

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 з змінною кількістю
параметрів (еліпсис). Рекурсія. Вбудовані функції.»
з дисципліни: «Основи програмування»

до:

ВНС Лабораторної Роботи № 2
ВНС Лабораторної Роботи № 3
ВНС Лабораторної Роботи № 7
Практичних Робіт до блоку № 3

Виконав:

Студент групи ІІІ-11
Цяпа Остап Андрійович

Львів 2024

Тема роботи:

Цикли. Вкладені Цикли. Завершення виконання циклів. Функції. Простір імен. Перевантаження функцій. Функції з змінною кількістю параметрів (еліпсис). Рекурсія. Вбудовані функції.

Мета роботи:

Навчитися працювати з функціями, циклами та перевантаженими функціями. Застосувати на практиці вивчений теоретичний матеріал.

Теоретичні відомості:

- 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++

1) Індивідуальний план опрацювання теорії:

- Тема №1: Введення в цикли та їх види в C++:
 - Джерела інформації:
 - Відео.
<https://www.youtube.com/watch?v=zBtcqNdiRf4&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=31>
 - Що опрацьовано:
 - Значення та роль циклів у програмуванні.
 - Огляд видів циклів: for, while, do-while.
 - Синтаксис та основи використання кожного типу циклу.
 - Приклади базових циклів для різних задач.
 - Запланований час на вивчення 1 година.
Витрачений час 1 година.
- Тема №2: Управління виконанням циклів:
 - Джерела інформації:
 - Відео.
<https://www.youtube.com/watch?v=rj1OLsBKazA&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=36>
<https://www.youtube.com/watch?v=UY295pIdeoQ&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=37>
 - Що опрацьовано:
 - Застосування операторів break та continue.
 - Приклади та вправи з управлінням циклами.
- Запланований час на вивчення 30 хвилин.
Витрачений час 30 хвилин.
- Тема №3: Вкладені цикли:
 - Джерела інформації:
 - Відео.
<https://www.youtube.com/watch?v=mBPHKQx21eE>
 - Що опрацьовано
 - Поняття та важливість вкладених циклів.
 - Реалізація вкладених циклів: приклади для різних сценаріїв.

- Практичні завдання на вкладені цикли.
 - Запланований час на вивчення 1 година.
 - Витрачений час 1 година.
- Тема №4 Основи функцій у C++:
 - Джерела інформації:
 - Відео.
 - <https://www.youtube.com/watch?v=G8P6SvdqU9s&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=43>
 - Що опрацьовано
 - Визначення та оголошення функцій.
 - Повернення значень з функцій.
 - Приклади створення та використання функцій.
 - Запланований час на вивчення 2 години.
 - Витрачений час 2 години.
- Тема № 5 Перевантаження функцій та простір імен:
 - Джерела інформації:
 - Відео.
 - <https://www.youtube.com/watch?v=hcYgFCgeZzQ>
 - Що опрацьовано
 - Концепція перевантаження функцій.
 - Запланований час на вивчення 1 година.
 - Витрачений час 1 година..
- Тема №6: Розширені можливості функцій:
 - Джерела інформації:
 - Статті.
 - <https://acode.com.ua/urok-15-funktsiyi-i-operator-return/>
 - Що опрацьовано
 - Функції зі змінною кількістю параметрів (еліпсис): синтаксис та приклади.
 - Рекурсія: основи, приклади рекурсивних функцій та їх аналіз.
 - Запланований час на вивчення 1 година.
 - Витрачений час 1 година.
- Тема №7: Вбудовані функції в C++:
 - Джерела інформації:
 - Відео.
 - https://www.youtube.com/watch?v=V_8XRRlus7Y&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=49
 - Що опрацьовано
 - Огляд вбудованих функцій у C++.
 - Приклади використання стандартних функцій у програмуванні.
 - Роль вбудованих функцій у спрощенні коду.
 - Практичні завдання для розуміння вбудованих функцій.
 - Запланований час на вивчення 1 година.
 - Витрачений час 1 година.

Виконання роботи:

1. Опрацювання завдання та вимог до програм.

Завдання №1

VNS LAB 2 – TASK 1

9) Знайти суму ряду з точністю $\epsilon=0.0001$, загальний член якого

$$a_n = \frac{10^n}{n!}$$

Завдання №2

VNS LAB 3 – TASK 1

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 ϵ ($\epsilon=0.0001$).

Для порівняння знайти точне значення функції.

9	$y = \frac{1}{4} \ln \frac{1+x}{1-x} + \frac{1}{2} \arctg X$	$0,1 \leq x \leq 0,8$	3	$S = x + \frac{x^5}{5} + \dots + \frac{x^{4n+1}}{4n+1}$
---	--	-----------------------	---	---

Завдання №3 і №4

VNS LAB 7 – TASK 1

9. Написати функцію `max` зі змінною кількістю параметрів, що знаходить мінімальне із чисел типу `int` або із чисел типу `double`, тип параметрів визначається за допомогою першого параметра функції. Написати викликаючу функцію `main`, що звертається до функції `min` не менше трьох разів з кількістю параметрів 5, 10, 12.

VNS LAB 7 – TASK 2

Написати перевантажені функції й основну програму, що їх викликає.

9.

а) для додавання десяткових дробів;

б) для додавання звичайних дробів.

Завдання №5

Class Practice work

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

Деталі

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.

Вимоги:

1. while: продовжувати працювати, доки користувач не вирішить вийти.
2. do while: Після кожної операції (позичити, повернути, перерахувати) запитуйте користувача, чи хоче він виконати іншу операцію. Якщо так, поверніться назад.
3. for: список усіх книг за допомогою циклу.
4. for each: перевірити наявність кожної книги.
5. goto: якщо користувач вводить неправильний вибір, використовуйте goto, щоб перенаправити його до головного меню.

Завдання №6

Self practice work algotester

Мале Бісеня та Дракон полюбляють проводити дозвілля разом. Сьогодні вони грають в одну дуже цікаву гру.

У них є дошка, що складається з n рядків та m стовпців, всі клітинки якої білі. Гравці по черзі вибирають одну білу клітинку та зафарбовують її в чорний колір. Бісеня ходить першим. Гравець, який не може зробити хід, тобто на початку ходу якого вся дошка чорна, програє.

Погостривши зубки, Бісеня зрозуміло, що у Дракона велика перевага, адже він двоголовий, а, як то кажуть, «одна голова добре, а дві — краще». Тому воно просить вас допомогти. Вам потрібно сказати за заданими n та m , хто виграє у цій напруженій грі.

