Львівський національний університет імені Івана Франка

Факультет прикладної математики та інформатики

**Паралельні та розподілені обчислення**

**Лабораторна робота №2**

Виконав:

Студент групи ПМі-33

Тимчишин Ярема

Львів 2023

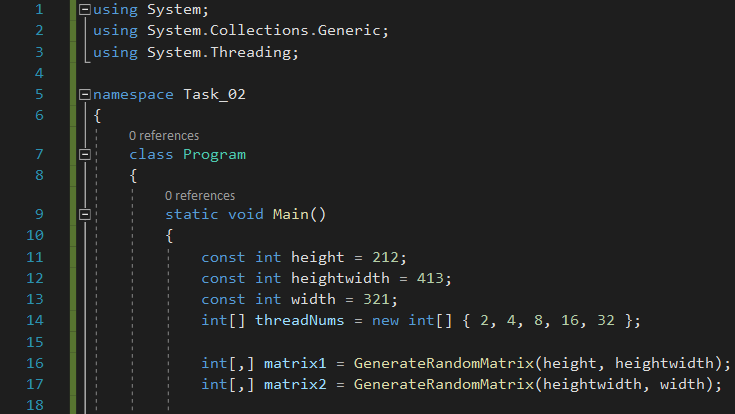
**Тема:** Множення матриць.

**Завдання:** Написати програму обчислення множення двох матриць (послідовний та паралельний алгоритми). Порахувати час роботи кожної з програм, обчислити прискорення та ефективність роботи паралельного алгоритму.

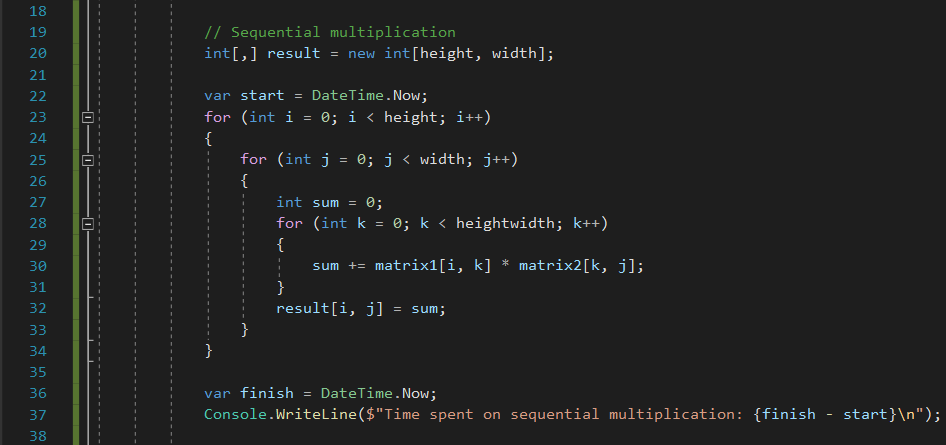
**Хід роботи**

Використовуючи мову програмування C# написав програму для обчислення множення двох матриць та порахував час роботи кожного із методів.

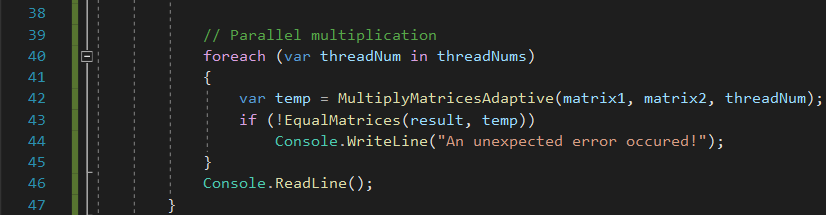
Почав із оголошення констант для висоти та ширини матриць, а також масиву “threadNums”, який зберігає кількість потоків для паралельного методу. Щоб дія множення двох матриць спрацювала коректно, необхідно щоб кількість колонок першої матриці відповідала кількості рядків другої матриці, тому я також створив змінну “heightwidth” щоб кількість колонок і рядків співпадала. Два двовимірні масиви “matrix1” та “matrix2” для зберігання випадкових цілих значень. Дані матриці заповнюються завдяки функції “GenerateRandomMatrix”, код якої я покажу в кінці.



