

Міністерство освіти і науки України  
Національний університет «Львівська політехніка»  
Кафедра систем штучного інтелекту



## Звіт

**про виконання лабораторних та практичних робіт блоку № 3**

На тему: «Цикли. Вкладені Цикли. Завершення виконання циклів. Функції.  
Простір імен. Перевантаження функцій. Функції зі змінною кількістю  
параметрів (еліпсис). Рекурсія. Вбудовані функції.»

**з дисципліни:** «Основи програмування»

до:

Практичних Робіт до блоку № 3

**Виконала:**

Студентка групи ШІ-12  
Лазаревич Юлія Дмитрівна

Львів 2024

## Тема роботи:

Цикли. Вкладені цикли. Завершення виконання циклів. Функції. Простір імен. Перевантаження функції. Функції зі змінною кількістю параметрів. Рекурсія. Вбудовані функції.

## Мета роботи:

Ознайомитись з циклами, вкладеними циклами, функціями, простором імен, перевантаженням функції, функціями зі змінною кількістю параметрів, рекурсією, вбудованими функціями в мовах C та C++.

## Теоретичні відомості:

### 1. Теоретичні відомості з переліком важливих тем:

Тема №1: Введення в Цикли та їх Види в C++

Тема №2: Управління Виконанням Циклів

Тема №3: Вкладені Цикли

Тема №4: Основи Функцій у C++

Тема №5: Перевантаження Функцій та Простір Імен

Тема №6: Розширені Можливості Функцій

Тема №7: Вбудовані Функції в C++

### 2. Індивідуальний план опрацювання теорії:

#### ○ Тема №1: Введення в Цикли та їх Види в C++.

##### ○ Джерела Інформації:

- [Цикли. Оператори циклу for, while, do...while](#)
- [Урок №6 - Цикли та оператори в них \(For, While, Do While\)](#)

##### ○ Що опрацьовано:

##### ▪ Оператори циклу у мові C++:

- for (ініціалізація; вираз; приріст){//}
- while (вираз){//}
- do{//} while (вираз);

##### ▪ Також, існують такі цикли:

- Вкладені цикли(далі детальніше)
- Нескінчений цикл – це цикл, який ніколи не закінчується. Його часто можна написати випадково/помилково.

##### ○ Статус: ознайомлена.

##### ○ Початок опрацювання теми: 20.09.24

- Завершення опрацювання теми 25.09.24
- Тема №2: Управління Виконанням Циклів.
  - Джерела Інформації:
    - [Урок №73. Оператори break і continue](#)
  - Що опрацьовано:
    - Оператор break - використовується для негайного припинення виконання циклу. Коли цикл зустрічає *break*, виконання виходить з циклу, і програма продовжує виконання коду, що йде після циклу.
    - Оператор continue - використовується для пропуску поточної ітерації циклу та переходу до наступної ітерації. Це корисно, коли необхідно пропустити частину коду в циклі за певної умови.
  - Статус: ознайомлена.
  - Початок опрацювання теми: 30.09.24
  - Завершення опрацювання теми 05.10.24
- Тема №3: Вкладені Цикли
  - Джерела Інформації:
    - [Цикли. Оператори циклу for, while, do...while](#)
  - Що опрацьовано:
    - Вкладені цикли - цикли, які знаходяться всередині інших циклів. Можуть використовуватися, при роботі з багатовимірними структурами даних, таких як матриці. Вони дозволяють виконувати операції з кожним елементом такої структури, проходячи через кожен рівень вкладеності.
  - Статус: ознайомлена.
  - Початок опрацювання теми: 20.09.24
  - Завершення опрацювання теми 25.09.24
- Тема №4: Основи Функцій у C++
  - Джерела Інформації:
    - [Урок №15. Функції](#)
    - [Функції. Частина 1.](#)
  - Що опрацьовано:
    - Функція - це блок коду, який виконує певне завдання і може бути викликаний з різних частин програми. Функція має ім'я, тип повертаємого значення, список параметрів (аргументів) і тіло.

- Загальна форма запису:
 

```
тип ім'я_функції(список_параметрів або void)
{
    тіло_функції
    [return] (вираз);
}
```
- Область видимості формальних параметрів функції визначається межами тіла функції, в якій вони описані.
- Статус: ознайомлена.
- Початок опрацювання теми: 27.09.24
- Завершення опрацювання теми 05.10.24

## ○ Тема №5: Перевантаження Функцій та Простір Імен

- Джерела Інформації:
  - [Перевантаження функцій.](#)
  - [Простори імен. Ключові слова namespace, using.](#)
- Що опрацьовано:
  - Перевантаження функцій – це оголошення функції з тим же іменем декілька разів. Щоб компілятор міг їх відрізнити, ці функції повинні відрізнитися між собою списком вхідних параметрів.
  - Відрізняються за:
    - кількістю параметрів;
    - якщо кількість параметрів однакова, то їх типами.
  - Простір імен – це область визначення змінних, типів, констант та функцій, яка об'єднана в єдиний іменований або неіменований блок. Використовується для організації коду і запобігання конфліктів імен у великих проектах. Дозволяє групувати пов'язані класи, функції і змінні під спільним ідентифікатором.
- Статус: ознайомлена.
- Початок опрацювання теми: 10.10.24
- Завершення опрацювання теми 17.10.24

## ○ Тема №6: Розширені Можливості Функцій

- Джерела Інформації:
  - Окремі відео на ютуб
  - Допомога копілота
- Що опрацьовано:
  - Функції в C++ можуть мати розширені можливості, які роблять код більш гнучким та ефективним. До таких можливостей належать аргументи за замовчуванням, шаблони функцій, рекурсія та функції як параметри.

- Аргументи за замовчуванням дозволяють задавати значення параметрів функції, які будуть використовуватись, якщо аргументи не передані при виклику функції. Це зменшує кількість перевантажень функцій і покращує читабельність коду.
- Шаблони функцій дозволяють створювати функції, які можуть працювати з різними типами даних, забезпечуючи повторне використання коду та зменшуючи кількість перевантажених функцій.
- Рекурсія дозволяє функції викликати саму себе, що особливо корисно для вирішення завдань, які можуть бути розбиті на менші підзадачі. Важливо уникати нескінченних рекурсій.
- Статус: ознайомлена.
- Початок опрацювання теми: 18.10.24
- Завершення опрацювання теми 25.10.24

