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

Іванов Олексій Олександрович

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:

1. Опрацювання завдання та вимог до програм та середовища:

Завдання № 1 VNS Practice Work - Task 1

- Варіант завдання - 2
- Деталі завдання

Варіант 2. $a = \frac{2\cos(x-\frac{\pi}{6})b}{\frac{1}{2}+\sin^2 y}$; $b = 1+\frac{z^2}{3+z^2/5}$, де x=1,45; y=-1,22; z=3,5.

Рисунок 1: Деталі завдання VNS Practice Work - Task 1

- Важливі деталі для врахування в імплементації програми
  Розбити програму на підпроцедури

Завдання № 2 VNS Practice Work - Task 2

- Варіант завдання - 24
- Деталі завдання

Варіант 24. Підрахувати, скільки разів функція $y = \cos x^2 \cdot e^{-x}$ приймає негативне значення, якщо $x \in [0{,}3;5]$; $h_x = 0{,}1$.

Рисунок 2: Деталі завдання VNS Practice Work - Task 2

- Важливі деталі для врахування в імплементації програми
  Врахувати неточність типу double і ввести певну змінну epsilon для того, аби коректно перевірити належність **x** проміжку

Завдання № 3 VNS Practice Work - Task 3.1

- Варіант завдання - 7
- Деталі завдання

Варіант 7. Обчислення площі трикутника, якщо відомі довжини двох його сторін і величина кута між цими сторонами.

Рисунок 3: Деталі завдання VNS Practice Work - Task 3.1

Завдання № 4 VNS Practice Work - Task 3.2

- Варіант завдання - 7
- Деталі завдання

Реалізувати визначення числа болтів в ящику з розмірами (*H*B*S*) м3, якщо один болт в середньому займає об'єм 2 см^3. Розміри ящика *H, B і S* повинні вводитися з клавіатури.

Рисунок 4: Деталі завдання VNS Practice Work - Task 3.2

Завдання № 5 VNS Practice Work - Task 4

- Варіант завдання - 19
- Деталі завдання

Варіант 19. Скласти програму, яка виводить таблицю значень функції $y = |x|$. Діапазон зміни аргументу -4 до 4, крок приросту аргументу 0,5.

Рисунок 5: Деталі завдання VNS Practice Work - Task 4

- Важливі деталі для врахування в імплементації програми

2. Дизайн та планована оцінка часу виконання завдань:

Програма № 1 VNS Practice Work - Task 1

- Блок-схема

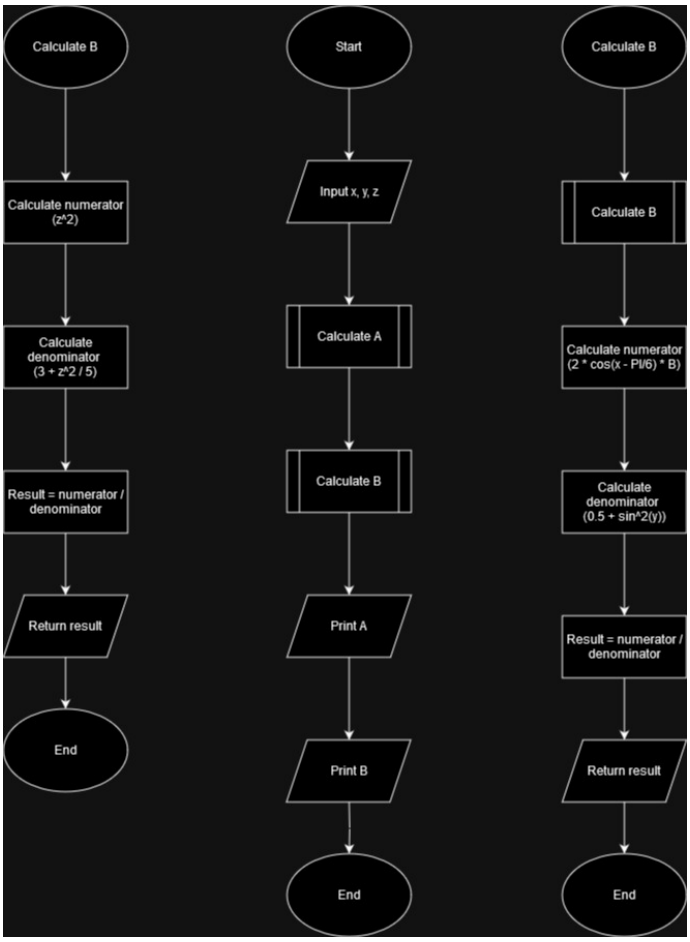

Рисунок 6: Блок-схема до програми № 1

- Планований час на реалізацію — 10 хвилин

Програма № 2 VNS Practice Work - Task 2

- Блок-схема

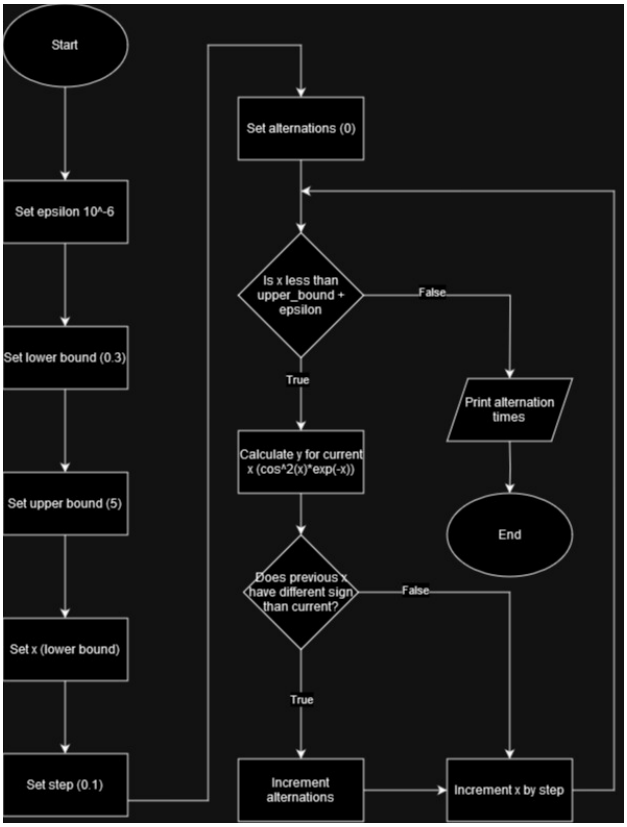

Рисунок 7: Блок-схема до програми № 2

Планований час на реалізацію — 5 хвилин

Програма № 3 VNS Practice Work - Task 3.1

- Блок-схема

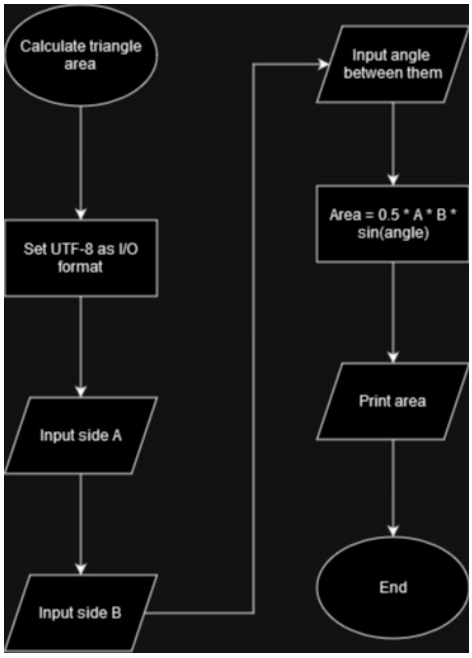

Рисунок 8: Блок-схема до програми № 3

- Планований час на реалізацію — 5 хвилин

Програма № 4 VNS Practice Work - Task 3.2

- Блок-схема

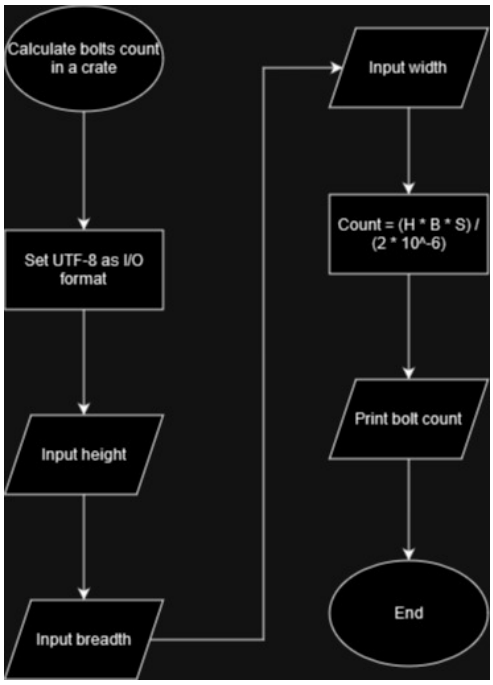

Рисунок 9: Блок-схема до програми № 4

- Планований час на реалізацію — 5 хвилин

Програма № 5 VNS Practice Work - Task 4

- Блок-схема

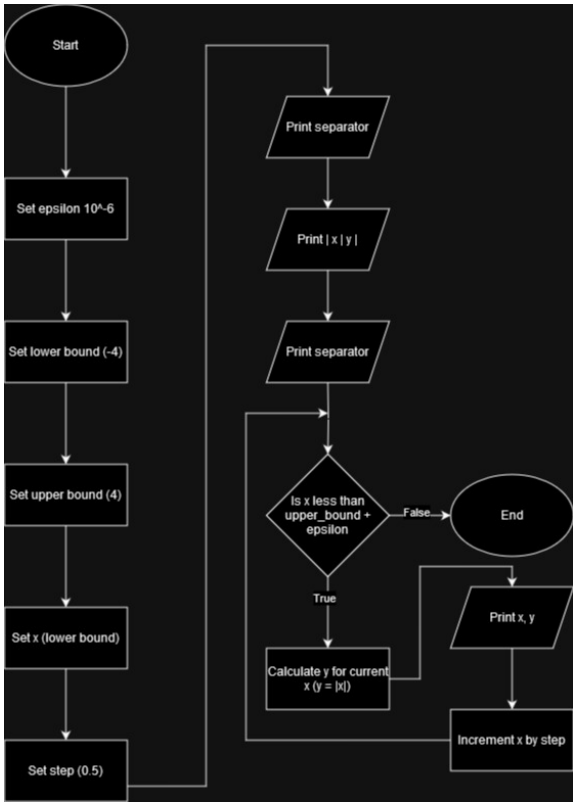

Рисунок 10: Блок-схема до програми № 5

- Планований час на реалізацію — 10 хвилин

3. Код програм з посиланням на зовнішні ресурси:

Завдання № 1 VNS Practice Work - Task 1

```c
#include <stdio.h>
#include <math.h>

// Calculate the value by formula
double calculateB(double z) {
    double numerator = pow(z, 2);
    double denominator = 3 + numerator / 5.0;
    return 1 + numerator / denominator;
}
double calculateA(double x, double y, double z) {
    double numerator = 2 * cos(x - M_PI / 6.0) * calculateB(z);
