МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**КУРСОВА РОБОТА**

з дисципліни «Алгоритмізація та програмування»

Варіант №

|  |  |  |
| --- | --- | --- |
| **Керівник:** |  | **Виконав:** |
| Допущено до захисту |  |
| І\_\_\_І \_\_\_\_\_\_\_\_\_ 2022 \_\_\_\_\_\_\_\_\_\_\_  підпис |  |
| Захищено з оцінкою  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_  оцінка підпис  І\_\_\_І \_\_\_\_\_\_\_\_\_ 2022 |  |

Київ-2022

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Факультет | ***біомедичної інженерії*** | | | | |
|  | (назва факультету, інституту) | | | | |
| Кафедра | ***біомедичної кібернетики*** | | | | |
|  | (назва кафедри) | | | | |
| Дисципліна | ***«Алгоритмізація та програмування»*** | | | | |
|  | (назва) | | | | |
| Курс | ***2*** | Група |  | Семестр | ***3*** |

**ЗАВДАННЯ**

**на курсовий проект (роботу) студента**

|  |
| --- |
| ***Романюка Михайла Дмитровича*** |
| (прізвище, ім’я, по батькові) |

.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. Тема роботи: | | ***Програмно-алгоритмічна реалізація наближеного обчислення*** | | | |
| ***значення функції через її розклад в степеневий ряд.*** | | | | | |
|  | | | | | |
|  | |  | | | |
| 2. Строк здачі студентом закінченого проекту (роботи) | | | | | ***09.12.2022*** |
|  | |  | | | |
| 3. Вихідні дані до проекту (роботи): | | | | ***Варіант №*** | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | |  | | | |
| 4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають | | | | | |
| розробці): | ***1. Організація діалогу з користувачем і управління процесом об-*** | | | | |
| ***числень. 2. Розробка коду для реалізації розрахунків значення функції через її*** | | | | | |
| ***розклад в степеневий ряд, функція sin(x)/x. 3. Візуалізація отриманих*** | | | | | |
| ***результатів.*** | | | | | |
|  | | | | | |
|  | | | | | |
|  | |  | | | |
| 5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): | | | | | |
| ***блок-схеми, графіки дослідження збіжності ряду та дослідження точності*** | | | | | |
| ***наближеного обчислення значень функції через її розклад у степеневий ряд.*** | | | | | |
|  | |  | | | |
| 6. Дата видачі завдання: | | | ***16.09.2022*** | | |

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| **№п/п** | **Назва етапів курсового проекту (роботи) та питань, які мають бути розроблені відповідно до завдання** | **Термін виконання етапу** | **Позначки керівника про виконання завдань** |
| 1. | Отримання завдання на курсову роботу | 16.09.2022 |  |
| 2. | Огляд технічної літератури за темою роботи | 23.09.2022 |  |
| 3. | Розробка першої частини курсової роботи (організацію діалогу з користувачем і управління процесом обчислень) | 14.10.2022 |  |
| 4. | Перший контроль за процесом виконання курсової роботи, консультація у викладача | 21.10.2022 |  |
| 5. | Розробка другої частини курсової роботи (розробку коду для реалізації розрахунків значення функції через її розклад в степеневий ряд) | 11.11.2022 |  |
| 6. | Другий контроль за процесом виконання курсової роботи, консультація у викладача | 18.11.2022 |  |
| 7. | Розробка третьої частини курсової роботи (візуалізація отриманих результатів) | 25.11.2022 |  |
| 8. | Оформлення пояснювальної записки | 09.12.2022 |  |
| 9. | Захист курсової роботи | 23.12.2022 |  |

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(підпис)

**Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(підпис) (прізвище ім'я, по батькові)

**«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 р.**

**Анотація**

ПІБ. Програмно-алгоритмічна реалізація наближеного обчислення значення функції через її розклад в степеневий ряд.

Курсова робота з дисципліни «Алгоритмізація та програмування» присвячена питанню чисельного обчислення значення аналітичної функції шляхом підсумовування наявних значень для цих функцій у степеневих рядах. У курсовій роботі була виконана програмно-алгоритмічна реалізація обчислення значення функції для дослідження швидкості збіжності ряду і точності одержуваних значень функції.

**Структура і обсяг роботи.** Курсова робота складається із вступу, трьох розділів, висновків і списку використаної літератури з 4 джерел. Загальний обсяг роботи становить 37 сторінок, основного тексту (без додатків) – 37 сторінок, ілюстрацій – 22, таблиць – 2.

**Annotation**

Surname Name. Software and algorithmic implementation for approximate calculation of the function value through its expression as power series.

The course work on the subject "Algorithms and Programming" is devoted to the issue of numerical calculation of the analytical function through summing the available values for these functions in power series. In the course work, a software and algorithmic implementation of the calculation of the function value was performed to study the rate of the series convergence and the accuracy of the obtained function values.

**The structure and the amount of work.** Coursework consists of an introduction, three partitions, conclusions and list of used literature with 4 references. The total volume of coursework is 37 pages, main text (without applications) – 37 pages, illustrations – 22, tables – 2.

**ЗМІСТ**

[**ВСТУП 6**](#_Toc122521387)

[**РОЗДІЛ 1 7**](#_Toc122521388)

[**ВИВІД ФОРМУЛИ ОБРАХУВАННЯ 7**](#_Toc122521389)

[**1.1.Вивід рекурентної формули 7**](#_Toc122521390)

[**1.2. Блок-схеми програмного застосунку 8**](#_Toc122521391)

[**РОЗДІЛ 2 19**](#_Toc122521392)

[**ПРОГРАМНА РЕАЛІЗАЦІЯ 19**](#_Toc122521393)

[**2.1. Лістинг 19**](#_Toc122521394)

[**2.2. Результати роботи програмного застосунку 26**](#_Toc122521395)

[**РОЗДІЛ 3 31**](#_Toc122521396)

[**ДОСЛІДЖЕННЯ ЗБІЖНОСТІ РЯДУ І ТОЧНОСТІ ОБРАХУНКУ 31**](#_Toc122521397)

[**3.1. Аналіз збіжності ряду 31**](#_Toc122521398)

[**3.2. Аналіз точності обчислення значень функції 33**](#_Toc122521399)

[**ВИСНОВКИ 36**](#_Toc122521400)

[**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 37**](#_Toc122521401)

# **ВСТУП**

Опираючись на сучасну моду до автоматизації процесів у всіх видах діяльності людини, можна зрозуміти, що машини виконують певні завдання набагато ефективніше за людей. Вони їх виконують завдяки алгоритмам, які написані людиною. Ці алгоритми впливають на наше життя, де б ми не знаходились і що б не робили. Так, наприклад, жодна людина не зрівняється з алгоритмами аналізу даних, які вже стали невід’ємною частиною нашого світу і улюбленим знаряддям корпорацій.

Важливим аспектом роботи більшості інженерно-технічних алгоритмів є використання функцій, законів та формул вищої математики. Необхідні обрахунки можна провести і власноруч, але це займе дуже велику кількість часу, у порівнянні з алгоритмом. У цьому і полягає мета та актуальність роботи – замінити декілька хвилин ручного обрахунку функції, через розкладання у ряд Тейлора, на долі секунд роботи алгоритму.

