Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

A blue and white logo

Description automatically generated

**Звіт**

**Звіт**

**про виконання розрахунково-графічних робіт блоку № 7**

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

до:

ВНС Розрахунково-графічних робіт № 1-4

Практичних Робіт № 7

**Виконав:**

Студент групи ШІ-14

Шелеп Андрій Іванович

# **Мета роботи:**

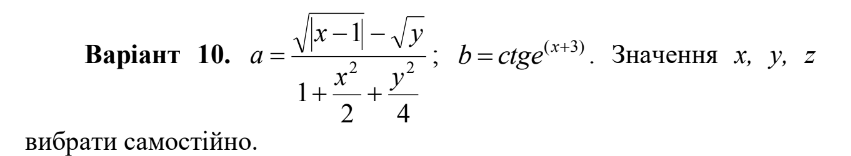
Одержати практичні навички в розробці і дослідженні алгоритмів розв’язання задач. Узагальнити та закріпити здобуті знання з курсу “Мови та парадигми програмування”.

# **Виконання роботи:**

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

Завдання №1 VNS Practice Work - Task 1

* Варіант 10
* Деталі завдання



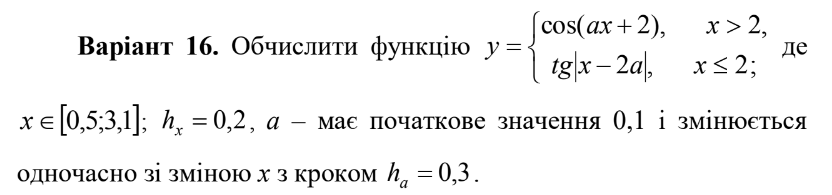
*Зображення №1. Деталі завдання №1 VNS Practice Work - Task 1*

* Важливі деталі для врахування в імплементації програми

Реалізовував функції для обчислення значення *a* та *b*

Завдання №2 VNS Practice Work - Task 2

* Варіант 16
* Деталі завдання



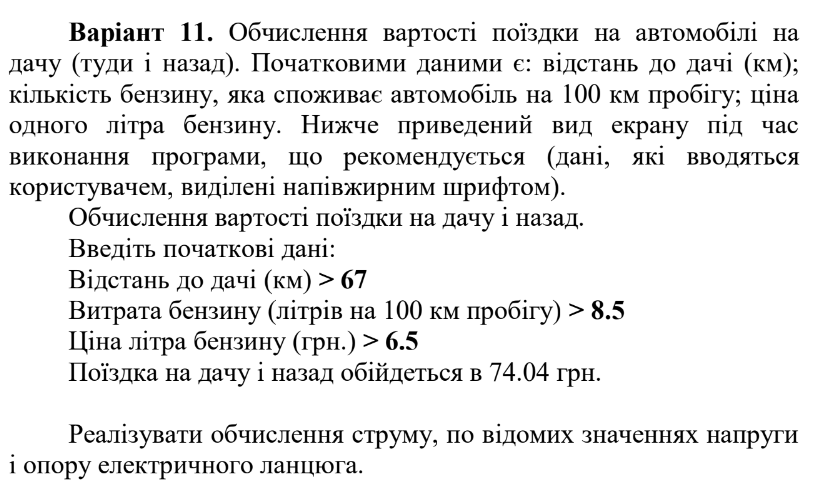
*Зображення №2. Деталі завдання №2 VNS Practice Work - Task 2*

* Важливі деталі для врахування в імплементації програми

Використовував цикл while для реалізації обчислення цієї функції

Завдання №3 VNS Practice Work - Task 3

* Варіант 11
* Деталі завдання



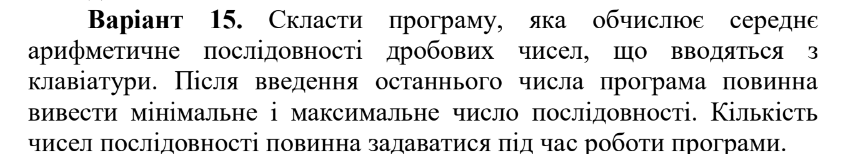
*Зображення №3. Деталі завдання №3 VNS Practice Work - Task 3*

