**Національний технічний університет України**
**“Київський політехнічний інститут ім. Ігоря Сікорського”**

**Факультет прикладної математики**

**Кафедра системного програмування і спеціалізованих комп’ютерних систем**

# ЛАБОРАТОРНА РОБОТА №1

**з дисципліни**

**“Алгоритми та методи обчислень”**

**Группа: КВ-33**
**Виконав: Козлов С. О.**
**Оцінка:**

Київ - 2025

# Лабораторна робота 1
# ОБЧИСЛЕННЯ ЗНАЧЕНЬ ФУНКЦІЇ

## Мета роботи

Дослідити особливості наближення функцій за допомогою степеневих рядів, виконати аналіз похибок наближення.

## Загальне завдання

1. Побудувати таблицю залежності довжини ряду n, що забезпечує точність функції не меншу за задане значення eps у точці x = (b + a)/2, від eps.
   Значення eps змінюється від $10^{-2}$ до $10^{-14}$ з кроком $10^{-3}$.
2. Для n (довжина ряду фіксована й дорівнює n), отриманого в п.1 при eps = 10-8 , у точках xi = a + hi, h = (b – a)/10, i = 0, ..., 10 обчислити абсолютну похибку та залишковий член ряду. Результати подати у вигляді таблиці.
3. Побудувати графік залежності абсолютної похибки від x (у логарифмічному масштабі).

## Завдання за варіантом:

Варіант №6
Функція – ch(x)
Інтервал – [-0.8, 1.9]

## Вихідний текст програми

main.c

```
#include <stdio.h>
#include "ch.c"
#include "log.c"

const double RANGE_START = -0.8;
const double RANGE_END = 1.9;
const double RANGE_MIDDLE = (RANGE_END + RANGE_START) / 2;
const double RANGE_STEP = (RANGE_END - RANGE_START) / 10;

const int N = 6;

void run_task_1() {
    FILE *logfile = fopen("./task1.log.csv", "w");

    char epsilon_str[50];
    char terms_str[50];
    char absolute_error_str[50];
    char remainder_term_str[50];

    log_columns(logfile, 4, "Epsilon", "Terms", "Absolute Error",
"Remainder Term");

    for (int power = -2; power >= -14; power -= 3) {
        double epsilon = pow(10, power);
        ChResult result = ch_with_precision(RANGE_MIDDLE, epsilon);

        sprintf(epsilon_str, "10^%d", power);
        sprintf(terms_str, "%d", result.terms);
        sprintf(absolute_error_str, "%.16f", result.absolute_error);
        sprintf(remainder_term_str, "%.16f", result.remainder_term);
        log_columns(logfile, 4, epsilon_str, terms_str,
absolute_error_str, remainder_term_str);

        printf("Epsilon: 10^%d\n", power);
        printf("N: %d\n", result.terms);
        printf("Precise value: %.16f\n", result.precise);
        printf("Approximate value: %.16f\n", result.approximate);
        printf("Absolute error: %.16f\n", result.absolute_error);
        printf("Remainder term: %.16f\n", result.remainder_term);
        printf("\n");
```

```
    }
}

void run_task_2() {
    FILE *logfile = fopen("./task2.log.csv", "w");

    char xi_str[50];
    char absolute_error_str[50];
    char remainder_term_str[50];

    log_columns(logfile, 3, "X", "Absolute Error", "Remainder Term");

    for (int i = 0; i <= 10; i++) {
        double x = RANGE_START + (RANGE_STEP * i);
        ChResult result = ch_with_terms(x, N);

        sprintf(xi_str, "%.10f", x);
        sprintf(absolute_error_str, "%.16f", result.absolute_error);
        sprintf(remainder_term_str, "%.16f", result.remainder_term);
        log_columns(logfile, 3, xi_str, absolute_error_str,
remainder_term_str);

        printf("X: %f\n", x);
        printf("N: %d\n", result.terms);
        printf("Precise value: %.16f\n", result.precise);
        printf("Approximate value: %.16f\n", result.approximate);
        printf("Absolute error: %.16f\n", result.absolute_error);
        printf("Remainder term: %.16f\n", result.remainder_term);
        printf("\n");
    }
}

int main() {
    run_task_1();
    run_task_2();

    return 0;
}
```

