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

Звіт

про виконання лабораторних та практичних робіт блоку № 1

На тему: «Програмування: алгоритм, програма, код. Системи числення. Двійкова система числення. Розробка та середовище розробки програми.»

з дисципліни: «Основи програмування»

до:

Практичних Робіт до блоку № 1

Виконав:
Студент групи ШІ-12
Макович Маркіян Володимирович

Львів 2024

Тема роботи:

- Налаштування VSCode (компілятора для C++)
- Освоєння Linux команд
- Створення дошок в Trello
- Робота з Git та GitHub
- Створення блок-схем в Draw.io
- Виконання задач на сайті Algotester.com
- Робота в команді

Мета роботи:

Навчитись створювати блок-схеми в Draw.io, користуватись дошками в Trello, вивчити Linux команди для роботи з консоллю, налаштувати Visual Studio Code для робити з C++, працювати з Git та GitHub, виконувати задачі на Algotester.com, освоїти двійкову систему числення та операції над нею. Покращити навички роботи в команді (SoftSkills).

Теоретичні відомості:

1. Для налаштування компілятора у VSCode використовував сайт: https://code.visualstudio.com/docs/cpp/config-mingw.
Витрачений час: близько двох днів, бо була проблема з шляхом куди встановлювався компілятор.

2. Вивчав Linux команди тут: https://kinsta.com/blog/linux-commands/, також звертався до ChatGPT для детальніших роз'яснень.
Витрачений час: 1.5 години.

3. Для освоєння Git та GitHub використовував міні-курс Андрія Кравця на YouTube: https://www.youtube.com/watch?v=oofAm4x6oOE&list=PL3o5sNxukLFCKfvxez4nEZQ_cpWcXtb06&pp=iAQB, а також сайт https://git-scm.com/book/uk/v2 де є пояснення до всіх речей пов'язаних з Git.
Витрачений час: 2 дні, бо були проблеми з клонуванням репозиторію за допомогою ключа SSH.

4. Створення дошок в Trello не зайняло багато часу, адже це інтуітивно зрозуміла платформа для контролю виконання завдань.
Витрачений час: 30 хвилин.

5. Створення блок-схем в Draw.io також не забрало багато часу. Для освоєння використовував такий ресурс: https://www.programiz.com/article/flowchart-programming.
Витрачений час: до 1 години.

Виконання роботи:

1. Опрацювання завдання та вимог до програм та середовища:

Завдання № 1 Practice Task

Обчислення складних відсотків за депозитом

Задача: *Обчислити складні відсотки для депозиту, який був відкритий в банку на певний період часу під фіксовані відсотки з різними варіантами виплати відсотків.*

Вимоги:

1. *Використати функції scanf та printf для для зчитування і форматування вводу/виводу;*
2. *В кінці програма має вивести повну інформацію про вкладені кошти, загальну суму інвестиції і суму самого заробітку.*

Завдання № 2 Self Practice Task

Щасливенькі числа

Задача: *Потрібно обчислити кількість щаслиеньких чисел, що не перевищують n. Щасливенькими називають числа в яких сума цифр є щаслива. Щасливі числа це ті, в яких десятковий запис мість тільки 4 та 7, наприклад 47,777,4747.*