2. Дизайн та планувальна оцінка часу виконання завдань:

Програма №1

- Блок-схема
- Важливі деталі для реалізації програми
Використати цикл for для того щоб пройти по числах від 0 до 100 і в середині циклу прописати if, щоб додавали тільки ті числа, які при діленні на два не дають остачі, тобто парні числа.
- Плановий час на реалізацію одна година.

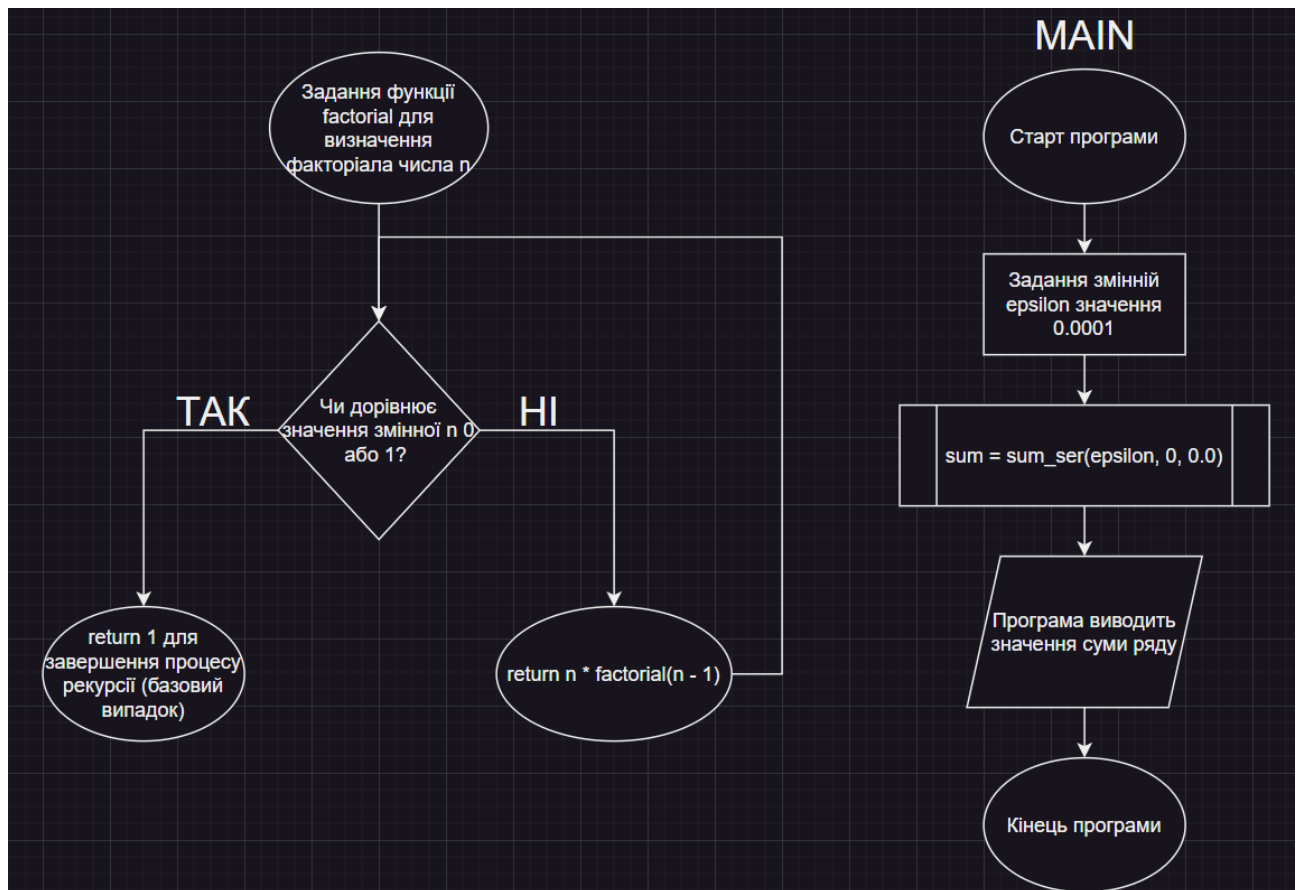


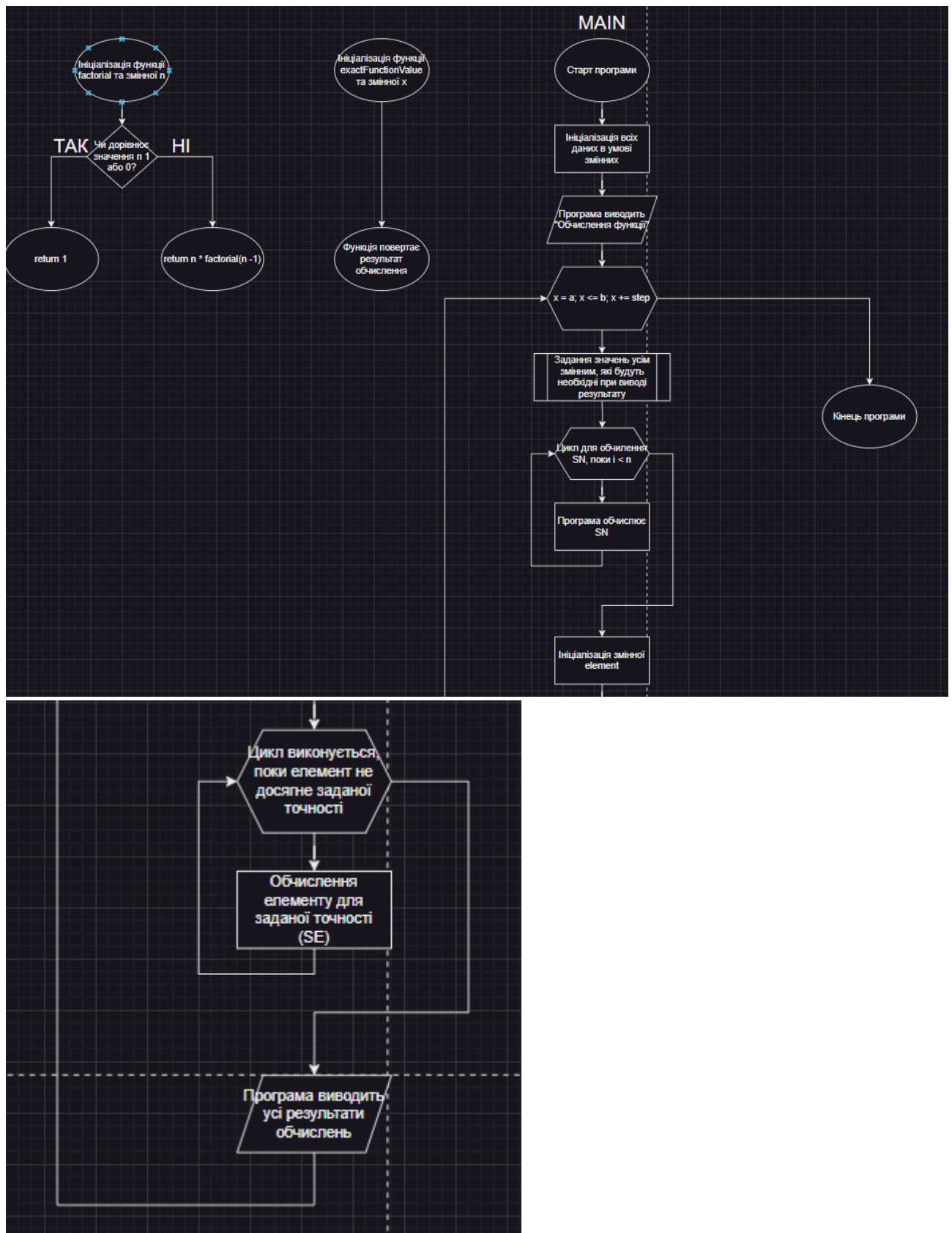
Рисунок 2.1. Блок-схема до програми 1

Програма №2

- Блок-схема
- Важливі деталі для реалізації програми

Написати функції для обрахунку значення з точністю ϵ ($\epsilon=0.0001$), функцію для значення суми SN, і функцію SE - для