Відповідно до мети, можна визначити задачі роботи:

* Реалізувати алгоритм розрахунку функції, шляхом розкладу у ряд Тейлора;
* Порівняти точність обрахунків алгоритму з вбудованою функцією.
* Проаналізувати отримані значення;

# **РОЗДІЛ 1**

# **ВИВІД ФОРМУЛИ ОБРАХУВАННЯ**

## 1.1.Вивід рекурентної формули

Вивід рекурентної формули функції можна отримати, завдяки ряду Тейлора. Формула розкладу функції у ряд Тейлора:

 (1)

Проаналізувавши формулу (1), можна дійти висновку, що елемент ряду дорівнює попередньому елементу, помноженому на рекурентний множник:

 (2)



, де

– елементи ряду

– рекурентний множник

– значення аргументу функції

– номер елементу

Підставивши рекурентний множник у формулу (2), отримаємо рекурентну формулу, за якою будемо проводити обрахунки у програмному застосунку:



Треба зазначити, що функція  не існує в точці.

## 1.2. Блок-схеми програмного застосунку

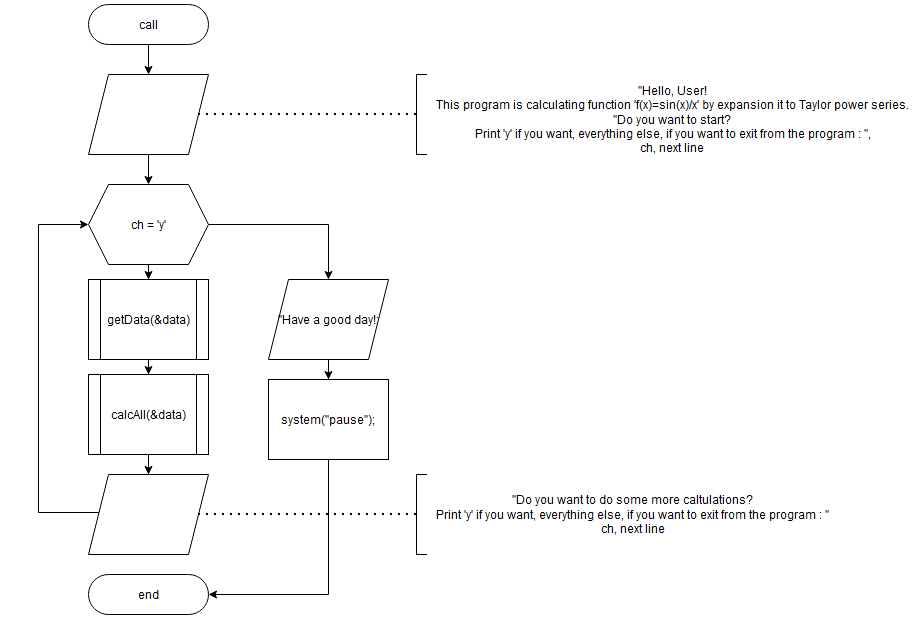


Рис. 1.1 Блок-схема функції main

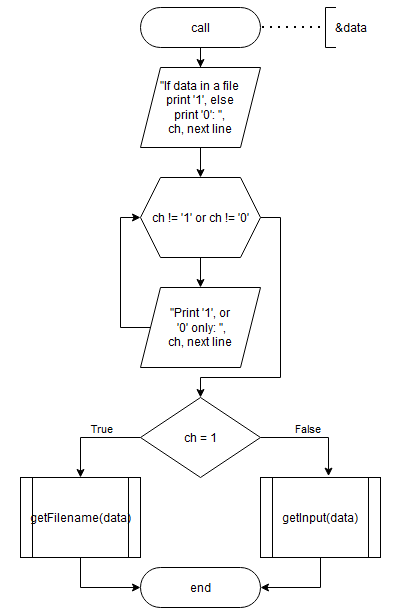


Рис. 1.2 Блок-схема функції getData

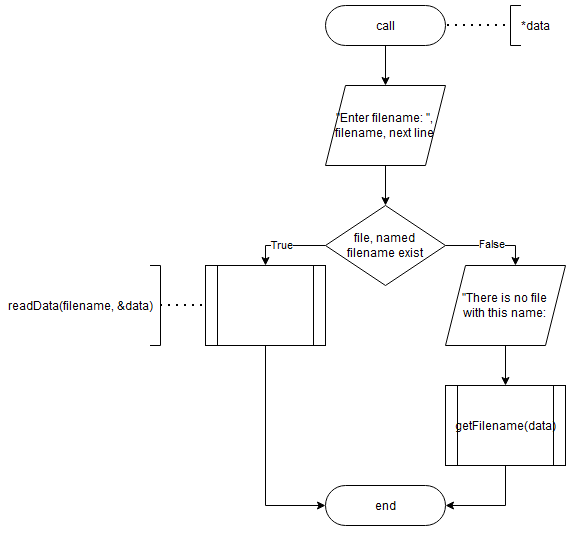


Рис. 1.3 Блок-схема функції getFilename

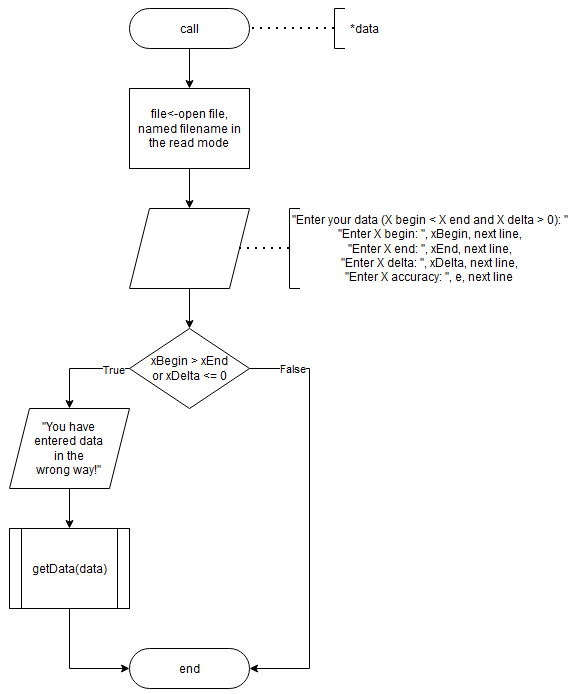


Рис. 1.4 Блок-схема функції getIntup

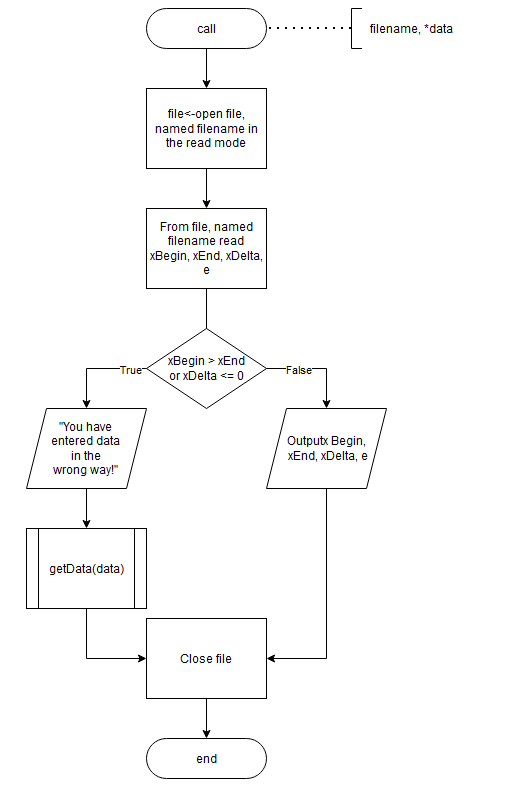


Рис. 1.5 Блок-схема функції readData

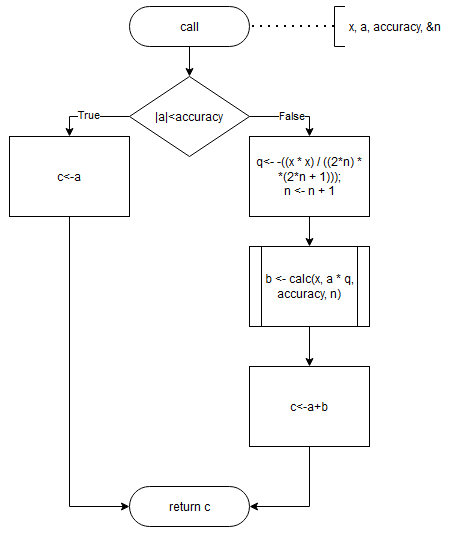


Рис. 1.6 Блок-схема функції calculate

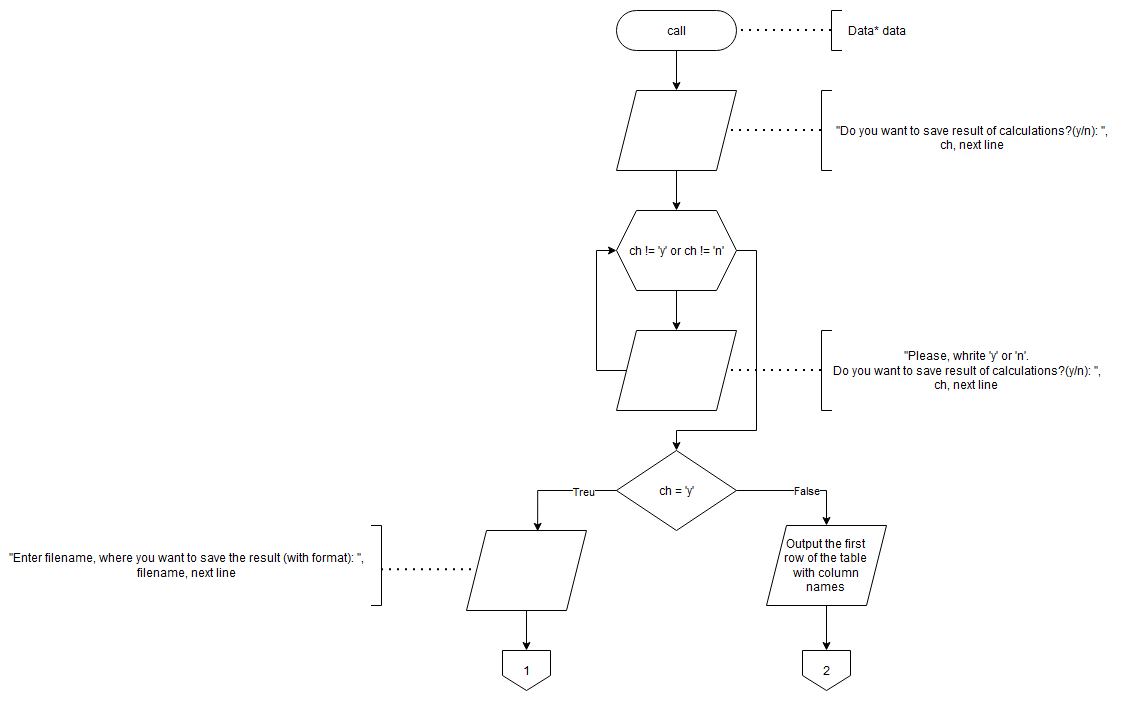


Рис. 1.7 Блок-схема функції calcAll частина 1

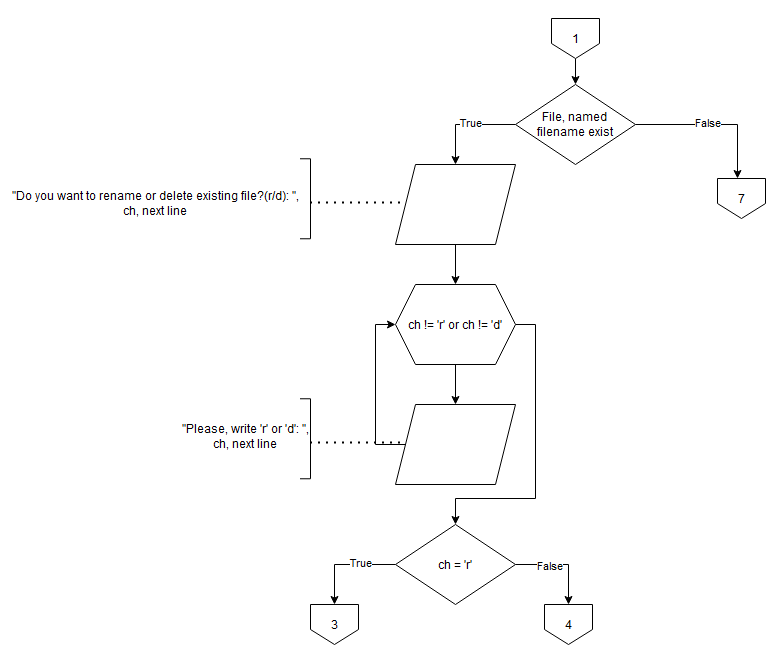


Рис. 1.8 Блок-схема функції calcAll частина 2

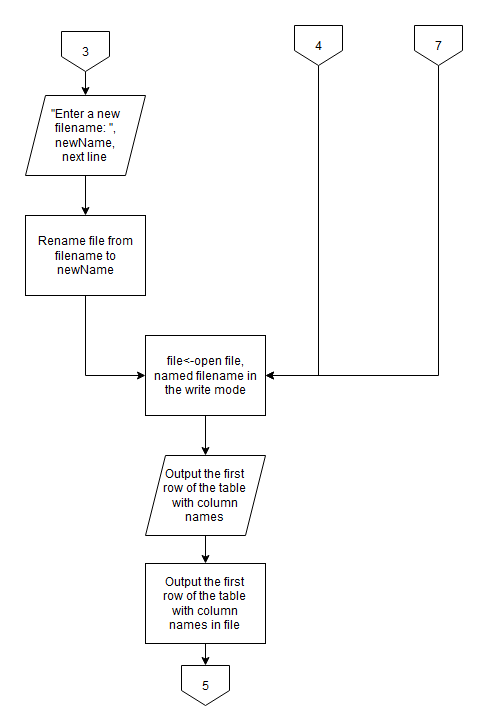


Рис. 1.9 Блок-схема функції calcAll частина 3

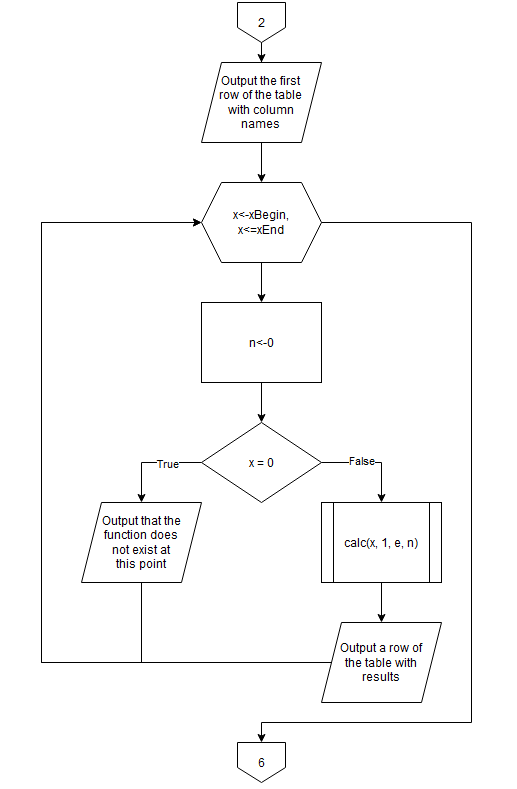


Рис. 1.10 Блок-схема функції calcAll частина 4

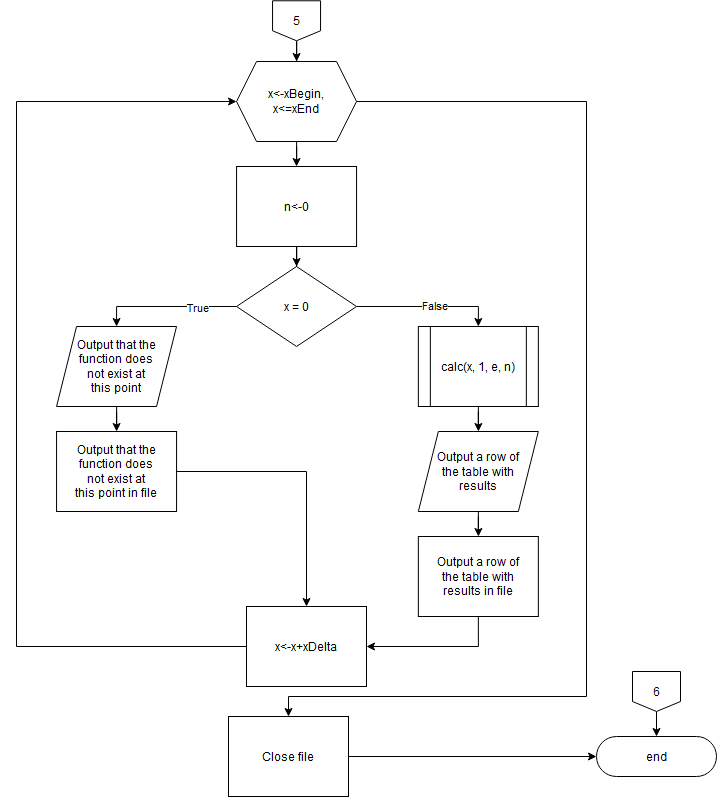


Рис. 1.11 Блок-схема функції calcAll частина 5

# **РОЗДІЛ 2**

# **ПРОГРАМНА РЕАЛІЗАЦІЯ**

## 2.1. Лістинг

/\* БС-14 Романюк Михайло

\* Реалізація розрахунків значення функції sin(x)/x

\* через розклад у степеневий ряд функції.

\* Для коректної роботи програми дані мають бути розташовані у

\* відповідному порядку: початок інтервалу -> кінець інтервалу -> крок інтервалу -> точність обрахунку;

\* в 1 строчку через пробіл. Наприклад, "-1 1 0.05 0.000001".

\*/

#include <iostream>

#include <cmath>

#include <cstdio>

#include <cstring>

