Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 3**

***з дисципліни:*** «Мови та парадигми програмування»

***з розділу***: «Цикли. Вкладені Цикли. Функції. Перевантаження функцій. Рекурсія»

***Виконав:***

студент групи ШІ-14

Вульчин Володимир Сергійович

# **Тема роботи:**

Цикли, використання циклів у С++. Вкладені цикли. Обчислення суми рядів за допомогою циклів. Функції та їх перевантаження. Рекурсія функції. Створення функції для обчислення виразу та функції зі змінною кількістю параметрів.

# **Мета роботи:**

Ознайомитися, використати та опрацювати цикли та вкладені цикли for, while, do while у С++. А також оператори переходу **continue**, **break**, **goto.** Ознайомитися та опрацювати функції та їхню організацію, перевантаження функції та рекурсію.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Цикли. Вкладені цикли
* Тема №2: Оператори переходу **continue**, **break**, **goto**.
* Тема №3: Функція. Перевантаження функції.
* Тема №4: Рекурсія

1. Індивідуальний план опрацювання теорії:

* Тема №1 Цикли. Вкладені цикли .
  + Джерела Інформації
    - ChatGPT / Команда, Одногрупнки
    - Відео.

Відео-теорія про цикл: [for.](https://youtu.be/QXaSSIjVor8?t=827)

Відео-теорія про цикл: [while.](https://youtu.be/QXaSSIjVor8?t=827)

* + - Стаття.

Теорія про цикли: [while, for, do while](https://www.bestprog.net/uk/2017/09/04/cycles-operators-of-the-cycle-for-while-do-while_ua/#q10).

Окрема теорія про цикл та вкладений цикли: [for](https://acode.com.ua/urok-72-tsykl-for/).

Окрема теорія про цикл та вкладений цикли: [while.](https://acode.com.ua/urok-70-tsykl-while/)

* + Що опрацьовано:
    - Самостійно опрацював теорію та приклади на цикли та вкладені цикли у мові програмування С++, а саме: for, while, do while. Присутнє розмуіння різниці між цими циклами. Ознайомлений з вкладенеми циклами.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 09.11.2023
  + Звершення опрацювання теми: 25.11.2023
* Тема №2: Оператори переходу: **continue**, **break**, **goto**.
  + Джерела Інформації:
    - ChatGPT / Команда, Одногрупники
    - Відео.

Відео-теорія про: [break](https://www.youtube.com/watch?v=rj1OLsBKazA&t=399s)

Теорія про: [continue](https://www.youtube.com/watch?v=UY295pIdeoQ&t=337s)

Теорія про: [goto](https://www.youtube.com/watch?v=k9LLwdp-tdg&list=PLiPRE8VmJzOoA_Td13YPAN_BF4gi2mhgc&index=44)

* + - Стаття.

Cтаття про: [break / continue / switch](https://acode.com.ua/urok-73-operatory-break-i-continue/#toc-1)

Cтаття про: [goto](https://acode.com.ua/urok-69-operator-goto)

* + Що опрацьовано:
    - Опрацював матеріал про оператори переходу **break**, **goto**, **continue**. **switch** та як їх використовувати в різних циклах.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 09.11.2023
  + Звершення опрацювання теми: 25.11.2023
* Тема №3: Функція. Перевантаження функції.
  + Джерела Інформації:
    - ChatGPT / Команда, Одногрупники
    - Відео.

Відео-теорія про: [функцію, як математичне значення.](https://youtu.be/qQmO_XJ2eAY)

Відео-теорія про: [функцію в мові програмування С++.](https://youtu.be/G8P6SvdqU9s)

Відео-теорія про: [перевантаження функції.](https://youtu.be/7F0KF-24Aoc)

* + - Стаття.

Теорія про: [функцію.](http://cpp.dp.ua/ponyattya-funktsiyi/)

Теорія про: [перевантаження функції.](https://acode.com.ua/urok-108-perevantazhennya-funktsij/)

* + Що опрацьовано:
    - Розібрався, що таке функція, що собою представляє перевантаження функції, для чого потрібно функції.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 17.11.2023
  + Звершення опрацювання теми: 28.11.2023
* Тема №4 Рекурсія.
  + Джерела Інформації:
    - ChatGPT / Команда, Одногрупники
    - Відео.

Відео-теорія про: [рекурсію.](https://youtu.be/V7q9w_s0nns?t=843)

Відео-теорія про [рекурсію та як її не боятись. 😊](https://www.youtube.com/watch?v=5o-6Bu1FbLE)

* + - Стаття.

Теорія про: [рекурсію.](https://acode.com.ua/urok-113-rekursiya-i-chysla-fibonachchi/)

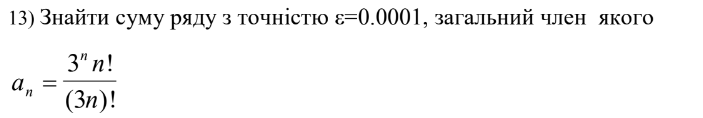
* + Що опрацьовано:
    - Що ж це таке рекурсія насправді? Навіщо вона потрібна для програміста?
  + Статус: Ознайомлений
  + Початок опрацювання теми: 17.11.2023
  + Звершення опрацювання теми: 26.11.2023

# **Виконання роботи:**

## **1. Опрацювання завдання та вимог до програм та середовища:**

Завдання **№1 VNS Lab 2 - Task 1**

* Варіант завдання: 13



*Image 1. VNS LAB 2 – Task 1*

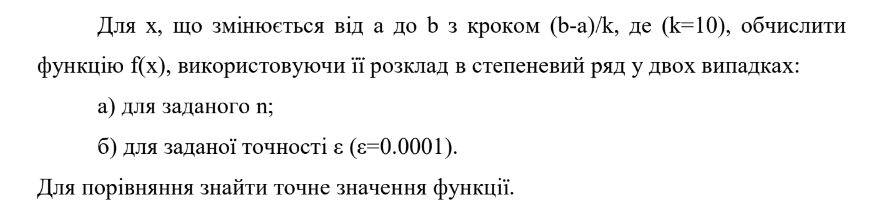
Успішно закодено програму, яка за умовою: , при введені користувачем значення n, виводить n результатів, та сумму ряду лише тих результатів n, які менше за .