заданої точності, а також виводити на екран X- значення параметра; SN- значення суми для заданого n; SE- значення суми для заданої точності; Y-точне значення функції.

- Плановий час на реалізацію 2.5 години.



Рисунки 2.2. Блок-схема до програми 2

Програма №3

- Блок-схема
- Важливі деталі для реалізації програми
- Використати функції із декількома параметрами min
- Плановий час на реалізацію 1.5 години.

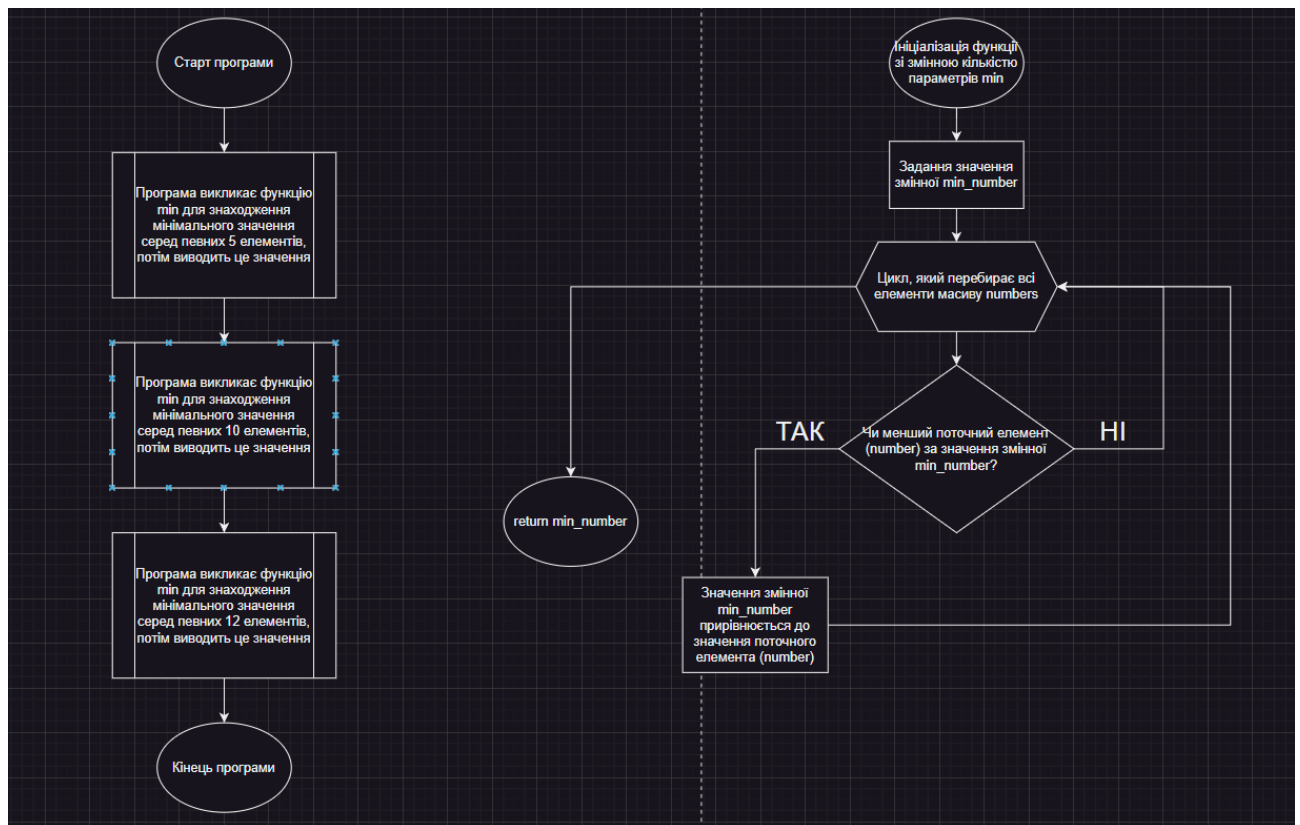


Рисунок 2.3. Блок-схема до програми 3

Програма №4

- Блок-схема
- Важливі деталі для реалізації програми
- Написати перевантажені функції та викликати їх у головній функції
- Плановий час на реалізацію 2 години.