#include <io.h>

#pragma warning( disable : 4996 )

#define \_CRT\_SECURE\_NO\_WARNINGS

using namespace std;

struct Data

{

/\* Структура для зберігання та передачі даних, необхідних для обрахунків фенкції на інтервалі

\* xBegin - початок інтервалу

\* xEnd - кінець інтервалу

\* xDelta - крок інтервалу

\* e - точність обрахунку \*/

double xBegin;

double xEnd;

double xDelta;

double e;

};

void readData(char[], Data\*);

void getData(Data\*);

double calcAndOut(Data);

void getFilename(Data\*);

void calcAll(Data\* data);

void getInput(Data\*);

double calc(double, double, double, int&);

//-----------------------------------------------------------------------------------------------------------------------------------------------

void main()

{

/\* Функція, що починає діалог з користувачем i викликає

\* необхідні функції для подальшої роботи програми \*/

char choice;

Data data;

printf("Hello, User!\nThis program is calculating function 'f(x)=sin(x)/x' by expansion it to Taylor power series.\n");

printf("Do you want to start? \nPrint 'y' if you want, everything else if you want to exit from the program : ");

cin >> choice;

while (choice == 'y')

{

getData(&data);

calcAll(&data);

printf("Do you want to do some more caltulations?\nPrint 'y' if you want, everything else if you want to exit from the program : ");

cin >> choice;

}

printf("\n\nHave a good day!\n\n");

system("pause");

}

//-----------------------------------------------------------------------------------------------------------------------------------------------

void getData(Data\* data)

{

/\* Функція питає в користувача, звідки буде вводитись інформація

\* Data\* data - структура з інформацією \*/

printf("If data in a file print '1', else print '0': ");

char ch;

cin >> ch;

while ((ch != '1') && (ch != '0'))

{

printf("Print '1', or '0' only: ");

cin >> ch;

}

if (ch == '1')

getFilename(data);

else

getInput(data);

}

//-----------------------------------------------------------------------------------------------------------------------------------------------

void getFilename(Data\* data)

{

/\* Рекурсивна функція, що бере в користувача назву файлу, в якому зберігається

\* інформація, необхідна для розрахунків.

\* У випадку існування файлу, відбувається виклик функції "readData(name, data)".