* Важливі деталі для врахування в імплементації програми

Використовую тип даних з плаваючою комою

Завдання №4 VNS Practice Work - Task 4

* Варіант 15
* Деталі завдання



*Зображення №4. Деталі завдання №4 VNS Practice Work - Task 4*

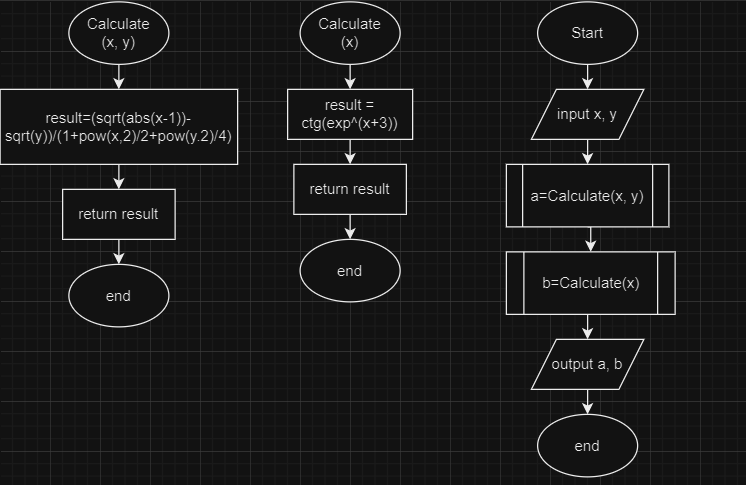
* Важливі деталі для врахування в імплементації програми

Для послідовності чисел використовував масив з виділеною динамічною пам’яттю

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

Програма №1 VNS Practice Work - Task 1

* Блок-схема



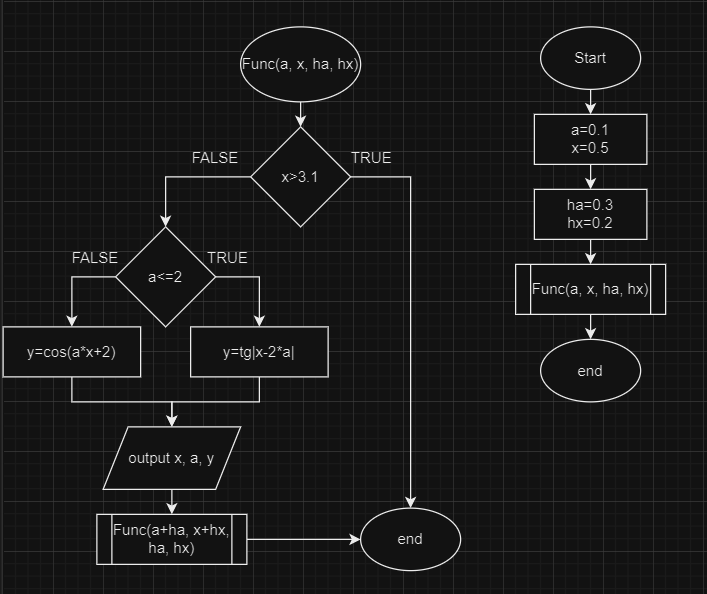
*Зображення №5. Блок-схема до програми №1 VNS Practice Work - Task 1*

* Планований час на реалізацію: 15 хв
* Важливі деталі для врахування в імплементації