    double denominator = 0.5 + pow(sin(y), 2);
    return numerator / denominator;
}

int main() {
    // Projected A: 2.82355
    // Projected B: 3,24771 (354/109)

    double x = 1.45, y = -1.22, z = 3.5;
    printf("A: %lf\nB: %lf\n", calculateA(x, y, z), calculateB(z));
    return 0;
}
```

Завдання № 2 VNS Practice Work - Task 2

```c
#include <stdio.h>
#include <math.h>

#define epsilon 1e-6

double y(double x) {
    return cos(pow(x, 2)) * exp(-x);
}

int main() {
    double  lower_bound = 0.3, upper_bound = 5.0,
            step = 0.1,
            x = lower_bound;

    int alternates = 0;

    double prev_y = y(x), curr_y;
//  printf("x = %lf, y(x) = %lf\n", x, curr_y);
    while (x <= upper_bound + epsilon) {
        curr_y = y(x);
//        printf("x = %lf, y(x) = %lf\n", x, curr_y);

        // If the product of the previous and current y values is negative,
```

```
        // then the y values alternate between positive and negative.
        if (prev_y * curr_y < 0) alternates++;

        x += step;
        prev_y = curr_y;
    }

    printf("The function y(x) alternates between "
           "positive and negative %d times "
           "in the interval [%.1lf, %.1f].\n",
           alternates, lower_bound, upper_bound);

    return 0;
}
```