\* У випадку, коли вказаного файлу не існує, функція викликає саму себе, щоб

\* користувач повторно ввів назуву файлу.

\* Data\* data - структура для зберігання інформації \*/

char filename[256];

FILE\* file;

printf("Enter filename: ");

scanf("%s", filename);

file = fopen(filename, "r");

if (file)

readData(filename, data);

else

{

printf("There is no file with this name\n");

getFilename(data);

}

}

//-----------------------------------------------------------------------------------------------------------------------------------------------

void getInput(Data\* data)

{

/\* Функція бере в користувача дані для обрахунків

\* Data\* data - структура для зберігання інформації \*/

printf("Enter your data (X begin < X end and X delta > 0):\n");

printf("Enter X begin: ");

scanf("%lf", &data->xBegin);

printf("Enter X end: ");

scanf("%lf", &data->xEnd);

printf("Enter X delta : ");

scanf("%lf", &data->xDelta);

printf("Enter accuracy: ");

scanf("%lf", &data->e);

if ((data->xBegin > data->xEnd) || (data->xDelta <= 0)) // перевірка на коректність введених даних

{

printf("Your data is wrong. Check it and come back please.\n");

getData(data);

}

}

//-----------------------------------------------------------------------------------------------------------------------------------------------

void readData(char filename[], Data\* data)

{

/\* Функція бере з файлу інформацію

\* та передає в структуру "Data"

\* char filename[] - назва файлу

\* Data\* data - структура для зберігання інформації \*/

FILE\* file = fopen(filename, "r");

fscanf(file, "%lf", &data->xBegin);

fscanf(file, "%lf", &data->xEnd);

fscanf(file, "%lf", &data->xDelta);

fscanf(file, "%lf", &data->e);

if ((data->xBegin > data->xEnd) || (data->xDelta <= 0)) // перевірка на коректність введених даних

{

printf("Your data is wrong. Check it and come back please.\n");

getData(data);

}

else

printf("Your input:\nX begin = %f \nX end = %f \nX delta = %f \nAccuracy = %f \n", data->xBegin, data->xEnd, data->xDelta, data->e);

}

//-----------------------------------------------------------------------------------------------------------------------------------------------

void calcAll(Data\* data)

{

/\* Функція, що проводить обрахунки та виводить на екран,

\* або на екран і у файл таблицю з результатами обрахунку.

