Оленчикова Т.Ю.

**Методические указания и задачи к выполнению лабораторной работы по дисциплине «**Программирование на языке Java для анализа данных**»**

**Лабораторная работа 4 Многопоточное программирование**

Цель: Научиться разрабатывать высокопроизводительные программы на Java, используя параллелизм потов исполнения.

**Теоретические материалы**

## [Многопоточность Thread, Runnable](https://java-online.ru/java-thread.xhtml)

## [Многопоточный пакет util.concurrent](https://java-online.ru/concurrent.xhtml)

## [Многозадачность](https://www.bestprog.net/ru/2021/01/13/java-multitasking-threads-of-execution-threads-basic-concepts-ru/). Потоки выполнения (Threads).

**Индивидуальные задания для лабораторной работы.**

В работе одно задание для всех вариантов и контрольный вопрос по варианту.

**Задание**

1. Напишите приложение, эмулирующее работу фабрики по сборке автомашин, используя легковесные потоки.
2. Напишите ответ на контрольный вопрос в соответствии с вашим вариантом

**1) Постановка задачи**

Напишите приложение, эмулирующее работу фабрики по сборке автомашин. Машина состоит из 3-х частей: кузов, двигатель и аксессуары. Машину надо собрать и отвезти на склад, откуда она поступает дилерам. Процесс работы фабрики показан на картинке:

Рисунок Схема работы фабрики по производству автомашин

**2) Возможная структура программы**

./factory – пакет содержит основные классы программы.

./threadpool – пакет для реализации пула потоков.

**3) Описание задачи**

1. Все склады имеют определенный размер, который нельзя превышать. Размеры складов, количествосборщиков, поставщиков и дилеров задаются в конфигурационном файле. Приложение предоставляет графический интерфейс (библиотека Swing), где можно смотреть основные параметры работы фабрики и контролировать процесс.
2. Каждый сборщик,поставщик и дилердолжен работать в отдельном потоке. Для синхронизации и ожидания событий необходимо использовать мониторы синхронизации (notify(), notifyAll(), wait()). Наличие процедуры ожидания в виде цикла автоматически ведет к непринятию задания. Каждаядеталь - это отдельный объект. Хранить просто количество изделий/деталей нельзя - необходимо хранить непосредственно объекты. Каждый объект должен иметь уникальный идентификатор для отслеживания.
3. Потоки, которые представляют поставщиков деталей, поставляют одну деталь раз в N миллисекунд. Если какой-то склад деталей полон, то поставщик ожидает освобождения места для деталей (используя методы wait(), notify()). Скорость работы поставщиков определяется 3-мя ползунками (для каждого типа деталей). Должно отображаться кол-во деталей на каждом из складов в текущий момент и кол-во деталей, произведенных поставщиками (для поставщиков аксессуаров можно общий показывать).
4. Потоки, которые представляют дилеров, запрашивают со склада готовой продукции 1 машину в M миллисекунд. Скорость запрашивания машин можно регулировать ползунком в интерфейсе окна. Интерфейс также должен показывать кол-во произведенных машин (вообще) и кол-во машин на складе в данный момент. При отправке машины дилеру информация о покупке должна писаться в лог работы фабрики (в файл) в виде строки:

<Time>: Dealer <Number>: Auto <ID> (Body: <ID>, Motor: <ID>, Accessory: <ID>)

Включение/отключение лога контролируется с помощью специального параметра в конфигурационном файле.

1. Поток контроллера склада готовой продукции просыпается при любом отправке машины со склада продукции. Он анализирует состояние склада и передает запрос на изготовление новых машин (в случае необходимости) на фабрику.
2. На фабрике работает несколько потоков (сборщиков) в рамках ThreadPool. Задачами для ThreadPool являются запросы на создание новых машин (от контроллера склада готовых изделий). При выполнении такой задачи сборщик должен взять по одной детали, необходимой для сборки машины, с соответствующих складов. Если на складе нет нужной детали, то поток ждет поставки. Собирая новую машину,рабочий создает новый объект и с помощью всехнеобходимых объектов, представляющих детали. После этого объект отправляется на склад готовой продукции. Если склад полон, то рабочий ждет освобождения места для новой машины. Интерфейс должен отображать, сколько всего было сделано машин и сколько задач еще ждут исполнителя (в очереди задач ThreadPool).
3. Конфигурационный файл должен предоставлять настройки для задания вместимости всех складов и количестве всех типов потоков. Примерный список параметров в конфигурационном файле (просьба использовать свои имена):

