**Постановка задачи**

В супермаркете продаются различные товары (продукты, одежда, бытовая техника и т.д.).

У всех товаров есть цена и название, другие параметры (ниже будем называть их «специфические параметры») товаров зависят от их вида (например, у бытовой техники есть гарантия, а у продуктов питания вес).

Мы создаём приложение, которое позволит наполнять «корзину» любыми товарами из магазина. В первоначальный функционал корзины входит:

1. Добавление любого товара
2. Вывод на экран «чека»:
3. Название товара №1 : цена
4. Название товара №2 : цена

….

n. Название товара № n : цена

Total price : итоговая сумма

Для тестирования системы предлагается реализовать следующие виды товаров: Электроника (имеет специфические параметры – Производитель, Гарантия); Продукты питания (специфические параметры – вес, калории).

И конкретные виды товаров:

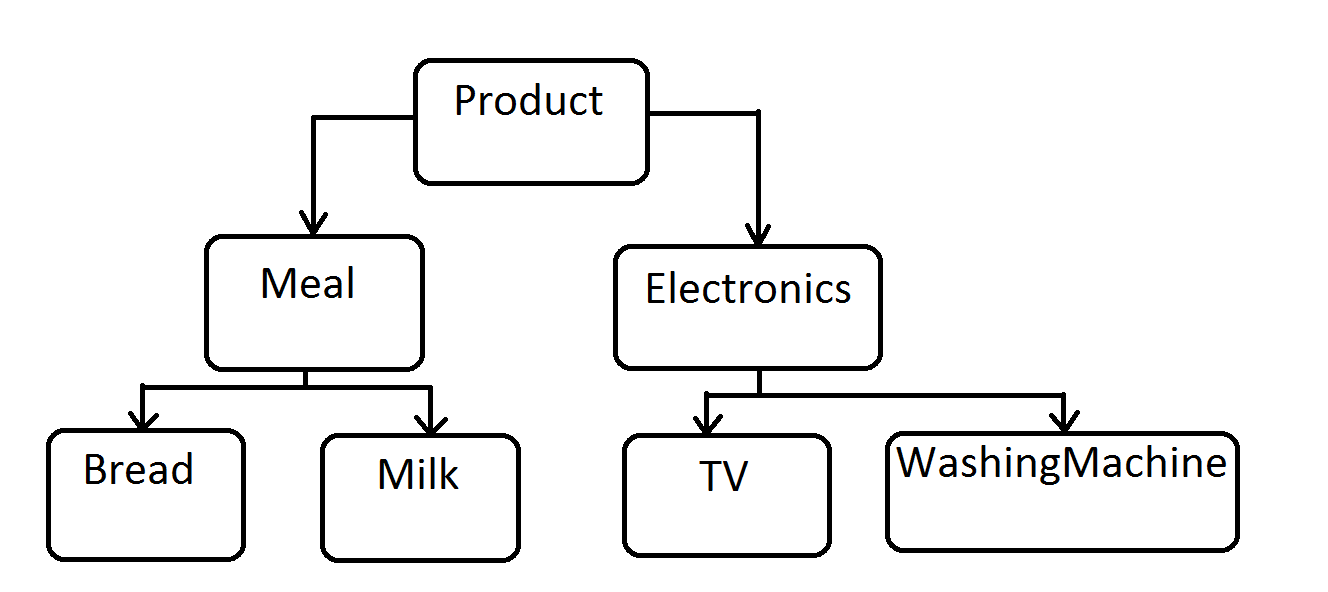
Хлеб, Молоко – являются подвидами класса Продукты питания. Хлеб имеет специфический атрибут – тип. Молоко имеет специфический атрибут – объём.

Телевизоры, стиральные машины – являются подвидами класса Электроника

Телевизор имеет специфический атрибут – диагональ экрана. Стиральная машинка имеет специфический атрибут - максимальный вес загружаемых вещей.

**Шаг 1: определяем иерархию классов**

Для товаров в магазине определим следующую иерархию наследования:



Легенда:

Product – товары

Meal - продукты питания

Bread – хлеб

Milk – молоко

Electronics – Бытовая техника

TV – телевизор

WashingMachine – стиральная машинка

Для создания такой иерархии используем механизм наследования.

* 1. **Класс Product**

Класс Product является базовым классом в иерархии и содержит 3 атрибута:

- name (название товара)

- price (цена товара)

- discount (скидка в процентах на товар)

Атрибуты заявлены со спецификаторами доступа private/protected, и для них написаны функции задания и получения значений (сеттеры и геттеры):

setName(String name), getName() – задать/получить значение атрибута name

setPrice (double price), getPrice() – задать/получить значение атрибута price. При этом при задании проверяется, что поступающее значение больше нуля. А при получении цена товара считается со скидкой.

setDiscount(double discount) – задать значение атрибута discount.

