Міністерство освіти і науки України Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



Звіт

про виконання

**Лабораторних та практичних робіт № 7**

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

***з розділу***: «Розробка, програмування та код. Середовища для розробки.»

***Виконав:***

студент групи ШІ-11

Дах Тарас Романович

Львів 2023

# Тема роботи:

**Розв'язання математичних задач за допомогою програмування**

# Мета роботи:

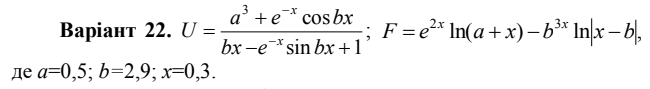
* Ознайомитися з основними математичними задачами, які можуть бути розв'язані за допомогою програмування.
* Вивчити основні алгоритми розв'язання математичних задач.
* Закріпити навички реалізації алгоритмів на мові програмування.

# Виконання роботи:

## Опрацювання завдання та вимог до програм та середовища:

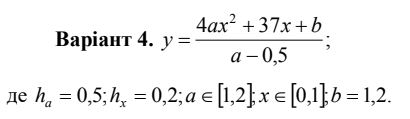
Завдання № 1 VNS Practice Work - Task 1

* + Варіант завдання - 22
  + Деталі завдання



*Рисунок 1: Деталі завдання VNS Practice Work - Task 1*

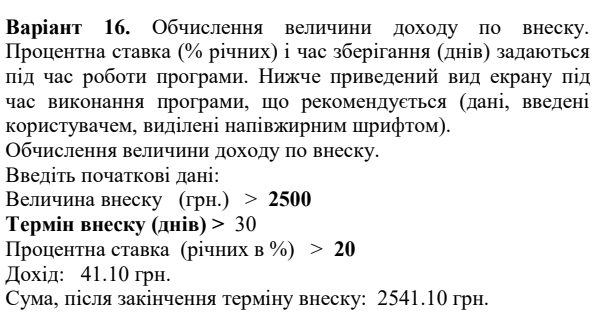
Завдання № 2 VNS Practice Work - Task 2

* + Варіант завдання - 24
  + Деталі завдання

*Рисунок 2: Деталі завдання VNS Practice Work - Task 2*

Завдання № 3 VNS Practice Work - Task 3.1

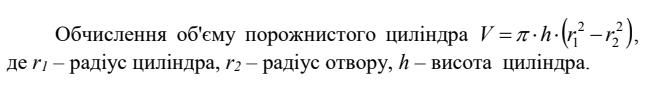
* + Варіант завдання - 7
  + Деталі завдання



*Рисунок 3: Деталі завдання VNS Practice Work - Task 3.1*

Завдання № 4 VNS Practice Work - Task 3.2

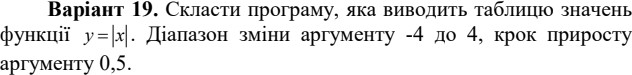
* + Варіант завдання - 7
  + Деталі завдання



*Рисунок 4: Деталі завдання VNS Practice Work - Task 3.2*

Завдання № 5 VNS Practice Work - Task 4

* + Варіант завдання - 19
  + Деталі завдання

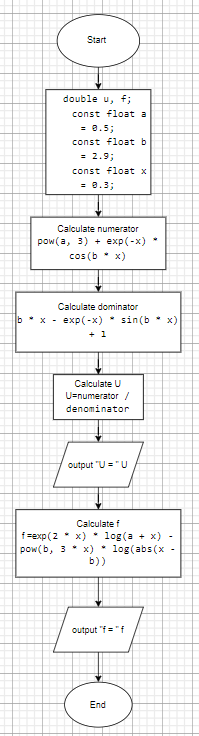


*Рисунок 5: Деталі завдання VNS Practice Work - Task 4*

## Дизайн та планована оцінка часу виконання завдань:

Програма № 1 VNS Practice Work - Task 1

* + Блок-схема

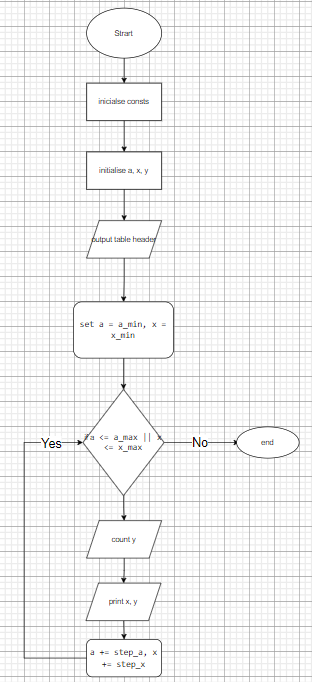


*Рисунок 6: Блок-схема до програми № 1*

* + Планований час на реалізацію — 10 хвилин

Програма № 2 VNS Practice Work - Task 2

* + Блок-схема

**

*Рисунок 7: Блок-схема до програми № 2*

Планований час на реалізацію — 5 хвилин Програма № 3 VNS Practice Work - Task 3.1