\* Data\* data - аргумент, в якому збережені дані для обрахунків \*/

printf("Do you want to save result of calculations?(y/n): ");

char ch;

cin >> ch;

while ((ch != 'y') && (ch != 'n'))

{

printf("Please, whrite 'y' or 'n'.\nDo you want to save result of calculations?(y/n): ");

cin >> ch;

}

if (ch == 'y')

{

printf("Enter filename, where you want to save the result (with format): ");

char filename[256];

cin >> filename;

if (!access(filename, 0)) // перевірка, чи існує файл з назвою filename

{

printf("Do you want to rename or delete existing file?(r/d): ");

cin >> ch;

while ((ch != 'r') && (ch != 'd'))

{

printf("Please, write 'r' or 'd': ");

cin >> ch;

} // кінець 'while ((ch != 'r') && (ch != 'd'))'

if (ch == 'r')

{

printf("Enter a new filename: ");

char newName[256];

cin >> newName;

rename(filename, newName);

} // кінець ' if (ch == 'r')'

} // кінець 'if (!access(filename, 0))'

FILE\* file = fopen(filename, "w");

printf("===================================================================================================\n");

printf("| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "x", "series(x)", "n", "f(x)", "Diff");

printf("===================================================================================================\n");

fprintf(file, "===================================================================================================\n");

fprintf(file, "| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "x", "series(x)", "n", "f(x)", "Diff");

fprintf(file, "===================================================================================================\n");

for (double x = data->xBegin; x <= data->xEnd; x += data->xDelta)

{

int n = 0;

if (float(abs(x)) == 0) //умова, при виконанні якої ввжається, що х = 0 і функція не обраховується

{

printf("| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "0", "---", "---", "---", "---");

printf("====================================================================================================\n");

fprintf(file, "| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "0", "---", "---", "---", "---");

fprintf(file, "====================================================================================================\n");

} // кінець 'if (abs(x) <= 1e-6)'

else

{

double res = calc(x, 1, data->e, n);

double fx = sin(x) / x;

printf("| % 8.4lf | % 24.10lf | % 4d | % 24.10lf | % 24.10lf |\n", x, res, n, fx, abs(res - fx));

printf("====================================================================================================\n");

fprintf(file, "| % 8.4lf | % 24.10lf | % 4d | % 24.10lf | % 24.10lf |\n", x, res, n, fx, abs(res - fx));

fprintf(file, "====================================================================================================\n");

} // кінець 'else' від 'if (abs(x) <= 1e-6)'

} // кінець 'for (double x = data->xBegin; x <= data->xEnd; x += data->xDelta)'

fclose(file);

} // кінець "if (ch == 'y')"

else

{

printf("====================================================================================================\n");

printf("| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "x", "series(x)", "n", "f(x)", "Diff");

printf("====================================================================================================\n");

for (double x = data->xBegin; x <= data->xEnd; x += data->xDelta)

{

int n = 0;

if (float(abs(x)) == 0) //умова, при виконанні якої ввжається, що х = 0 і функція не обраховується

{

printf("| % 8s | % 24s | % 4s | % 24s | % 24s |\n", "0", "---", "---", "---", "---");

printf("====================================================================================================\n");

} // кінець 'if (abs(x) <= 1e-6)'

else {

double res = calc(x, 1, data->e, n);

double fx = sin(x) / x;

printf("| % 8.4lf | % 24.10lf | % 4d | % 24.10lf | % 24.10lf |\n", x, res, n, fx, abs(res - fx));

printf("====================================================================================================\n");

} // кінець 'else' від 'if (abs(x) <= 1e-6)'

} // кінець 'for (double x = data->xBegin; x <= data->xEnd; x += data->xDelta)'

} // кінець 'else' від 'if (ch == 'y')'

} // кінець "calcAndOut(Data\* data)"

//-----------------------------------------------------------------------------------------------------------------------------------------------

double calc(double x, double a, double accuracy, int& n)

{

/\* Рекурсивна функція, що проводить обрахунки функції через степеневий ряд:

\* x - значення x

\* a - член ряду

\* accuracy - точність

\* n - кількість кроків \*/

if (abs(a) < accuracy)

return a;

n += 1;

double q = -((x \* x) / ((2 \* n) \* (2 \* n + 1))); // рекурентний множник

return a + calc(x, a \* q, accuracy, n);

}

## 2.2. Результати роботи програмного застосунку

Одразу після запуску програми, користувача зустрічає консольний інтерфейс, який проводить з ним діалог (рис. 2.1). Після уточнення того, чи хоче користувач проводити розрахунки ініціалізується структура «Data», яка буде зберігати в собі необхідні для обрахунків значення, а саме: xBegin -початкове значення проміжку; xEnd – кінцеве значення проміжку; xDelta - крок проміжку; e – точність, з якою буде обрахований ряд.

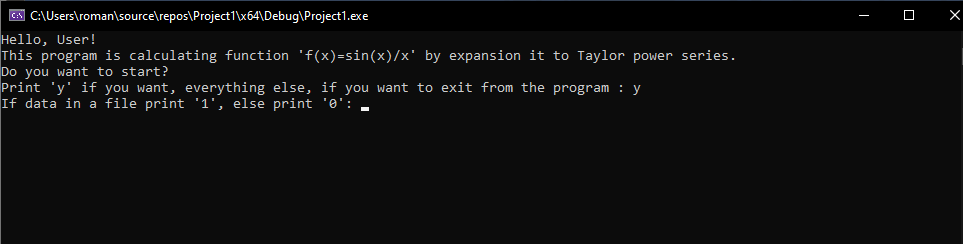


Рис. 2.1 Початок діалогу між користувачем та програмою

Задати необхідні для розрахунків значення можна, як з клавіатури так і з файлу.

Розглянемо ввід даних з файлу. Для цього слід йти за інструкціями, які надає консольний інтерфейс і у моменті, який зображено на рис.2.1 ввести «1» (рис.2.2).