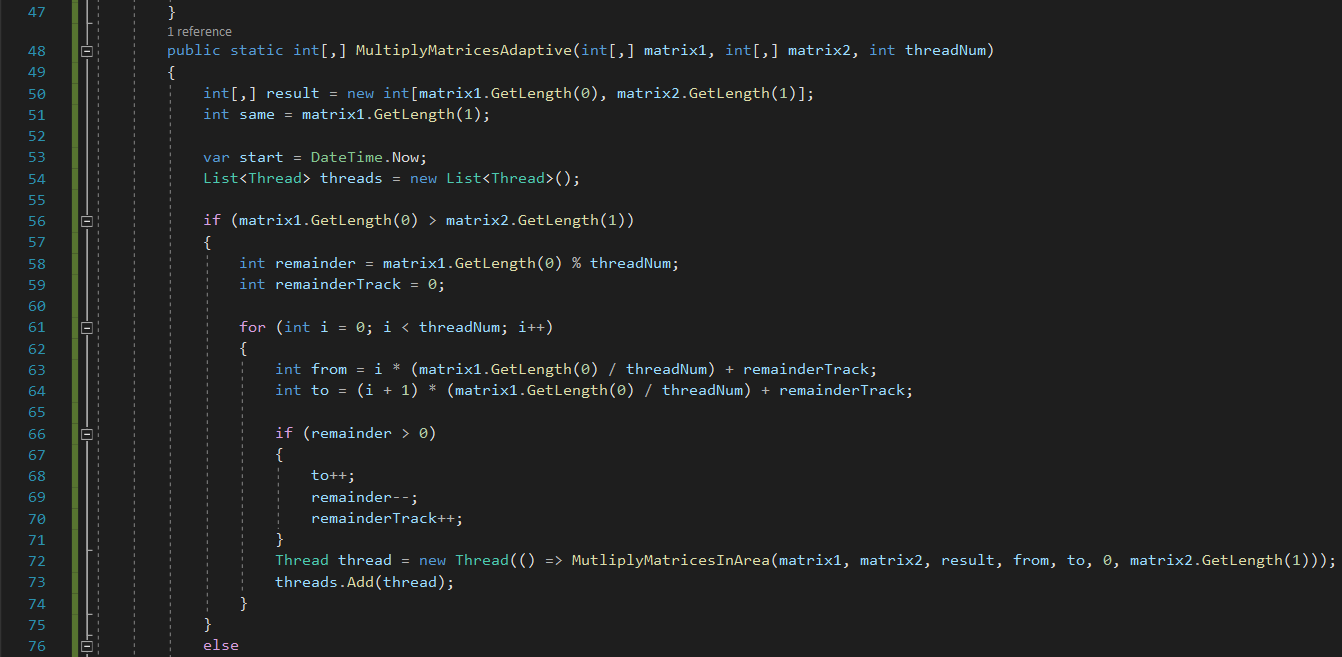
Для послідовного методу, почав із оголошення нового двовимірного масиву “result” в якому буде зберігатися результат множення матриць. Після чого за допомогою “DateTime.Now” записав в змінну “start” початок множення матриць. Потім за допомогою ще трьох вкладених циклів “for” відбувається послідовне множення елементів двох матриць. Знову за допомогою “DateTime.Now” присвоїв в змінну “finish” кінець множення двох матриць. Після чого, обчислив час витрачений на паралельне множення матриць та вивів його на консоль.

