Используя язык Java необходимо написать метод объединения имеющихся цен с вновь импортированными из внешней системы. Также необходимо написать unit тесты, для проверки хотя бы некоторых возможных вариантов.

Требования к оформлению:

1) Сборка проекта и запуск тестов при помощи maven или gradle.

2) Получившийся код должен быть размещён в системе контроля версий. Предпочтительно использование Git на публичных сервисах (GitLab, GitHub, Bitbucket).

Каждый продаваемый товар, имеет свою цену. Цен у товара может быть несколько, каждая цена имеет свой номер, принадлежность к отделу, период действия и значение в валюте.

В базе данных для каждого товара хранится история цен. **В один момент времени может действовать только одна цена из цен, с одинаковым номером и отделом.** Обычно товар продается по первой цене, вторая, третья и четвертая могут используются для применения скидок (т.е. сработало условие скидки, товар будет продан по цене номер 2). Касса может обслуживать какой-то отдел, тогда она будет использовать при продаже цены указанные для этого отдела.

Правила объединения цен:

* если товар еще не имеет цен, или имеющиеся цены не пересекаются в периодах действия с новыми, то новые цены просто добавляются к товару;
* если имеющаяся цена пересекается в периоде действия с новой ценой, то:
* если значения цен одинаковы, период действия имеющейся цены увеличивается согласно периоду новой цены;
* если значения цен отличаются, добавляется новая цена, а период действия старой цены уменьшается согласно периоду новой цены.

Метод в качестве параметров получает коллекцию имеющихся цен, новых цен и должен вернуть коллекцию объединенных цен, для дальнейшего сохранения в БД.

Пример полей класса «цена»:

long id; // идентификатор в БД

String productCode; // код товара

int number; // номер цены

int depart; // номер отдела

Date begin; // начало действия

Date end; // конец действия

long value; // значение цены в копейках

Примеры объединений:

1.

Имеющиеся цены

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Product\_code | Number | Depart | Begin | End | value |
| 122856 | 1 | 1 | 01.01.2013 00:00:00 | 31.01.2013 23:59:59 | 11000 |
| 122856 | 2 | 1 | 10.01.2013 00:00:00 | 20.01.2013 23:59:59 | 99000 |
| 6654 | 1 | 2 | 01.01.2013 00:00:00 | 31.01.2013 00:00:00 | 5000 |

Новые цены

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 122856 | 1 | 1 | 20.01.2013 00:00:00 | 20.02.2013 23:59:59 | 11000 |
| 122856 | 2 | 1 | 15.01.2013 00:00:00 | 25.01.2013 23:59:59 | 92000 |
| 6654 | 1 | 2 | 12.01.2013 00:00:00 | 13.01.2013 00:00:00 | 4000 |

Результат:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 122856 | 1 | 1 | 01.01.2013 00:00:00 | 20.02.2013 23:59:59 | 11000 |
| 122856 | 2 | 1 | 10.01.2013 00:00:00 | 15.01.2013 00:00:00 | 99000 |
| 122856 | 2 | 1 | 15.01.2013 00:00:00 | 25.01.2013 23:59:59 | 92000 |
| 6654 | 1 | 2 | 01.01.2013 00:00:00 | 12.01.2013 00:00:00 | 5000 |
| 6654 | 1 | 2 | 12.01.2013 00:00:00 | 13.01.2013 00:00:00 | 4000 |
| 6654 | 1 | 2 | 13.01.2013 00:00:00 | 31.01.2013 00:00:00 | 5000 |

Для более наглядного понимания можно представить действие цены (одного товара, с одним номером в одном отделе) в виде отрезков времени:

**Пример 1:**

Имеем цену:

|  |
| --- |
| 50 |

Пришедшая цена:

|  |
| --- |
| 60 |

Результат:

|  |  |  |
| --- | --- | --- |
| 50 | 60 | 50 |

**Пример 2:**

Имеем цены:

|  |  |
| --- | --- |
| 100 | 120 |

Пришедшая цена:

|  |
| --- |
| 110 |

Результат:

|  |  |  |
| --- | --- | --- |
| 100 | 110 | 120 |

**Пример3:**

Имеем цены:

|  |  |  |
| --- | --- | --- |
| 80 | 87 | 90 |

Пришедшие цены:

|  |  |
| --- | --- |
| 80 | 85 |

Результат:

|  |  |  |
| --- | --- | --- |
| 80 | 85 | 90 |