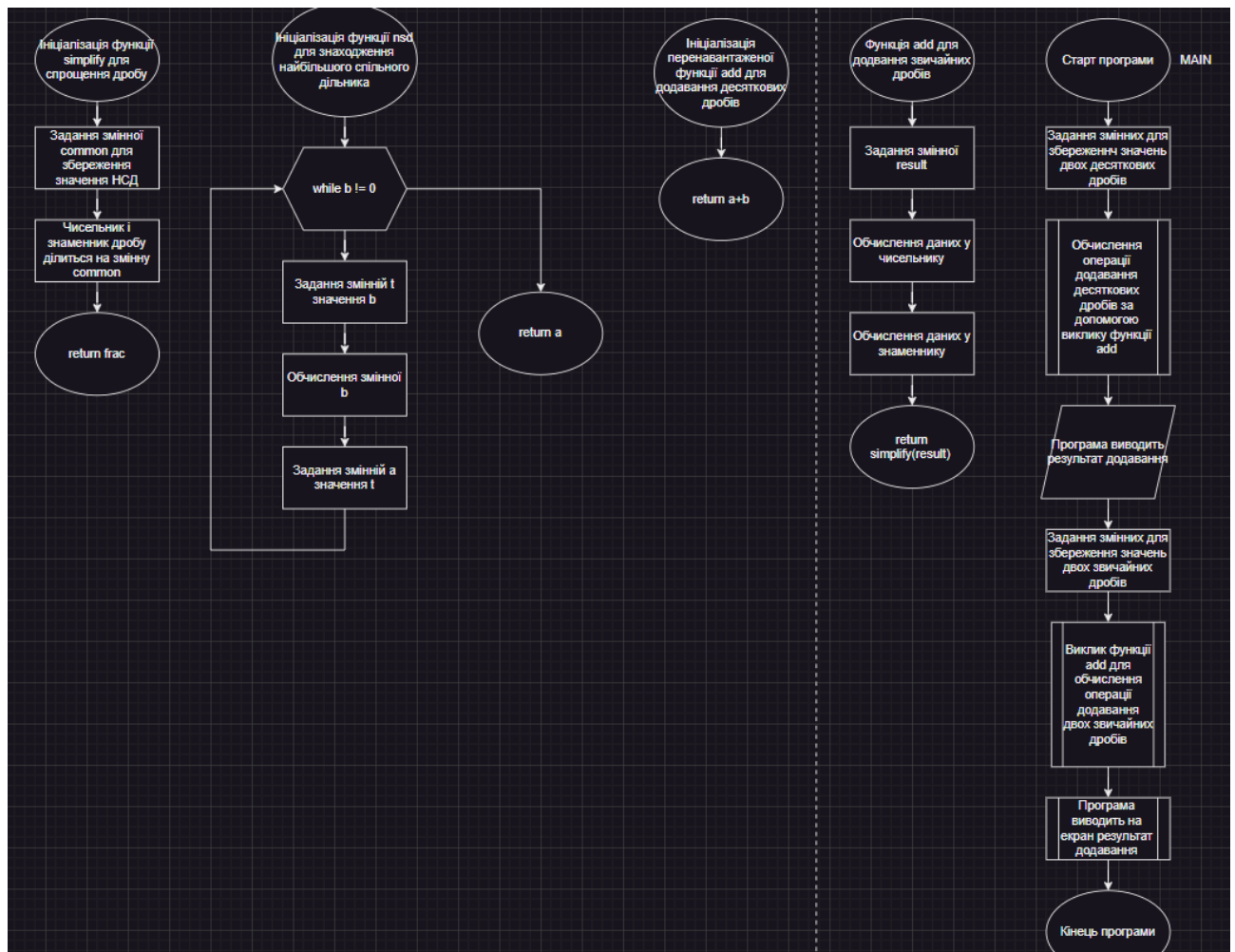


Рисунок 2.4. Блок-схема до програми 4

Програма №5

- Блок-схема

- Плановий час на реалізацію 1 день

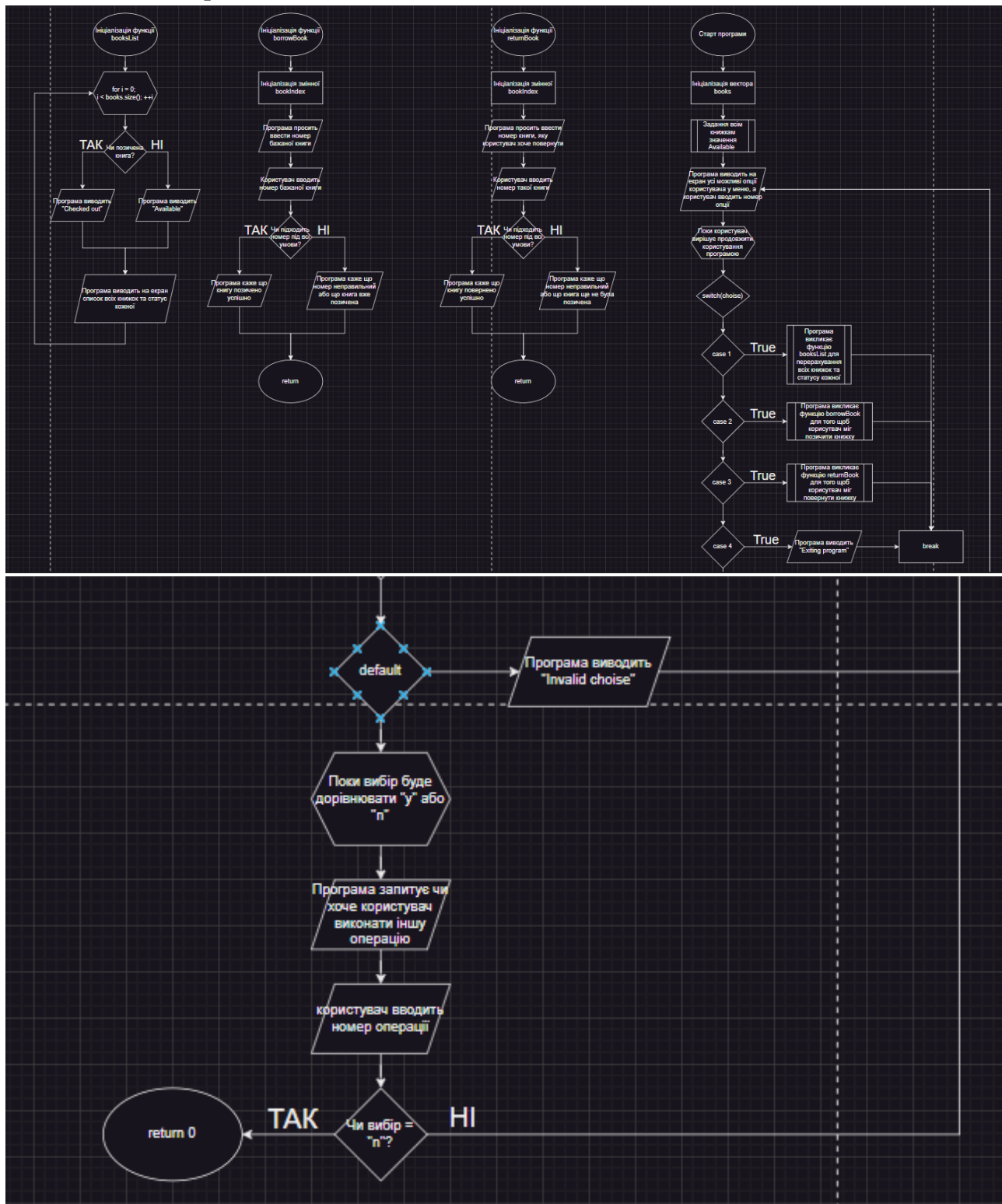


Рисунок 2.5. Блок-схема до програми №5