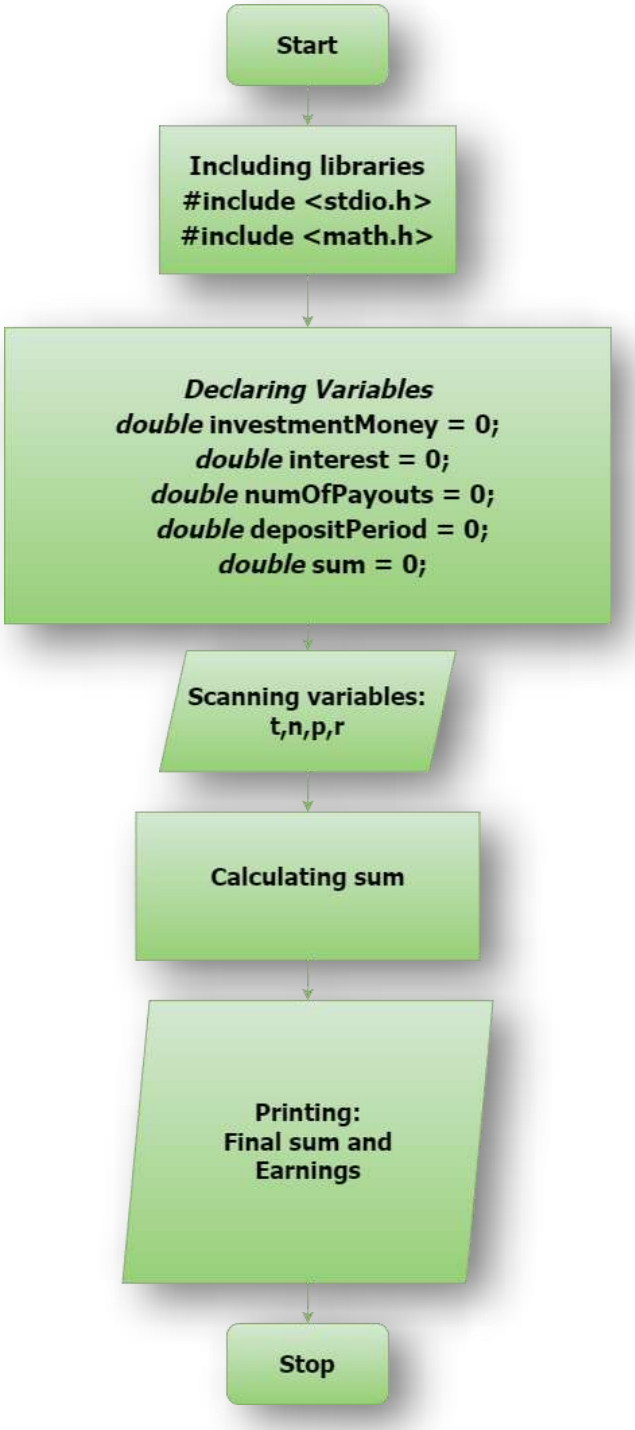
Вимоги:

1. *Обмеження в часі та розмірі програми:* 4 сек., 256 МіБ
2. *Число n є цілим,* $1 \leq n \leq 10^6$.

Завдання № 3 Калькуляції у двійковій системі

	Завдання на калькуляції в двійковій системі
1	Згенерувати в рандомайзері десяткове число y від 20 до 99
2	Згенерувати в рандомайзері десяткове число x від 20 до 99
3	Перевести y у двійкову систему числення
4	Перевести x у двійкову систему числення
5	Додати два двійкових числа x та y
6	Відняти від більшого двійковго числа менше двійкове число
7	Більше двійкове число поділити на менше двійкови число число
8	Більше двійкове число помножити на менше двійкови число число
9	Згенерувати в рандомайзері десяткове число k від 20 до 99
10	Перевести k у 16-ву систему числення

2. Графічне представлення Practice Task за допомогою Draw.io



Я очікував, що виконаю це завдання за 45 хвилин, враховуючи ознайомелення з функціями scanf, printf та бібліотеки <math.h>.

3. Конфігурація середовища до виконання завдань:

Скріншоти конфігурації середовища з підписами, що на цьому скіншоті

1.Встановлення VScode:

2.Встановлення розширення для C/C++:

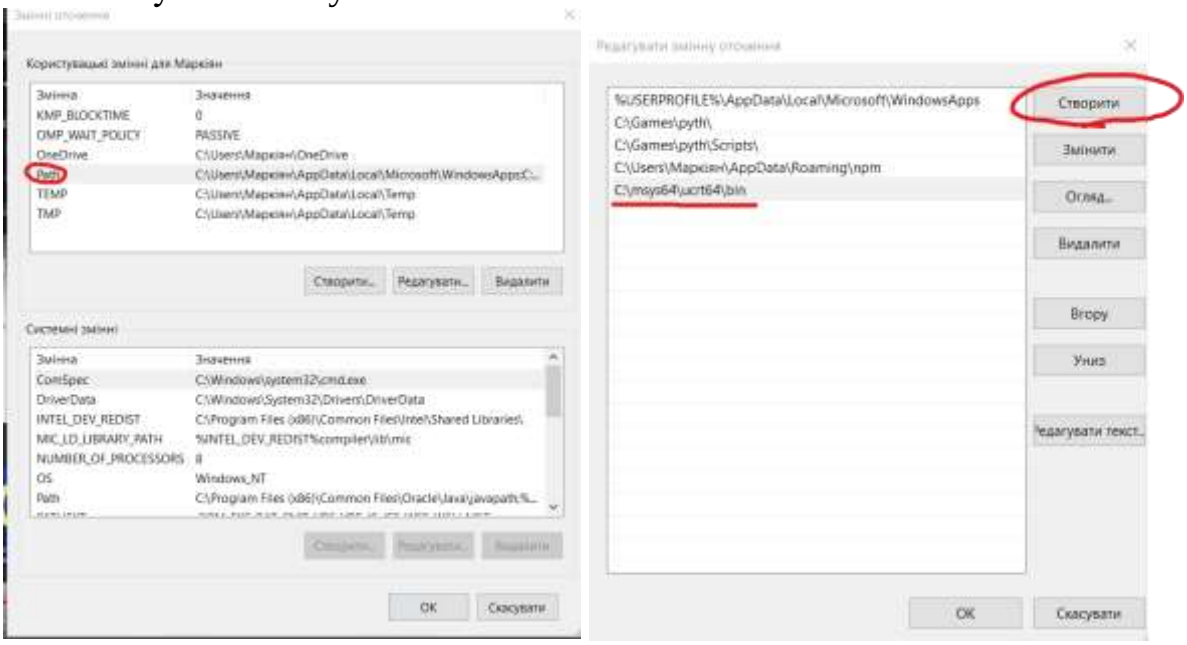
3.Встановлюємо MSYS2 і вводимо туди таку командну:
pacman -S --needed base-devel mingw-w64-ucrt-x86_64-toolchain

4.Встановлюємо шлях до нашого компілятора

В пошуку шукаємо: Змінити змінні оточення для вашого облікового запису

Заходимо в Path

Створюємо новий шлях
Вказуємо C:\msys64\ucrt64\bin



Компілятор встановлено та налаштовано!

4. *Код програм з посиланням на зовнішні ресурси:*

Завдання №1 Practice Task:

```cpp
#include <stdio.h>
#include <math.h>
using namespace std;

int main()
{
    // Декларація змінних
    double investmentMoney = 0;
    double interest = 0;
    double numOfPayouts = 0;
    double depositPeriod = 0;
    double sum = 0;

    // Введення змінних

    printf("Investment: ");
    scanf("%lf", &investmentMoney);
    printf("Annual rate (0<r<1): ");
    scanf("%lf", &interest);
    printf("Number of payouts: ");
    scanf("%lf", &numOfPayouts);
    printf("Deposit period: ");
    scanf("%lf", &depositPeriod);

    sum = investmentMoney * pow(1.0 + (interest / numOfPayouts), numOfPayouts * depositPeriod); // Формула розрахунку загальної суми

    printf("Money invested: %lf$\n", investmentMoney); // Інвестовані кошти
    printf("Final sum: %lf$\n", sum);                  // Загальна сума
    printf("Earnings: %lf$\n", sum - investmentMoney); // Сума заробітку

    return 0;
}
```

В цьому завданні використав змінні типу *double*, бо результати виводились некоректно при заданні інших типів даних. Також освоїв функції *printf*, *scanf* і які в них є переваги над іншими функціями вводу/виводу.

