infotecs
INTERN

ТЕСТОВОЕ ЗАДАНИЕ

для стажера по направлению

Разработчик Java

Описание задачи

В идеальном мире и в идеальном городе живут счастливые люди. Эти люди делятся на два типа "Трудяги" - `Worker` и "Транжиры" - `Spender`.

Так же в этом городе есть замечательный банк - `Bank` который безвозмездно дает кредиты, но увы, так же безвозмездно принимает вклады.

Обычный день в этом городе проходит следующим образом: сначала открываются банки. Транжиры нанимают работяг и платят им деньги. Как только деньги у транжир заканчиваются, они идут в банк и берут безвозмездный кредит. Работяги получают за свою работу деньги и как только денег у них становится очень много, они идут в банк и отдают свои накопленные деньги. Банк работает не целый день, а очень часто уходит на обед.

Так же в этом городе есть СМИ - `Media`, которые установили прослушки `Wiretapping` не только в банки, но и в телефоны трудяг и транжир. Они в определенный момент времени выводят информацию о количестве денег в банках и в кошельках их клиентов - `Client`. После того, как они выведут сводку, так же уходят на обед.

Для получения различной информации о городе существует справочная доска - `HelpDesk`.

Вам необходимо смоделировать один из таких рабочих дней. Следует учесть, что исходя из конфигурации приложения, город может оказаться очень большим. Может существовать большое количество банков, трудяг и транжир.

Процесс выбора банка для получения/отдачи денег, а так же процесс найма трудяг транжирами должен быть реализован совершенно случайным образом.

Транжира может нанять только одного трудягу. После того, как транжира заплатит трудяге, он сможет нанять нового работягу. Один банк может работать только с одним клиентом, причем не важно кто он - трудяга или работяга. Если клиент выбрал банк, а он занят в этот момент, то он должен подождать, пока банк не освободится. Аналогично происходит процесс выбора трудяг транжирами. Если транжира выбрал работягу, а он в этот момнет чем то занят, необходимо подождать, пока он не освободится.

Требования

1. Необходимые конфигурационные параметры должны задаваться в `.properties` либо вводиться из консоли. В конфигурационные параметры входят следующие значения:
   - Продолжительность рабочего дня - long (милисекунды)
   - Продолжительность работы одного работяги - long (милисекунды)
   - Продолжительность обеда - long (милисекунды)
   - Количество банков - int
   - Количество работяг - int
   - Количество транжир - int
   - Начальное количество денег в банках - int
   - Начальное количество денег у трудяг и работяг - int
   - Размер заработной платы трудяг - int
   - Предельное количество денег у работяг, по достижении которого они отнесут свои деньги в банк - int
2. У банков, трудяг и транжир должно быть имя. Оно генерируется в зависимости от порядка их инициализации. Пример - `Bank - 1` или `Worker - 252`
3. Необходимо реализовать корректное завершение всех потоков.
4. Программа должна завершаться с завершением главного потока main.
5. Поток СМИ `Media` должен быть потоком демоном.
6. Создание банков, трудяг и транжир реализовать с помощью порождающего шаблона проектирования - `Abstract Factory`
7. `HelpDesk` реализовать через порождающий шаблон проектирования - `Singleton`
8. Поток денег в системе цикличен, следовательно количество денег на момент старта приложения и на момент его завершения должно совпадать.
9. Разрешается использовать только низкоуровневые способы синхронизации: механизм `wait-notify`, `synchronized`, `join`, `volatile`.
10. Реализовать механизм выбора случайного банка/трудяги одним методом с помощью механизма Generic.

Пример вывода

```
Total money amount in city on day start: 270$
Good news for everyone! Total amount money in city is: 270$
This Bank-1 has money: 110$
This Bank-2 has money: 120$
This Worker-1 has money: 0$
This Worker-2 has money: 0$
This Worker-3 has money: 0$
This Spender-1 has money: 10$
This Spender-2 has money: 10$
This Spender-3 has money: 10$
This Spender-4 has money: 10$
Good news for everyone! Total amount money in city is: 270$
This Bank-1 has money: 110$
This Bank-2 has money: 120$
This Worker-1 has money: 0$
This Worker-2 has money: 1$
This Worker-3 has money: 1$
This Spender-1 has money: 9$
This Spender-2 has money: 9$
This Spender-3 has money: 10$
This Spender-4 has money: 10$
Thread Spender-3 has been stopped.
Thread Spender-2 has been stopped.
Thread Bank-1 has been stopped.
Thread Bank-2 has been stopped.
Thread Spender-1 has been stopped.
Thread Spender-4 has been stopped.
Thread Spender-1 has been stopped.
Thread Spender-2 has been stopped.
Total money amount in city on day end: 270$
```

ТРЕБОВАНИЯ К ПРИСЫЛАЕМЫМ РЕШЕНИЯМ

Язык программирования: Java 8+.

Оформление кода должно соответствовать общепринятым нормам (например https://google.github.io/styleguide/javaguide.html).

Исходники необходимо упаковать в ZIP-архив вместе с кратким описанием решения и инструкцией по сборке/запуску.

В поставке не должно быть скомпилированных .class-файлов и привязок к среде разработки (.eclipse, .iml и т.п.).

Приветствуется наличие руководства пользователя/документации для api/javadoc для драйвера.

Всё, что явно не указано в условиях, остаётся на усмотрение автора решения.

Максимальное время на выполнение задания

1 неделя