#### ○ Тема №7: Вбудовані Функції в C++

- Джерела Інформації:
  - [Урок №107. Вбудовані функції](#)
- Що опрацьовано:
  - Вбудована функція - це функція, яку компілятор замінює її викликами на сам код функції під час компіляції. Щоб оголосити функцію як вбудовану, використовується ключове слово `inline` перед визначенням функції.
  - Переваги:
    - Вбудовані функції дозволяють зменшити витрати на виклик функції, що може значно підвищити продуктивність в програмі з частими викликами невеликих функцій;
    - Уникаючи витрат на збереження і відновлення стеку викликів і переходів до функції, програма виконується швидше.
  - Недоліки:
    - Оскільки код вбудованої функції копіюється в кожне місце її виклику, це може збільшити розмір виконуваного файлу, що може бути проблематичним для великих програм;
    - Вбудовані функції повинні бути невеликими і простими, інакше вигоди від їх використання можуть бути знижені.
- Статус: ознайомлена.
- Початок опрацювання теми: 12.10.24
- Завершення опрацювання теми 22.10.24

## Виконання роботи:

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

#### Завдання №1 - VNS Lab 2 - Task 1- 10.

*Деталі завдання:*

- Використовуючи оператор циклу, знайти суму елементів, зазначених у конкретному варіанті. Результат надрукувати, надавши відповідний заголовок.
- Знайти суму ряду з точністю  $\epsilon=0.0001$ , загальний член якого

$$a_n = \frac{n!}{(2n)!}$$

#### Завдання №2 - VNS Lab 3 - Task 1 - 10.

*Деталі завдання:*

- Для  $x$ , що змінюється від  $a$  до  $b$  з кроком  $(b-a)/k$ , де  $(k=10)$ , обчислити функцію  $f(x)$ , використовуючи її розклад в степеневий ряд у двох випадках:
  - а) для заданого  $n$ ;
  - б) для заданої точності  $\epsilon$  ( $\epsilon=0.0001$ ).
- Для порівняння знайти точне значення функції.

Функція	Діапазон зміни аргументу	$n$	Сума
$e^{\cos x} \cos(\sin x)$	$0.1 \leq x \leq 1$	20	$S = 1 + \frac{\cos x}{1!} + \dots + \frac{\cos n x}{n!}$

#### Завдання №3 – VNS Lab 7 - Task 1 - 10.

*Деталі завдання:*

- Розв'язати зазначене у варіанті завдання, використовуючи функції зі змінною кількістю параметрів:
- Написати функцію `kvaдр` зі змінною кількістю параметрів, що визначає кількість чисел, що є точними квадратами (2, 4, 9, 16, . . . ) типу `int`. Написати викликаючу функцію `main`, що звертається до функції `kvaдр` не менше трьох разів з кількістю параметрів 3, 7, 11.

#### Завдання №4 - VNS Lab 7 - Task 2 - 10.

*Деталі завдання:*

- Написати перевантажені функції й основну програму, що їх викликає.
- а) для віднімання десяткових дробів;
- б) для віднімання звичайних дробів.

#### Завдання №5 – Class Practice Work - Менеджмент бібліотеки.

*Деталі завдання:*

- Ви створюєте просту програму керування бібліотекою. Книги в бібліотеці є, користувачі можуть їх взяти або повернути.

*Програма повинна вміти*

- Перерахувати всі книги.
- Дозволити взяти книгу (за наявності).
- Дозволити повернення книги.

*Структури даних*

- Використовуйте масив або вектор для зберігання назв книг.
- Використовуйте інший масив або вектор для збереження стану доступності кожної книги.

*Мета Задачі*

Навчитися користуватися операторами циклів та функцією переходу на мітку:

- `for( ) { ... }`
- `for each`
- `while( ) { ... }`
- `do { ... } while( )`
- `go to`

*Вимоги:*

- `while`: продовжувати працювати, доки користувач не вирішить вийти.

- `do while`: Після кожної операції (позичити, повернути, перерахувати) запитуйте користувача, чи хоче він виконати іншу операцію. Якщо так, поверніться назад.
- `for`: список усіх книг за допомогою циклу.
- `for each`: перевірити наявність кожної книги.
- `goto`: якщо користувач вводить неправильний вибір, використовуйте `goto`, щоб перенаправити його до головного меню.

### Завдання №6 - Self Practice Work - Офісна Вулиця.

Зустрілися якось працівники великих компаній і почали... Обговорювати план вулиці. Виявляється, всі приміщення, які орендуватимуть ці компанії, збудують вздовж однієї вулиці.  $i$ -та компанія орендуватиме офіс довжиною  $l_i$  метрів. Офіси будуватимуть один за одним, починаючи з точки 0. Всі працівники приїжджатимуть на стоянку, яку побудують в точці 0, та будуть йти до офісів своїх компаній.

Тобто, якщо офіси будуть збудовані в порядку  $p_1, p_2, \dots, p_n$ , то перший офіс почнеться в точці 0 і закінчиться в точці  $l_{p_1}$ , другий почнеться в  $l_{p_1}$  і закінчиться в  $l_{p_1} + l_{p_2}$  і т.д. Двері кожного офісу завжди є в кінці будинку, який є ближчим до стоянки.

Ваше завдання — допомогти розмістити офіси компаній на цій вулиці в такому порядку, щоб сумарна відстань від точки 0 до усіх офісів була мінімальною.

#### ***Вхідні дані***

У першому рядку задане ціле число  $n$  — кількість компаній.

У наступному рядку задано  $n$  цілих чисел  $l_i$  через пробіл — довжини офісів усіх компаній.