Реалізовував функції для обчислення значення *a* та *b*

Програма №2 VNS Practice Work - Task 2

* Блок-схема



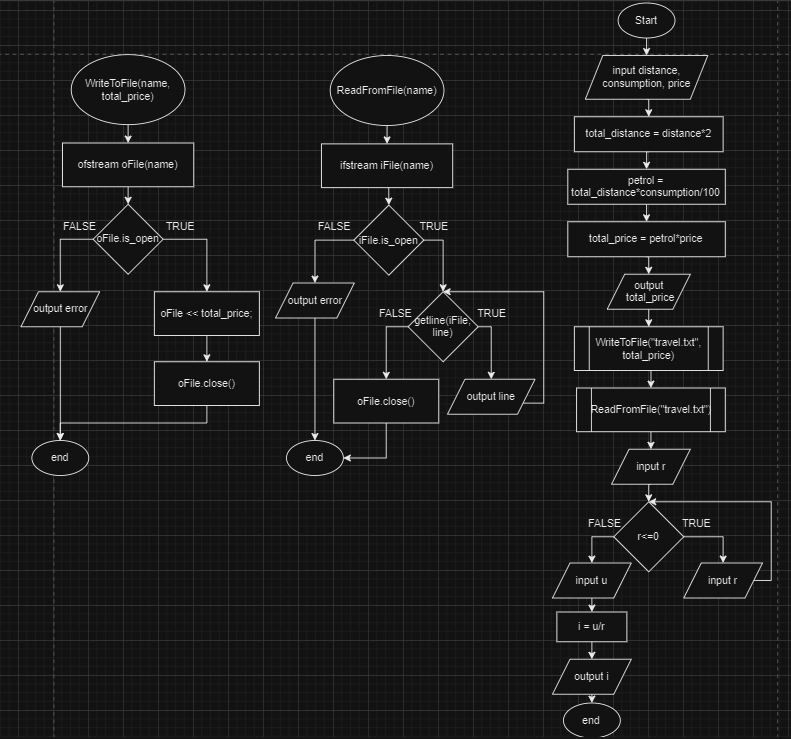
*Зображення №6. Блок-схема до програми №2 VNS Practice Work - Task 2*

* Планований час на реалізацію: 15 хв
* Важливі деталі для врахування в імплементації

Використовував цикл while для реалізації обчислення цієї функції

Програма №3 VNS Practice Work - Task 3

* Блок-схема



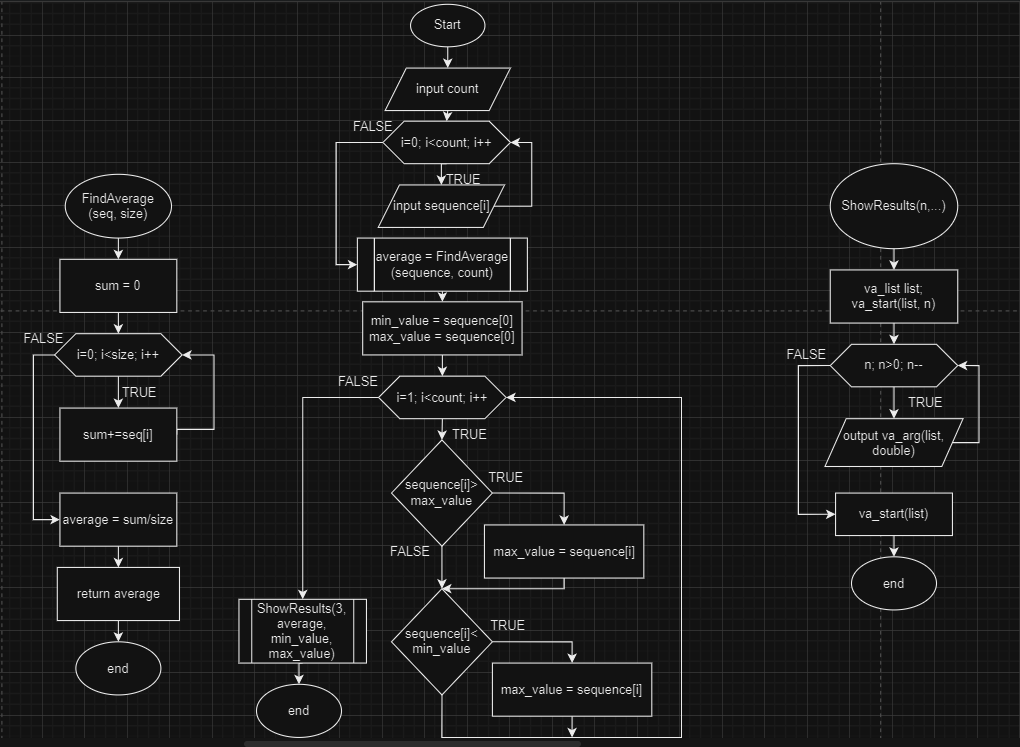
*Зображення №7. Блок-схема до програми №3 VNS Practice Work - Task 3*