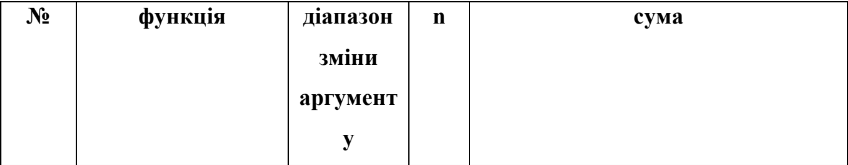
До прикладу: Користувач вводить значення n = 4. Програма виводить обчислені значення, які вміщає в собі n = 4, тобто (n = 1, n = 2, n = 3 до n = 4).

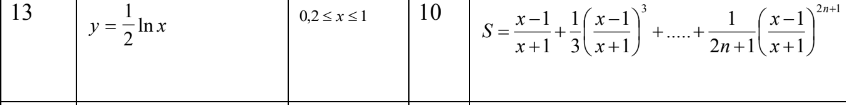
Наступним кроком, програма виводить «Суму рядку меншу за », якщо таких значень не існує, то виводиться число «0»

Завдання **№2 VNS Lab 3 - Task 1**

* Варіант 13







*Image 2. VNS LAB 3 – Task 1*

* Обчислити функцію f(x), використовуючи її розклад в степеневий ряд у двох випадках:

а) для заданого n = 10;

б) для заданої точності ε (ε=0.0001).

Де умова завдання має вигляд:

Функція :

Діапазон зміни аргументу y: 1

Значення:

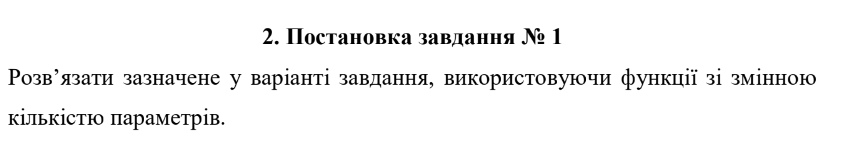
Сума:

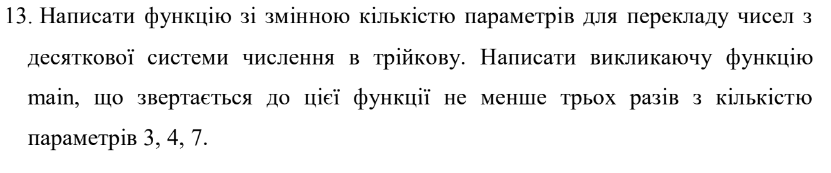


* В результаті у виводі маємо отримати значення: х, y, sum(n), sum(epsilon).

Завдання **№3 VNS Lab 7 - Task 1**

* Варіант завдання: 13





*Image 3. VNS LAB 7 – Task 1*

Використовуючи алгоритм дій (**допоміг** **ЧатЖПТ**)

1. Задати десяткове число.

2. Ділити число на 3 і записувати залишок.

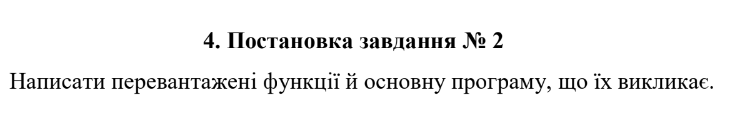
3. Ділити результат попереднього ділення на 3 і записувати залишок.

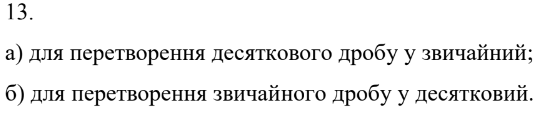
4. Продовжувати цей процес до тих пір, поки результат ділення не стане 0.

5. Отримані залишки, відчитані від найменш значущого до найбільш значущого, будуть утворювати трійкове представлення числа.

Завдання **№4 VNS Lab 7 - Task 2**

* Варіант завдання: 13





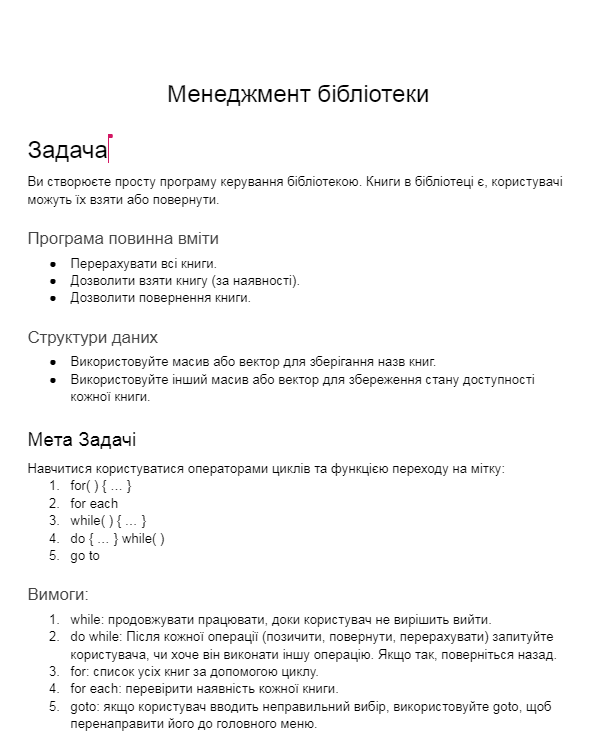
*Image 4. VNS LAB 7 – Task 2*

Використовуючи алгоритм дій (**ЧатЖПТ допоміг:**)

