middle+ | java one day offer

Правила выполенения

- форк проекта https://github.com/smlab-org/spring-day-offer-middle-plus в свой github аккаунт
- git clone к себе локально
- разработка
- git commit
- git push
- отправка ссылки на форк ведущему вас hr

Контекст

Для организации работы сотрудников розничного магазина разрабатывается система управления задачами, распределяющая задания между сотрудниками. Примером задания может быть помощь покупателям, инвентаризация, приемка товара, клиентский сервис и т.п. Задание имеет наименование, приоритет, время выполнения, статус и назначенного исполнителя.

Программист, разрабатывающий систему, только приступил к реализации, и ушёл в незапланированный отпуск. Ваша задача подхватить разработку и доделать начатые задачи.

Бизнес цель

Повысить эффективность сотрудников розничного магазина путем оптимизации их рабочего времени.

Бизнес требования

В магазине множество сотрудников постоянно находятся без заданий.

Необходимо придумать способ управления их рабочим временем.

Необходимое программное обеспечение

- Java 17
- Git 2.32 и выше
- Maven 3.6.3 и выше
- IDE (например, IntelliJ IDEA)
- Учётка на GitHub

Задача: разработать алгоритм планирования времени сотрудников.

Функциональные требования:

- Дано N заданий. Каждое задание имеет свой приоритет и оценку времени выполнения.
- Приоритет задания указан в диапазоне от 1 до 10, где 1 - самое приоритетное (TaskDTO.priority).
- Время выполнения указано в минутах (TaskDTO.leadTime).

Условия:

- Необходимо распределить задания по сотрудникам таким образом, чтобы в первую очередь были выполнены задания с приоритетом 1, затем 2, и т.д.

- Алгоритм должен предусматривать минимальную оптимизацию загрузки сотрудников, таким образом, чтобы каждый сотрудник был загружен минимум на 6 часов, даже если для этого потребуется добирать менее приоритетные, но и менее ёмкие задания. Данное условие выполняется в рамках предоставленных тестовых данных.
- Максимальная загрузка сотрудника - не более 7 часов.

Технические аспекты:

- Для решения задачи необходимо реализовать интерфейс TaskDistributor. Метод distribute принимает коллекцию сотрудников и коллекцию заданий. Результат работы метода: на сотрудников назначены задания.
- Для получения списка заданий и списка сотрудников воспользуйтесь классом utils.DataGenerator (методы getTasks и getEmployees соответственно).
- Тест DistributionTest должен пройти. Тест проверяет, что загрузка каждого сотрудника в пределах от 6 до 7 часов.
- Воспользуйтесь employee.getTotalLeadTime() для получения текущей загрузки сотрудника.