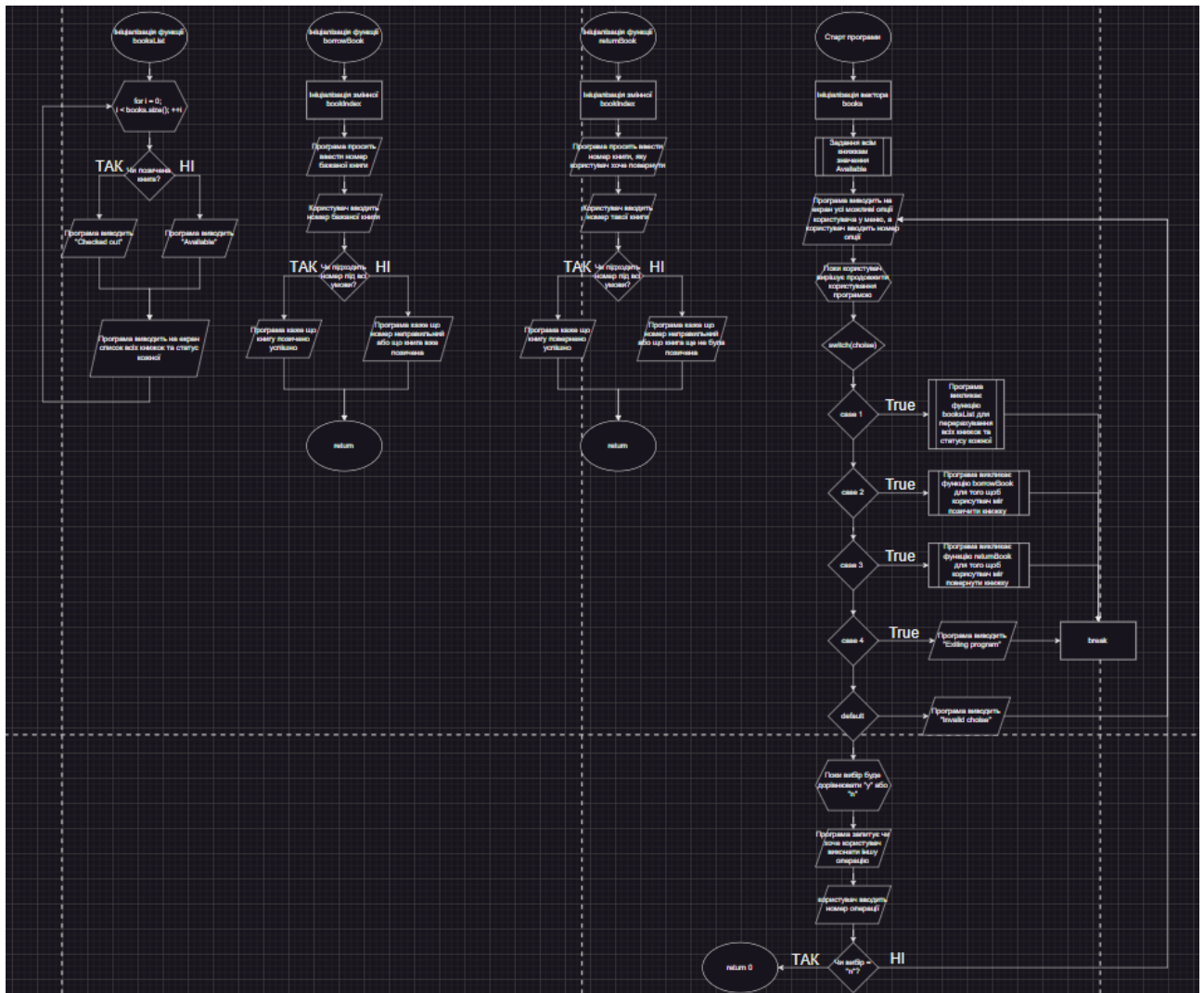


Рисунок 2.6. Загальна блок-схема до програми №5

Програма №6

- Блок-схема
- Важливі деталі для реалізації програми
- Використати цикл для перевірки елементів у векторі.
- Плановий час на реалізацію 1 день