#### ***Вихідні дані***

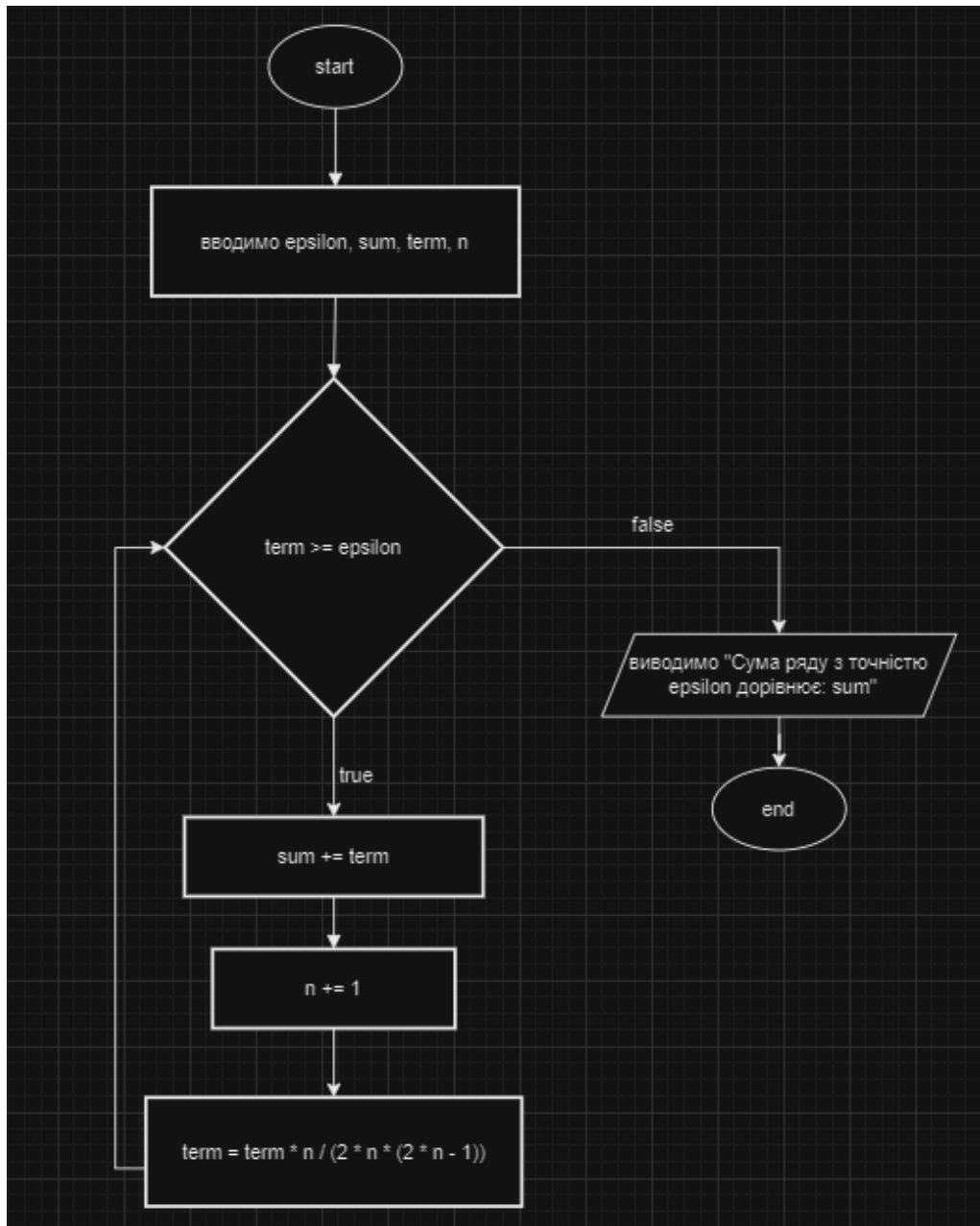
У єдиному рядку виведіть  $n$  чисел від 1 до  $n$  — порядок компаній, в якому варто будувати офіси.

Якщо існує декілька оптимальних порядків — виведіть будь-який із них.



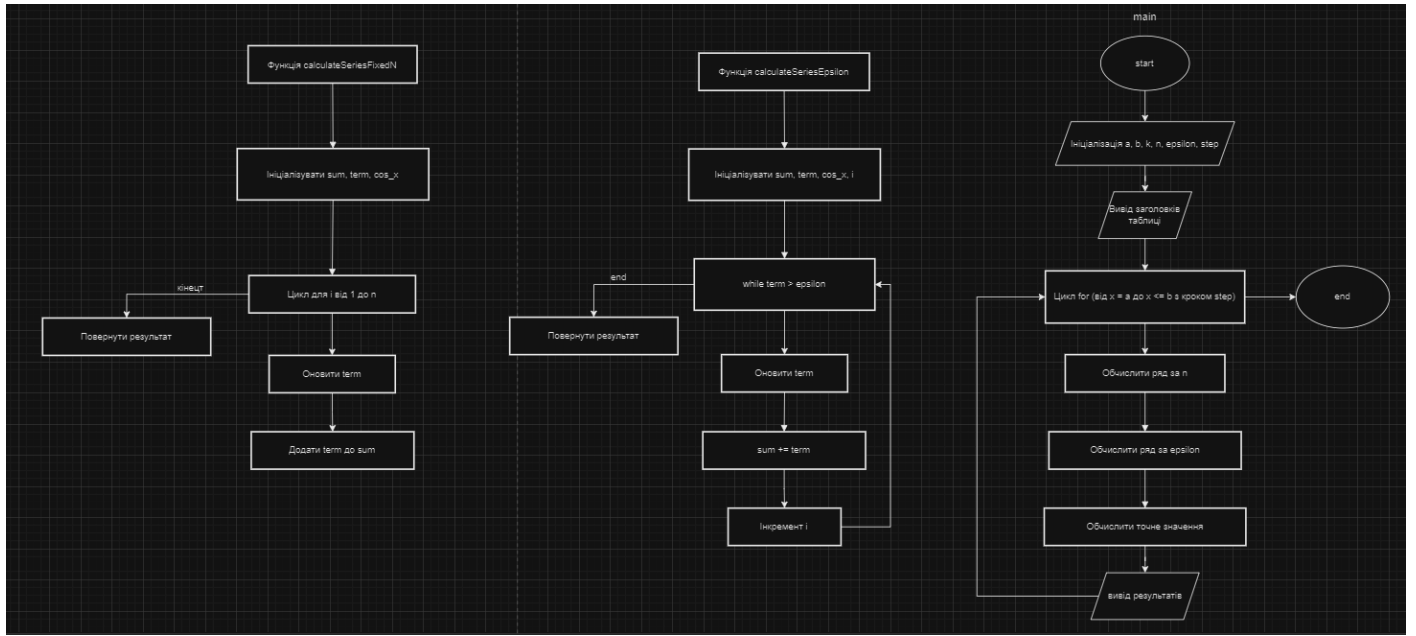
## 2. Дизайн та планована оцінка часу виконання завдань:

Програма №1 - VNS Lab 2 - Task 1- 10.