1. Показати як використовується цикл з використанням алгоритму Евкілда для обчислення (НСД)

Завдання **№5 Class Practice Work**

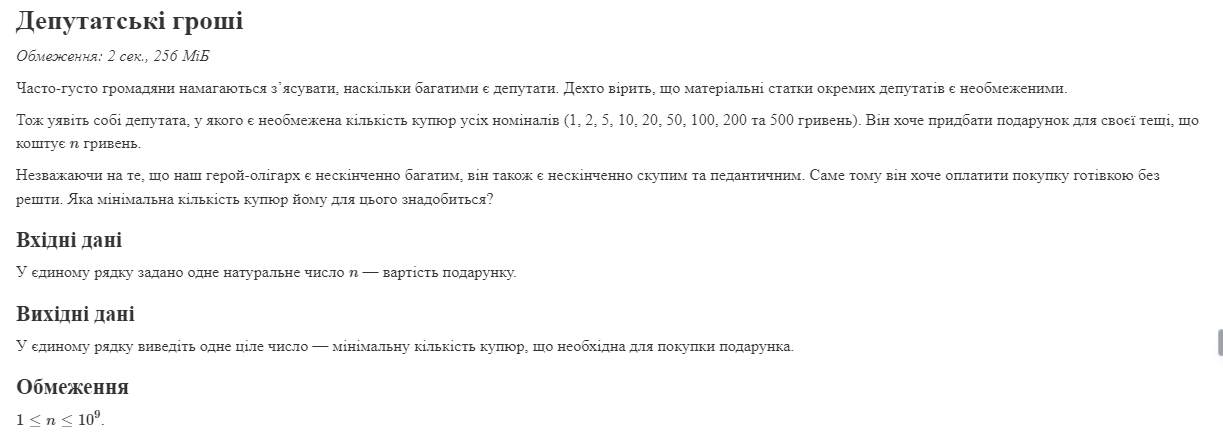
* Варіант відсутній.



*Image 5. Class Practice Work*

Завдання **№6 Self Practice Work**

* [Задача Algotester 0021](https://algotester.com/uk/ArchiveProblem/DisplayWithEditor/6) - Депутатські гроші



*Image 6. Self Practice Work*

Використано масив, який складається з купюр та циклу for.

## **2. Дизайн та планована оцінка часу виконання завдань:**

Програма **№1 VNS Lab 2 - Task 1**

* Блок-схема: (відсутня)
* Планований час на реалізацію: *2* год => Затрачений час: *4* год.
* Важливі деталі для врахування в імплементації: Виведення формули Через цикли for. Перевірка рівності , умовним оператором if.
* Програма **№2 VNS Lab 3 - Task 1**
* Блок-схема (відсутня)
* Планований час на реалізацію: *2* год => Затрачений час: *6* год.

Важливі деталі для врахування в імплементації: Використання 3-ьох перевантажених функцій для обчислення точного значення , обчислення при підстановці значення *,* обчислення значення з заданою точністю до ., відповідно.

* Програма **№3 VNS Lab 7 - Task 1**
* Блок-схема (відсутня)
* Планований час на реалізацію: *1* год => Затрачений час: *3* год.