Завдання № 3 VNS Practice Work - Task 3.1

```
#include <stdio.h>
#include <math.h>
#include <fcntl.h>

void set_utf8() {
    _setmode(_fileno(stdout), _O_U8TEXT);
    _setmode(_fileno(stdin), _O_U8TEXT);
}

double area(double a, double b, double angle_degrees) {
    return 0.5 * a * b * sin(angle_degrees * M_PI / 180);
}

int main() {
    set_utf8();

    double a, b, angle;

    wprintf(L"Введіть (через пропуск) довжини двох сторін (см) трикутника: ");
    wscanf(L"%lf %lf", &a, &b);

    wprintf(L"Введіть величину кута між сторонами трикутника (градуси): ");
    wscanf(L"%lf", &angle);

    wprintf(L"Площа трикутника: %.2lf (см^2)\n", area(a, b, angle));

    return 0;
}
```

Завдання № 4 VNS Practice Work - Task 3.2

```
#include <stdio.h>
#include <fcntl.h>
// 1 cubic meter - 1 000 000 cubic centimeters
#define BOLT_VOLUME 2e-6
```

```c
void set_utf8() {
    _setmode(_fileno(stdout), _O_U8TEXT);
    _setmode(_fileno(stdin), _O_U8TEXT);
}

double calc_bolt_count(double H, double B, double S) {
    double box_volume = H * B * S;
    return box_volume / BOLT_VOLUME;
}

int main() {
    set_utf8();

    double H, B, S;
    wprintf(L"Введіть висоту, ширину та глибину ящика (м, в одному рядку): ");
    wscanf(L"%lf %lf %lf", &H, &B, &S);

    double bolt_count = calc_bolt_count(H, B, S);
    wprintf(L"Максимальна кількість болтів у ящику: %.2lf\n", bolt_count);

    return 0;
}
```

Завдання № 5 VNS Practice Work - Task 4

```c
#include <stdio.h>
// may include math.h, but
#define abs(x) ((x) < 0 ? -(x) : (x))

#define epsilon 1e-6

int main() {
    double  lower_bound = -4, upper_bound = 4, step = 0.5,
            x = lower_bound, y;

    char* separator = "+----------------------+\n";

    printf("%s", separator);
    printf("|  x\t|\t y\t|\n");
    printf("%s", separator);

    while (x <= upper_bound + epsilon) {
        y = abs(x);
        printf("|%.2lf\t|\t%.2lf\t|\n", x, y);
        x += step;
    }

    printf("%s", separator);
    return 0;
}
```

Посилання на pull request

4. Результати виконання завдань, тестування та фактично витрачений час:

Завдання № 1 VNS Practice Work - Task 1

```
"C:\Users\oleks\CLionProjects\Epic 7\VNS Practice Work - Task 1\cmake-build-debug\VNS_Practice_Work___Task_1.exe"
A: 2.823555
B: 3.247706

Process finished with exit code 0
```

Рисунок 11: Результати виконання VNS Practice Work - Task 1

Час, витрачений на виконання завдання — 5 хвилин

Завдання № 2 VNS Practice Work - Task 2

```
"C:\Users\oleks\CLionProjects\Epic 7\VNS Practice Work - Task 2\cmake-build-debug\VNS_Practice_Work___Task_2.exe"
The function y(x) alternates between positive and negative 8 times in the interval [0.3, 5.0].

Process finished with exit code 0
```

