Задания

Multithreading. **Lesson 1.**

1. Создать класс *NewThread* расширяющий *Thread*. Переопределить метод *run()*. В цикле for вывести на консоль символ 100 раз. Создать экземпляр класса и запустить новый поток.
2. Создать класс, реализующий интерфейс *Runnable*. Переопределить*run()* метод. Создать цикл *for*. В цикле распечатываем значения от 0 до 100 делящиеся на 10 без остатка. Используем статический метод *Thread.sleep(100)* чтобы сделать паузу. Создать три потока, выполняющих задачу распечатки значений.
3. Написать 3 класса, реализующие интефейс Runnable, в конструкторе принимают массив int. В главном потоке вывести: Закончили выполнение в главном потоке. Использовать join так, чтобы сообщения шли в порядке функциональности, описанной ниже(1,2,3). Сообщение из главного потока должно быть в самом конце. Каждый поток имеет свою функциональность.
   1. Ищет в массиве минимальное значение. (можно использовать Stream API) Выводит результат на экран.
   2. Ищет в массиве максимальное значение. (можно использовать Stream API) Выводит результат на экран.
   3. Находит стреднее значение. Выводи результат на экран.

Multithreading. **Lesson 2.**

1. Необходимо создать три потока, которые изменяют один и тот же объект. Каждый поток должен вывести на экран одну букву 100 раз, и затем увеличить значение символа на 1.   
   a) Создать класс расширяющий Thread.   
   б) Переопределить метод run() - здесь будет находиться синхронизированный блок кода.   
   в) Для того чтобы три объекта потока имели доступ к одному объекту, создаем конструктор принимающий на вход StringBuilder объект.   
   г) Синхронизированный блок кода будет получать монитор на объект StringBuilder из пункта в).   
   д) Внутри синхронизированного блока кода выведите StringBuilder на экран 100 раз, а потом увеличьте значение символа на 1.   
   е) В методе main() создайте один объект класса StringBuilder, используя символ ‘a’. Затем создайте три экземпляра объекта нашего класса и запустите потоки.
2. Изменить *MyQueue*:  
   a) Вместо int n добавить *Queue<T>* (*MyQueue* сделать обобщенным), которая будет содержать объекты создаваемые *Producer*.   
   б) Добавьте еще один объект Consumer, который будет запускаться тоже отдельным потоком.   
   в) Выводите на консоль какой из объектов *Consumer* обработал объект из очереди.   
   г) Измените цикл for на бесконечный цикл.
3. Обедающие философы. Пять безмолвных философов сидят вокруг круглого стола, перед каждым философом стоит тарелка спагетти. Вилки лежат на столе между каждой парой ближайших философов. Каждый философ может либо есть, либо размышлять. Философ может есть только тогда, когда держит две вилки — взятую справа и слева. Взятие каждой вилки и возвращение её на стол являются раздельными действиями, которые должны выполняться одно за другим.