ch.c

```c
#include <math.h>

typedef struct {
    int terms;
    double precise;
    double approximate;
    double absolute_error;
    double remainder_term;
} ChResult;

double term(int n, double radians) {
    double result = 1.0;

    for (int k = 1; k < n; k++) {
        result *= (radians * radians) / ((2 * k) * (2 * k - 1));
    }

    return result;
}

double remainder_term(int n, double radians) {
    return (2.0/3.0) * term(n, radians);
}

ChResult ch_with_precision(double radians, double epsilon) {
    int n = 0;
    double approximate = 0.0;

    do {
        n++;
        approximate += term(n, radians);
    } while (remainder_term(n, radians) >= epsilon);

    double precise = cosh(radians);
    double absolute_error = fabs(precise - approximate);
    double remainder_term_value = remainder_term(n, radians);

    return (ChResult){n, precise, approximate, absolute_error,
remainder_term_value};
}

ChResult ch_with_terms(double radians, int N) {
    int n = 0;
```

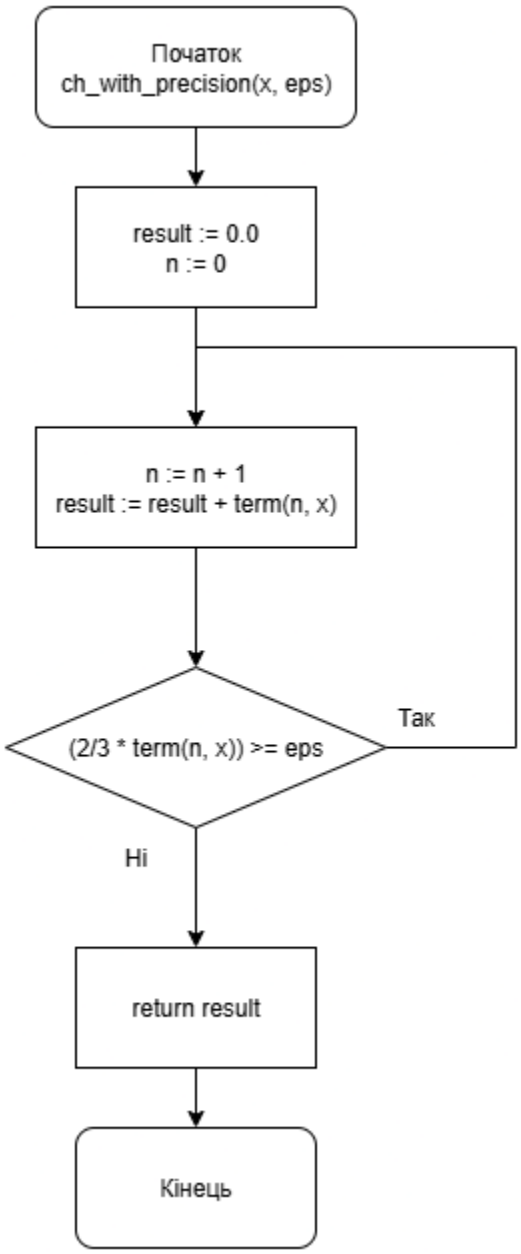
```
    double approximate = 0.0;

    do {
        n++;
        approximate += term(n, radians);
    } while (n < N);

    double precise = cosh(radians);
    double absolute_error = fabs(precise - approximate);
    double remainder_term_value = remainder_term(n, radians);

    return (ChResult){n, precise, approximate, absolute_error,
remainder_term_value};
}
```

