Разработать консольное приложение, реализующее функционал формирования чека в магазине.

1. Приложение запускается java RunnerClassName <набор\_параметров>, где набор параметров в формате itemId-quantity (itemId - идентификатор товара, quantity - его количество.

Например: **java CheckRunner 3-1 2-5 5-1** **card-1234** должен сформировать и вывести в консоль чек содержащий в себе наименование товара с id=3 в количестве 1шт, то же самое с id=2 в количестве 5 штук, id=5 - одна штука и т. д. **Card-1234** означает, что была предъявлена скидочная карта с номером 1234. Необходимо вывести в консоль сформированный чек (вариант на рисунке), содержащий в себе список товаров и их количество с ценой, а также рассчитанную сумму с учетом скидки по предъявленной карте (если она есть).

1. Среди товаров, предусмотреть акционные. Если их в чеке больше пяти, то сделать скидку 10% по этой позиции. Данную информацию отразить в чеке.
2. Набор товаров и скидочных карт может задаваться прямо в коде, массивом или коллекцией объектов. Их количество и номенклатура имеет тестовый характер, поэтому наименование и количество свободные.
3. Готовое задание можно передать zip архивом с исходным кодом (набор .java файлов), которые можно скомпилировать при помощи **javac** и полученный .class файл запустить через **java.** Можно использовать сборщики (gradle, maven, ant и пр.). В любом случае, лучше приложить инструкцию по запуску вашего приложения. Кроме того, кодовую базу можно разместить в любом из публичных репозиториев (Bitbucket, github, gitlab).
4. В данном задание важно показать понимание ООП, умение строить модели (выделять классы, интерфейсы, их связи), разделять функционал между ними а также знать синтаксис самого языка. Обратить внимание на устойчивость к изменениям в логике и избегать копипаста. Только после этого можно перейти к необязательным заданиям отмеченных \*.
5. \* Организовать чтение исходных данных (товары и скидочные карты) из файлов (в таком случае, можно передавать имя файла в набор параметров команды **java**)/.
6. \* Реализовать вывод чека в файл.
7. \* Реализовать обработку исключений (например, товара с id или файла не существует и т. д.).
8. \* Покрыть функционал тестами
9. \* \* Реализовать web - интерфейс (например, получать чек по GET <http://localhost/check?itemId=1&itemId=1>). UI не обязателен.
10. \*\*\* Расширить функционал на свое усмотрение.
11. 