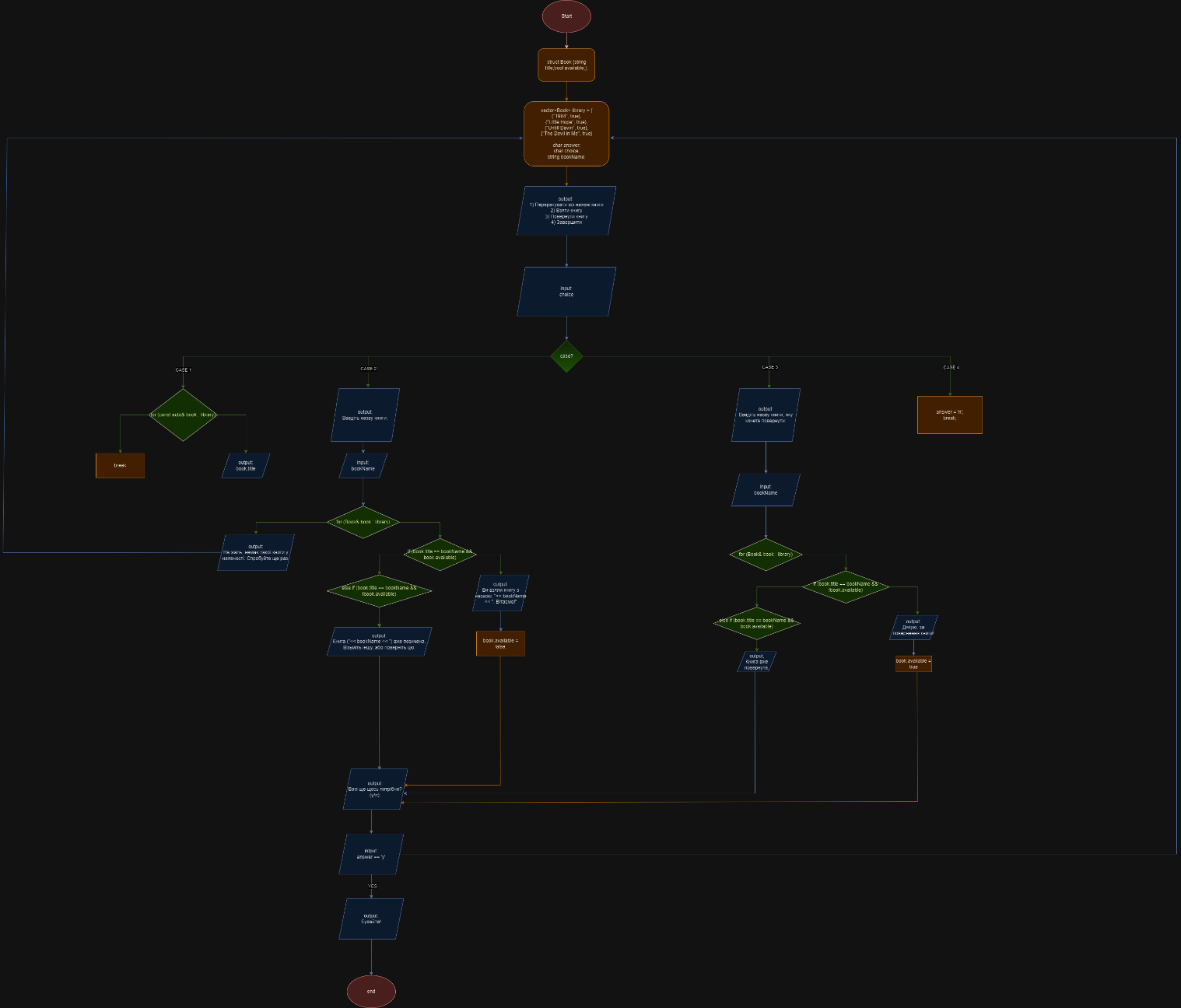
Важливі деталі для врахування в імплементації. Оголошено перевантажену функцію з двома параметрами: к-сть чисел, які будуть конвертовані в трійкову систему та масив цілих чисел, які потрібно конвертувати. Викликано функцію не менше трьох разів з кількістю параметрів 3, 6, 7.

Програма **№4 VNS Lab 7 - Task 2**

* Блок-схема (відсутня)
* Планований час на реалізацію: *2.5* год => Затрачений час: *4* год.
* Важливі деталі для врахування в імплементації: Використання алгоритму Евкліда для обчислення (НСД)

Програма **№5 Class Practice Work**

* Блок-схема

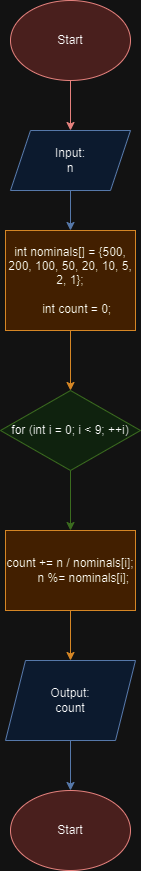


*Image 7. Flowchart for Class Practice Work.*

* Планований час на реалізацію: *2* *дні* => Затрачений час: *3 дні*.

Програма **№6 Self Practice Work**

* Блок-схема



*Image 8. Блок-схема до програми №6 Self Practice Work*

- Планований час на реалізацію: *2.5* год => Затрачений час: *1* год

## **3. Конфігурація середовища до виконання завдань:**

Використано налаштування середовища з попередніх лабораторних робіт.

## **4. Код програм з посиланням на зовнішні ресурси:**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-VNS-Lab-2-Task-1.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-e6070b7daf8fba29af57cc991ce7767a7d15d3092d103df2101122be075bf259)

Завдання **№1 VNS Lab 2 - Task 1**

#include <iostream>

#include <math.h>

using namespace std;

// n! = n\*(n-1)!

// a\_n = (3^n\*n!)/(3\*n)!

// a\_n+1 / an = 1/9n^2+9n+2

int main()

{

    int temp; // тимчасове число, яке = n

    double epsilon = 0.0001; // значення епсілом

    double sum = 0; // сума всіх чисел

    double a\_n; // відповідь при якомусь n, якщо n = 1, то відповідь: 0.5, якщо n = 2, то відповідь: 0.025. Тобто, якщо введешь n = 2, то воно виведе n = 1 і n = 2.

    double factorial\_n = 1; // виведення n!

    double factorial\_3n = 1; // виведення (3\*n)!

    int n; // значення, яке вводить користувач

    cout << "Введіть значення n: ";

    cin >> n;

    for(int i = 1 ; i <= n; i++){ // цикл, який відбувається n разів, int i = 1 - починаємо з числа 1, при умові i <= n - цикл продовжується до тих пір поки i > n; i++ - виконується в самому кінці

    factorial\_n = 1; // вказали значення 1, щоб забути факторіал попереднього n.

    factorial\_3n = 1; // вказали значення 1, щоб забути факторіал попереднього n.

    temp = i; // робиму копію i, якщо шукаємо n = 1 то temp = 1, і т.д.

    for (int i = 1; i <= temp; i++){ // цикл, який відбувається temp разів, int i = 1 - починаємо з числа 1, при умові i <= temp - цикл продовжується до тих пір поки i > temp; i++ - виконується в самому кінці

        factorial\_n \*= i; // домножуємо до факторіалу i temp разів.