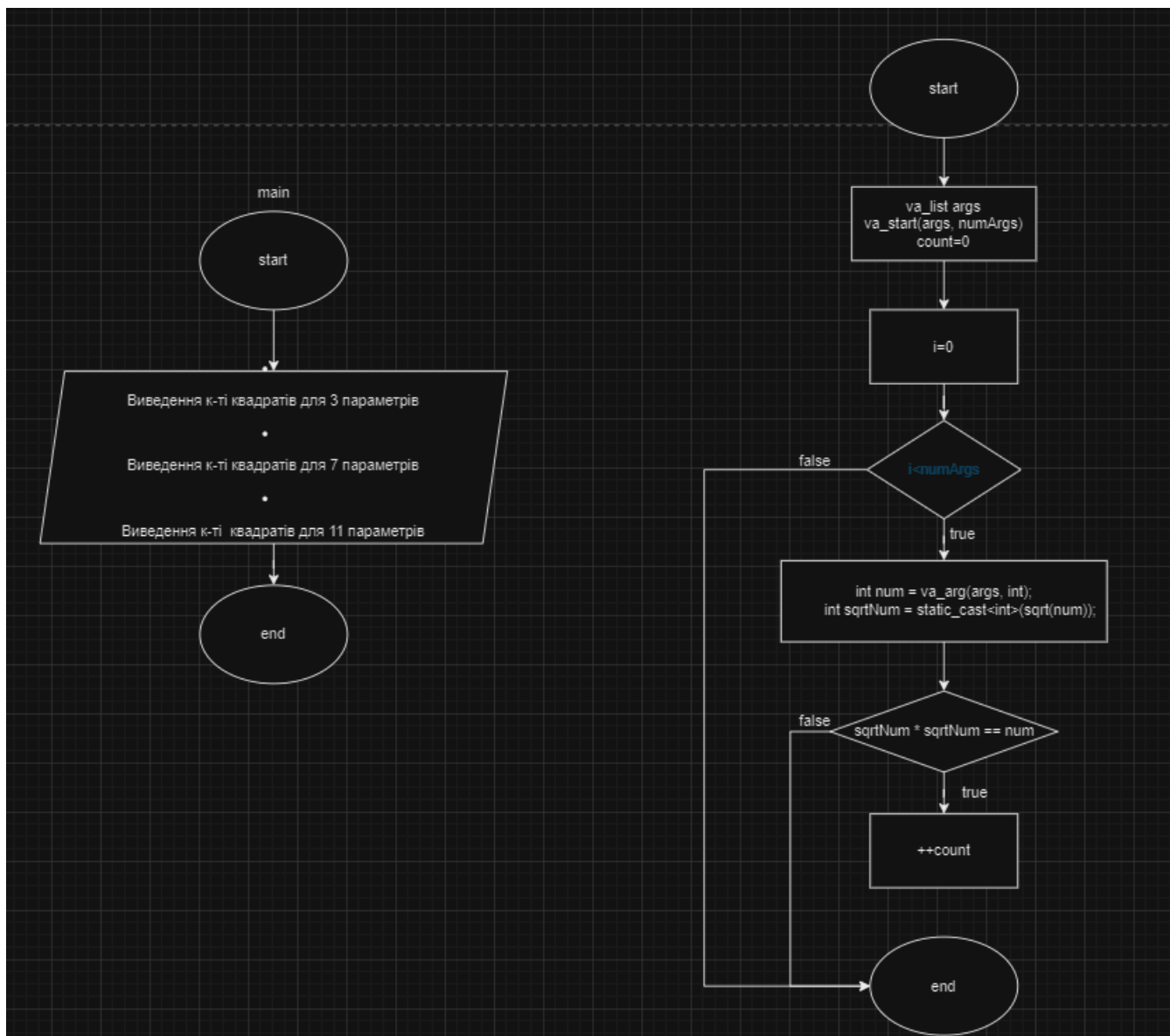
Плановий час на реалізацію: 30 хвилин

## Програма №2 - VNS Lab 3 - Task 1 - 10.



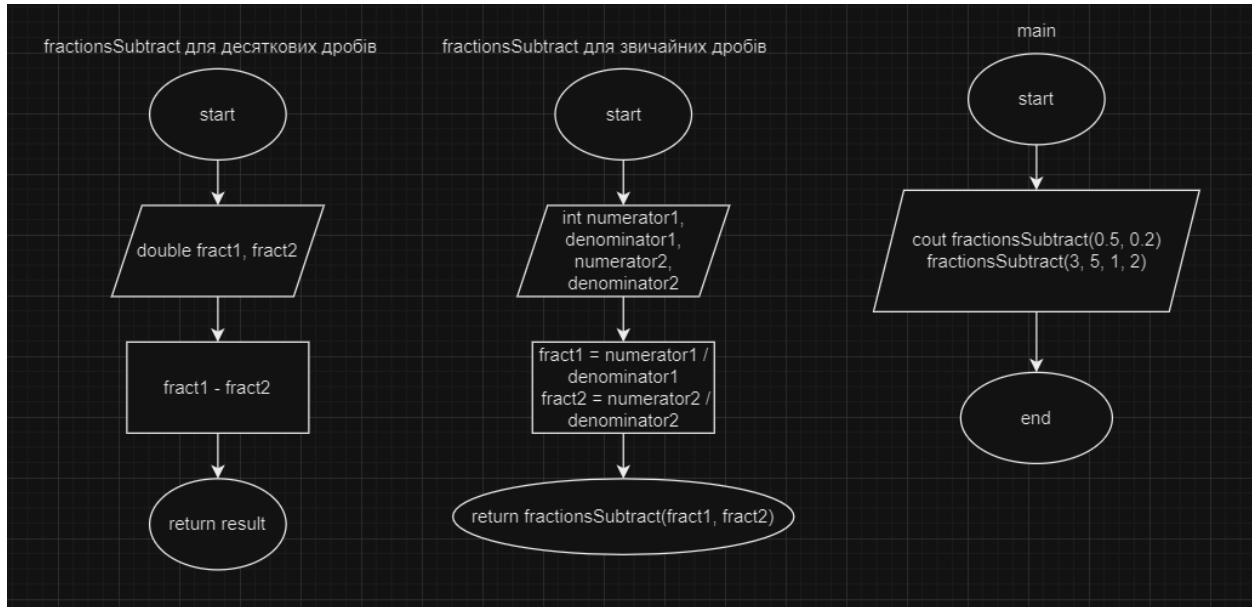
Плановий час на реалізацію: 2 години

Програма №3 – VNS Lab 7 - Task 1 - 10.