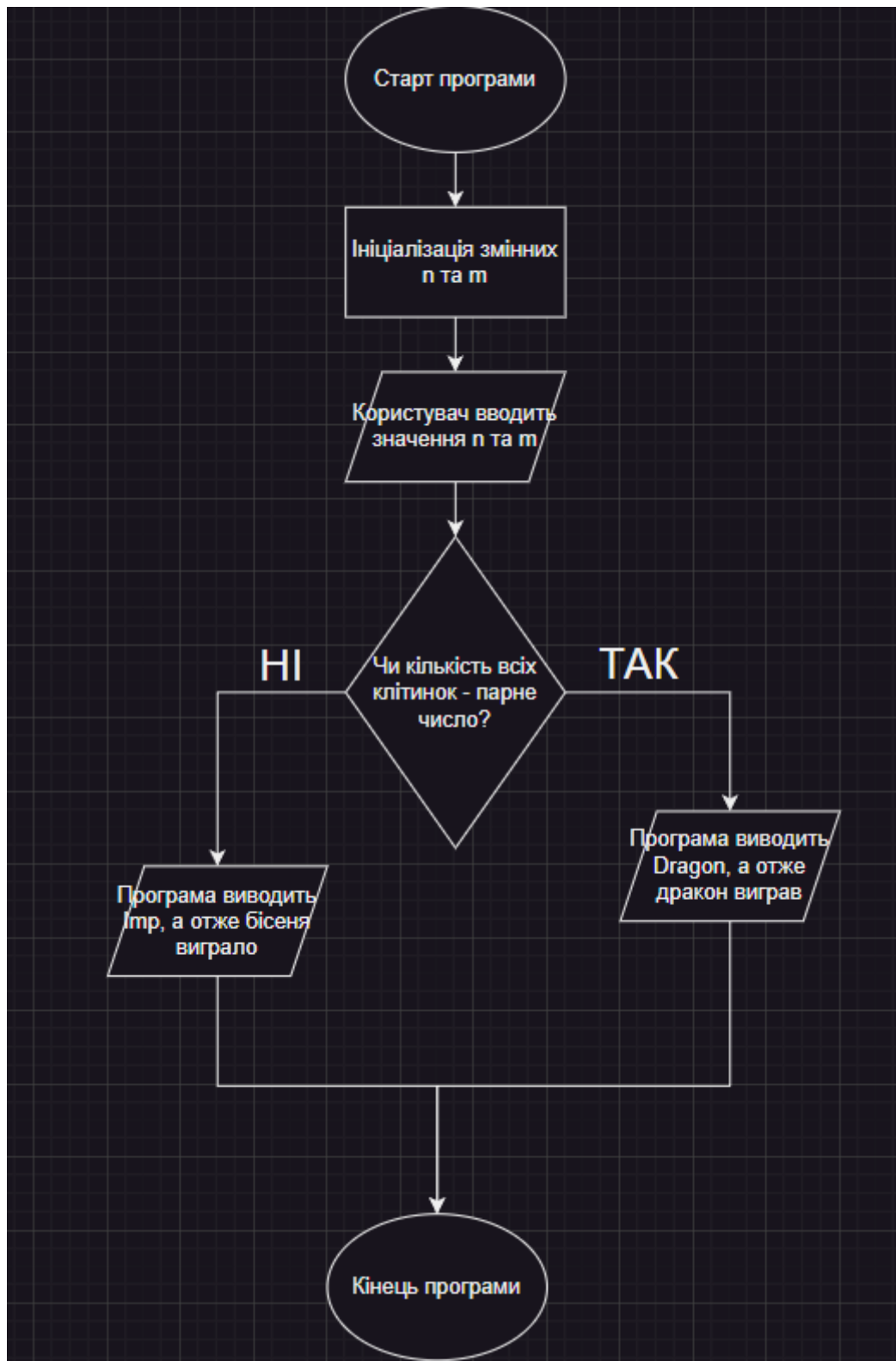


Рисунок 2.7. Блок-схема до програми №6

3. Код програм з посиланням на зовнішні ресурси та фактично затрачений час:

Завдання №1

```

1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  // Функція для обчислення факторіалу
7  double factorial(int n) {
8      if (n == 0 || n == 1) return 1;
9      return n * factorial(n - 1);
10 }
11
12 int main() {
13     double epsilon = 0.0001;
14     double suma = 0.0;
15     double element;
16     int n = 0;
17
18     do {
19         element = pow(10, n) / factorial(n);
20         suma += element;
21         n++;
22     } while (element >= epsilon);
23
24     cout << "Сума ряду з точністю  $\epsilon = 0.0001$ : " << suma << endl;
25
26     return 0;
27 }
28

```

Рисунок 3.1. Код до програми №1

Сума ряду з точністю $\epsilon = 0.0001$: 22026.5

Рисунок 3.2. Приклад виконання програми №1

На початку ініціалізуємо функцію для знаходження факторіалу, після того відбувається ініціалізація змінних для зберігання точності, загальної суми ряду, номера та значення поточного члена ряду. За допомогою циклу do while програма обчислює суму та виводить її значення.

Фактично затрачений час 25 хвилин.

Завдання №2

```

1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  // Функція для обчислення факторіалу
7  double factorial(int n) {
8      if (n == 0 || n == 1) return 1;
9      return n * factorial(n - 1);
10 }
11
12 // Точне значення функції
13 double exactFunctionValue(double x) {
14     return (1.0 / 4.0) * log((1.0 + x) / (1.0 - x)) + (1.0 / 2.0) * atan(x);
15 }
16
17 int main() {
18     double a = 0.1;        // Початкове значення x
19     double b = 0.8;        // Кінцеве значення x
20     int k = 10;             // Кількість кроків
21     int n = 3;             // Задане значення n
22     double epsilon = 0.0001; // Задана точність
23
24     double step = (b - a) / k;
25
26     cout << "Обчислення функції" << endl;
27
28     for (double x = a; x <= b; x += step) {
29         double SN = 0.0;
30         double SE = 0.0;
31         double exactValue = exactFunctionValue(x);
32
33         // Обчислення SN (заданого n)
34         for (int i = 0; i < n; ++i) {
35             SN += pow(x, 4*i + 1) / (4*i + 1);
36         }
37
38         // Обчислення SE (заданої точності епсилон)
39         int i = 0;
40         double element;
41         do {
42             element = pow(x, 4*i + 1) / (4*i + 1);
43             SE += element;
44             ++i;
45         } while (element >= epsilon);
46
47         // Вивід результатів
48         cout << "X=" << x << " SN=" << SN << " SE=" << SE << " Y=" << exactValue << endl;
49     }
50
51     return 0;
52 }
53

```

Рисунок 3.3. Код до програми №2

Обчислення функції

```
X=0.1 SN=0.100002 SE=0.100002 Y=0.100002
X=0.1 SN=0.100002 SE=0.100002 Y=0.100002
X=0.17 SN=0.170028 SE=0.170028 Y=0.170028
X=0.17 SN=0.170028 SE=0.170028 Y=0.170028
X=0.24 SN=0.24016 SE=0.24016 Y=0.24016
X=0.24 SN=0.24016 SE=0.24016 Y=0.24016
X=0.31 SN=0.310576 SE=0.310576 Y=0.310576
X=0.38 SN=0.381603 SE=0.381603 Y=0.381603
X=0.45 SN=0.453775 SE=0.453775 Y=0.453777
X=0.52 SN=0.527913 SE=0.527929 Y=0.52793
X=0.59 SN=0.605261 SE=0.605342 Y=0.60535
X=0.66 SN=0.687687 SE=0.688084 Y=0.688093
X=0.73 SN=0.778003 SE=0.779632 Y=0.779653
```

Рисунок 3.4. Приклад виконання програми №2

Цей код дозволяє обчислити значення функції $f(x)$ за допомогою її розкладу в степеневий ряд для заданих умов і порівняти їх з точним значенням.

Фактично затрачений час 2 години.

