МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Західноукраїнський національний університет

Факультет комп’ютерний інформаційних технологій

Кафедра інформаційно-обчислювальних систем та управління

Лабораторна робота № 5

з дисципліни «Основи комп’ютерних наук»

Виконала:

Студентка групи

КН-12

Ряба М.В.

Тернопіль

2023 рік

**Теоретичні основи побудови алгоритмів**

**Мета:** Вивчити основні методи побудови алгоритмів. Навчитися будувати блок-схеми та представляти алгоритм програми у вигляді псевдо-коду.

**Хід роботи:**

1. Власна система найменувань псевдокоду

**Оголошення функції findLCM:** ця функція отримує два аргументи a та b, які представляють числа, для яких ми шукаємо НСК. Повертає вона ціле число, яке є НСК a та b;

**Оголошення функції** **main**;

**Оголошення змінних num1, num2, result:** num1 та num2 зберігають числа, введені користувачем, а result буде містити результат обчислення НСК;

**Введення першого числа:** користувачеві пропонується ввести перше число, яке зберігається в змінній num1;

**Введення другого числа:** користувач вводить друге число, яке зберігається в змінній num2;

**Виклик функції findLCM:** Функція findLCM викликається з аргументами num1 та num2, результат записується в змінну result;

**Функція findLCM:**

* Визначаються maxNum та minNum як максимальне та мінімальне введені числа;
* Змінна lcm ініціалізується значенням maxNum;
* Використовується цикл while, який перевіряє, чи число lcm є кратним як minNum, так і maxNum. Якщо так, цикл припиняє роботу;
* Значення lcm повертається як результат обчислення НСК;

**Вивід результату.**

1. Опис одного із алгоритмів, які подано у варіантах завдань, відповідно до порядкового номера у журналі групи:

#include <iostream>

using namespace std;

int main() {

setlocale(LC\_CTYPE, "ukr");

int num1, num2;

cout << "Введiть перше число: ";

cin >> num1;

cout << "Введiть друге число: ";

cin >> num2;

int maxNum = (num1 > num2) ? num1 : num2;

int minNum = (num1 < num2) ? num1 : num2;

int lcm = maxNum;

while (true) {

if (lcm % minNum == 0 && lcm % maxNum == 0) {

break;

}

lcm += maxNum;

