альфаЛАБ

лабораторная
информационная
система

Автор:	Тимофей Косачев
Дата:	12.05.2022
Версия:	1.1

Тестовое задание

Разработчик Java

www.alfalabsystem.ru
info@alfalabsystem.ru

1 Предисловие

Наша компания проводит собеседование по следующему алгоритму:

- Соискателю даются небольшие задачи для самостоятельного выполнения перед собеседованием.
- Соискатель выполняет их, уведомляет HR и согласовывает дату и время собеседования. Высылать решение нам не нужно.
- На собеседовании мы смотрим решения задач, задаем дополнительные вопросы и даем небольшие дополнения к задачам, чтобы соискатель в режиме online внес правки в код для их решения.

Предполагается, что все решения должны компилироваться и запускаться на компьютере соискателя. Для решения задач необходимо использовать Java 8 или выше и Hibernate 5 или выше. Использование прочих библиотек и фреймворков – на усмотрение соискателя.

2 Задача 1. Hibernate

2.1 Предисловие

В качестве основного фреймворка доступа к БД в наших проектах используется Hibernate. Поэтому от соискателя требуется уверенное владение и понимание особенностей его работы (кеширование, взаимодействие с БД, распределение внутренних слоев архитектуры приложения и т.п.).

2.2 Структура базы данных

Создайте базу данных (любую) и в ней схему для сущностей «Человек» и «Документ». Набор полей на ваше усмотрение исходя из личного жизненного опыта и понимания предметной области.

Необходимо учесть, что по стандарту компании имена таблиц и полей в БД пишутся с заглавной буквы. Если в названии несколько слов, то они разделяются знаком подчеркивание, например LAST_NAME.

2.3 Модель данных

Создайте классы для персистентных сущностей «Человек» и «Документ» в соответствии со схемой базы данных из предыдущего пункта.

2.4 Задача

Напишите программу, которая находит и печатает в консоли данные (ФИО гражданина, тип документа и номер документа) всех граждан и их документов, у которых в номере документа есть «777». Должна выводиться информация только по активным документам.

3 Задача 2. Stream API

3.1 Предисловие

Stream API позволяет сделать код понятнее и убрать нагромождение циклов. Поэтому мы всячески приветствуем использование данного механизма в коде.

3.2 Задача

Имеется интерфейс:

```
public interface NamedObject {
    public String getName();
}
```

Необходимо написать класс Grouper с одним методом groupByName, который на входе получает коллекцию объектов, реализующих интерфейс NamedObject и преобразует ее в Map с ключом – имя объекта, и значением – коллекция объектов с таким именем.

4 Задача 3. Многопоточность

4.1 Предисловие

В наших проектах часто приходится выполнять определенные действия в фоне и впоследствии собирать данные для выдачи из разных потоков, поэтому понимание многопоточности, синхронизационных примитивов необходимо.

4.2 Задача

Необходимо написать программу, запускающую 2 одинаковых потока (или задания). Потоки работают параллельно (одновременно).

Каждый поток должен дописывать в один и тот же файл на диске (Result.txt) разделённые пробелами текстовые записи простых чисел, упорядоченных по возрастанию и ограниченных сверху числом 1000000 (миллион).

Числа в файле не должны дублироваться или быть пропущенными.

В отдельном собственном файле (Thread1.txt или Thread2.txt) каждый поток должен сохранить те числа (так же через пробел), которые записал в общий файл (Result.txt) именно он.

Простое число — это натуральное число, имеющее ровно два различных натуральных делителя единицу и самого себя.

Пример, как могли бы выглядеть файлы по окончании работы программы, если вместо 1000000 использовать ограничение 20:

```
Result.txt
      2 3 5 7 11 13 17 19
Thread1.txt
      2 3 5 13 19
Thread2.txt
      7 11 17
```

4.3 Уточнения по реализации

Потоки должны работать, как написано, то есть проверять числа и записывать в файл. Запрещается использовать память для получения всех чисел, последующей сортировки и вывода в файл.

Многопоточное выполнение должно выполнять задачу значительно быстрее, чем если бы работал один поток.

При решении задачи и оценке используемых ресурсов учитывайте, что сейчас ограничение 10^6, но «заказчик» может запросить и 2^64.