Рисунок 12: Результати виконання VNS Practice Work - Task 2

Час, витрачений на виконання завдання — 10 хвилин

Завдання № 3 VNS Practice Work - Task 3.1

```
PS C:\Users\oleks>  & 'c:\Users\oleks\.vscode\extensions\ms-vscode.cpptools-1.18.5-win32-x64\debugAdapters\bin\Windo
wsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-projgwfs.2nh' '--stdout=Microsoft-MIEngine-Out-khzz3nng.21w' '--
stderr=Microsoft-MIEngine-Error-xyx4pizd.4f1' '--pid=Microsoft-MIEngine-Pid-u4a30ynf.l51' '--dbgExe=C:\msys64\ucrt64
\bin\gdb.exe' '--interpreter=mi'
Введіть (через пропуск) довжини двох сторін (см) трикутника: 25 17
Введіть величину кута між сторонами трикутника (градуси): 30
Площа трикутника: 106.25 (см^2)
PS C:\Users\oleks>
```

Рисунок 13: Результати виконання VNS Practice Work - Task 3.1

Час, витрачений на виконання завдання — 5 хвилин

Завдання № 4 VNS Practice Work - Task 3.2

```
"C:\Users\oleks\CLionProjects\Epic 7\VNS Practice Work - Task 3.2\cmake-build-debug\VNS_Practice_Work___Task_3_2.exe"
Введіть висоту, ширину та глибину ящика (м, в одному рядку):0.4
0.3
0.1
 Максимальна кількість болтів у ящику: 6000.00

Process finished with exit code 0
```

Рисунок 14: Результати виконання VNS Practice Work - Task 3.2

Час, витрачений на виконання завдання — 5 хвилин

Завдання № 5 VNS Practice Work - Task 4

```
"C:\Users\oleks\CLionProjects\Epic 7\VNS Practice Work - Task 4\cmake-build-debug\VNS_Practice_Work___Task_4.exe"
+----------------------+
|  x    |       y      |
+----------------------+
|-4.00  |       4.00   |
|-3.50  |       3.50   |
|-3.00  |       3.00   |
|-2.50  |       2.50   |
|-2.00  |       2.00   |
|-1.50  |       1.50   |
|-1.00  |       1.00   |
|-0.50  |       0.50   |
|0.00   |       0.00   |
|0.50   |       0.50   |
|1.00   |       1.00   |
|1.50   |       1.50   |
|2.00   |       2.00   |
|2.50   |       2.50   |
|3.00   |       3.00   |
|3.50   |       3.50   |
|4.00   |       4.00   |
+----------------------+

Process finished with exit code 0
```

Рисунок 15: Результати виконання VNS Practice Work - Task 4

Час, витрачений на виконання завдання — 15 хвилин

5. Відповіді на контрольні запитання:

1. Назвіть основні властивості алгоритму.

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

2. Що таке алгоритм?

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

3. Визначте основні етапи розробки алгоритмів.

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

4. Перелічить базові конструкції.

   Алфавіт, лексеми, вирази та оператори

5. Перелічіть складні базові конструкції.

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

Цикли: while, do-while, for

6. Дайте визначення конструкції розгалуження.

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

7. Дайте визначення конструкції цикл.

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

8. Сформулюйте правило виконання циклу з передумовою.

   8.1. Спочатку обчислюється значення виразу-умови.

   8.2. Якщо значення виразу-умови істинне, то виконується тіло циклу.

   8.3. Після виконання тіла циклу значення виразу-умови обчислюється знову.

   8.4. Якщо значення виразу-умови істинне, то цикл повертається на крок 2.

   8.5. Якщо значення виразу-умови хибне, то цикл завершується.

9. Сформулюйте правило виконання циклу з відомою кількістю повторювань тіла циклу.

   9.1. Перед початком виконання циклу ініціалізується лічильник, який буде контролювати кількість повторень тіла циклу.

   9.2. Потім виконується тіло циклу.

   9.3. Після виконання тіла циклу лічильник інкрементується (збільшується на одиницю).

   9.4. Якщо лічильник не досяг заданої кількості повторень, то цикл повертається на крок 2.

   9.5. Якщо лічильник досяг заданої кількості повторень, то цикл завершується.

10. Що таке обчислювальна складність алгоритму?

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

11. 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.

12. 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.