    }

    for (int i = 1; i <= temp \* 3; i++){ // цикл, який відбувається temp разів, int i = 1 - починаємо з числа 1, при умові i <= temp \* 3 - цикл продовжується до тих пір поки i > temp \* 3; i++ - виконується в самому кінці

        factorial\_3n \*= i; // домножуємо до факторіалу i (temp \* 3) разів

    }

    a\_n = (factorial\_n \* pow(3,temp)) / factorial\_3n; // підрахунок нашої формули на даних момент a\_n при 1 = 0.5 і т.д.

    cout << a\_n << " "; // виводимо а\_n

    if(a\_n < epsilon) { // перевіряємо чи а\_n є меншим за епсілон

    sum += a\_n; // якщо а\_n є меншим за епсілон, то ми додаємо його до суми, якщо  не а\_n є меншим за епсілон то ми його пропускаємо.

    }

    }

    cout << "\nСума ряду менша за epsilon: " << sum;

    return 0;

}

*Блок №1. Код до програми №1 VNS Lab 2 - Task 1*

Завдання **№2 VNS Lab 3 - Task 1**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-VNS-Lab-3-Task-1.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-fb6c9337aaba1c7e083d5d16a4f652435eb6121deebc972ccb0d04a7ab90f73f)

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

// Функція для обчислення точного значення ln(x/2)

double y(double x) { //

    return 0.5 \* log(x);

}

// Функція для обчислення значення ln(x/2) через розклад у степеневий ряд

double sum\_n(int i, double x) {

    double sum = 0.0;

    double term = 1.0;

    for (int n = 0; n <= i; ++n) {

        term \*= pow((x - 1) / (x + 1),2\*n+1);

        sum += term / (2 \* n + 1);

    }

    return sum;

}

// Функція для обчислення значення ln(x/2) з заданою точністю ε

double sum\_e(double epsilon, double x) {

    double sum = 0.0;

    double term = 1.0;

    int n = 1;

    while (term > epsilon) {

        term \*= pow((x - 1) / (x + 1),2\*n+1);

        sum += term / (2 \* n + 1);

        n++;

    }

    return sum;

}

int main() {

    // Задані параметри

    double a = 0.2;

    double b = 1.0;

    int k = 10;

    int n = 10;

    double epsilon = 0.0001;

    // Обчислення та вивід результатів для кожного x

    for (int i = 1; i <= k; i++) {

        double x = a + i \* (b - a) / k;

        double function = y(x);

        double ap\_sum\_N = sum\_n(n, x);

        double ap\_sum\_E = sum\_e(epsilon, x);

        // Вивід результатів

        cout << fixed << setprecision(7);

        cout << " | X: " << x <<  " | Y: " << function <<  " | Sum n (10): " << ap\_sum\_N << " | Sum epsilon (0.0001): " << ap\_sum\_E << "\n";

    }

    return 0;

}

*Блок №2. Код до програми №2 VNS Lab 3 - Task 1*

Завдання **№3 VNS Lab 7 - Task 1**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-VNS-Lab-7-Task-1.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-9e5cfb33b8964f6f785ed82b01b3830053f9c5bea05308d5212d2588064d266a)

#include <iostream>

using namespace std;

/\* Оголошення функції dec\_to\_ter з двома параметрами.

int count: к-сть чисел, які будуть конвертовані в трійкову систему.

int numbers[]: масив цілих чисел, які потрібно конвертувати.\*/

void dec\_to\_ter(int count, int numbers[]) {

    for (int i = 0; i < count; ++i) { // Цикл, який проходить через кожен елемент масиву чисел з 0 до count - 1.

        int dec\_num = numbers[i]; // Отримуємо поточне число для конвертації з масиву.

        int ter\_num[32]; // Масив, для збереження трійкового числа

        int index = 0; // Ініціалізація індексу для масиву ter\_num

        while (dec\_num > 0) {

            ter\_num[index] = dec\_num % 3; // Залишок від ділення dec\_num(десяткових чисел) на 3, який записується в ter\_num за поточним індексом

            dec\_num /= 3; // dec\_num (десяткове число) знову ділимо на 3, щоб продовжити перетворення у трійкову систему.

            ++index; // Інкрементація index`a, для переходу до наступної позиції в ter\_num

        }

        cout << "Decimal to Ternary: "; // Виведення повідомлення на консоль.

        for (int j = index - 1; j >= 0; --j) { //  Цикл for, який виводить елементи масиву ter\_num з кінця до початку.

            cout << ter\_num[j]; // Виведення поточного елементу на консоль.

        }

        cout << endl; // Перехід на новий рядок після виведення всіх елементів трійкового числа.

    }

}

int main() {

    // Три масиви чисел (num1, num2, num3) для подальшої конвертації у трійкову систему.

    int num1[] = {10, 15, 21};

    int num2[] = {7, 12, 18, 25};

    int num3[] = {30, 42, 56, 63, 77, 89, 99};

    // Виклик функції dec\_to\_ter тричі, яка передає різні масиви чисел та їхню кількість.

    dec\_to\_ter(3, num1);

    dec\_to\_ter(4, num2);

    dec\_to\_ter(7, num3);

    return 0;

}

*Блок №3. Код до програми №3 VNS Lab 7 - Task 1*

Завдання **№4 VNS Lab 7 - Task 2**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-VNS-Lab-7-Task-2.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-f29e8dd78abfba7cf67cae2d72a22690ec5bc56695aaa50672f25c50cd413377)

#include <iostream>

#include <cmath>

using namespace std;

// Функція яка обчислює (НСД) для двох цілих чисел а і b

int calculateGCD(int a, int b) { //

    while (b != 0) { // Цикл з використанням алгоритму Евкілда для обчислення (НСД)

        int temp = b;

        b = a % b; // Обчислення нового значення b як залишку від ділення a на b.

        a = temp; //  Присвоєння a значення змінної temp.