* Планований час на реалізацію: 10 хв
* Важливі деталі для врахування в імплементації

Використовую тип даних з плаваючою комою

Програма №4 VNS Practice Work - Task 4

* Блок-схема



*Зображення №8. Блок-схема до програми №4 VNS Practice Work - Task 4*

* Планований час на реалізацію: 15 хв
* Важливі деталі для врахування в імплементації

Для послідовності чисел використовував масив з виділеною динамічною пам’яттю

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

Посилання на файл програми у пул-запиті GitHub: <https://github.com/artificial-intelligence-department/ai_programming_playground/pull/872>

Завдання №1 VNS Practice Work - Task 1

#include <iostream>

#include <cmath> //підключаю заголовковий файл cmath, щоб мати змогу прауювати з математичними функціями

*double* Calculate(*double* *x*, *double* *y*) // функція для обчислення значення a

{

*double* result = (sqrt(abs(*x*-1))-sqrt(*y*))/(1+pow(*x*,2)/2+pow(*y*,2)/4); //використовую sqrt для кореня та функцію pow для підняття до степеня

return result;

}

*double* Calculate(*double* *x*) // перевантажена функція для обчислення значення b

{

*double* result = 1/tan(exp(*x*+3)); //реалізовую ctg (котангенс) через вираз 1/tg

return result;

}

using *namespace* std;

*int* main(){

*double* x, y;

cout << "Введіть значення x: ";

cin >> x; //вводимо значення x

cout << "Введіть значення y: ";

cin >> y; //вводимо значення y

*double* a = Calculate(x, y); //викликаємо функцію для обчислення значення a

*double* b = Calculate(x); //викликаємо функцію для обчислення значення b

cout << endl << " Results" << endl; //виводимо результати

cout << "a = " << a << endl;

cout << "b = " << b << endl;

return 0;

}

*Блок №1. Код до завдання №1 VNS Practice Work - Task 1*

Завдання №2 VNS Practice Work - Task 2

#include <iostream>

#include <cmath> //підключаю заголовковий файл cmath, щоб мати змогу прауювати з математичними функціями

using *namespace* std;

const *double* eps = 1e-9;

*void* Func(*double* *a*, *double* *x*, const *double* *ha*, const *double* *hx*)

{

*double* y;

if(*x*>3.1+eps) //умова виходу з рекурсивної функції

{

return;

} else{

if(*a*<=2) //якщо а<=2, то функція обчислюється за одною формулою, в іншому випадку - за іншою

{

y = tan(abs(*x*-2\**a*));

} else{

y = cos(*a*\**x*+2);

}

cout << "x = " << *x* << "\t\ta = " << *a* << "\t\ty = " << y << endl; //виводжу результати

Func(*a*+*ha*, *x*+*hx*, *ha*, *hx*); //рекурсивно викликаю функцію зі збільшенням значення а та х на їх кроки

}

}

*int* main(){

*double* a=0.1, x=0.5, y; //ініціалізую змінним a, x початкові значення

const *double* hx = 0.2, ha = 0.3; //hx - крок, на який змінюється х; ha - крок, на який змінюється а; eps - змінна з дуже малим значенням для порівняння чисел типу double

