисления Sn (рис. 1.3.2), функции вычисления Se (рис. 1.3.3), функции вычисления f(x)(рис. 1.3.4).

******

***Рис. 1.3.1. Блок-схема 1 задания***

******

***Рис. 1.3.2. Блок-схема операции “Вычисление Sn”***

******

***Рис. 1.3.3. Блок-схема операции “Вычисление Se”***



***Рис. 1.3.4. Блок-схема операции “Вычисление f(x)”***

## Программа (листинг)

using System;

using System.Linq.Expressions;

namespace Лаба1

{

internal class task3

{

/// <summary>

/// main function

/// </summary>

static void Main(string[] args)

{

FirstTask();

}

/// <summary>

/// function count Se, Sn, f(x) for different x

/// </summary>

/// <param ="xa">start point for x</param>

/// <param ="xb">finish point for x</param>

/// <param ="sn">amount power series expansions for baced n</param>

/// <param ="se">amount power series expansions for accuracy e</param>

/// <param ="k">count point for count Se, Sn, f(x)</param>

/// <param ="n">count series expansions for Sn</param>

/// <param ="e">count accuracy series expansions for Se</param>

static void FirstTask()

{

double xa = 0.1;

double xb = 0.8;

double f;

double sn, se;

double k = 10;

double x = xa;

int n = 30;

double e = 0.0001;

// automatic table formation

Console.WriteLine(string.Join("", Enumerable.Repeat("\_", 107)));

Console.WriteLine("|" + string.Join("", Enumerable.Repeat(" ", 25)) + "|" +

string.Join("", Enumerable.Repeat(" ", 26)) + "|" +

string.Join("", Enumerable.Repeat(" ", 26)) + "|" +

string.Join("", Enumerable.Repeat(" ", 25)) + "|");

Console.WriteLine("|" + string.Join("", Enumerable.Repeat(" ", 12)) + "X" + string.Join("", Enumerable.Repeat(" ", 12)) + "|" +

string.Join("", Enumerable.Repeat(" ", 12)) + "Sn" + string.Join("", Enumerable.Repeat(" ", 12)) + "|" +

string.Join("", Enumerable.Repeat(" ", 12)) + "Se" + string.Join("", Enumerable.Repeat(" ", 12)) + "|" +

string.Join("", Enumerable.Repeat(" ", 12)) + "Y" + string.Join("", Enumerable.Repeat(" ", 12)) + "|");

Console.WriteLine("|" + string.Join("", Enumerable.Repeat("\_", 25)) + "|" +

string.Join("", Enumerable.Repeat("\_", 26)) + "|" +

string.Join("", Enumerable.Repeat("\_", 26)) + "|" +

string.Join("", Enumerable.Repeat("\_", 25)) + "|");

Console.WriteLine("|" + string.Join("", Enumerable.Repeat(" ", 25)) + "|" +

string.Join("", Enumerable.Repeat(" ", 26)) + "|" +

string.Join("", Enumerable.Repeat(" ", 26)) + "|" +

string.Join("", Enumerable.Repeat(" ", 25)) + "|");

// main cycle, selecting the value of x from xa to xb with step = (xb - xa) / k

do

{

// calculating different functions in different ways

sn = countSN(n, x);

se = countSE(e, x);

f = countY(x);

// automatic table formation

string bufx, bufn, bufe, bufy;

bufx = string.Join("", Enumerable.Repeat(" ", (20 - x.ToString().ToArray().Length)));

bufn = string.Join("", Enumerable.Repeat(" ", (20 - sn.ToString().ToArray().Length)));

bufe = string.Join("", Enumerable.Repeat(" ", (20 - se.ToString().ToArray().Length)));

bufy = string.Join("", Enumerable.Repeat(" ", (20 - f.ToString().ToArray().Length)));

// data output in automatic table

Console.WriteLine($"| X = {x}{bufx}| Sn = {sn}{bufn}| Se = {se}{bufe}| y = {f}{bufy}|");

x += (xb - xa) / k;

} while (x < xb);

// automatic table formation

Console.WriteLine("|" + string.Join("", Enumerable.Repeat("\_", 25)) + "|" +

string.Join("", Enumerable.Repeat("\_", 26)) + "|" +

string.Join("", Enumerable.Repeat("\_", 26)) + "|" +

string.Join("", Enumerable.Repeat("\_", 25)) + "|");

}

/// <summary>

/// count amount Sn with a recurrent numerator for baced n

/// </summary>

/// <param ="amount">amount Sn</param>

/// <param ="an">the part of a numerical series</param>

/// <returns>total amount</returns>

static double countSN(int n, double x)

{

double amount = x;

double an = x;

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

{

an \*= (x \* x \* x \* x) / (double)(4 \* i + 1);

amount += an;

}

return amount;

}

/// <summary>

/// count amount Sn with a recurrent numerator for accuracy e

/// </summary>

/// <param ="amount">amount Sn</param>

/// <param ="an">the part of a numerical series</param>

/// <returns>total amount</returns>

static double countSE(double e, double x)

{

double amount = 0;

double an = x;

int n = 0;

while (Math.Abs(an) > e)

{

amount += an;

n += 1;

an \*= (x \* x \* x \* x) / (double)(4 \* n + 1);

};

return amount;

}

/// <summary>

/// count f(x)

/// </summary>

/// <param ="f">math function</param>

/// <returns>the value of the function at the point x</returns>

static double countY(double x)

{

double f = Math.Log((1 + x) / (1 - x)) / 4 + Math.Atan(x) / 2;

return f;

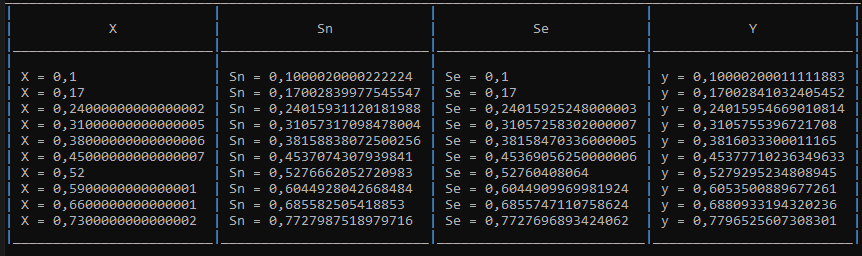
}

}

}

## Результаты работы программы

В результате работы программы получается следующая таблица (рис. 1.5.1).



***Рис. 1.5.1. Выходные данные программы***

Проанализировав полученные результаты, могу сделать вывод, что при разных способах подсчета результату незначительно разнятся. Это происходить в результате работы с дробными числами и разными подходами к вычислению.