    }

    return a;

}

// Функція для перетворення десяткового дробу у звичайний

void convertDecToFraction(double dec) {

    cout << "a) Перетворення десяткового дробу у звичайний: ";

    cout << dec << " = "; // Виводить десятковий дріб перед перетворенням.

    // Визначення знаменника

    int precision = 1000000;  // Встановлює точність для апроксимації при перетворенні десяткового дробу.

    int num = round(dec \* precision); // Обчислює чисельник десяткового дробу.

    int denom = precision; // Задає знаменник десяткового дробу.

    // Скорочення дробу

    int gcd = calculateGCD(num, denom);

    num /= gcd;

    denom /= gcd;

    // Виведення результату

    cout << num << "/" << denom << endl;

}

// Функція для перетворення звичайного дробу у десятковий

void convertFractionToDec(int num, int denom) {

    cout << "б) Перетворення звичайного дробу у десятковий: ";

    // Виведення звичайного дробу

    cout << num << "/" << denom << " = ";

    // Перетворення у десятковий дріб

    double dec = static\_cast<double>(num) / denom;

    // Виведення результату

    cout << dec << endl;

}

int main() {

    // a) Перетворення десяткового дробу у звичайний

    convertDecToFraction(9.1240);

    // b) Перетворення звичайного дробу у десятковий

    convertFractionToDec(1, 5);

    return 0;

}

*Блок №4. Код до програми №4 VNS Lab 3 - Task 2*

Завдання **№5 Class Practice Work.**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-Class-Practice-Work.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-0ac69349d636954ea0b9683540f3884b8f0de40e9918e789b83098c4023cf739)

#include <iostream>

#include <string>

#include <vector>

using namespace std;

// Оголошуємо структуру Book з двома полями: title (назва книги) та available (доступність книги).

struct Book {

    string title;

    bool available;

};

int main() {

// Створюємо вектор з книгами та їх доступністю.

    vector<Book> library = {

        {"1984", true},

        {"Little Hope", true},

        {"Untill Dawn", true},

        {"The Devil in Me", true}

    };

// змінні для вводу та збереження введених даних.

    char answer;

    char choice;

    string bookName;

    // Початок циклу, який буде виконуватися, доки користувач буде вводити значення 'y'.

    do {

        cout << "1) Перерахувати всі наявні книги" << endl;

        cout << "2) Взяти книгу" << endl;

        cout << "3) Повернути книгу" << endl;

        cout << "4) Завершити" << endl;

        cin >> choice; // Зчитування вибору користувача

        //  Перехід до блоку коду в залежності від вибору користувача.

        switch (choice) {

            // Якщо користувач обрав опцію '1', виводить усі наявні книги.

            case '1':

                for (const auto& book : library) {

                    cout << book.title << endl;

                }

                break;

            // Якщо користувач обрав опцію '2', дозволяє взяти книгу, перевіряючи доступність та назву книги.

            case '2':

                cout << "Введіть назву книги: ";

                cin.ignore(); // Ігнор попереднього сивола нового рядка в буфері введення.

                getline(cin, bookName); // Зчитуємо повністю введену назву книги у змінну bookName.

                for (Book& book : library) { // Перевірка кожної книги у бібліотеці.

                    if (book.title == bookName && book.available) { //  Перевірка, чи книга доступна та чи вона має вказану користувачем назву.

                        cout << "Ви взяли книгу з назвою: "<< bookName << ". Вітаємо!" << endl;

                        book.available = false; // Позначає книгу як недоступну.

                        goto continueMenu; // Повернення до меню.

                    } else if (book.title == bookName && !book.available) { // Перевірка на введеня правильного значення книги, яке наявне в векторі книг.

                        cout << "Книга ("<< bookName << ") вже позичена. Візьміть іншу, або поверніть цю."<< endl;

                        goto continueMenu;

                    }

                }

                cout << "На жаль, немає такої книги у наявності. Спробуйте ще раз." << endl;

                break;

            // Якщо користувач обрав опцію '3', дозволяє повернути книгу, перевіряючи наявність та стан книги.

            case '3':

                cout << "Введіть назву книги, яку хочете повернути: ";

                cin.ignore(); // Ігнор попереднього сивола нового рядка в буфері введення.

                getline(cin, bookName); // Зчитуємо повністю введену назву книги у змінну bookName

                for (Book& book : library) { // Перевіряє кожну книгу у бібліотеці для повернення.

                    if (book.title == bookName && !book.available) { // Перевіряє, чи книга з введеною назвою вже була позичена.

                        cout << "Дякую, за повернення книги!" << endl;

                        book.available = true; // Позначає книгу як доступну для позичення.

                        goto continueMenu; // Повернення до меню (запит на продовження).

                    } else if (book.title == bookName && book.available) { // Перевіряє, якщо користувач взяв і віддав книгу, або не брав її, і хоче повернути. То виводить cout нижче.

                        cout << "Книга вже повернута." << endl;

                        goto continueMenu; // Повернення до меню (запит на продовження).

