EXTERNAL JAVA TRAINING. TASK MULTITHREADING

Deadline – 10.00, 20.09.19

Task. Многопоточность

Разработать многопоточное приложение, использующее разделяемые ресурсы. Любая сущность, желающая получить доступ к разделяемому ресурсу, должна быть потоком. В приложении должна быть реализована функциональность, определенная индивидуальным заданием.

Требования

- Программа должна использовать возможности синхронизации, поставляемые библиотеками **java.util.concurrent** и **java.util.concurrent.locks**.
- **Не использовать** synchronized, volatile, BlockingQueue и другие ограниченно потокобезопасные коллекции.
- Классы и другие сущности приложения должны быть грамотно структурированы по пакетам и иметь отражающую их функциональность название.
- Использовать шаблон **State** для описания состояний объекта, если только этих состояний больше двух.
- Вместо Thread.sleep использовать только возможности перечисления TimeUnit.
- Данные инициализации матрицы и потоков считывать из текстового файла.
- **В приложении должен присутствовать потокобезопасный Singleton.**
- Для записи логов использовать Log4J2.
- Разрешается для вывода работы потоков использовать метод main

Задание

Матрица. Создано ***Y*N*** потоков. Инициализирована целочисленная матрица размерности ***NxN***. Именем каждого потока является некоторое уникальное целое число. Каждый поток записывает в диагональ матрицы свое имя-число и одновременно изменяет один из элементов строки или столбца в котором находится изменяемый диагональный элемент. Только один поток может изменить конкретный элемент стоящий на диагонали и элемент в его строке или столбце. Далее каждый поток должен посчитать сумму всех элементов в строке и столбце с номером своей диагонали. После того как отработают первые ***N*** потоков производится запись матрицы и результатов вычисления в файл, и к работе с матрицей допускаются следующие ***N*** потоков.