log.c

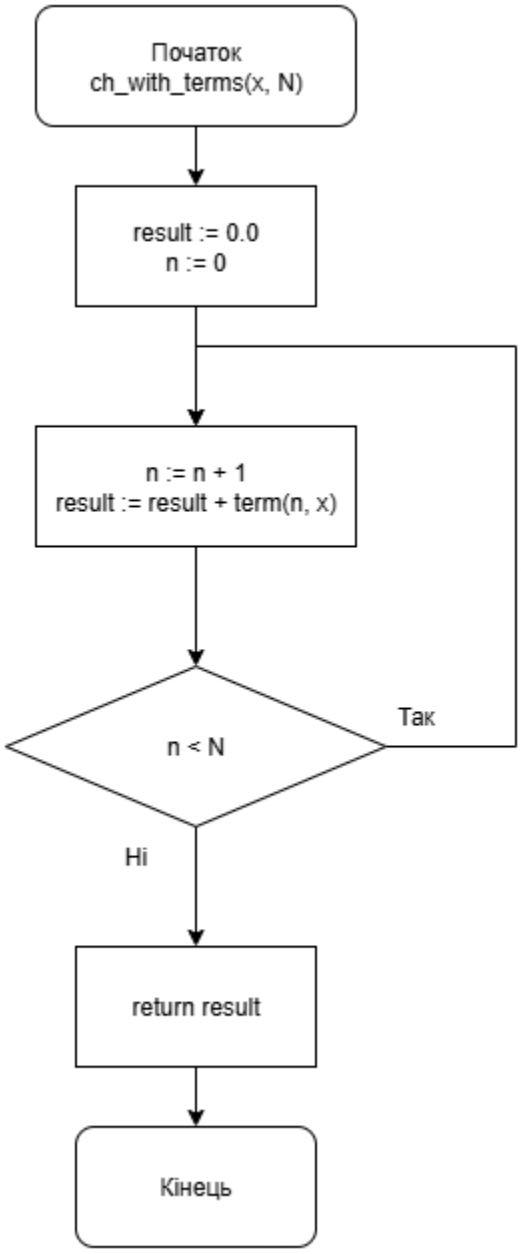
```c
#include <stdio.h>
#include <stdarg.h>

void log_columns(FILE *f, int columns, ...) {
    va_list args;
    va_start(args, columns);

    for (int i = 0; i < columns; ++i) {
        const char *header = va_arg(args, const char*);
        fprintf(f, "%s", header);
        if (i < columns - 1) {
            fprintf(f, ",");
        }
    }

    fprintf(f, "\n");
    va_end(args);
}
```