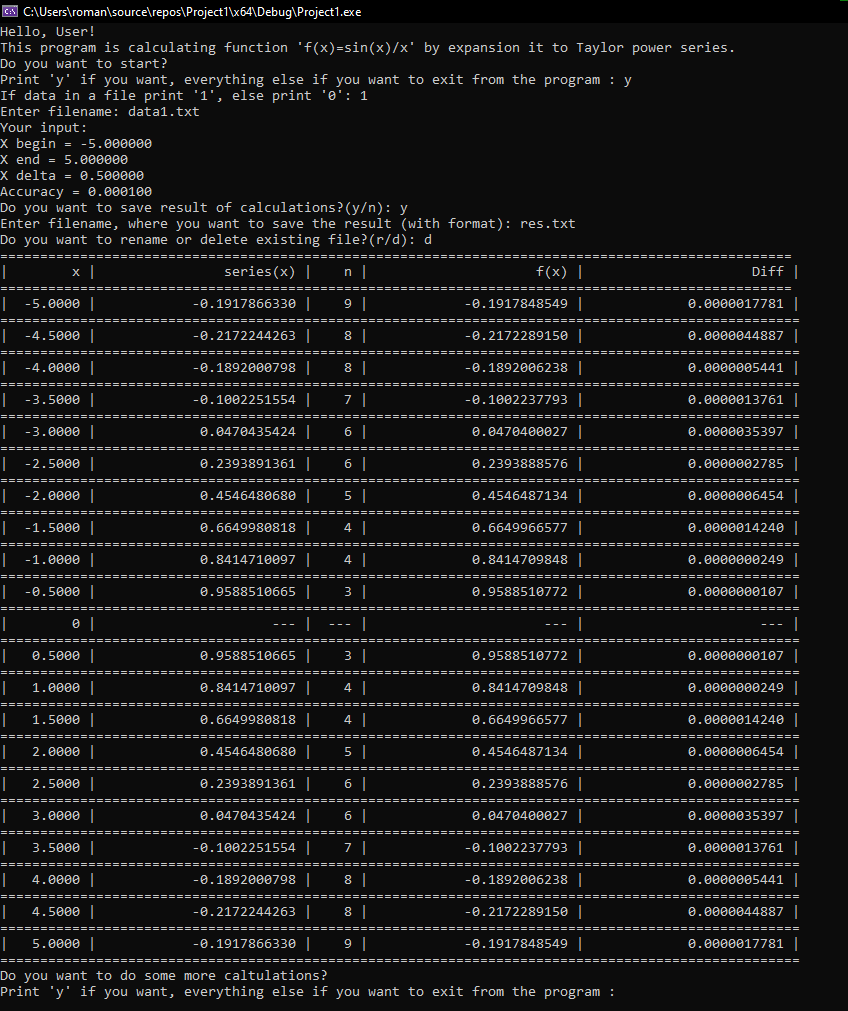


Рис. 2.2 Обрахунки з вводом даних з файлу

Як ми бачимо, то у програмному застосунку перевіряється чи існує файл з даними. У випадку, коли його немає, користувача просять ввести іншу назву. Сам файл має містити 4 значення у відповідному порядку (xBegin xEnd xDelta e), записаних у один рядок через пробіл (рис.2.3).

Відомо, що функція не існує в точці . Щоб програмний застосунок працював коректно, була зроблена перевірка, чи .

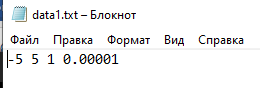


Рис. 2.3 Вміст файлу з даними

Також користувачу пропонується зберегти таблицю результатів обрахунку у файл (рис.2.4). Якщо файл вже існує, то пропонується або видалити, або перейменувати файл. Відповідно до вибору, існуючий файл або видаляється, або перейменовується на назву, яку введе користувач.

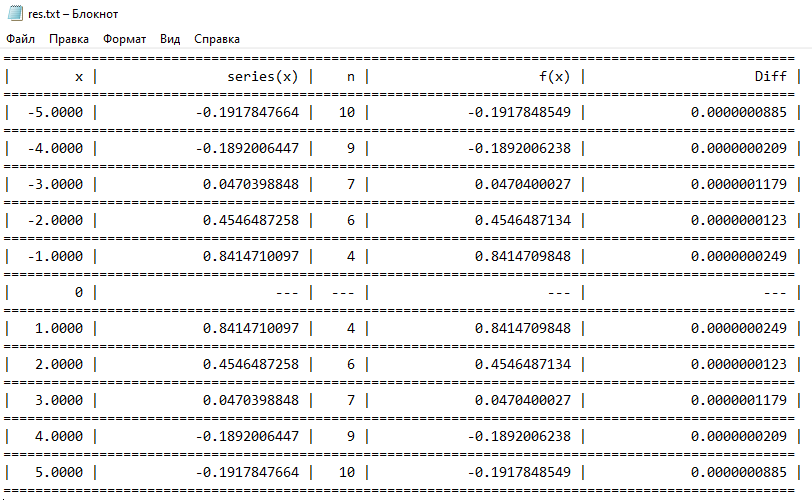


Рис. 2.4 Вміст файлу з результатами обрахунку

Після обрахунків, консольний інтерфейс програмного застосунку пропонує провести ще одні обрахунки (рис.2.5).



Рис. 2.5 Пропозиція прорахувати інші значення

Скористаємось цією пропозицією та спробуємо другий варіант вводу даних – через клавіатуру. Цього разу не будемо зберігати результати обрахунку у файл (рис.2.6).

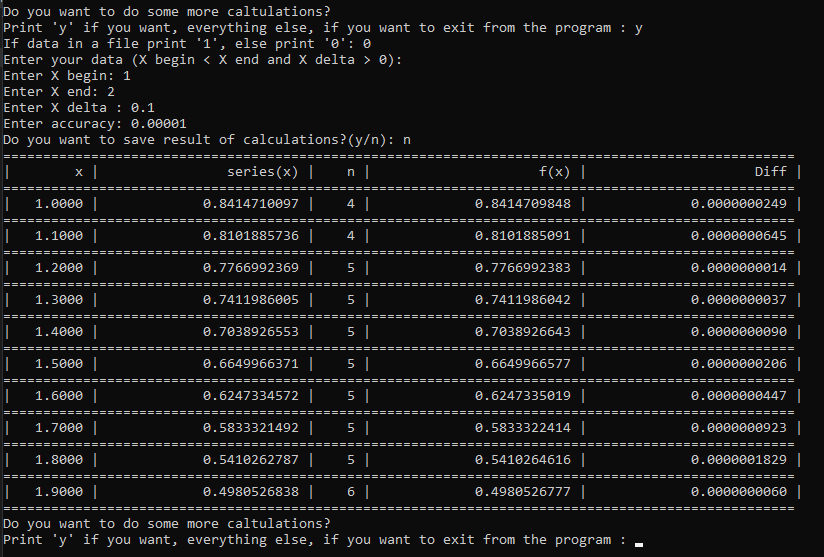


Рис. 2.6 Обрахунки з вводом даних з клавіатури

Зробивши необхідні обрахунки, можемо завершувати виконання програмного застосунку, для цього, відповідно до наданих консольним інтерфейсом інструкцій, введемо будь-що, окрім «y» (рис. 27).

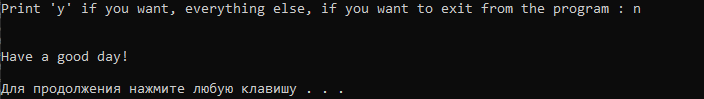


Рис. 2.7 Завершення виконання програмного застосунку

# **РОЗДІЛ 3**

# **ДОСЛІДЖЕННЯ ЗБІЖНОСТІ РЯДУ І ТОЧНОСТІ ОБРАХУНКУ**

## 3.1. Аналіз збіжності ряду

Щоб дослідити збіжність ряду, подивимось на залежність кількості членів ряду функції від точності. Для цього зробимо розрахунки два рази, змінивши тільки точність наближення. Оберемо для цього значення: -5 – для початкового значення; 5 – для кінцевого значення ; 0,5 – для кроку ряду; 0,0001 – точність наближення першого розрахунку; 0,01 – точність наближення другого розрахунку. Кількість членів ряду в точці будемо вважати рівним 0.

