Задание на практическую работу №29

В процессе написания тестовых заданий ознакомьтесь с принципами создания динамических структур в Java, механизмом исключений и концепцией интерфейсов.

Замечание: в процессе выполнения задания НЕЛЬЗЯ пользоваться утилитными классами Java (за исключением java.util.HashMap).

1. Создайте класс Drink – напитка. Класс описывает сущность – напиток и характеризуется следующими свойствами - стоимостью, названием и описанием. Класс должен быть определен как неизменяемый (Immutable class). Создайте класс Dish, описывающий посуду.

2. Создайте интерфейс Item – для работы с позициями заказа. Интерфейс определяет 3 метода: возвращает стоимость, возвращает название, возвращает описание позиции. Классы Drink и Dish, которые должны реализовывать этот интерфейс.

3. Создайте класс InternetOrder, который моделирует сущность интернет-заказ в ресторане или кафе. Класс основан на циклическом двусвязном списке с выделенной головой и может хранить как блюда, так и напитки.

Конструкторы: не принимающий параметров (для списка создается только головной элемент, сам список пуст), принимающий массив позиций заказа (создаем список из N позиций). Класс InternetOrder моделирует сущность ""нтернет-заказ"" в ресторане или кафе. Класс основан на циклическом двусвязном списке с выделенной головой и может хранить как блюда, так и напитки. Внимание: список вы реализуется самостоятельно! Класс Dish сделайте неизменяемым (аналогично Drink). Класс Order должен хранить (удалять и добавлять) не только экземпляры класса Dish, но и Drink (для этого разработайте классы Order и TablesOrderManager). Методы: −добавляющий позицию в заказ (принимает ссылку типа Item). Пока этот метод возвращает истину после выполнения операции добавления элемента.

4. Переименуйте класс Order из предыдущего задания в RestaurantOrder.

Создайте интерфейс Order – позиции заказа.

Интерфейс должен определять следующие методы:

- добавления позиции в заказ (принимает ссылку типа Item), при этом возвращает логическое значение;
- удаляет позицию из заказа по его названию (принимает название блюда или напитка в качестве параметра). Возвращает логическое значение;
- удаляет все позиции с заданным именем (принимает название в качестве параметра). Возвращает число удаленных элементов;
- возвращает общее число позиций заказа в заказе;

- возвращает массив позиций заказа;
- возвращает общую стоимость заказа;
- возвращает число заказанных блюд или напитков (принимает название в качестве параметра);
- возвращает массив названий заказанных блюд и напитков (без повторов);
- возвращает массив позиций заказа, отсортированный по убыванию цены.
Замечание: Классы InternetOrder и RestaurantOrder должны реализовывать интерфейс Order.
5. Переименуйте класс TablesOrderManager в OrderManager. Добавьте ему еще одно поле типа java.util.HasMap<String, Order>, которое содержит пары адрес-заказ, и методы (работающие с этим полем).
Методы класса:
- перегрузка метода добавления заказа. В качестве параметров принимает строку – адрес и ссылку на заказ;
- перегрузка метода получения заказа. В качестве параметра принимает строку – адрес;
- перегрузка метода удаления заказа. В качестве параметра принимает строку – адрес заказа;
- перегрузка метода добавления позиции к заказу. В качестве параметра принимает адрес и Item;
- возвращающий массив имеющихся на данный момент интернет-заказов;
- возвращающий суммарную сумму имеющихся на данный момент интернет-заказов;
- возвращающий общее среди всех интернет-заказов количество заказанных порций заданного блюда по его имени. Принимает имя блюда в качестве параметра. Методы должны работать с интерфейсными ссылками Order и Item.
6. Создайте пользовательское исключение OrderAlreadyAddedException, выбрасываемое при попытке добавить заказ столику или по адресу, если со столиком или адресатом уже связан такой заказ.
Конструктор классов Drink и Dish должен выбрасывать исключение java.lang.IllegalArgumentException при попытке создать блюдо или напиток со стоимостью меньше 0, без имени или описания (если параметры имя и описание - пустые строки).
Создайте не объявляемое исключение IllegalTableNumber, выбрасываемое в методах, принимающих номер столика в качестве параметра, если столика с таким номером не существует.