                    }

                }

                cout << "Ця книга не з бібліотеки." << endl;

                break;

            // Якщо користувач обрав опцію '4', завершує цикл.

            case '4':

                answer = 'n';

                break;

            //  Випадок за замовчуванням, який викликається, якщо введене значення не відповідає жодному з case. Виводить повідомлення про невірний вибір.

            default:

                cout << "Неправильне значення. Будь ласка, вкажіть номер від 1 до 4!" << endl;

                break;

        }

    // Запитує користувача, чи він хоче виконати ще якусь операцію.

    continueMenu:

        cout << "Вам ще щось потрібно? (y/n): ";

        cin >> answer;

    } while (answer == 'y'); // Кінець циклу. Повторює виконання, якщо користувач введе 'y'.

    cout << "Бувайте!" << endl;

    return 0;

}

*Блок №5. Код до програми №5 Class Practice Work*

Завдання **№6 Self Practice Work**

Посилання на файл програми у пул-запиті GitHub: [pull-request-link-Self-Practice-Work.](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/567/commits/ac88db874b21bf3daf01e686d9fb4adf3cae1107#diff-e6070b7daf8fba29af57cc991ce7767a7d15d3092d103df2101122be075bf259)

#include <iostream>

using namespace std;

int main() {

    long long n;

    cin >> n;

    int nominals[] = {500, 200, 100, 50, 20, 10, 5, 2, 1};

    int count = 0;

    for (int i = 0; i < 9; ++i) {

        count += n / nominals[i];

        n %= nominals[i];

    }

    cout << count << endl;

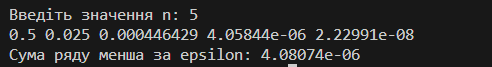
    return 0;

}

*Блок №6. Код до програми №6 Self Practice Work (Algotester 0021)*

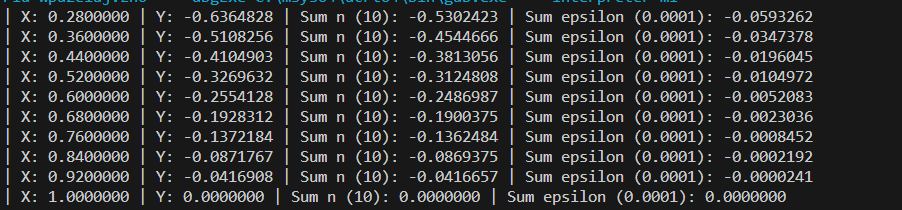
## **5. Результати виконання завдань, тестування та фактично затрачений час:**

Завдання **№1 VNS Lab 2 - Task 1**



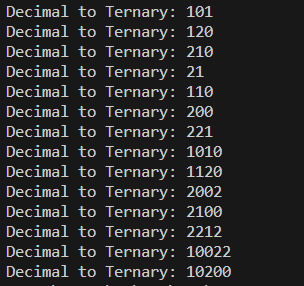
*Image 9. Результати виконання завдання №2 VNS Lab 3 - Task 1*

Час затрачений на виконання завдання: *4 год.*

Завдання **№2 VNS Lab 3 - Task 1**. *Image 10. Результати виконання завдання №2 VNS Lab 3 - Task 2*