Func(a, x, ha, hx); //викликаю функцію для виведення результатів обчислення

return 0;

}

*Блок №2. Код до завдання №2 VNS Practice Work - Task 2*

Завдання №3 VNS Practice Work - Task 3

#include <iostream>

#include <string> //для роботи з рядками

#include <iostream>

#include <string> //для роботи з рядками

#include <iomanip> //підключаю для використання setprecision

#include <fstream> //для роботи з файлами

using *namespace* std;

*struct* *Travel* //структура Travel

{

*int* distance;

*float* consumption;

*float* price;

};

*void* WriteToFile(const *char*\* *name*, *double* *total\_price*)

{

*ofstream* ofile(*name*); //відкриваю файл для запису

if(ofile.is\_open()) //перевіряю чи файл відкрито

{

ofile << "Вартість поїздки: " << fixed << setprecision(2) << *total\_price* << " грн."; //записую у файл вартість поїздки

cout << "Запис у файл пройшов успішно!" << endl;

} else {

cout << "error opening file" << endl;

}

ofile.close(); // закриваю файл

}

*void* ReadFromFile(const *char*\* *name*)

{

*ifstream* ifile(*name*); //відкриваю файл для зчитування

if(ifile.is\_open()) //перевіряю чи файл відкрито

{

*string* line;

while(getline(ifile, line)) //зчитую файл поки не дійду до кінця

{

cout << line << endl; //виводжу вмістиме

}

} else {

cout << "error opening file" << endl;

return;

}

ifile.close();

}

*int* main(){

*Travel* travel;

cout << " Обчислення вартості поїздки на дачу і назад" << endl;

cout << "Введіть початкові дані:" << endl;

cout << "Відстань до дачі (км): ";

cin >> travel.distance; //вводимо відстань до дачі

cout << "Витрата бензину (літрів на 100 км пробігу): ";

cin >> travel.consumption; //вводимо розхід палива

cout << "Ціна літра бензину (грн): ";

cin >> travel.price; //вводимо ціну 1 літрa палива

*int* total\_distance = travel.distance\*2; //шукаємо загальну відстань (відстань до дачі і назад)

*float* petrol = total\_distance\*travel.consumption/100; //обчислюємо скільки літрів палива витратиться на поїздку

*float* total\_price = petrol\*travel.price; //обчислюємо ціну на цей об'єм палива

cout << "Поїздка на дачу і назад обійдеться в " << fixed << setprecision(2) << total\_price << " грн." << endl; //виводимо результат; використано setprcision для виведення 2-ох знаків після коми

WriteToFile("travel.txt", total\_price);

ReadFromFile("travel.txt");

cout << endl << " Обчислення сили струму" << endl; //завдання з обчисленням сили струму

*float* i, u, r;

cout << "Введіть значення опору у Ом: ";

cin >> r; //вводимо значення опору

while(r<=0) //створив цикл; якщо користувач введе від'ємний або нульовий опір, то просимо його ввести ще раз поки він не введе додатнє значення

{

cout << "Опір не може бути від'ємним чи дорівнювати нулю. Введіть ще раз значення опору: ";

cin >> r;

}

cout << "Введіть значення напруги у В: ";

cin >> u; //вводимо значення напруги

i = u/r; //знаходимо силу струму за законом Ома

cout << "Сила струму дорівнює " << setprecision(3) << i << " A" << endl; //виводимо силу струму у амперах, використовуючи setprecision для виведення трьох знаків після коми

return 0;

}

*Блок №3. Код до завдання №3 VNS Practice Work - Task 3*

Завдання №4 VNS Practice Work - Task 4

#include <iostream>

#include <cstdarg> //для роботи з функціями зі змінною кількістю параметрів

using *namespace* std;

*void* showResults(*int* *n*,...) //функція зі змінною кількістю параметрів, що виводить їх значення. Використовую еліпсис у цій функції