* 1. **Класс Electronics**

Класс Electronics наследуется от класса Product, а следовательно содержит все методы и атрибуты класса Product. Кроме того, в нём добавляются следующие атрибуты:

warrant – гарантия на товар

producer – производитель товара

Также для удобства объявлен конструктор класса. В качестве параметров конструктор принимает атрибуты для задания значения атрибутам класса Product, а также два атрибута для задания значений атрибутам warrant и producer.

Примечание: так как нами уже написаны методы для задания атрибутов класса Product будем использовать их. Собственно говоря, атрибуты price и name мы не можем задать иным образом, так как они заявлены как private и следовательно доступны только классу, в котором они объявлены. Однако основная причина состоит в том, что методы уже реализованы (и мы подразумеваем, что они валидны).

* + 1. **Класс TV**

Класс TV наследуется от класса Electronics (который в свою очередь наследуется от Product), а следовательно содержит все методы и атрибуты классов Product и Electronics. Кроме того, в нём добавляются следующие атрибуты:

Screen – размер диагонали экрана

Так как в классе Electronics уже реализован конструктор, задающий значения все атрибутам класса, то в конструкторе для класса TV вызовем, конструктор класса Electronics (при помощи синтаксиса: super(<parameters>) и зададим значение атрибута screen).

* + 1. **Класс WashingMachine**

Класс WashingMachine наследуется от класса Electronics (который в свою очередь наследуется от Product), а следовательно содержит все методы и атрибуты классов Product и Electronics. Кроме того, в нём добавляются следующие атрибуты:

volume – размер диагонали экрана

Так как в классе Electronics уже реализован конструктор, задающий значения все атрибутам класса, то в конструкторе для класса WashingMachine вызовем, конструктор класса Electronics (при помощи синтаксиса: super(<parameters>) и зададим значение атрибута volume).

* 1. **Класс Meal**

Класс Meal наследуется от класса Product, а следовательно содержит

все методы и атрибуты класса Product. Кроме того, в нём добавляются следующие атрибуты:

calories – количество калорий

weight – вес

Для удобства объявлен конструктор класса. В качестве параметров конструктор принимает атрибуты для задания значения атрибутам класса Product, а также два атрибута для задания значений атрибутам calories и weight.

Примечание: так как нами уже написаны методы для задания атрибутов класса Product будем использовать их. Собственно говоря, атрибуты price и name мы не можем задать иным образом, так как они заявлены как private и следовательно доступны только классу, в котором они объявлены. Однако основная причина состоит в том, что методы уже реализованы (и мы подразумеваем, что они валидны).

* + 1. **Класс Bread**

Класс Bread наследуется от класса Meal (который в свою очередь наследуется от Product), а следовательно содержит все методы и атрибуты классов Product и Meal. Кроме того, в нём добавляются следующие атрибуты:

type – сорт хлеба

Так как в классе Meal уже реализован конструктор, задающий значения все атрибутам класса, то в конструкторе для класса Bread вызовем, конструктор класса Meal (при помощи синтаксиса: super(<parameters>) и зададим значение атрибута type).

* + 1. **Класс Milk**

Класс Milk наследуется от класса Meal (который в свою очередь наследуется от Product), а следовательно содержит все методы и атрибуты классов Product и Milk. Кроме того, в нём добавляются следующие атрибуты:

volume - объём

Так как в классе Meal уже реализован конструктор, задающий значения все атрибутам класса, то в конструкторе для класса Milk вызовем, конструктор класса Milk (при помощи синтаксиса: super(<parameters>) и зададим значение атрибута type).

**Шаг 2: создаём корзину**

Корзина (класс Bin) – по сути, массив с добавленными туда товарами. Для реализации такой структуры используем класс ArrayList из пакета java.util.

Атрибут класса Bin myBin – контейнер товаров, реализованный при помощи класса ArrayList:

*ArrayList<Product> myBin=new ArrayList<Product>();*

Ссылка myBin инициализируется пустым объектом класса ArrayList .

Класс myBin содержит методы:

add(Product p) – добавление в контейнер товара p

printCheck() – вывод на экран чека

**Шаг 3: Точка входа**

Тестирование проведём при помощи точки входа (класс Buyer)

1. Создадим 3 объекта;
2. Создадим корзину;
3. Добавим объекты в корзину;
4. Выведем чек.