*Таблиця 3.1*

**Залежність кількості членів ряду функції від точності**

|  |  |  |
| --- | --- | --- |
| x | Кількість кроків при точності е | |
| е = 0.01 | е = 0.0001 |
| -5.0 | 7 | 9 |
| -4.5 | 7 | 8 |
| -4.0 | 6 | 8 |
| -3.5 | 5 | 7 |
| -3.0 | 5 | 6 |
| -2.5 | 4 | 6 |
| -2.0 | 4 | 5 |
| -1.5 | 3 | 4 |
| -1.0 | 2 | 4 |
| -0.5 | 2 | 3 |
| 0.0 | 0 | 0 |
| 0.5 | 2 | 3 |
| 1.0 | 2 | 4 |
| 1.5 | 3 | 4 |
| 2.0 | 4 | 5 |
| 2.5 | 4 | 6 |
| 3.0 | 5 | 6 |
| x | Кількість кроків при точності е | |
| е = 0.01 | е = 0.0001 |
| 3.5 | 5 | 7 |
| 4.0 | 6 | 8 |
| 4.5 | 7 | 8 |
| 5.0 | 7 | 9 |

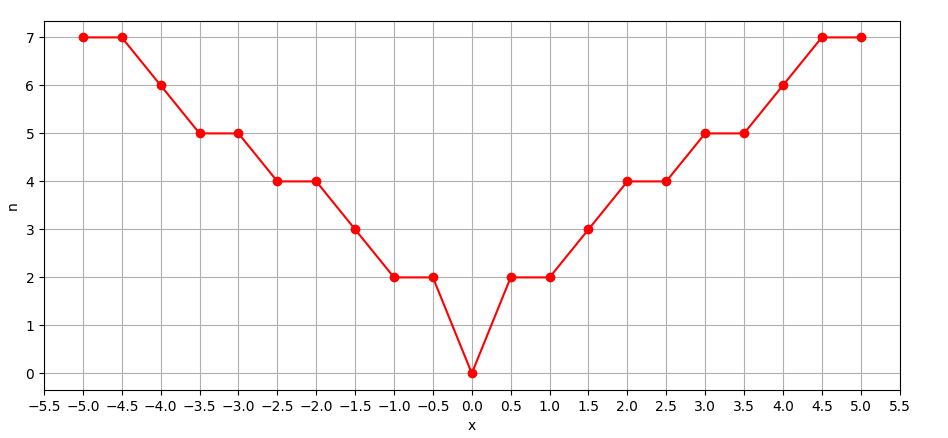
******

Рис. 3.1 Залежність кількості членів ряду від *х* при точності

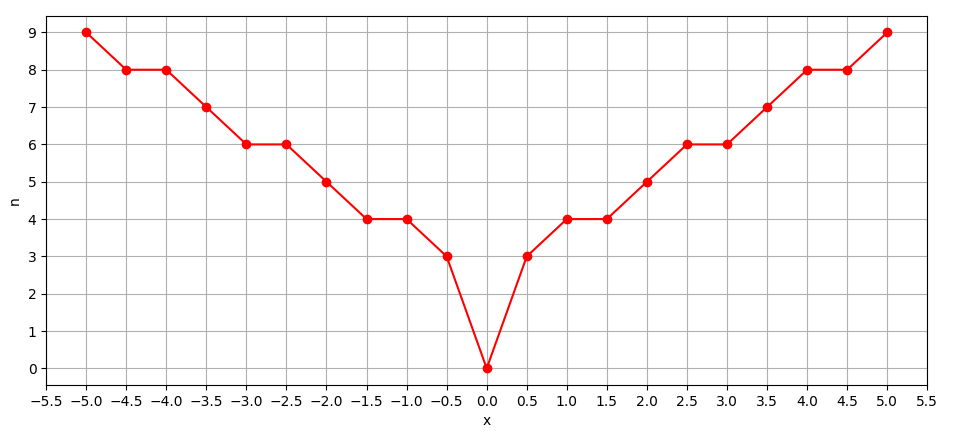


Рис. 3.2 Залежність кількості членів ряду від *х* при точності

З таблиці 3.1.1 та рис.3.1.1, рис. 3.1.2, можна дійти висновку, що при більшій точності необхідно обрахувати більшу кількість елементів ряду для її досягнення. Також видно, що при збільшенні модуля *х*, також збільшується необхідна кількість обрахунків. Числа, що рівні за модулем, мають однакову кількість членів ряду, необхідних для обрахунків з однаковою точністю.

## 3.2. Аналіз точності обчислення значень функції

Для наочного зображення відхилення значень, знайдених рекурсивним методом, від значень вбудованої функції, можемо зобразити їх на одному графіку.

*Таблиця 3.2*

**Порівняння значень функцій**

|  |  |  |  |
| --- | --- | --- | --- |
| х | Значення вбудованої функції | Значення функції при рекурсивному обрахунку ряду з точністю е | |
| е=0.01 | е=0.0001 |
| -5.0 | -0.1917848549 | -0.1921842681 | -0.1917866330 |
| -4.5 | -0.2172289150 | -0.2173039199 | -0.2172244263 |
| -4.0 | -0.1892006238 | -0.1890068779 | -0.1892000798 |
| -3.5 | -0.1002237793 | -0.1007361701 | -0.1002251554 |
| -3.0 | 0.0470400027 | 0.0469581981 | 0.0470435424 |
| -2.5 | 0.2393888576 | 0.2396184797 | 0.2393891361 |
| -2.0 | 0.4546487134 | 0.4546737213 | 0.4546480680 |
| -1.5 | 0.6649966577 | 0.6649274554 | 0.6649980818 |
| -1.0 | 0.8414709848 | 0.8416666667 | 0.8414710097 |
| -0.5 | 0.9588510772 | 0.9588541667 | 0.9588510665 |
| 0.0 | --- | --- | --- |
| 0.5 | 0.9588510772 | 0.9588541667 | 0.9588510665 |
| 1.0 | 0.8414709848 | 0.8416666667 | 0.8414710097 |
| 1.5 | 0.6649966577 | 0.6649274554 | 0.6649980818 |
| 2.0 | 0.4546487134 | 0.4546737213 | 0.4546480680 |
| 2.5 | 0.2393888576 | 0.2396184797 | 0.2393891361 |
| х | Значення вбудованої функції | Значення функції при рекурсивному обрахунку ряду з точністю е | |
| е=0.01 | е=0.0001 |
| 3.0 | 0.0470400027 | 0.0469581981 | 0.0470435424 |
| 3.5 | -0.1002237793 | -0.1007361701 | -0.1002251554 |
| 4.0 | -0.1892006238 | -0.1890068779 | -0.1892000798 |
| 4.5 | -0.2172289150 | -0.2173039199 | -0.2172244263 |
| 5.0 | -0.1917848549 | -0.1921842681 | -0.1917866330 |

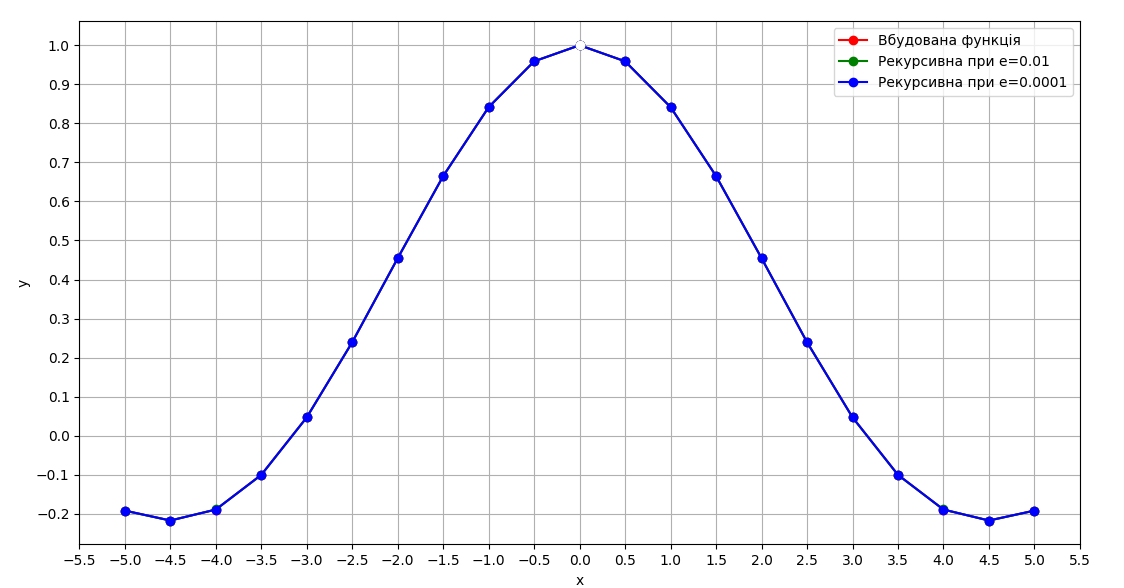


Рис.3.3 Зображення функцій на одному графіку

Видно, що при такому віддалені функції перекривають одна-одну на графіку. Приблизимо, щоб подивитись поближче.