{

*va\_list* list;

va\_start(list, n);

for(; *n*>0; *n*--)

{

cout << " " << va\_arg(list, *double*) << "\t";

}

va\_end(list);

}

*double* FindAverage(*double*\* *seq*, *int* *size*) // функція для обчислення середнього арифметичного послідовності чисел

{

*double* average;

*double* sum = 0;

for(*int* i = 0; i < *size*; i++){

sum+=*seq*[i]; //додаю всі числа

}

average = sum/*size*; //ділю суму всіх чисел на їх кількість та отримую середнє

return average;

}

*int* main(){

*int* count;

cout << "Введіть кількість чисел: ";

cin >> count; //вводимо кількість чисел

*double*\* sequence = new *double*[count]; //використовую динамічний масив для дробових чисел

cout << "Введіть числа: " << endl;

for(*int* i = 0; i < count; i++){

cin >> sequence[i]; //вводимо ці числа

}

*double* average = FindAverage(sequence, count); //викликаємо функцію знаходження середнього арифметичного

*double* min\_value = sequence[0]; //ініціалізовую для мінімального і максимального значення перше число

*double* max\_value = sequence[0];

*int* i = 0;

for(*int* i = 1; i < count; i++)

{

if(sequence[i]<min\_value) //якщо наступне число більше за мінімальне, то оновлюю мінімальне значення

min\_value = sequence[i];

if(sequence[i]>max\_value) //якщо наступне число більше за максимальне, то оновлюю максимальне значення

max\_value=sequence[i];

}

cout << "\tRESULTS" << endl;

cout << "average\t min\t max" << endl;

showResults(3, average, min\_value, max\_value); //виклик функції для виведення результатів

delete[] sequence; //звільняю пам'ять, виділену для масиву

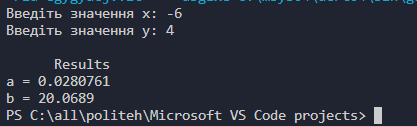
return 0;

}

*Блок №4. Код до завдання №4 VNS Practice Work - Task 4*

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

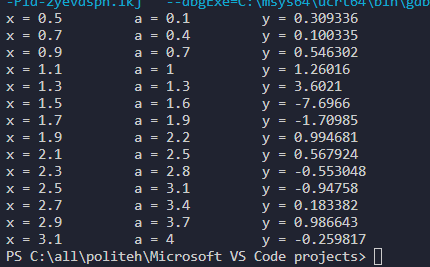
Завдання №1 VNS Practice Work - Task 1



*Зображення №9. Результати виконання завдання №1 VNS Practice Work - Task 1*

Час затрачений на виконання завдання: 15 хв

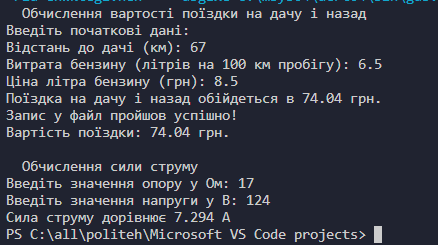
Завдання №2 VNS Practice Work - Task 2



*Зображення №10. Результати виконання завдання №2 VNS Practice Work - Task* *2*

Час затрачений на виконання завдання: 15 хв

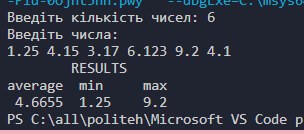
Завдання №3 VNS Practice Work - Task 3



*Зображення №11. Результати виконання завдання №3 VNS Practice Work - Task 3*

Час затрачений на виконання завдання: 10 хв

Завдання №4 VNS Practice Work - Task 4



*Зображення №12. Результати виконання завдання №4 VNS Practice Work - Task 4*

Час затрачений на виконання завдання: 15 хв

# **Висновки:**

За допомогою здобутих знань з курсу написано та реалізовано програми для поставлених задач у цій розрахунковій роботі.