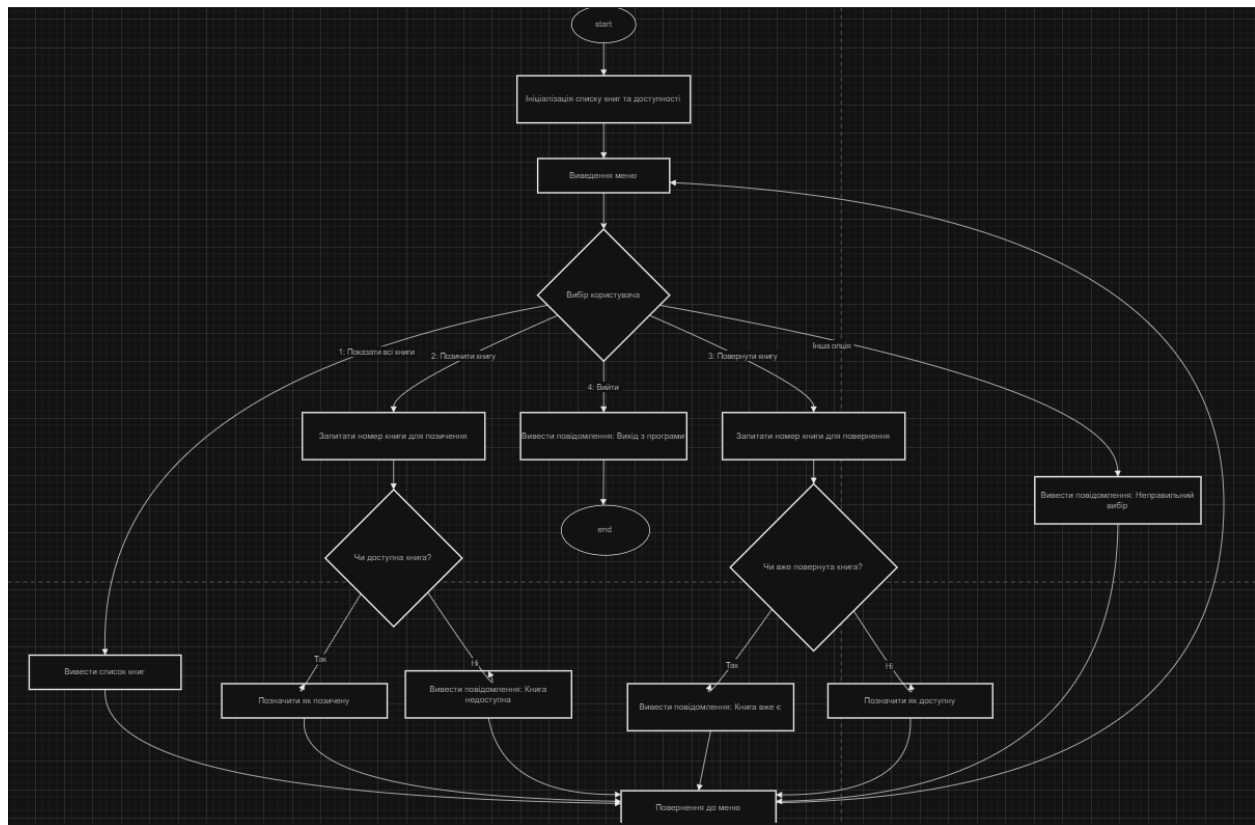
Плановий час на реалізацію: 30 хвилин

## Програма №4 - VNS Lab 7 - Task 2 - 10.



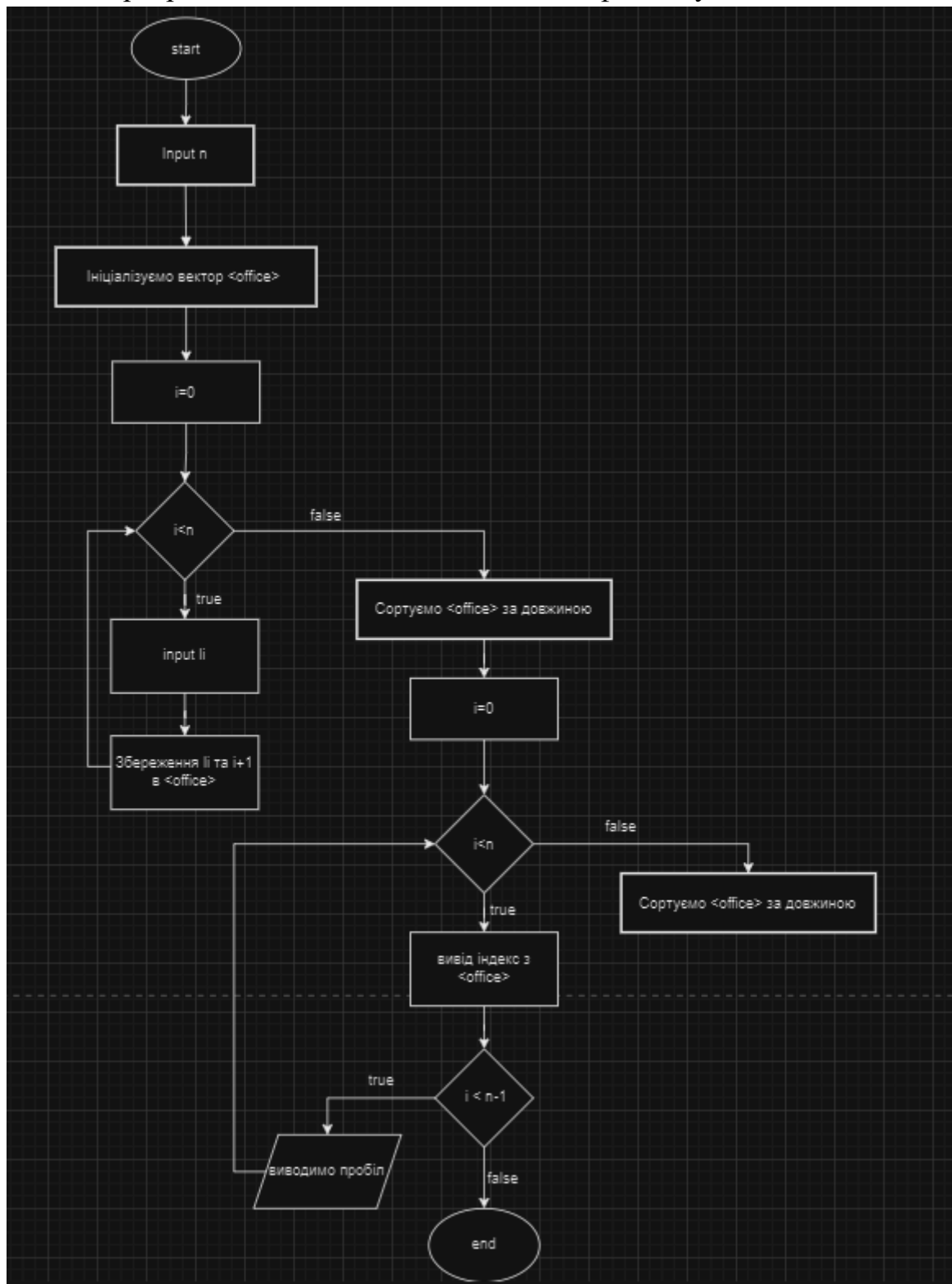
Плановий час на реалізацію: 30 хвилин

## Програма №5 - Class Practice Work – Менеджмент бібліотеки



Плановий час на реалізацію: 1 година

## Програма №6- Self Practice Work – Офісна Вулиця.



Плановий час на реалізацію: 2 години

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

#### Завдання №1 - VNS Lab 2 - Task 1- 10.

```
vns_lab_2_task_1_variant_1_yuliia_lazarevych.cpp > ...
1  #include <iostream>
2  using namespace std;
3
4  const double epsilon = 0.0001;
5
6  //використовую рекурсивну функцію для обчислення суми ряду
7  double calculateSum(double term, int n, double sum) {
8      if (term < epsilon) {
9          return sum;
10     }
11     term = term * static_cast<double>(n) / (2 * n * (2 * n - 1)); //новий член ряду
12     return calculateSum(term, n + 1, sum + term);
13 }
14
15 int main() {
16     double sum = 1.0;
17     double term = 1.0;
18     int n = 1;
19
20     sum = calculateSum(term, n, sum);
21
22     cout << "Сума ряду з точністю " << epsilon << " дорівнює: " << sum << endl;
23     return 0;
24 }
```