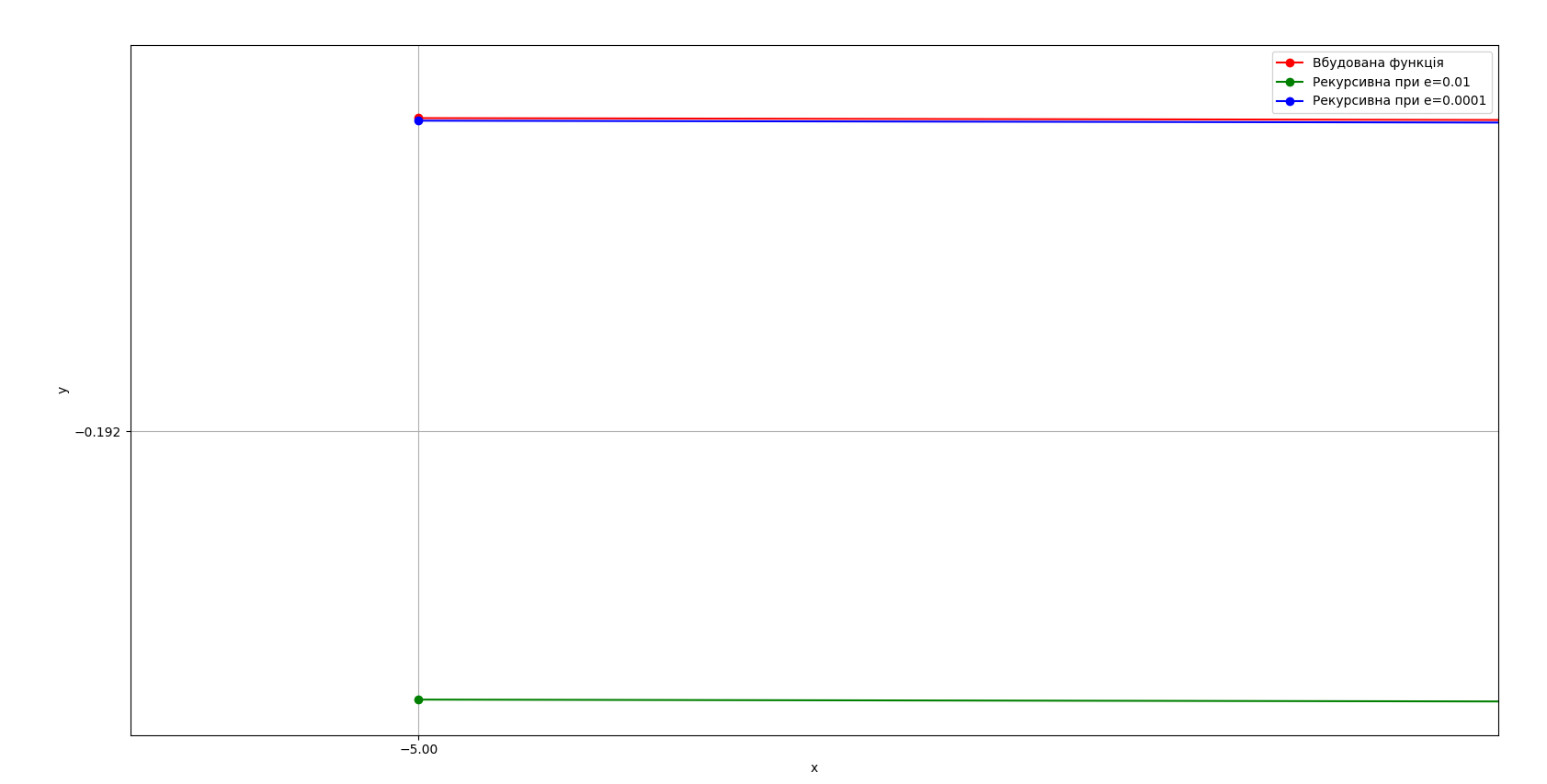


Рис.3.4 Наближене зображення функцій на одному графіку

Можна побачити, що з таким великим наближенням, при точності 0.01, різниця між вбудованою функцією доволі помітна, в той момент, як при точності 0.0001, графіки майже співпадають.

# **ВИСНОВКИ**

Аналізуючи хід виконання роботи, можна зрозуміти, що ми провели велику кількість часу, розробляючи програмний застосунок для реалізації обрахунків функції, шляхом її розкладу у степеневий ряд та навчилися спрощувати формули степеневих рядів. Завдяки рекурентному множнику, ми зменшили кількість необхідних арифметичних дій, тим самим оптимізувавши розрахунковий процес.

Важливим аспектом виконання завдання була мова програмування, а саме – С++. Завдяки якій розрахунки проводяться швидко і ефективно, з точки зору ресурсозатратності процесу. Також ця мова надає всі необхідні можливості для реалізації консольного інтерфейсу та збереження результатів.

Виходячи з результатів обрахунку, можна сказати, що реалізований програмний застосунок гарно виконує поставлені завдання і відповідає поставленим вимогам. І найголовніше це те, що він позбавлений від будь-яких факторів, які можуть впливати на коректність роботи програми, на відміну від розрахунків, які проводяться людьми. Відсутність людського фактору під час виконання обрахунків і є найголовнішим плюсом алгоритмів. Але ще залишається можливі помилки, спричинені цим фактором, у процесі розробки програмного застосунку.

Розробка алгоритму рекурсивного обрахунку степеневого ряду Тейлора виявилась доволі цікавим досвідом, оскільки ми з довгого арифметичного обрахунку зробили автоматичний рекурсивний обрахунок з приємним та логічним консольним інтерфейсом, який дає змогу користувачу без проблем зрозуміти, як користуватися цим програмним застосунком.

# **СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Алгоритмізація та програмування мовою високого рівня С++ : комп’ютерний практикум / С. М. Алхімова. - Київ : КПІ ім. Ігоря Сікорського, Вид-во Політехніка», 2018. - 156 с.
2. Уроки програмування на С++ [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: <https://acode.com.ua/uroki-po-cpp/>.
3. C++ reference [Електронний ресурс] – Режим доступу до ресурсу: <https://en.cppreference.com/w/cpp>.
4. Бакун, В. В. Математичний аналіз. Частина ІІІ. Числові й функціональні ряди. Інтеграли, залежні від параметра [Електронний ресурс] : підручник у 3-х ч. / В. В. Бакун ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,23 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2021. – 435 с. – Назва з екрана.