Завдання №3

```
1  #include <iostream>
2  #include <initializer_list>
3  #include <algorithm>
4
5  using namespace std;
6
7  auto min (initializer_list<int> numbers)
8  {
9      auto min_number = *numbers.begin();
10     for (auto number : numbers)
11     {
12         if (number < min_number)
13         {
14             min_number = number;
15         }
16     }
17     return min_number;
18 }
19
20 int main()
21 {
22     cout << "Мінімальне значення (1, 2, 3, 4, 5): " << min({1, 2, 3, 4, 5}) << endl;
23     cout << "Мінімальне значення (74, 23, 81, 56, 34, 68, 97, 12, 45, 89): " << min({74, 23, 81, 56, 34, 68, 97, 12, 45, 89}) << endl;
24     cout << "Мінімальне значення (3, 59, 24, 77, 51, 36, 82, 15, 92, 68, 40, 11): " << min({3, 59, 24, 77, 51, 36, 82, 15, 92, 68, 40, 11}) << endl;
25     return 0;
26 }
27
```

Рисунок 3.5. Код до програми №3

```
Мінімальне значення (1, 2, 3, 4, 5): 1
Мінімальне значення (74, 23, 81, 56, 34, 68, 97, 12, 45, 89): 12
Мінімальне значення (3, 59, 24, 77, 51, 36, 82, 15, 92, 68, 40, 11): 3
```

Рисунок 3.6. Приклад виконання програми №3

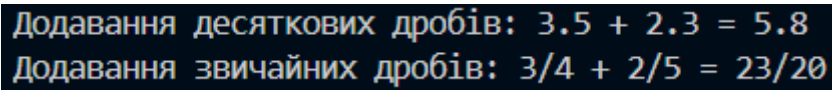
Спочатку створюємо функцію `min` зі змінною кількістю параметрів типу `int`, яка буде шукати серед них найменше значення. У функції за допомогою `initializer_list numbers` створили список для збереження чисел. У циклі, який перебирає всі елементи у списку, за допомогою `if` програма визначає найменше значення, після чого функція

повертає шуканий елемент . У головній функції ми викликаємо функцію `min`, яка знаходить мінімальне значення серед заданих попередньо чисел.
Фактично затрачений час 30 хвилин.

Завдання №4

```
1  #include <iostream>
2
3  using namespace std;
4
5  // Структура для звичайних дробів
6  struct drib {
7      int numerator; // Чисельник
8      int denominator; // Знаменник
9  };
10
11 // Функція для знаходження найбільшого спільного дільника
12 int nsd(int a, int b) {
13     while (b != 0) {
14         int t = b;
15         b = a % b;
16         a = t;
17     }
18     return a;
19 }
20
21 // Спрощення дробу
22 drib simplify(drib frac) {
23     int common = nsd(frac.numerator, frac.denominator);
24     frac.numerator /= common;
25     frac.denominator /= common;
26     return frac;
27 }
28
29 // Перевантажена функція для додавання десяткових дробів
30 double add(double a, double b) {
31     return a + b;
32 }
33
34 // Перевантажена функція для додавання звичайних дробів
35 drib add(drib f1, drib f2) {
36     drib result;
37     result.numerator = f1.numerator * f2.denominator + f2.numerator * f1.denominator;
38     result.denominator = f1.denominator * f2.denominator;
39     return simplify(result);
40 }
41
42 int main() {
43     // Виклик функції додавання десяткових дробів
44     double dec1 = 3.5;
45     double dec2 = 2.3;
46     double decResult = add(dec1, dec2);
47     cout << "Додавання десяткових дробів: " << dec1 << " + " << dec2 << " = " << decResult << endl;
48
49     // Виклик функції додавання звичайних дробів
50     drib frac1 = {3, 4};
51     drib frac2 = {2, 5};
52     drib fracResult = add(frac1, frac2);
53     cout << "Додавання звичайних дробів: " << frac1.numerator << "/" << frac1.denominator
54         << " + " << frac2.numerator << "/" << frac2.denominator
55         << " = " << fracResult.numerator << "/" << fracResult.denominator << endl;
```

Рисунок 3.7. Код до програми №4



Додавання десяткових дробів: $3.5 + 2.3 = 5.8$
Додавання звичайних дробів: $3/4 + 2/5 = 23/20$

Рисунок 3.8. Приклад виконання програми №4

Спочатку створюємо 3 перевантажені функції `addition_real_numbers`, які будуть додавати три цілі числа, потім 3 перевантажені функції `addition_complex_numbers`, які відповідно будуть додавати комплексні числа. У головній функції ми будемо викликати тричі одну функцію і тричі другу, щоб додати різну кількість чисел.

Завдання №5

```

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>& availability) {
8      for (size_t i = 0; i < books.size(); ++i) {
9          cout << i + 1 << ". " << books[i] << " - " << (availability[i] ? "Available" : "Checked out") << endl;
10     }
11 }
12
13 void borrowBook(vector<bool>& availability) {
14     int bookIndex;
15     cout << "Enter the number of the book you want to borrow: ";
16     cin >> bookIndex;
17     if (bookIndex < 1 || bookIndex > static_cast<int>(availability.size()) || !availability[bookIndex - 1]) {
18         cout << "Invalid choice or book not available." << endl;
19         goto mainMenu;
20     } else {
21         availability[bookIndex - 1] = false;
22         cout << "Book borrowed successfully!" << endl;
23     }
24     return;
25
26     mainMenu:
27     cout << "Returning to main menu." << endl;
28 }
29
30 void returnBook(vector<bool>& availability) {
31     int bookIndex;
32     cout << "Enter the number of the book you want to return: ";
33     cin >> bookIndex;
34     if (bookIndex < 1 || bookIndex > static_cast<int>(availability.size()) || availability[bookIndex - 1]) {
35         cout << "Invalid choice or book already returned." << endl;
36         goto mainMenu;
37     } else {
38         availability[bookIndex - 1] = true;
39         cout << "Book returned successfully!" << endl;
40     }
41     return;
42
43     mainMenu:
44     cout << "Returning to main menu." << endl;
45 }
46
47 int main() {
48     vector<string> books = {"The Great Gatsby", "1984", "To Kill a Mockingbird", "Pride and Prejudice", "The Hobbit", "Surgeon"};
49     vector<bool> availability(books.size(), true);
50
51     char choice;
52     do {
53         cout << "\nLibrary Management System" << endl;
54         cout << "1. List all books" << endl;
55         cout << "2. Borrow a book" << endl;