Час затрачений на виконання завдання: *6 год.*

Завдання **№3 VNS Lab 7 - Task 1**

**

*Image 12. Результати виконання завдання №3 VNS Lab 7 - Task 1*

Час затрачений на виконання завдання: *3 год.*

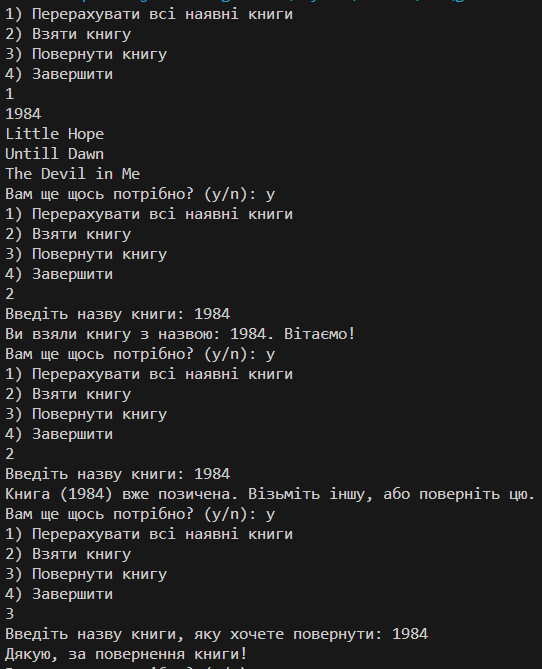
Завдання **№4 VNS Lab 7 - Task 2**

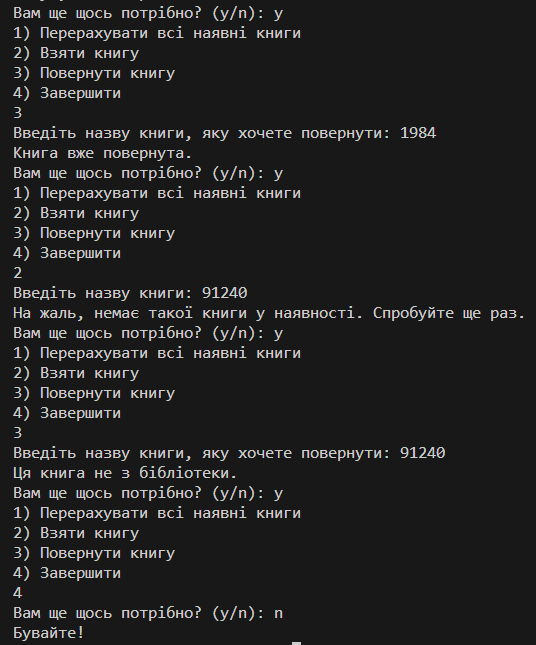


*Image 13. Результати виконання завдання №4 VNS Lab 7 - Task 2*

Час затрачений на виконання завдання: *4 год.*

Завдання **№5 Class Practice Work**





*Image 14. Результати виконання завдання №6 Class Practice Work*

Час затрачений на виконання завдання: *3 дні.*

Завдання **№6 Self Practice Work**

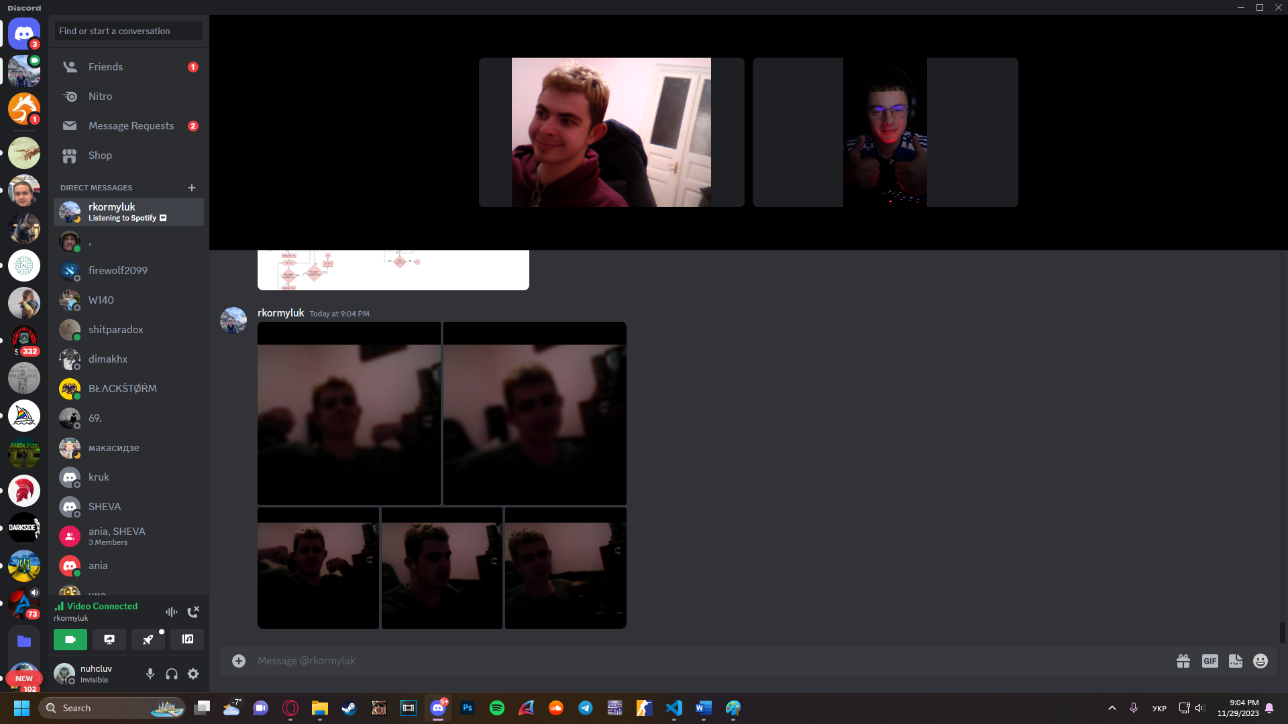


*Image 15. Результати виконання завдання №6 Self Practice Work*

Час затрачений на виконання завдання: *1 год.*

## **6. Кооперація з командою:**

Час затрачений на обговорення з командою: *18 год*.



*Image 16. Кооперація з командою (11.29.2023)*



*Image 17. Кооперація з командою (11.29.2023)*



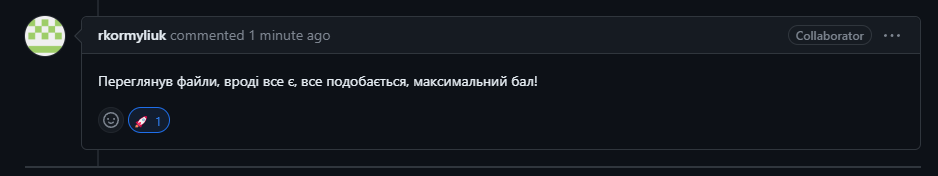
*Image 18. Кооперація з командою (11.29.2023)*



*Image 19. Кооперація з командою (11.28.2023)*

**

*Image 20. Кооперація з командою (11.27.2023)*

**

*Image 21. Коментар від побратима по команді (11.30.2023)*

# **Висновки:**

Я покращив:

1. Свої знання в області циклів, які дозволяють ефективно виконувати повторювані завдання у коді.
2. Свої знання функцій, для організації та структурування коду, роблячи його більш читабельним та підтримуваним.
3. Розуміння концепції перевантаженої функцій, що дозволяє визначити функції з однаковим ім'ям, але з різними параметрами. Це робить код більш гнучким та адаптованим до різних сценаріїв.
4. Сприйняття конструкцій вкладених циклів, що важливо для роботи з більш складними завданнями. Вивчення рекурсії дало мені інструмент для вирішення завдань, які можуть бути ефективно поділені на менші частини.