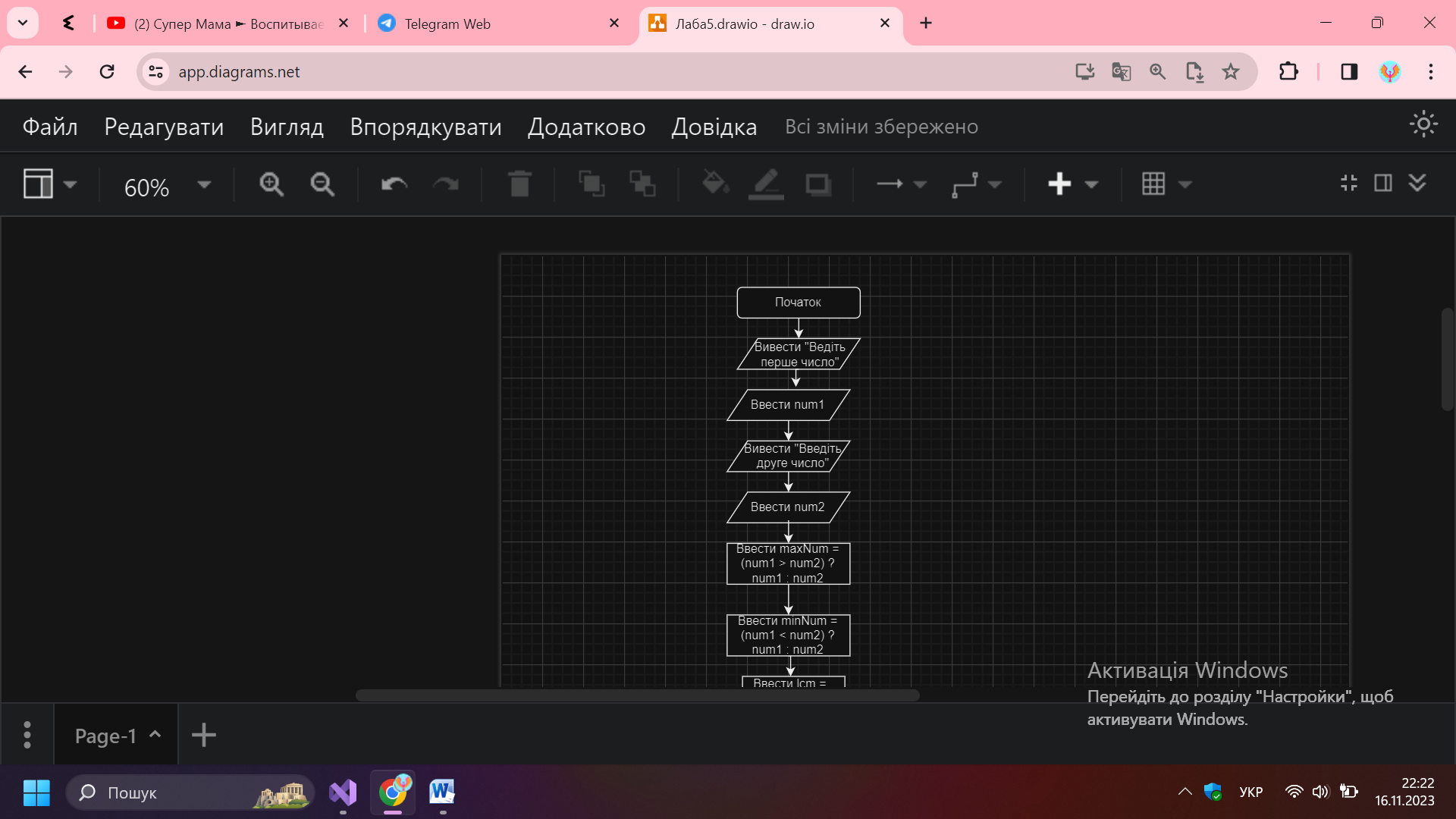
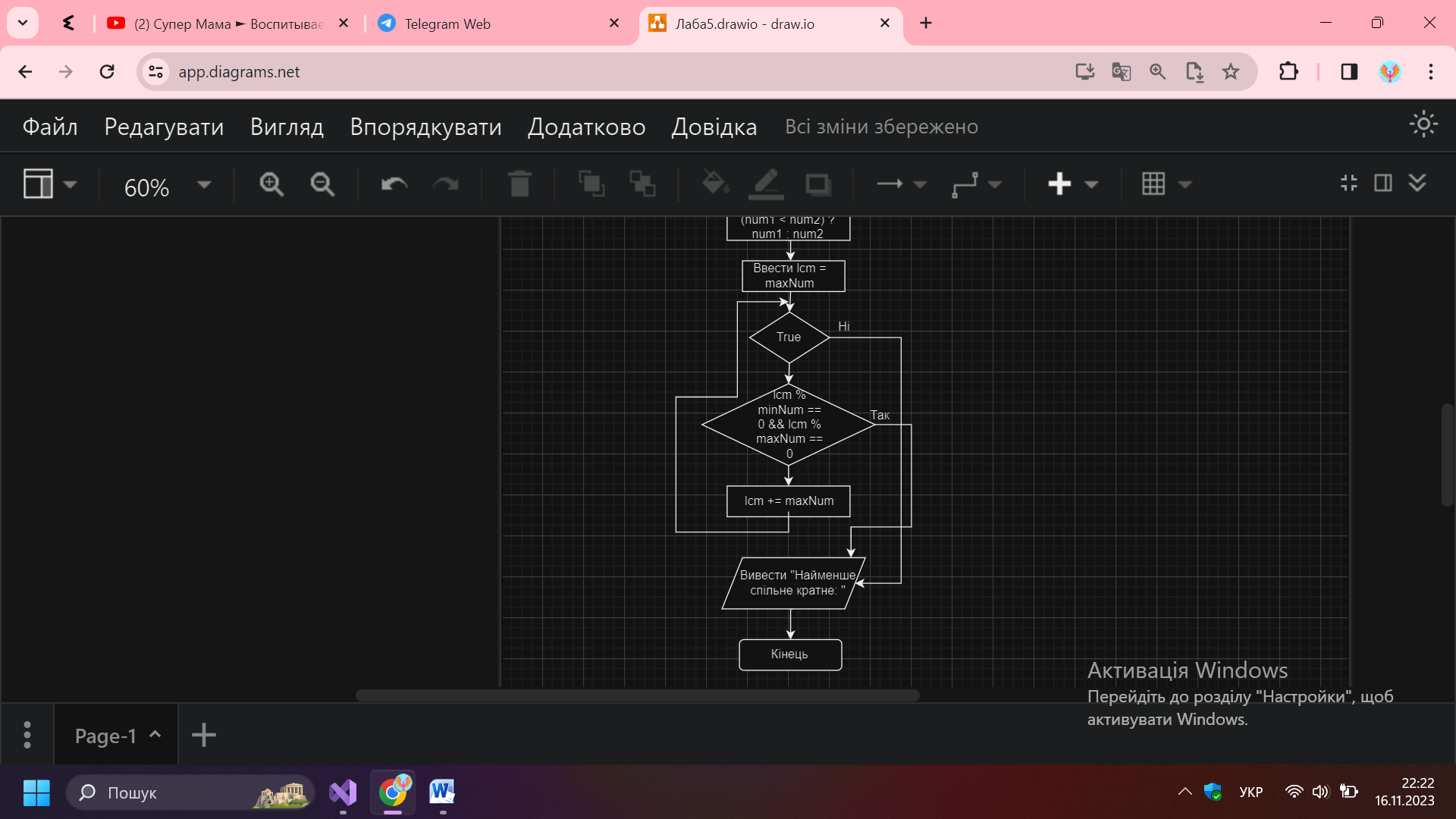
}

cout << "Найменше спiльне кратне: " << lcm << endl;

return 0;

