ГУАП
КАФЕДРА № 51

ПРЕПОДАВАТЕЛЬ

доцент, к.т.н.		Линский Е. М.
должность , уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЫ №8
СОЗДАНИЕ ПРОГРАММЫ НА ЯЗЫКЕ JAVA

по курсу: ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	5022		С.А.Баландюк
		подпись, дата	**инициалы, фамилия**

Санкт-Петербург 2022

Задание

Реализовать класс ParallelMatrixProduct для многопоточного умножения матриц UsualMatrix. В конструкторе класс получает число потоков, которые будут использованы для перемножения (число потоков может быть меньше, чем число строк у первой матрицы).

В функции main сравнить время перемножения больших случайных матриц обычным и многопоточным способом. Получить текущее время можно с помощью методов класса System.

Дополнительное задание

Реализуйте многопоточную функцию для подсчета количества нечетных чисел в ArrayList.

int calcOddNum(ArrayList<int> array, int threadNum);

Функция в качестве параметра получает число потоков.

Инструкция

Класс ParallelMatrixProduct содержит в себе метод public UsualMatrix multiThreadingMultiply(UsualMatrix firstMatrix, UsualMatrix secondMatrix), который возвращает произведение матриц firstMatrix на secondMatrix. Данный метод позволяет пользователю воспользоваться многопоточным умножением матриц.

Тестирование

firstMatrix и secondMatrix заполняются в main случайным образом с помощью класса Random(). Помимо сравнения скорости работы многопоточного и обычного умножения, в main также с помощью метода equals проверяется, что результат многопоточного умножения равен результату обычного умножения. Время выполнения выводится на экран в миллисекундах.

1. Тест 1

matrixSize = 1500 x 1500
amountOfThreads = 12

Result:

```
Time for multi threading multiplication of matrix: 3418
Time for default multiplication of matrix: 16995
Result of multithreading and default multiplication is equals: true
```

2. Тест 2

matrixSize = 1500 x 1500
amountOfThreads = 4

Result:

```
Time for multi threading multiplication of matrix: 4552
Time for default multiplication of matrix: 17214
Result of multithreading and default multiplication is equals: true
```

3. Тест 3

matrixSize = 1500 x 1500
amountOfThreads = 1

Result:

```
Time for multi threading multiplication of matrix: 15375
Time for default multiplication of matrix: 17359
Result of multithreading and default multiplication is equals: true
```

Тестирование дополнительного задания

Массив чисел заполняется случайным образом с помощью класса Random(). Помимо сравнения скорости работы многопоточного и обычного алгоритма нахождения количества нечетных чисел, в main также проверяется, что результат многопоточного алгоритма равен результату обычного алгоритма. Время выполнения выводится на экран в миллисекундах.

1. Тест 1

sizeOfArray = 150000000
amountOfThreads = 12

Result:

```
Amount off odd numbers(multithreading algorithm): 83333636
Time for multithreading calculate amount of odd numbers: 163
Amount off odd numbers(default algorithm): 83333636
Time for default calculate amount of odd numbers: 782
```

2. Тест 2

sizeOfArray = 150000000
amountOfThreads = 4

Result:

```
Amount off odd numbers(multithreading algorithm): 83323294
Time for multithreading calculate amount of odd numbers: 265
Amount off odd numbers(default algorithm): 83323294
Time for default calculate amount of odd numbers: 744
```

3. Тест 3

sizeOfArray = 150000000
amountOfThreads = 1

Result:

```
Amount off odd numbers(multithreading algorithm): 83344040
Time for multithreading calculate amount of odd numbers: 815
Amount off odd numbers(default algorithm): 83344040
Time for default calculate amount of odd numbers: 739
```