StorageBodySize=100

StorageMotorSize=100

StorageAccessorySize=100

StorageAutoSize=100

AccessorySuppliers=5

Workers=10

Dealers=20

LogSale=true

**Требования к реализации**

1. Классы объектной модели (деталь, склад, поставщик, сборщик и т.д.) не должны зависеть от библиотеки графического интерфейса SWING.
2. Пул потоков (ThreadPool)нужно реализоватьв отдельном пакете. ThreadPoolдолжен выполнять абстрактные задачи и не зависеть от реализации (задача на сборку автомашины в фабрике).
3. Программа должна корректно завершаться. При получении сигнала о закрытии окна все потоки должны корректно прерываться, и лог-файл закрываться.

**Контрольные вопросы**

1. Какие средства для работы с многопоточностью знаете?
2. Что такое процесс и поток? Чем отличается процесс от потока?
3. Расскажите о синхронизации между потоками. Для чего используют методы wait(), notify() - notifyAll(), join()?
4. Как остановить поток?
5. Как между потоками обмениваться данными?
6. В чем заключается отличие класса Thread от интерфейса Runnable?
7. Есть потоки Т1, Т2 и Т3. Как реализовать их последовательное выполнение?
8. Опишите жизненный цикл потока.
9. Если объявить метод synchronized, то какой эффект будет этим достигнут?
10. Опишите механизмы синхронизации потоков
11. Как работают static synchronized методы?
12. Если один поток начал исполнение synchronized-блока, указав ссылку на некий объект, может ли другой поток обратиться к полю этого объекта? К методу?
13. Почему метод wait требует обработки InterruptedException, а методы notify и notifyAll– нет?
14. Может ли поток никогда не выйти из метода wait, даже если будет вызван метод notify? notifyAll?

**Требования к оформлению отчета.**

Отчет по ЛР состоит из

**а)** Титульный лист: название ЛР, ФИО студента и номер группы, № варианта.

б)Текст задания; в) UML-диаграмму классов (создаете в Umlet):

г) Исходный текст программы;

д) Скрины выполнения;

е) Выводы.

**Критерии оценивания.**

За решение задачи вы можете получить до 2 баллов: задание выполнено полностью и правильно -2 балла; имеются незначительные ошибки -1 балл; приложение не работоспособно – 0 баллов;

Ответ на контрольный вопрос должен продемонстрировать понимание механизмов Java, за ответ вы можете получить до 2 баллов: ответ полный и правильный -2 балла; ответ не полный -1 балл; ответ не по существу – 0 баллов.

Итого, максимальная оценка - 4 балла

**Внимание!** Полученная оценка автматически снижается на 2% за каждую полную неделю задержки сдачи отчета по работе.

**Библиографический список**

1. Программирование на языке Java. Конспект лекций : учебно-методическое пособие / А. В. Гаврилов, С. В. Клименков, Ю. А. Королёва [и др.]. — Санкт-Петербург : НИУ ИТМО, 2019. — С. 5-17, 26-35. — URL: https://e.lanbook.com/book/136549 (дата обращения: 21.08.2023). — Режим доступа: для авториз. пользователей.
2. Пономарчук, Ю. В. Программирование на языке Java : учебное пособие / Ю. В. Пономарчук, И. В. Кузнецов. — Хабаровск : ДВГУПС, 2021. — 103 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/259451 (дата обращения: 21.08.2023). — Режим доступа: для авториз. пользователей.

**Электронные ресурсы**

Руководство по языку программирования Java [Электронный ресурс]. – URL: <https://metanit.com/java/tutorial//> – Свободный доступ.

Курсы, статьи по Java. – URL: <https://javarush.com/> – Свободный доступ.

Учебное пособие по Java [Электронный ресурс]. – URL: <https://www.bestprog.net/ru/sitemap_ru/java/> – Свободный доступ.