[ai-programming-playground 2024/ai 12/yuliia lazarevych/epic 3/codes/vns lab 2 task 1 variant 1 yuliia lazarevych.cpp](https://ai-programming-playground-2024/ai-12/yuliia-lazarevych/epic-3/codes/vns-lab-2-task-1-variant-1-yuliia-lazarevych.cpp) at epic 3 practice and labs yuliia lazarevych · artificial-intelligence-

#### Завдання №2 - VNS Lab 3 - Task 1- 10.

[ai-programming-playground 2024/ai 12/yuliia lazarevych/epic 3/codes/vns lab 3 task 1 variant 10 yuliia lazarevych.cpp](https://ai-programming-playground-2024/ai-12/yuliia-lazarevych/epic-3/codes/vns-lab-3-task-1-variant-10-yuliia-lazarevych.cpp) at epic 3 practice and labs yuliia lazarevych · artificial-intelligence-department/ai-programming-playground-2024

```

G+ vns_lab_3_task_1_variant_10_yuliia_jazarevych.cpp > ...
1  #include <iostream>
2  #include <cmath>
3  #include <iomanip>
4
5  using namespace std;
6
7  //використовую простір імен для обчислень ряду
8  namespace SeriesCalculations {
9      //використовую перевантажену функцію для обчислень ряду для фіксованого n
10     double calculateSeries(double x, int n) {
11         double sum = 1.0;
12         double cos_x = cos(x); //використала вбудовану ф-ю cos
13
14         for (int i = 1; i <= n; ++i) {
15             double term = 1.0;
16             for (int j = 1; j <= i; ++j) { //використовую вкладений цикл
17                 term *= cos_x / j;
18             }
19             sum += term;
20         }
21
22         return sum * cos(sin(x));
23     }
24
25     //використовую перевантажену функцію для обчислення ряду з точністю epsilon
26     double calculateSeries(double x, double epsilon) {
27         double sum = 1.0;
28         double term = 1.0;
29         double cos_x = cos(x);
30         int i = 1;
31
32         while (fabs(term) > epsilon) {
33             term *= cos_x / i;
34             sum += term;
35             ++i;
36         }
37
38         return sum * cos(sin(x));
39     }
40 }
41
42 int main() {
43     double a = 0.1, b = 1.0;
44     int k = 10;
45     int n = 20;
46     double epsilon = 0.0001;
47     double step = (b - a) / k;
48
49     cout << "x\tSeries (fixed n)\tSeries (epsilon)\tExact" << endl;
50     cout << fixed << setprecision(6);
51
52     for (double x = a; x <= b; x += step) {
53         //викликаю перевантажені функції
54         double seriesFixedN = SeriesCalculations::calculateSeries(x, n);
55         double seriesEpsilon = SeriesCalculations::calculateSeries(x, epsilon);
56         double exactValue = exp(cos(x)) * cos(sin(x));
57
58         cout << x << "\t" << seriesFixedN << "\t\t" << seriesEpsilon << "\t\t" << exactValue << endl;
59
60         if (fabs(seriesFixedN - exactValue) < epsilon) {
61             cout << "Досягнута точність для x = " << x << "." << endl;
62         }
63     }
64
65     return 0;
66 }

```

```

vns_lab_7_task_1_variant_10_yuliia_lazarevych.cpp > ...
1  #include <iostream>
2  #include <cmath>
3  #include <cstdarg>
4
5  using namespace std;
6
7  //Ф-я зі змінною кількістю параметрів (еліпсис)
8  int kvadr(int numArgs, ...) {
9      va_list args;
10     va_start(args, numArgs);
11
12     int count = 0;
13     for (int i = 0; i < numArgs; ++i) {
14         int num = va_arg(args, int);
15         int sqrtNum = static_cast<int>(sqrt(num));
16         if (sqrtNum * sqrtNum == num) {
17             ++count;
18         }
19     }
20
21     va_end(args);
22     return count;
23 }
24
25 int main() {
26     cout << "Кількість точних квадратів (3 параметри): " << kvadr(3, 2, 4, 9) << endl;
27
28     cout << "Кількість точних квадратів (7 параметрів): " << kvadr(7, 1, 4, 8, 9, 16, 23, 25) << endl;
29
30     cout << "Кількість точних квадратів (11 параметрів): " << kvadr(11, 3, 6, 9, 12, 16, 25, 36, 48, 49, 64, 100) << endl;
31
32     return 0;
33 }

```

[ai-programming-playground-2024.ai-12.yuliia-lazarevych/epic-3/codes/vns-lab-7-task-1-variant-10-yuliia-lazarevych.cpp](https://ai-programming-playground-2024.ai-12.yuliia-lazarevych/epic-3/codes/vns-lab-7-task-1-variant-10-yuliia-lazarevych.cpp) at epic 3 practice and labs yuliia-lazarevych · artificial-intelligence-department/ai-programming-playground-2024

#### Завдання №4 - VNS Lab 7 - Task 2- 10.

```

vns_lab_7_task_2_variant_10_yuliia_lazarevych.cpp > ...
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5  //юзаю перевантаження ф-ї
6  double fractionsSubtract(double fract1, double fract2)
7  {
8      return fract1 - fract2;
9  }
10
11 //юзаю перевантаження ф-ї
12 double fractionsSubtract(int numerator1, int denominator1, int numerator2, int denominator2)
13 {
14     double fract1 = (double)numerator1 / denominator1;
15     double fract2 = (double)numerator2 / denominator2;
16     return fractionsSubtract(fract1, fract2);
17 }
18
19 int main()
20 {
21     cout << "Віднімання десяткових дробів: " << fractionsSubtract(0.5, 0.2) << endl;
22     cout << "Віднімання звичайних дробів: " << fractionsSubtract(3,5,1,2) << endl;
23     return 0;
24 }

```

[ai-programming-playground-2024.ai-12.yuliia-lazarevych/epic-3/codes/vns-lab-7-task-2-variant-10-yuliia-lazarevych.cpp](https://ai-programming-playground-2024.ai-12.yuliia-lazarevych/epic-3/codes/vns-lab-7-task-2-variant-10-yuliia-lazarevych.cpp) at epic 3 practice and labs yuliia-lazarevych · artificial-intelligence-department/ai-programming-playground-2024



