Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Інститут комп’ютерних наук та інформаційних технологій

Кафедра систем штучного інтелекту

Лабораторна робота №13

на тему:

“ Паралельне виконання.
Багатопоточність. Ефективність використання..”

з курсу:

“Об’єктно-орієнтоване програмування ”

Виконала:
ст. гр. КН-110
Гелетій Софія
Прийняв:
Гасько Р.Т

Львів 2018

Мета

- Ознайомлення з моделлю потоків Java.
- Організація паралельного виконання декількох частин програми.
- Вимірювання часу паралельних та послідовних обчислень.
- Демонстрація ефективності паралельної обробки.

Вимоги

1. Використовуючи програми рішень попередніх задач, продемонструвати можливість паралельної обробки елементів контейнера: створити не менше трьох додаткових потоків, на яких викликати відповідні методи обробки контейнера.
2. Забезпечити можливість встановлення користувачем максимального часу виконання (таймаута) при закінченні якого обробка повинна припинятися незалежно від того знайдений кінцевий результат чи ні.
3. Для паралельної обробки використовувати алгоритми, що не змінюють початкову колекцію.
4. Кількість елементів контейнера повинна бути досить велика, складність алгоритмів обробки колекції повинна бути зіставна, а час виконання приблизно однаковий, наприклад:
   - ○ пошук мінімуму або максимуму;
   - ○ обчислення середнього значення або суми;
   - ○ підрахунок елементів, що задовольняють деякій умові;
   - ○ відбір за заданим критерієм;
   - ○ власний варіант, що відповідає обраній прикладної області.
5. Забезпечити вимірювання часу паралельної обробки елементів контейнера за допомогою розроблених раніше методів.
6. Додати до алгоритмів штучну затримку виконання для кожної ітерації циклів поелементної обробки контейнерів, щоб загальний час обробки був декілька секунд.
7. Реалізувати послідовну обробку контейнера за допомогою методів, що використовувались для паралельної обробки та забезпечити вимірювання часу їх роботи.

8. Порівняти час паралельної і послідовної обробки та зробити висновки про ефективність розпаралелювання:

○ результати вимірювання часу звести в таблицю;

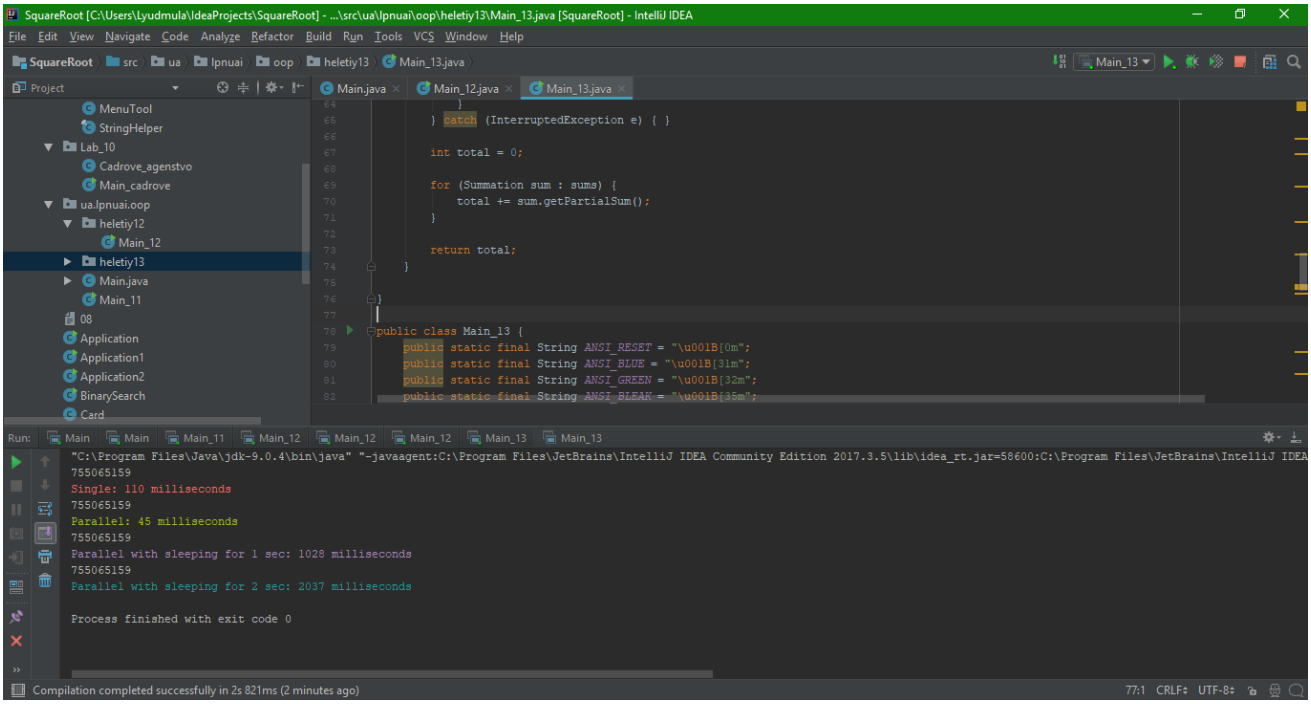
○ обчислити та продемонструвати у скільки разів паралельне виконання швидше послідовного.

Текст основних моментів програми

```
public static int parallelSum(int[] arr)
{
    return parallelSum(arr, Runtime.getRuntime().availableProcessors());
}

public static int parallelSum(int[] arr, int threads)
{
    int size = (int) Math.ceil(arr.length * 1.0 / threads);

    Summation[] sums = new Summation[threads];

    for (int i = 0; i < threads; i++) {
        sums[i] = new Summation(arr, i * size, (i + 1) * size);
        sums[i].start();
    }

    try {
        for (Summation sum : sums) {
            sum.join();
        }
    } catch (InterruptedException e) { }

    int total = 0;
```

```
        for (Summation sum : sums) {
            total += sum.getPartialSum();
        }

        return total;
    }

}
```

Приклад використання програми



Висновок: Я ознайомилась з моделлю потоків Java, організацією паралельного виконання декількох частин програми, вимірювання часу паралельних та послідовних обчислень та демонстрацією ефективності паралельної обробки.