Завдання №2 Self Practice Task:

Завдання взято з сайту Algotester.com

```cpp
#include <iostream>

using namespace std;

int sumOfDigits(int n) // Функція розрахунку суми цифр
{
    int sum = 0;
    while (n != 0)
    {
        sum += n % 10;
        n /= 10;
    }
    return sum;
};

int main()
{

    int n;
    int count = 0;
    cin >> n;
    for (int i = 0; i <= n; i++)
    {
        if (sumOfDigits(i) == 4 || sumOfDigits(i) == 7 || sumOfDigits(i) == 47 || sumOfDigits(i) == 44) // Перевірка чи сума чисел є дозволеною

            count++;
    }

    if (n == 4) // Якщо число дорівнює 4
    {
        count = 1;
    }
    if (n == 7) // Якщо число дорівнює 7
    {
        count = 2;
    }

    cout << count << endl;
    return 0;
```

Задача	Компілятор	Результат	Час (сек.)	Пам'ять (МБ)	№
1402 · Щасливенькі числа	C++ 23	Зараховано	0.044	1.324	1769649

Ввівши певне число, програма виводить кількість чисел менших за це число, сума цифр яких складається тільки з цифр 4 та 7. Оскільки в умові задачі стоять обмеження на розмір введеного числа, то можна спростити задачу для перевірки не на всі можливі числа з цифр 4 та 7, а лише на числа менші 54 (бо

сума цифр 999999 = 54, а це максимальна можлива сума цифр).

Також під час роботи над цією задачою я освоїв створення окремих функцій.

Посилання на мою гілку де знаходяться ці програми:
https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/9

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

Завдання №1 Practice Task

```
PS C:\Users\Марсіян> cd "c:\PLUSPLUS\epic_1_practice_and_labs_markiian_makovych\ai_programming_playground_2024\ai_12\marki
_1_markiian_makovych } ; if ($?) { .\practice_work_task_1_markiian_makovych }
Investment: 1000
Annual rate (0<r<1): 0.05
Number of payouts: 4
Deposit period: 2
Money invested: 1000.000000$
Final sum: 1104.486101$
Earnings: 104.486101$
PS C:\PLUSPLUS\epic_1_practice_and_labs_markiian_makovych\ai_programming_playground_2024\ai_12\markiian_makovych\epic_1>
```

На це завдання я витратив *близько години*, бо ознайомлювався з новими для мене функціями.

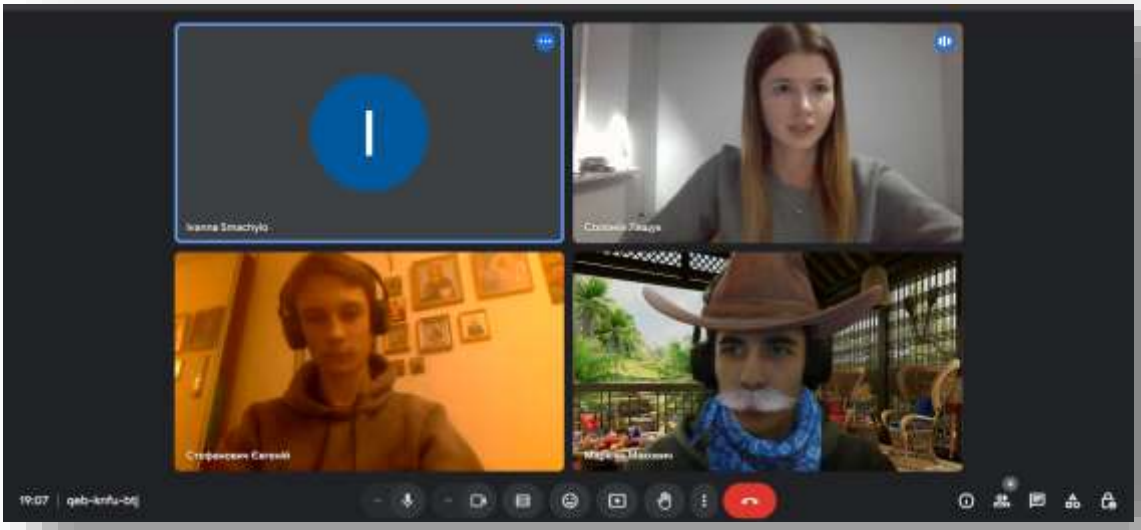
Завдання №2 Self Practice Task

```
1445
86
PS C:\PLUSPLUS\epic_1_practice_and_labs_markiian_makovych\ai_programming_playground_2024\ai_12\markiia
ovych\epic_1\" ; if ($?) { g++ self_practice_work_algotester_task_1_markiian_makovych.cpp -o self_prac
1236732
5218
PS C:\PLUSPLUS\epic_1_practice_and_labs_markiian_makovych\ai_programming_playground_2024\ai_12\markiia
ovych\epic_1\" ; if ($?) { g++ self_practice_work_algotester_task_1_markiian_makovych.cpp -o self_prac
47
10
PS C:\PLUSPLUS\epic_1_practice_and_labs_markiian_makovych\ai_programming_playground_2024\ai_12\markiia
```