## Завдання №5 – Class Practice Work

```
practice_work_task_1_yulia_lazarevych.cpp > ...
1  #include <iostream>
2  #include <vector>
3  #include <string>
4
5  using namespace std;
6
7  void listBooks(const vector<string>& books, const vector<bool>& available) {
8      cout << "Список книг:" << endl;
9      for (size_t i = 0; i < books.size(); ++i) {
10         cout << i + 1 << ". " << books[i] << " - " << (available[i] ? "доступна" : "позичена") << endl;
11     }
12 }
13
14 void borrowBook(vector<bool>& available) {
15     int index;
16     cout << "Введіть номер книги, яку хочете позичити: ";
17     cin >> index;
18     if (index < 1 || index > available.size() || !available[index - 1]) {
19         cout << "Книга недоступна для позичення" << endl;
20     } else {
21         available[index - 1] = false;
22         cout << "Ви позичили книгу" << endl;
23     }
24 }
25
26 void returnBook(vector<bool>& available) {
27     int index;
28     cout << "Введіть номер книги, яку хочете повернути: ";
29     cin >> index;
30     if (index < 1 || index > available.size() || available[index - 1]) {
31         cout << "Ця книга вже є в бібліотеці" << endl;
32     } else {
33         available[index - 1] = true;
34         cout << "Ви повернули книгу" << endl;
35     }
36 }
37
38 int main() {
39     vector<string> books = {"Книга 1", "Книга 2", "Книга 3"};
40     vector<bool> available(books.size(), true);
41     int choice;
42
43     do {
44         menu:
45         cout << "\nМеню:\n";
46         cout << "1. Показати всі книги\n";
47         cout << "2. Позичити книгу\n";
48         cout << "3. Повернути книгу\n";
49         cout << "4. Вийти\n";
50         cout << "Виберіть опцію: ";
51         cin >> choice;
52
53         switch (choice) {
54             case 1:
55                 listBooks(books, available);
56                 break;
57             case 2:
58                 borrowBook(available);
59                 break;
60             case 3:
61                 returnBook(available);
62                 break;
63             case 4:
64                 cout << "Вихід з програми" << endl;
65                 break;
66             default:
67                 cout << "Неправильний вибір." << endl;
68                 goto menu;
69         }
70
71         char again;
72         cout << "Хочете виконати іншу операцію? (y/n): ";
73         cin >> again;
74         if (again == 'n' || again == 'N') break;
75     } while (true);
76
77     return 0;
78 }
79 }
```

[ai\\_programming\\_playground\\_2024/ai\\_12/yuliia\\_lazarevych/epic\\_3/codes/practice\\_work\\_task\\_1\\_yuliia\\_lazarevych.cpp](https://ai_programming_playground_2024/ai_12/yuliia_lazarevych/epic_3/codes/practice_work_task_1_yuliia_lazarevych.cpp) at epic\_3\_practice\_and\_labs\_yuliia\_lazarevych · artificial-intelligence-department/ai\_programming\_playground\_2024

### Завдання №6 - Self Practice Work – Офісна Вулиця.

```
self_practice_work_algotester_task_1_yuliia_lazarevych.cpp > ...
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4
5  using namespace std;
6
7  int main() {
8      int n;
9      cin >> n;
10
11     vector<pair<int, int>> office(n);
12
13     for (int i = 0; i < n; ++i) {
14         int li;
15         cin >> li;
16         office[i] = {li, i + 1};
17     }
18
19     sort(office.begin(), office.end());
20
21     for (int i = 0; i < n; ++i) {
22         cout << office[i].second;
23         if (i < n - 1) {
24             cout << " ";
25         }
26     }
27
28     return 0;
29 }
```

[ai\\_programming\\_playground\\_2024/ai\\_12/yuliia\\_lazarevych/epic\\_3/codes/self\\_practice\\_work\\_algotester\\_task\\_1\\_yuliia\\_lazarevych.cpp](https://ai_programming_playground_2024/ai_12/yuliia_lazarevych/epic_3/codes/self_practice_work_algotester_task_1_yuliia_lazarevych.cpp) at epic\_3\_practice\_and\_labs\_yuliia\_lazarevych · artificial-intelligence-department/ai\_programming\_playground\_2024

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

Завдання №1 - VNS Lab 2 - Task 1- 10.

```
Сума ряду з точністю 0.0001 дорівнює: 1.59226
PS D:\shi\epics\epic_3> █
```

Витрачений час: 40 хвилин

Завдання №2 - VNS Lab 3 - Task 1- 10.

x	Series (fixed n)	Series (epsilon)	Exact
0.100000	2.691268	2.691265	2.691268
Досягнута точність для $x = 0.100000$ .			
0.190000	2.622330	2.622328	2.622330
Досягнута точність для $x = 0.190000$ .			
0.280000	2.515252	2.515250	2.515252
Досягнута точність для $x = 0.280000$ .			
0.370000	2.376117	2.376116	2.376117
Досягнута точність для $x = 0.370000$ .			
0.460000	2.212426	2.212415	2.212426
Досягнута точність для $x = 0.460000$ .			
0.550000	2.032383	2.032377	2.032383
Досягнута точність для $x = 0.550000$ .			
0.640000	1.844196	1.844192	1.844196
Досягнута точність для $x = 0.640000$ .			
0.730000	1.655450	1.655448	1.655450
Досягнута точність для $x = 0.730000$ .			
0.820000	1.472641	1.472640	1.472641
Досягнута точність для $x = 0.820000$ .			
0.910000	1.300891	1.300886	1.300891
Досягнута точність для $x = 0.910000$ .			
1.000000	1.143836	1.143834	1.143836
Досягнута точність для $x = 1.000000$ .			

Витрачений час: 3 години

### Завдання №3 – VNS Lab 7 - Task 1- 10.

Кількість точних квадратів (3 параметри): 2  
Кількість точних квадратів (7 параметрів): 5  
Кількість точних квадратів (11 параметрів): 7

Витрачений час: 30 хвилин

### Завдання №4 – VNS Lab 7 - Task 2- 10.

Віднімання десяткових дробів: 0.3  
Віднімання звичайних дробів: 0.1

Витрачений час: 30 хвилин

### Завдання №5 - Class Practice Work - Особистий поради́ник.

```

Меню:
1. Показати всі книги
2. Позичити книгу
3. Повернути книгу
4. Вийти
Виберіть опцію: 1
Список книг:
1. Книга 1 - доступна
2. Книга 2 - доступна
3. Книга 3 - доступна
Хочете виконати іншу операцію? (y/n): y

Меню:
1. Показати всі книги
2. Позичити книгу
3. Повернути книгу
4. Вийти
Виберіть опцію: 2
Введіть номер книги, яку хочете позичити: 1
Ви позичили книгу
Хочете виконати іншу операцію? (y/n): y

Меню:
1. Показати всі книги
2. Позичити книгу
3. Повернути книгу
4. Вийти
Виберіть опцію: 3
Введіть номер книги, яку хочете повернути: 3
Ця книга вже є в бібліотеці
Хочете виконати іншу операцію? (y/n): n
PS D:\shi\epics\epic_3>