* + Блок-схема

*Рисунок 8: Блок-схема до програми № 3*

* + Планований час на реалізацію — 5 хвилин

Програма № 4 VNS Practice Work - Task 3.2

* + Блок-схема

*Рисунок 9: Блок-схема до програми № 4*

* + Планований час на реалізацію — 5 хвилин Програма № 5 VNS Practice Work - Task 4
  + Блок-схема

*Рисунок 10: Блок-схема до програми № 5*

* + Планований час на реалізацію — 10 хвилин

## Код програм з посиланням на зовнішні ресурси:

Завдання № 1 [VNS Practice Work - Task 1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779/files#diff-320957b217537f6252b4da92cdf2169470de1f90b74b800663f0f8a7a7fff153)

// Підключення необхідних бібліотек

#include <iostream>

#include <cmath>

// Основна функція програми

int main(){

    // Оголошення змінних для обчислень та констант

    double u, f;

    const float a = 0.5;

    const float b = 2.9;

    const float x = 0.3;

    // Обчислення чисельника для виразу U

    double numerator = pow(a, 3) + exp(-x) \* cos(b \* x);

    // Обчислення знаменника для виразу U

    double denominator = b \* x - exp(-x) \* sin(b \* x) + 1;

    // Обчислення виразу U та виведення результату

    u = numerator / denominator;

    std::cout << "U = " << u << std::endl;

    // Обчислення виразу F

    f = exp(2 \* x) \* log(a + x) - pow(b, 3 \* x) \* log(abs(x - b));

    // Виведення результату обчислення виразу F

    std::cout << "F = " << f << std::endl;

    // Повертається 0, що вказує на успішне завершення програми

    return 0;

}

Завдання № 2 [VNS Practice Work - Task 2](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779/files#diff-87d734962dc86c2b61aa8aca8ca0c616d657ab32f901c61ee0624c50c92b984d)

#include <iostream>

#include <cmath>

int main() {

    // Задані значення констант

    const double b = 1.2;

    const double step\_a = 0.5;

    const double step\_x = 0.2;

    // Діапазон для a та x

    const double a\_min = 0.5;

    const double a\_max = 2.0;

    const double x\_min = 0.0;

    const double x\_max = 1.0;

    // Змінні для збереження значень a, x та y

    double a, x, y;

    // Вирішив зробити Вивід за допомогою таблиці

    std::cout << "a     |     x     |     y" << std::endl;

    std::cout << "-----------------------------" << std::endl;

    // Цикл для зміни значень a та x одночаснор

    for (a = a\_min, x = x\_min; a <= a\_max || x <= x\_max; a += step\_a, x += step\_x) {

        // Розрахунок функції для поточних значень a та x

        y = (4 \* a \* pow(x,2) + 37 \* x + b) / (a - 0.5);  // Якщо знаменник при a = 0.5 , значення виразу знайти неможливо

        // Вивід значень у форматі таблиці

        std::cout << a << "     |     " << x << "     |     " << y << std::endl;

    }

    return 0;

}

Завдання № 3 [VNS Practice Work - Task 3.1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779/files#diff-05abdea726dd50362734d36c6069977ef69f5dbeb5b5e3c26866dcaaeb887a5a)

#include <iostream>

#include <iomanip> // Для використання функції setprecision

using namespace std;

int main() {

    // Введення початкових даних

    double principalAmount, interestRate, income;

    int depositTerm;

    cout << "Обчислення величини доходу по внеску." << endl;

    cout << "Величина внеску (грн.) = ";

    cin >> principalAmount;

    cout << "Термін внеску (днів) = ";

    cin >> depositTerm;

    cout << "Процентна ставка (річних в %) = ";

    cin >> interestRate;

    // Обчислення доходу

    income = (principalAmount \* interestRate \* depositTerm) / (365 \* 100);

    // Обчислення суми після закінчення терміну внеску

    double finalAmount = principalAmount + income;

    // Виведення результату з форматуванням до двох знаків після коми

    cout << "Дохід: " << fixed << setprecision(2) << income << " грн." << endl; // setprecision використав для виведення 2 знаків піля коми

    cout << "Сума, після закінчення терміну внеску: " << fixed << setprecision(2) << finalAmount << " грн." << endl;//fixed вказує на використання фіксованого формату для виведення числа з плаваючою комою. Формат фіксованої точності забезпечує постійне відображення заданої кількості знаків після десяткової коми, навіть якщо вони нульові.