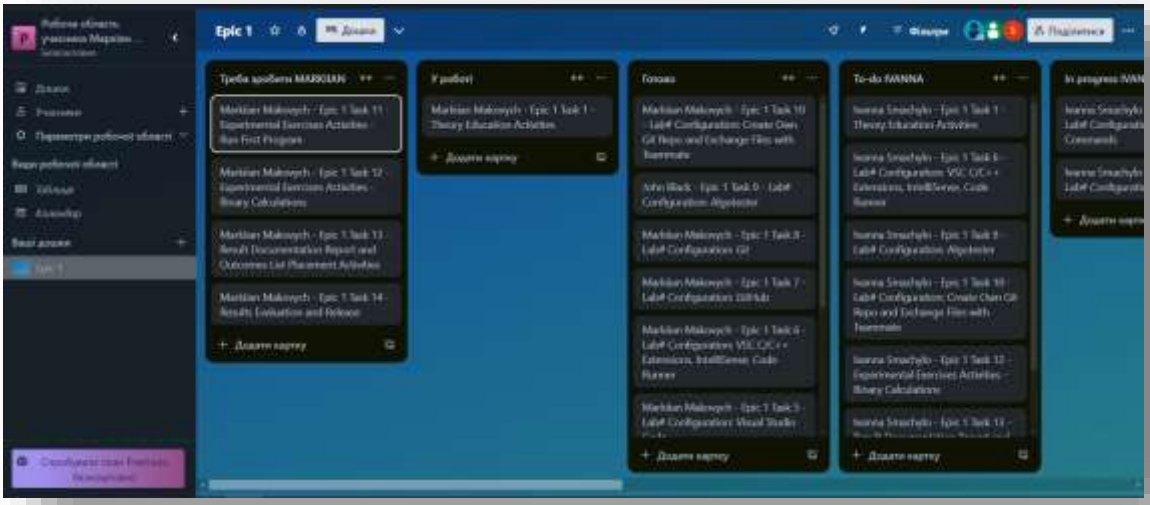
Виконуючи цю задачу на Algotester.com, я витратив *близько трьох годин*, оскільки під час виконання виникали помилки, які я довгий час не міг помітити.

6. Кооперація з командою:

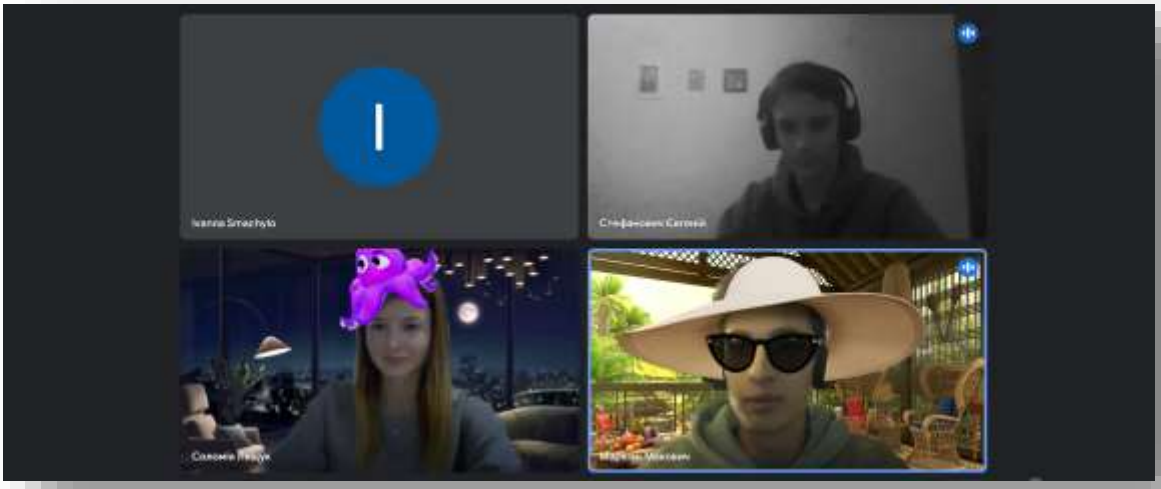
Зустріч з командою №1:



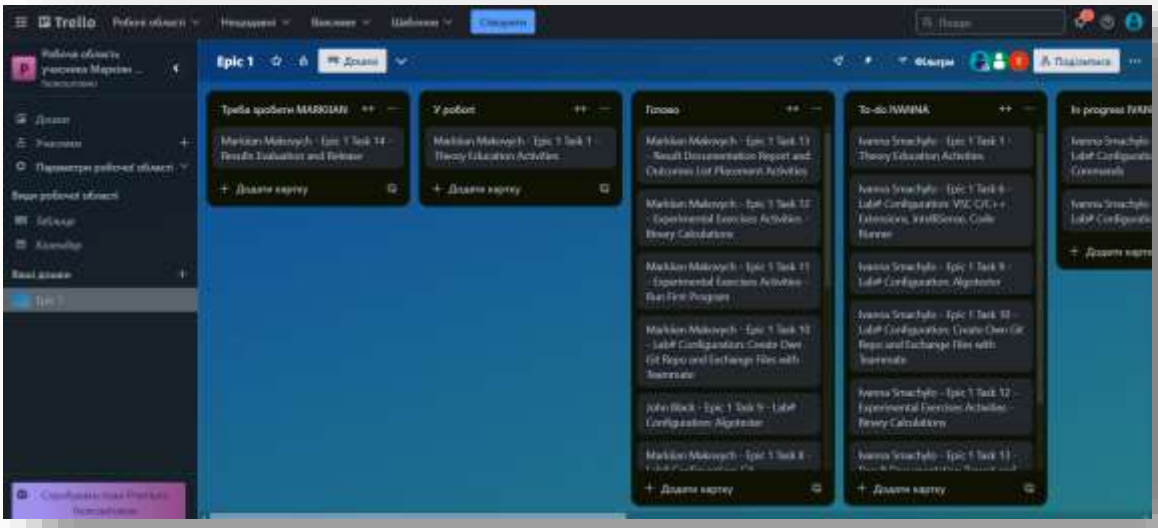
Прогрес Trello:



Зустріч з командою №2:



Прогрес Trello:



Висновки:

Працюючи над **Epic-1** я налаштував середовище для програмування, дізнався про Linux команди та освоїв частину з них. Створив дошку в Trello. Дізнався про сайт Draw.io, та навчився створювати блок-схеми. Зареєструвався на сайті Algotester.com, та виконав задачу. Налаштував Git та розібрався з потрібними мені командами. Створив свою гілку та Pull Request на сайті GtiHub. Навчився проводити калькуляції над числами в двійковій системі, та розвинув свої SoftSkills працюючи в команді з одногрупниками.