```

Витрачений час: 2 години

## Завдання №6 - Self Practice Work –Щасливий результат

```

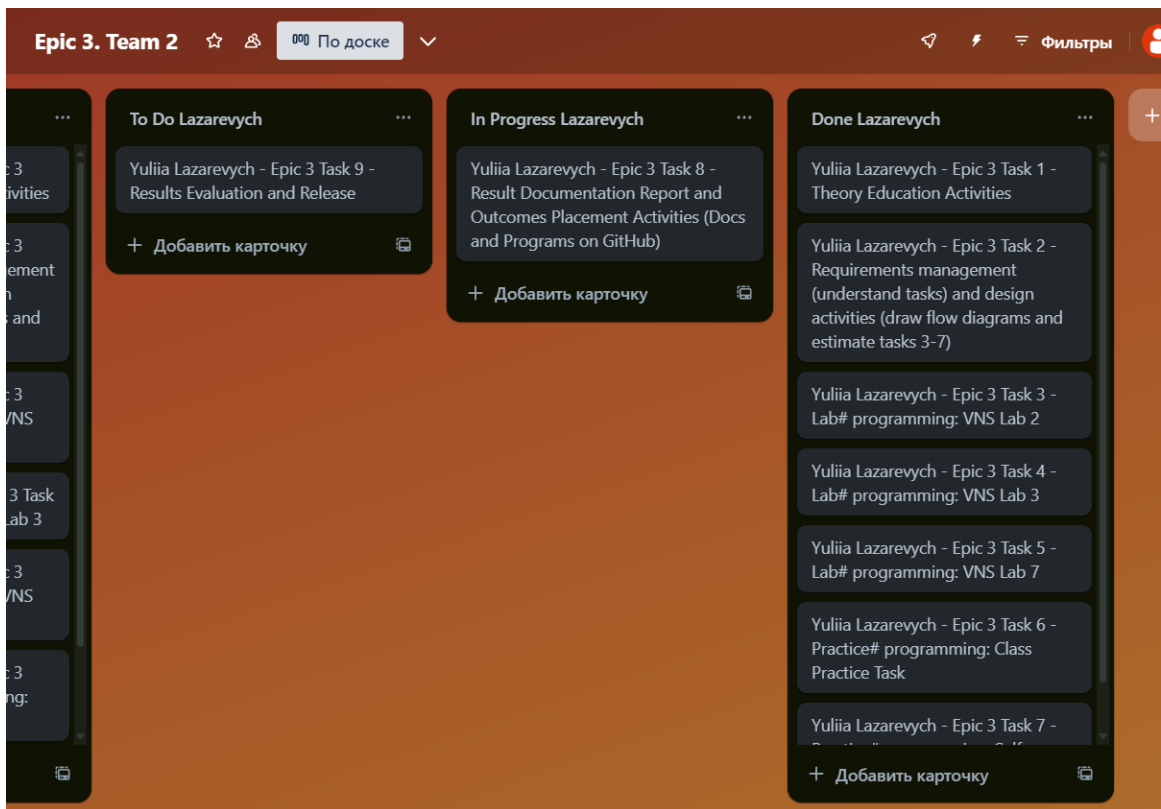
3
5
10
8
1 3 2

```

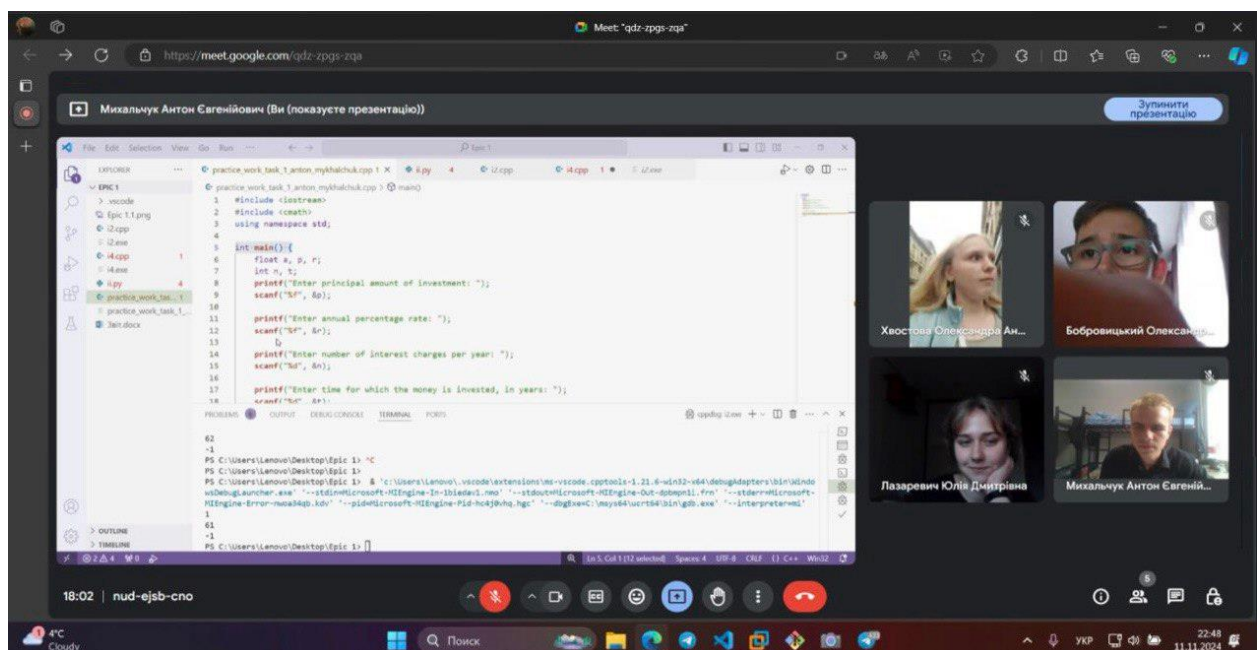
Витрачений час: 30 хвилин

### 6. Кооперація з командою:

- Скрін прогресу по Трелло



- Скрін з 2-ї зустрічі по обговоренню задач Епіку та Скрін прогресу по Тrello



**Висновки:** Виконуючи третій епік я ознайомилась з циклами, вкладеними циклами, функціями, простором імен, перевантаженням функції, функціями зі змінною кількістю параметрів, рекурсією, вбудованими функціями в мовах С та С++. Також я написала шість кодів де застосувала нові знання на практиці.