```

```

56     cout << "3. Return a book" << endl;
57     cout << "4. Exit" << endl;
58     cout << "Enter your choice: ";
59     cin >> choice;
60
61     switch (choice) {
62     case '1':
63         listBooks(books, availability);
64         break;
65     case '2':
66         borrowBook(availability);
67         break;
68     case '3':
69         returnBook(availability);
70         break;
71     case '4':
72         cout << "Exiting program." << endl;
73         break;
74     default:
75         cout << "Invalid choice." << endl;
76         goto mainMenu;
77     }
78
79     do {
80         cout << "Do you want to perform another operation? (y/n): ";
81         cin >> choice;
82     } while (choice != 'y' && choice != 'n');
83
84     } while (choice == 'y');
85
86     return 0;
87
88     mainMenu:
89     cout << "Returning to main menu." << endl;
90     return main();
91 }
92

```

Рисунок 3.9. Код до програми №5

```
Library Management System
1. List all books
2. Borrow a book
3. Return a book
4. Exit
Enter your choice: 1
1. The Great Gatsby - Available
2. 1984 - Available
3. To Kill a Mockingbird - Available
4. Pride and Prejudice - Available
5. The Hobbit - Available
6. Surgeon - Available
Do you want to perform another operation? (y/n): y

Library Management System
1. List all books
2. Borrow a book
3. Return a book
4. Exit
Enter your choice: 2
Enter the number of the book you want to borrow: 6
Book borrowed successfully!
Do you want to perform another operation? (y/n): y

Library Management System
1. List all books
2. Borrow a book
3. Return a book
4. Exit
Enter your choice: 1
1. The Great Gatsby - Available
2. 1984 - Available
3. To Kill a Mockingbird - Available
4. Pride and Prejudice - Available
5. The Hobbit - Available
6. Surgeon - Checked out
Do you want to perform another operation? (y/n): n
```

Рисунок 3.10. Приклад виконання програми №5

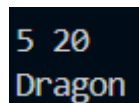
Програма №5 це онлайн бібліотека. Користувачу на екран програма виводить можливі дії і, залежно від бажання користувача, програма буде виводити різні результати на екран та робити різні дії.

Фактично затрачений час 3 години.

Завдання №6

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6      int n, m;
7      cin >> n >> m;
8
9      // Загальна кількість клітинок
10     int total_cells = n * m;
11
12     // Визначаємо переможця
13     if (total_cells % 2 == 0) {
14         cout << "Dragon" << endl;
15     } else {
16         cout << "Imp" << endl;
17     }
18
19     return 0;
20 }
21
```

Рисунок 3.11. Код до програми №6



```
5 20
Dragon
```

Рисунок 3.12. Приклад виконання програми №6

Створено	Компілятор	Результат	Час (сек.)	Пам'ять (МіБ)	Дії
декілька секунд тому	C++ 23	Зараховано	0.003	1.152	Перегляд

Рисунок 3.13. Приклад виконання програми №6 в алготестері

Програма зчитує та обчислює розміри дошки і залежно від результату визначає переможця.

Фактично затрачений час 30 хвилин.

Посилання на пул реквест: [Додав файли з кодом та блок-схеми by Ostap2007ter · Pull Request #364 · artificial-intelligence-department/ai_programming_playground 2024](#)

4. Робота з командою:

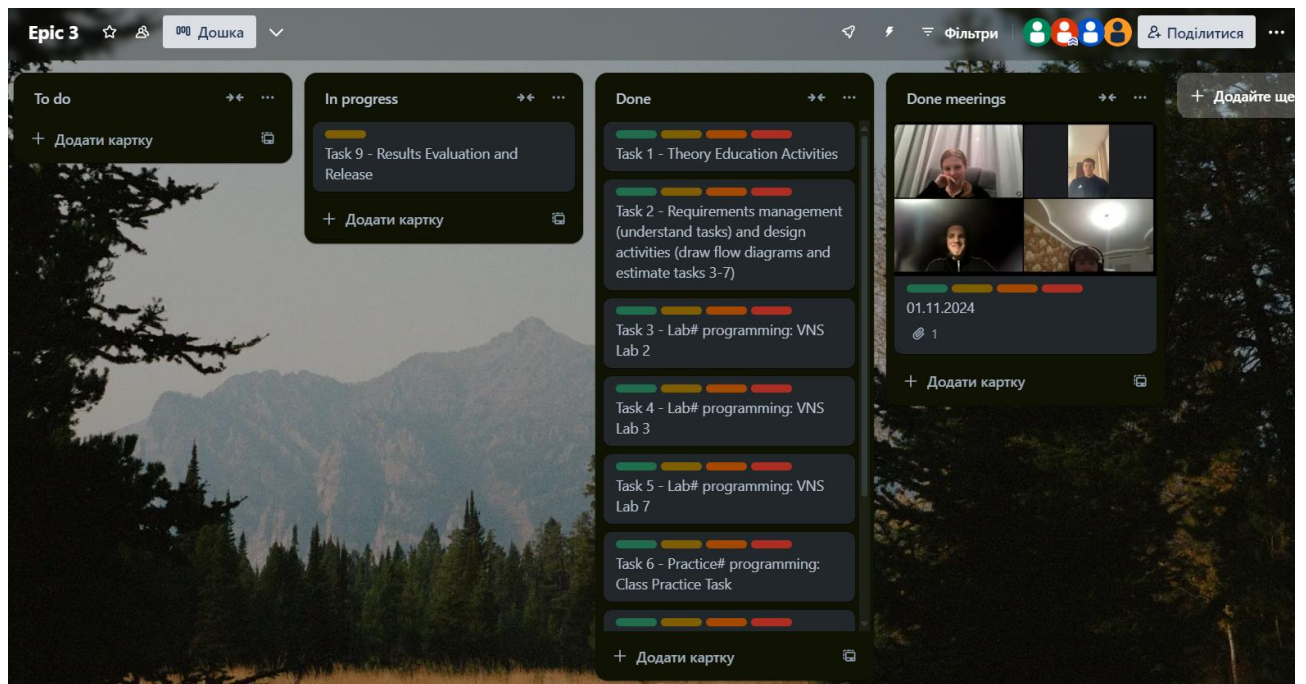


Рисунок 4.1. Командна дошка в Trello

Висновок: Під час виконання практичних і лабораторних робіт з блоку №3 я освоїв такі нові поняття, як функція, перевантажена функція, функція зі змінною кількістю параметрів, цикли, вкладені цикли та рекурсія. Деякі з цих концепцій я застосував на практиці, що допомогло мені краще зрозуміти їхню роботу. Для більш детального розуміння функціонування програм я створював блок-схеми в Draw.io, завдяки чому також вдосконалив свої навички роботи з цією програмою. Крім того, я створив нову дошку в Trello для командної роботи.