    return 0;

}

Завдання № 4 [VNS Practice Work - Task 3.2](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779/files#diff-82699cbcaafb5430e939974881e484a229e58b344adb42568ca00f48dbb46f85)

// Підключення необхідних бібліотек

#include <iostream>

#include <cmath>

// Визначення константи для числа Пі

const double PI = 3.14159265358979323846;

// Основна функція програми

int main() {

    // Оголошення змінних для радіусів та висоти циліндра

    double r1, r2, h;

    // Виведення повідомлення та отримання значення для радіусу циліндра (r1)

    std::cout << "Enter the cylinder radius (r1): ";

    std::cin >> r1;

    // Виведення повідомлення та отримання значення для радіусу отвору (r2)

    std::cout << "Enter the hole radius (r2): ";

    std::cin >> r2;

    // Виведення повідомлення та отримання значення для висоти циліндра (h)

    std::cout << "Enter the cylinder height (h): ";

    std::cin >> h;

    // Обчислення об'єму вихолодженого циліндра за заданими значеннями

    double volume = PI \* h \* (std::pow(r1, 2) - std::pow(r2, 2));

    // Виведення результату обчислення об'єму на екран

    std::cout << "Volume of the hollow cylinder: " << volume << std::endl;

    // Повертається 0, що вказує на успішне завершення програми

    return 0;

}

Завдання № 5 [VNS Practice Work - Task 4](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779/files#diff-206acd9c2f6d71542fd552af4b04ce736947d5fcb778f601576af3208b4b4a41)

#include <iostream>

#include <cmath>

#include <iomanip>

int main() {

    // Виведення заголовку таблиці

    std::cout << " x    |   y = |x|" << std::endl;

    std::cout << "-------------------" << std::endl;

    // Виведення значень функції для заданого діапазону та кроку

    for (double x = -4.0; x <= 4.0; x += 0.5) {

        double y = std::abs(x);

        // Виведення значень у форматі таблиці

        std::cout << std::fixed << std::setprecision(2) << std::setw(3) << x << " | " << std::setw(4) << y << std::endl;////fixed вказує на використання фіксованого формату для виведення числа з плаваючою комою. Формат фіксованої точності забезпечує постійне відображення заданої кількості знаків після десяткової коми, навіть якщо вони нульові. setw(5) встановлює ширину поля виведення для змінної x як 5 символів. Аналогічно, setw(10) встановлює ширину поля для змінної y як 10 символів. Це використовується для форматування виведення так, щоб значення x та y були вирівняні у вигляді стовпців.В даному випадку, вибрані ширини 5 та 10 символів

    }

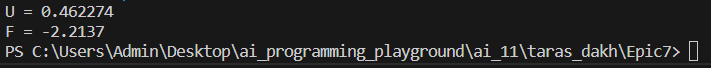
    return 0;

}

**[Посилання на pull request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/779)**

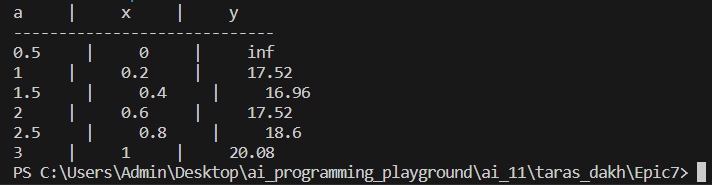
## Результати виконання завдань, тестування та фактично витрачений час:

Завдання № 1 VNS Practice Work - Task 1



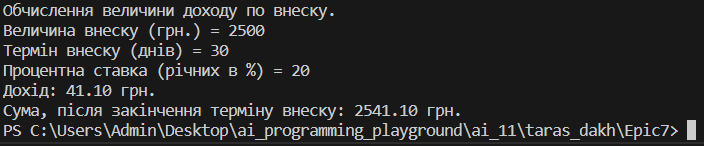
*Рисунок 11: Результати виконання VNS Practice Work - Task 1*

Час, витрачений на виконання завдання — 5 хвилин Завдання № 2 VNS Practice Work - Task 2



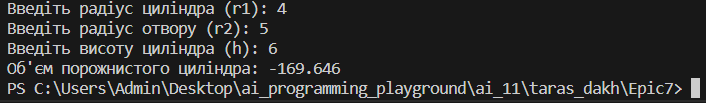
*Рисунок 12: Результати виконання VNS Practice Work - Task 2*

Час, витрачений на виконання завдання — 30 хвилин Завдання № 3 VNS Practice Work - Task 3.1