## Діаграми дій для основних обчислень.

Функція ch_with_precision(x, eps) для обчислення ch(x) з заданою точністю eps

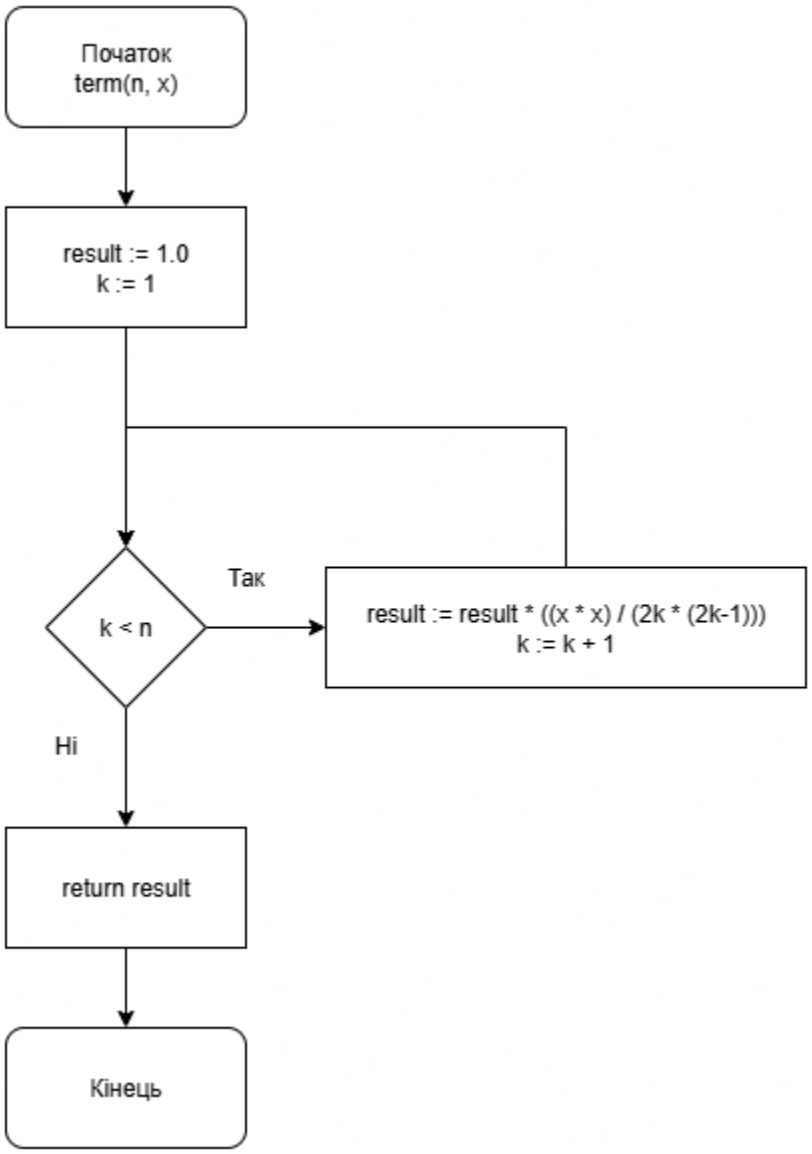


Функція ch_with_terms(x, N) для обчислення ch(x) за заданою кількістю членів ряду N



Функція term(n, x) для обчислення члена ряду Тейлора $u_n$ для функції ch(x) за формулою

$$u_1 = 1, \quad u_{k-1} = \frac{x^2}{2k(2k-1)} u_k, \quad k = (1, 2, 3, ..., n-1)$$



## Результати виконання

Таблиця 1.

Таблиця залежності довжини ряду n, що забезпечує точність функції не меншу за задане значення eps у точці x = (b + a)/2, від eps.

| ε | n | Абсолютна похибка | Залишковий член |
|---|---|---|---|
| $10^{-2}$ | 3 | 0.0000386537072743 | 0.0025418402777778 |
| $10^{-5}$ | 5 | 0.0000000006996157 | 0.0000001384489714 |
| $10^{-8}$ | 6 | 0.0000000000016021 | 0.0000000004653424 |
| $10^{-11}$ | 7 | 0.0000000000000024 | 0.0000000000010664 |
| $10^{-14}$ | 8 | 0.0000000000000002 | 0.0000000000000018 |

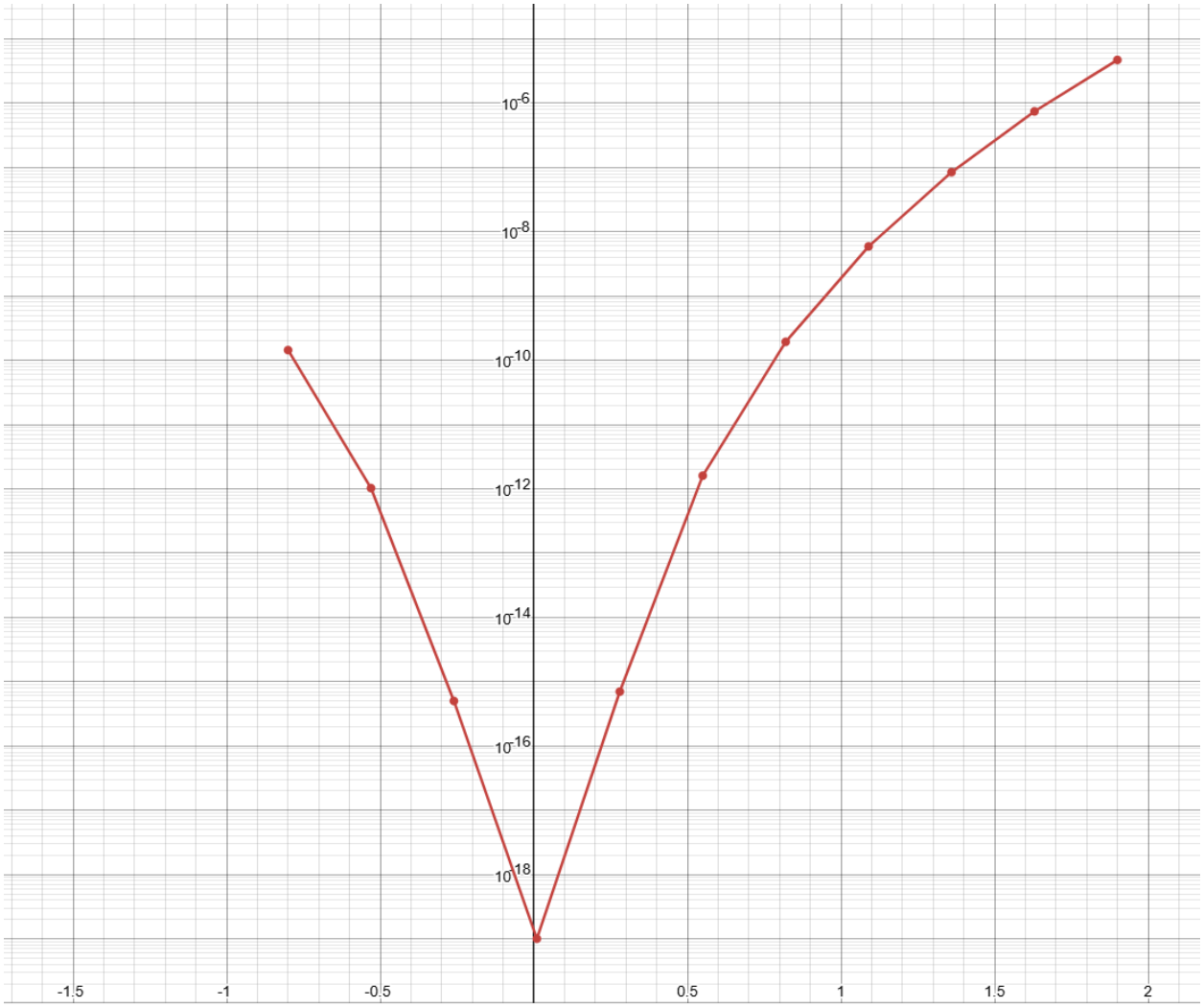
Таблиця 2.

