**Лабораторная работа №11**

**Задача Регистрация**

**Lab 11.1** Создайте класс Registration, который пока будет проверять только введенный логин. Под логином мы будем подразумевать почту пользователя, поэтому необходимо будет сделать некоторые проверки.

В классе Registration необходимо реализовать:

1. метод __init__ принимающий один аргумент логин пользователя. Метод __init__ должен сохранить переданный логин через сеттер (см пункт 3). То есть когда отработает данный код
   ```
   def __init__(self, логин):
       self.login = логин # передаем в сеттер login значение логин
   ```
   должно сработать свойство сеттер login из пункта 3 для проверки валидности переданного значения
2. Свойство *геттер login*, которое возвращает значение self.__login;
3. Свойство *сеттер login*, принимает значение нового логина. Новое значение мы должны проверить на следующее:
   1. логин, так как является почтой, должен содержать один символ собаки «@». В случае, если в логине отсутствует символ «@», вызываем исключение при помощи строки raise ValueError("Логин должен содержать один символ '@'")
   2. логин должен содержать символ точки «.» после символа «@».В случае, если после @ нету точки, вызываем исключение при помощи строки raise ValueError("Логин должен содержать символ '.'")

**Задача «Оформление заказа»**

**Lab 11.2** Создайте класс Product. Это класс, описывающий товар. В нем должно быть реализовано:

1. метод __init__, принимающий на вход имя товара и его стоимость. Эти значения необходимо сохранить в атрибутах name и price

Далее для оформления заказа нам нужен пользователь. Для этого создайте класс User, который содержит:

1. метод __init__, принимающий на вход логин пользователя и необязательный аргумент баланс его счета (по умолчанию 0). Логин необходимо сохранить в атрибуте login, а баланс необходимо присвоить сеттеру balance (см. пункт 4)
2. метод __str__, возвращающий строку вида «Пользователь {login}, баланс - {balance}»
3. Свойство *геттер* balance, которое возвращает значение self.__balance;
4. Свойство *сеттер* balance, принимает новое значение баланса и устанавливает его в атрибут self.__balance ;
5. метод deposit принимает числовое значение и прибавляет его к атрибуту self.__balance ;
6. метод payment принимает числовое значение, которое должно списаться с баланса пользователя. Если на счете у пользователя не хватает средств, то необходимо вывести фразу «Не хватает средств на балансе. Пополните счет» и вернуть False. Если средств хватает, списываем с баланса у пользователя указанную сумму и возвращаем True