*Рисунок 13: Результати виконання VNS Practice Work - Task 3.1*

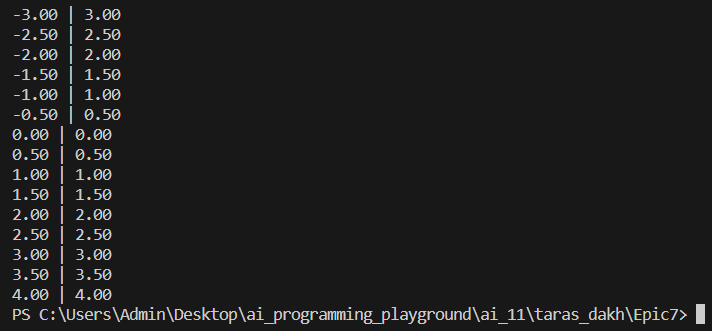
Час, витрачений на виконання завдання — 15 хвилин Завдання № 4 VNS Practice Work - Task 3.2



*Рисунок 14: Результати виконання VNS Practice Work - Task 3.2*

Час, витрачений на виконання завдання — 10 хвилин

Завдання № 5 VNS Practice Work - Task 4



*Рисунок 15: Результати виконання VNS Practice Work - Task 4*

Час, витрачений на виконання завдання — 40 хвилин

## Відповіді на контрольні запитання:

1. Назвіть основні властивості алгоритму.

Дискретність (атомарність), визначеність, виконуваність, скінченність, результативність, масовість, ефективність.

1. Що таке алгоритм?

Точно описана, скінченна послідовність дій, яка при виконанні дає необхідний виконавцю результат.

1. Визначте основні етапи розробки алгоритмів.

Постановка завдання, вибір методу вирішення, розробка алгоритму, аналіз алгоритму.

1. Перелічить базові конструкції. Алфавіт, лексеми, вирази та оператори
2. Перелічіть складні базові конструкції.

Оператори галуження: If (else-if, else), switch-case, goto;

Цикли: while, do-while, for

1. Дайте визначення конструкції розгалуження.

Конструкції розгалуження дозволяють виконувати певну частину алгоритму лише при досягненні деяких необхідних умов.

1. Дайте визначення конструкції цикл.

Цикли використовуються для виконання певної частини алгоритму, доки умова вірна, знову і знову.

1. Сформулюйте правило виконання циклу з передумовою.
   1. Спочатку обчислюється значення виразу-умови.
   2. Якщо значення виразу-умови істинне, то виконується тіло циклу.
   3. Після виконання тіла циклу значення виразу-умови обчислюється знову.
   4. Якщо значення виразу-умови істинне, то цикл повертається на крок 2.
   5. Якщо значення виразу-умови хибне, то цикл завершується.
2. Сформулюйте правило виконання циклу з відомою кількістю повторювань тіла циклу.
   1. Перед початком виконання циклу ініціалізується лічильник, який буде контролювати кількість повторень тіла циклу.
   2. Потім виконується тіло циклу.
   3. Після виконання тіла циклу лічильник інкрементується (збільшується на одиницю).
   4. Якщо лічильник не досяг заданої кількості повторень, то цикл повертається на крок 2.
   5. Якщо лічильник досяг заданої кількості повторень, то цикл завершується.
3. Що таке обчислювальна складність алгоритму?

Оцінка росту ресурсів (часу та пам’яті) для виконання алгоритму.

1. Як оцінити обчислювальну складність?

Для цього можна використати велику О-нотацію. Вона дозволяє взяти точну складність і відкинути всі члени, що мають степінь, менший за степінь

многочлена, що виражає точну складність. Це допоможе нам оцінити ріст ресурсів, необхідних для роботи програми.

1. Рекурсивні функції. Переваги їх використання.

Рекурсивні функції — функції, які викликають самі себе під час власного виконання. У деяких випадках це може бути ефективніше, ніж використання цикла з лічильником, а деякі задачі (наприклад, пов’язані з сортуванням, деревами і графами) неможливо розв’язати без рекурсії за прийнятний час.

Використання рекурсії може зменшити кількість часу, необхідного для виконання функції.

# Висновки:

Деталі по результатам виконання робіт та висновки згідно тем та завдань

Виконання лабораторної роботи дозволило ознайомитися з основними математичними задачами, які можуть бути розв'язані за допомогою програмування. Були вивчені основні алгоритми розв'язання математичних задач. Закріпили навички реалізації алгоритмів на мові програмування.

У результаті виконання лабораторної роботи було отримано наступні результати:

* Отримані теоретичні знання про розв'язання математичних задач за допомогою програмування.
* Володіння навичками реалізації алгоритмів на мові програмування.
* Уміння розв'язувати математичні задачі за допомогою програмування.