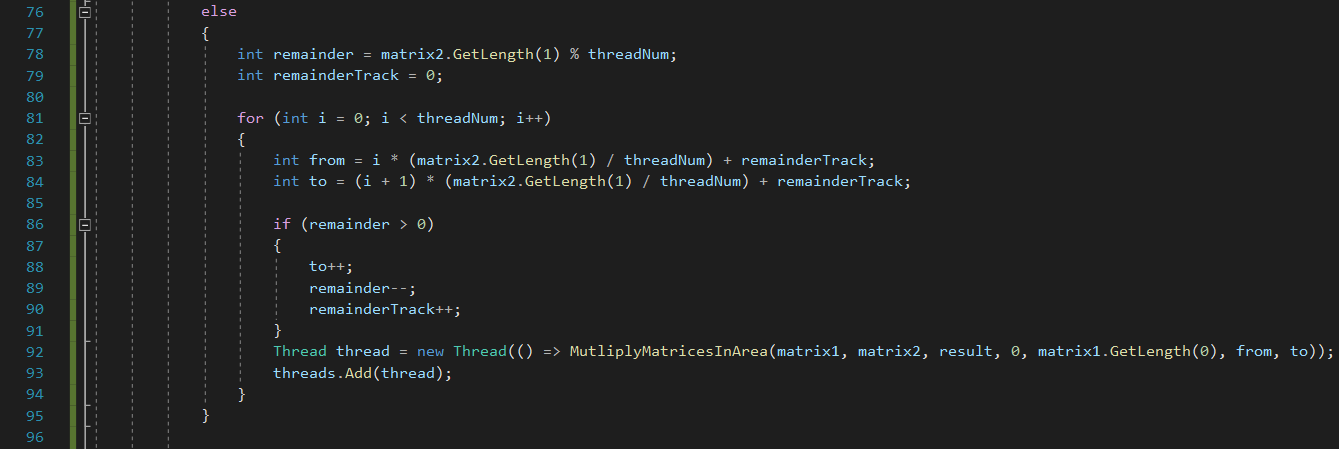


Для паралельного методу, почав із циклу “foreach” який перебирає значення масиву “threadNums” та викликає функцію “MultiplyMatricesAdaptive”, яка кожного разу бере різну кількість потоків і про яку я розкажу далі.

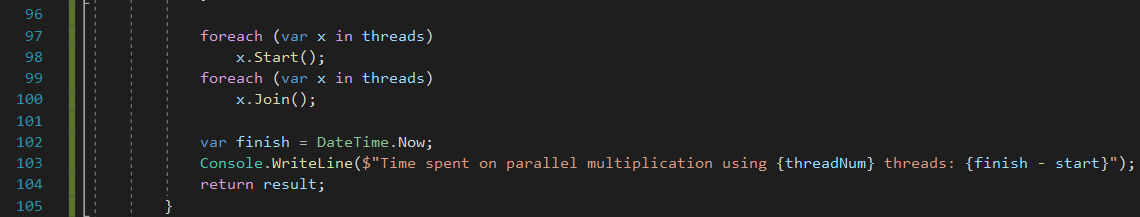


Функція “MultiplyMatricesAdaptive” відповідає за виконання паралельного множення матриць. Двовимірний масив “result” для зберігання множення двох матриць та “DateTime.Now” для засікання часу роботи алгоритму, усе аналогічно до того як це було в послідовному множення. Список “<Thread>” під назвою “threads” для зберігання об’єктів потоку для паралельної обробки. Після чого, використовую “if” для того щоб порівняти кількість рядків у “matrix1” та кількість стовпців у “matrix2”, щоб вирішити, розпаралелювати рядки чи стовпці. Якщо кількість рядків у “matrix1” є більшою за кількість стовпців у “matrix2”, програма розпаралелює рядки. На основі “threadNum” програма обчислює кількість рядків, над якими працюватиме кожен потік. Програма регулює значення “from” і “to” для кожного потоку, щоб рівномірно розподілити роботу та враховує будь-які залишки рядків і розподіляє їх між потоками. Якщо ж кількість стовпців у “matrix2” більша або дорівнює кількості рядків у “matrix1”, програма розпаралелює стовпці. Сам процес є подібний до розпаралелювання рядків, але цього разу програма розподіляє стовпці між потоками, зберігаючи рядки незмінними. Самі потоки створюються за допомогою об’єктів “Thread” і додаються до списку потоків.

