Лабораторна робота №6.

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.

Розробник: Гринишин Анастасія , КН-108, номер варіанту індивідуального завдання – 9.

Задача: Паралельне виконання. Багатопоточність. Ефективність використання.

Ієрархія та структура класів:

1.Клас Main, який містить функцію – main, doexample.

2.Клас Filego, який містить 2 функції – doFile, make_info.

3.Клас Demain,який містить 3 поля – numberFlight, date, numberFlight і їхні гетери та сетери.

4.Клас Station, який містить 4 поля – nameStation, dateArrival, dateDeparture, freeSeat.

5.Клас ConstructorsForXML , який містить 2 функції – WriteParamXML, read.

6.Клас SimpleArray, якиц є контейнером.

7.Інтерфейс Simple.

8.Клас ArrayIterator, який містить 2 функції – hasNext, next.

9.Клас ReDex, який містить 5 функції – time, station, seat, date, nameF.

10.Клас Search, який містить 2 функції – search, run.

10.Клас Max, який містить 2 функції – max, run.

10.Клас Minimum, який містить 2 функції – minimum,run.

Фажливий фрагмент коду:

```
129
130         }
131         static void doexample()
132         {
133             Max max = new Max();
134             Minimum miminum = new Minimum();
135             Search search = new Search();
136             max.timeStart=startTime;
137             max.timeout=MILLIS_TO_WAIT;
138             miminum.timeStart=startTime;
139             miminum.timeout=MILLIS_TO_WAIT;
140             search.timeStart=startTime;
141             search.timeout=MILLIS_TO_WAIT;
142             search.search=numberF;
143             max.start();
144             miminum.start();
145             search.start();
146         }
147
148
149     }
```

Висновок

У ході даної роботи дізналась про багатопоточність, як і навіщо її використовувати, і чим паралельна обробка даних краща за послідовну.