}

1. Блок-схема до алгоритму:

**Відповіді:**

1. Алгоритм - це набір інструкцій, які виконують певні дії для досягнення конкретної мети. В інформатиці це послідовність кроків для вирішення задачі.
2. Основні властивості алгоритмів:

* **Явність:** Алгоритм повинен бути чітко визначений та зрозумілий для виконавця.
* **Дискретність:** Алгоритм повинен складатися з окремих, дискретних кроків чи операцій.
* **Визначеність:** Для кожного вхідного набору даних алгоритм повинен мати однозначно визначений результат.
* **Кінцевість:** Алгоритм повинен завершувати свою роботу за скінчений час.
* **Ефективність:** Алгоритм повинен працювати ефективно, забезпечуючи оптимальне використання ресурсів.
* **Універсальність:** Алгоритми можуть застосовуватися для різних вхідних даних та у різних контекстах.
* **Масштабованість:** Алгоритм може бути масштабованим, тобто застосовуватися до великої кількості даних чи завдань

1. Існують різні методи представлення алгоритмів. Основні з них включають:

* **Текстовий вигляд:** Алгоритм може бути описаний у вигляді послідовності текстових інструкцій, які вказують кроки вирішення задачі.
* **Схеми або блок-схеми:** Використовують графічні елементи, такі як блоки, стрілки та ромби, щоб візуально показати послідовність дій та рішень в алгоритмі.
* **Псевдокод:** Це специфікація алгоритму, що використовує мову програмування, але має більше загальний та людський вигляд, ніж справжній код.
* **Діаграми Варда:** Графічний метод, який використовує стрілки та блоки для представлення послідовності дій.
* **Умовні таблиці:** Подібні до таблиць істинності, вони використовуються для представлення умов та відповідних дій в алгоритмі.
* **Програмний код:** Пряме представлення алгоритму за допомогою мови програмування, яке може бути виконано на комп'ютері.

1. Блок-схема - це графічний спосіб представлення алгоритму за допомогою блоків, які представляють окремі дії або операції, і з'єднаних стрілками, що вказують напрямок виконання. Вона надає візуальне уявлення про послідовність дій у алгоритмі, допомагаючи зрозуміти логіку та порядок виконання операцій.
2. Порядок створення блок-схем визначається за допомогою стандартів, таких як "ГОСТ 19.701-90. Електронно-обчислювальна техніка, системи та мережі. Об'єкти автоматизації. Блок-схеми. Правила складання" в СНД, який визначає основні правила створення, оформлення та використання блок-схем для різних об'єктів автоматизації.
3. Основні елементи блок-схеми включають:

* **Процес (Блок):** Представляє виконання конкретного дії або операції. Зазвичай позначається прямокутником.
* **Рішення (Умова):** Визначає точку вибору між альтернативними шляхами виконання програми. Позначається ромбом.
* **Введення/Виведення:** Показує обмін даними між алгоритмом та зовнішнім середовищем, таким як користувач або інша програма. Зображується паралелограмом.
* **Паралельні лограф (збір або розгалуження потоку):** Використовується для позначення збору даних або подій.
* **Стрілки:** Показують напрямок виконання програми, з'єднуючи різні блоки.
* **Підписи та коментарі:** Використовуються для надання додаткової інформації або пояснень щодо блок-схеми.

1. Псевдокод - це неформальний спосіб опису алгоритму, що використовує природну мову та елементи структурованого програмування. Він не є конкретною мовою програмування, але служить для узагальненого представлення алгоритмів перед їхнім фактичним кодуванням. Псевдокод призначений для полегшення розуміння та аналізу алгоритмів, а також для їхнього опису без прив'язки до конкретного програмного коду.
2. Головна відмінність полягає у формі представлення: блок-схема - це графічний спосіб зображення алгоритму, тоді як псевдокод - це текстовий опис, який може містити структуровану мову програмування, але не є конкретним кодом.

**Висновок:** Вивчення основних методів побудови алгоритмів, а також навичок створення блок-схем і написання псевдокоду були цільовими завданнями. В процесі цього вдалося зрозуміти принципи створення послідовних кроків для вирішення завдань або проблеми. Вивчення алгоритмів дозволило не лише освоїти техніку їх побудови, а й розвинути навички логічного мислення та аналітичного механізму. Вміння представляти алгоритми у вигляді блок-схем та псевдокоду стало корисним інструментом при проектуванні програм та розв'язанні задач, надаючи можливість систематизувати дії й виразити їх у чіткій послідовності.