Абсолютна похибка та залишковий член ряду для n = 6 у точках
xi = a + hi, h = (b – a)/10, i = 0, ..., 10.

| $x_i$ | Абсолютна похибка | Залишковий член |
|---|---|---|
| -0.8000000000 | 0.0000000001439697 | 0.0000000197262975 |
| -0.5300000000 | 0.0000000000010272 | 0.0000000003212953 |
| -0.2600000000 | 0.0000000000000000 | 0.0000000000002593 |
| 0.0100000000 | 0.0000000000000000 | 0.0000000000000000 |
| 0.2800000000 | 0.0000000000000007 | 0.0000000000005442 |
| 0.5500000000 | 0.0000000000016021 | 0.0000000004653424 |
| 0.8200000000 | 0.0000000001936580 | 0.0000000252513285 |
| 1.0900000000 | 0.0000000059104546 | 0.0000004349213100 |
| 1.3600000000 | 0.0000000844413388 | 0.0000039768095382 |
| 1.6300000000 | 0.0000007452092152 | 0.0000243232583168 |
| 1.9000000000 | 0.0000047137308137 | 0.0001126371666998 |

Примітка: Значення, що дорівнюють 0 в таблиці коректно інтерпретувати як такі, що менше за $10^{-16}$. Це пов'язано з обмеженнями стандартного типу double, який може гарантувати точність лише до 15-16 значущих цифр.

Графік 1. Залежність абсолютної похибки від x




Примітка: Значення в точках x = 0.01 та x = -0.26 для наочності наведені приблизно, виходячи з симетрії та форми графіку. Істинні дані наведені в табл. 2.

## Висновки

Під час виконання роботи була розроблена програми для обчислення функції ch(x) шляхом розкладання в степеневий ряд. Також були виконані обчислення абсолютної похибки обчислення на проміжку [-0.8, 1.9] для різних значень точності ε та кількості обчислюваних членів n.

Дані з таблиці 1 відповідають очікуванням і доводять правильність алгоритму. Як і очікувалось, значення абсолютної похибки < ε.

Дані з табл. 2 і граф. 1 демонструють цікаву закономірність. При фіксованій кількості обчислюваних членів ряду, абсолютна похибка зменшується з наближенням до x=0. Іншими словами, чим більший |x|, тим більше членів послідовності треба врахувати для отримання певної точності.

Це пояснюється тим, що функція завжди ch(x) зростає швидше за її апроксимацію через ряд Тейлора. Це можна строго довести, використовуючи властивості похідних, і той факт, що ряд, що розглядається є знакопостійним.

Важливо зазначити, що на абсолютну похибку вимірювань окрім точності обраного методу обчислення, впливає і методична похибка пов’язана з обмеженнями точності стандартного типу double мови програмування C. Цю похибку можна оцінити як $1*10^{-16}$.