Задача 1 (25 точки)

Да се направи чисто абстрактен клас `**Consumable**`, който представлява хранителен продукт и има единствено индикатор за това дали е отворена опаковката на самия продукт. Класът да има методи *readLabel()*, *consume(количество)* и quantity*()*.
Да се реализират класове `**Eatable**` и `**Drinkable**`, който наследяват Consumable, като добавят съответно свойствата(член-данни) грамаж(в грамове) и брой порции(парчета) и обем(в литри). Eatable приема грамажа на единично парче в опаковката и съответният брой(в конструктора) и се яде (чрез метода consume) само парче по парче. Drinkable приема обем на съдържанието и се пие в произволни количества, не надвишаващи обема на останалата течност.

Да се реализират следните класове, наследяващи Eatable:

- `**Chocolate**`: добавящ свойства описващи съдържание на какао и приемащ в конструктора брой редове и колони на блокчето.
- `**Croissant**`: добавящ свойство описващо пълнеж(пр. солен) и състоящ се от едно единствено парче.
- `**Cereal**`: добавящ свойства описващи марка и съдържание на ядки(**bool**).

Да се реализират следните класове, наследяващи Drinkable:

- `**Beer**`: добавящ свойства описващи съдържание на алкохол(%) и марка.
- `**MineralWater**`: добавящ свойства описващи pH(1-14), минерализация(висока или ниска) и отново марка?.
- `**CannedCoffee**`: добавящ свойство описващо тип(точно едно измежду black, latte, espresso, cappuccino, mochaccino, macchiato). Опаковката тежи с 10% повече, заради металното кенче.

За всеки клас методът quantity да връща оставащото в опаковката количество, а методът *readLabel* да извежда на стандартния изход всички свойства на обекта в удобен за четене формат.

Задача 2 (30 точки)

Да се напише шаблонен клас `**VendingMachine**` представляващ автомат за продажба на хранителни продукти. Автоматът да разполага с определен брой слотове за продукти, всеки с определен максимален капацитет (в брой продукти), като съответните бройки се задават при конструиране. Самите слотове да са запазени в масив с точно необходимата големина и за всеки да се пази текущият брой на продуктите в него. Всеки слот да е представен чрез свързана динамична структура реализираща функционалността на опашка - продукти се добавят само отзад на слота и се взимат само отпред. За класа да са реализирани необходимите методи и оператори, така че да е налична следната функционалност (**vm** е инстанция от класа):

- **vm[ind]** - връща (чрез указател) продукта най-отпред на слота с индекс **ind**.
- **vm(ind, product)** - добавя копие на **product** най-отзад на слота с индекс **ind**.
- **vm.clear(ind)** - унищожава всички продукт на слота с индекс **ind** (изпразва слота).
- **vm.empty(ind)** - показва дали слота с индекс **ind** е празен.
- **vm.numProd(ind)** - връща броя продукти на слота с индекс **ind**.

За всички класове да се осигури правилното функциониране на методите от "голямата четворка".