14. Паралельне виконання. Ефективність використання

Мета: Вимірювання часу паралельних та послідовних обчислень.

Демонстрація ефективності паралельної обробки.

1 ВИМОГИ

1.1 Розробник

Інформація про розробника:

- Кулик Даніїл Ігорович
- НТУ “ХПІ” КІТ-118в
- Варіант 11

1.2 Загальне завдання

1. Забезпечити вимірювання часу паралельної обробки елементів контейнера за допомогою розроблених раніше методів.
2. Додати до алгоритмів штучну затримку виконання для кожної ітерації циклів поелементної обробки контейнерів, щоб загальний час обробки був декілька секунд.
3. Реалізувати послідовну обробку контейнера за допомогою методів, що використовувались для паралельної обробки та забезпечити вимірювання часу їх роботи.
4. 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.

2 ОПИС ПРОГРАМИ

2.1 Засоби ООП

Були розроблені додаткові функції для класу-контейнера. Розроблене діалогове меню та можливість зчитування даних з файлу.

2.2 Ієрархія та структура даних

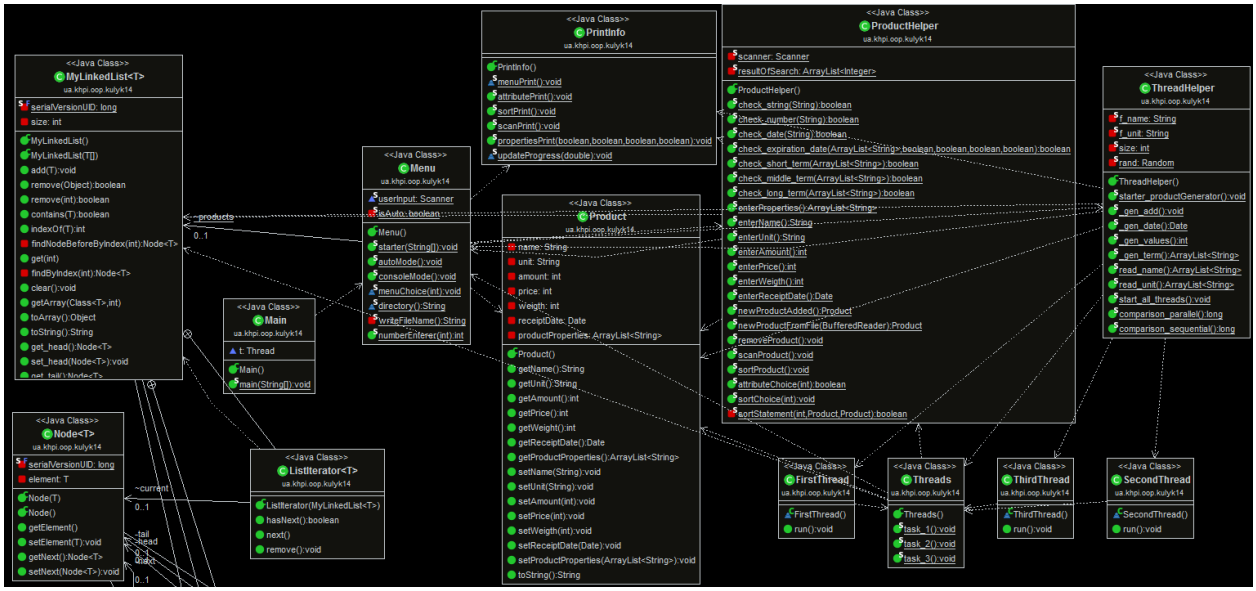

Рисунок 1 – Діаграма класів

2.3 Важливі фрагменти програми

```
    public static void task_1() throws InterruptedException {
            int count = 0;
for (Product product : Menu.products) {
    if (!Thread.currentThread().isInterrupted()) {
        if(ProductHelper.check_long_term(product.getProductProperties())) {
            count++;
        }
                            Thread.sleep(100);
    } else {
        throw new InterruptedException();
    }
}
System.out.println("First Task finished. Products with long term : " + count);
    }
    public static void task_2() throws InterruptedException {
            int count = 0;
for (Product product : Menu.products) {
    if (!Thread.currentThread().isInterrupted()) {
        if(product.getName().indexOf("Banana") >= 0) {
            count++;
        }
                            Thread.sleep(100);
    } else {
        throw new InterruptedException();
    }
}
System.out.println("Second Task finished. Banana products : " + count);
    }
```

Рисунок 2 – Алгоритми для розрахунку

```
    public static long comparison_parallel() {
            long time_start = System.currentTimeMillis();
            System.out.println("Starting all threads...");
            FirstThread first = new FirstThread();
Thread t1 = new Thread(first,"FirstThread");

            SecondThread second = new SecondThread();
Thread t2 = new Thread(second,"SecondThread");

            ThirdThread third = new ThirdThread();
Thread t3 = new Thread(third,"ThirdThread");

            t1.start();
            t2.start();
            t3.start();

            try {
                    t1.join();
                    t2.join();
                    t3.join();
            } catch (InterruptedException e) {
                    e.printStackTrace();
            }
            System.out.println("Finishing all threads...");
            return System.currentTimeMillis() - time_start;
    }
```

Рисунок 3 – Вимірювання паралельного виконання

```
    public static long comparison_sequential() {
            long time_start = System.currentTimeMillis();
            System.out.println("Starting sequence...");
            try {
                    Threads.task_1();
                    Threads.task_2();
                    Threads.task_3();
            } catch (InterruptedException e) {
                    e.printStackTrace();
            }
            System.out.println("Finishing sequence...");
            return System.currentTimeMillis() - time_start;
    }
```

Рисунок 4 – Вимірювання послідовного виконання

3 ВАРІАНТИ ВИКОРИСТАННЯ

Програма дозволяє створювати об’єкти – товари – , що заносяться у запис каталогу, тобто створюється масив об’єктів. Користувач може додавати об’єкти до масиву, видаляти елементи вибірково, а також очистити увесь масив одним викликом відповідної кнопки меню. Також присутня можливість серіалізувати/десеріалізувати об’єкти з файлу.

```
Finished
[100] User> 13
Starting....
Starting all threads...
First Thread started
Second Thread started
Third Thread started
First Task finished. Products with long term : 36
First Thread finished
Third Task finished. Products with long term : 25
Third Thread finished
Second Task finished. Banana products : 0
Second Thread finished
Finishing all threads...
Starting sequence...
First Task finished. Products with long term : 36
Second Task finished. Banana products : 0
Third Task finished. Products with long term : 25
Finishing sequence...
-----------------------------------------------
Time of the parallel execution   | 10007 ms
-----------------------------------------------
Time of the sequential execution | 30016 ms
-----------------------------------------------
```

Рисунок 5 – Порівняння паралельного та послідовного виконання

ВИСНОВКИ

В даній лабораторній роботі організував паралельне та послідовне виконання декількох частин програми, а також визначив, що паралельне виконання спрацьовує набагато швидше послідовного.