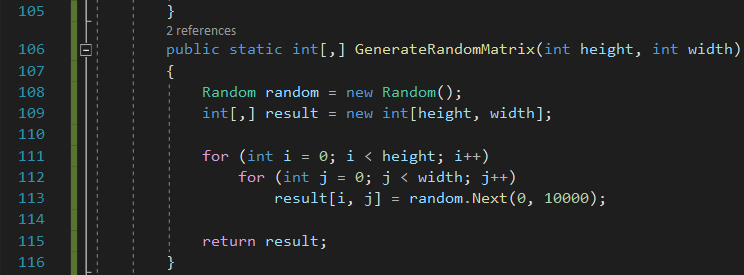




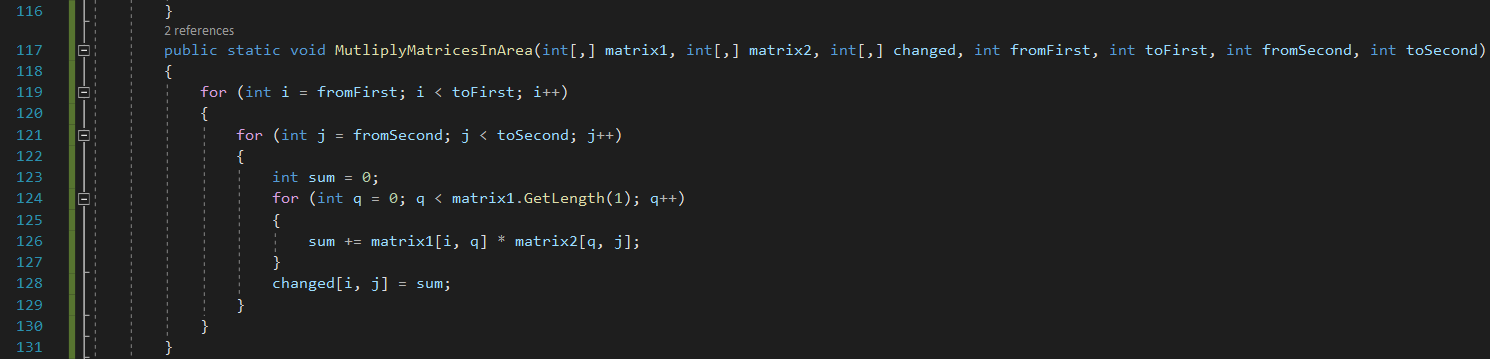
Після створення усіх потоків, програма запускає їх за допомогою методу “Start()”, потім чекає щоб всі потоки завершили свою роботу та з’єднює їх результати за допомогою методу “Join()”. Програма записує час закінчення, тоді коли всі потоки вже завершили свою роботу. Після чого програма обчислює час, витрачений на паралельне множення, та виводить його на консоль разом з кількістю використаних потоків.



Додаткова функція “GenerateRandomMatrix” генерує матрицю з заданими розмірами та заповнює цю матрицю випадковими цілими числами від 0 до 10000.



Додаткова функція “MultiplyMatricesInArea”. Ця функція відповідає за множення матриць у певній області, визначеній “fromFirst”, “toFirst”, “fromSecond” і “toSecond”. Вона використовується паралельними потоками для оновлення зміненої матриці результатами множення в межах призначеної їм області.



Додаткова функція “EqualMatrices” спочатку порівнює розміри (кількість рядків та стовпців) двох матриць за допомогою “matrix1.GetLength(0)” і “matrix2.GetLength(0)” для рядків та “matrix1.GetLength(1)” і “matrix2.GetLength(1)” для стовпців. Якщо ж розміри не співпадають, функція повертає “false”. Якщо ж розміри збігаються, функція переходить до порівняння елементів двох матриць. Якщо хоч один елемент не співпадає, функція повертає “false”. Якщо функція успішно завернує і перевірку розмірності, і порівняння по елементах, не знаходячи жодних невідповідностей, вона робить висновок що матриці дійсно рівні і повертає “true”.

