**Подвиг 6 (про модуль abc).** В языке Python есть еще один распространенный способ объявления абстрактных методов класса через декоратор abstractmethod модуля abc:

from abc import ABC, abstractmethod

Чтобы корректно работал декоратор abstractmethod сам класс должен наследоваться от базового класса ABC. Например, так:

class Transport(ABC):

@abstractmethod

def go(self):

"""Метод для перемещения транспортного средства"""

@classmethod

@abstractmethod

def abstract\_class\_method(cls):

"""Абстрактный метод класса"""

Мы здесь имеем два абстрактных метода внутри класса Transport, причем, первый метод go() - это обычный метод, а второй abstract\_class\_method() - это абстрактный метод уровня класса. Обратите внимание на порядок использования декораторов classmethod и abstractmethod. Они должны быть записаны именно в такой последовательности.

Теперь, если объявить какой-либо дочерний класс, например:

class Bus(Transport):

def \_\_init\_\_(self, model, speed):

self.\_model = model

self.\_speed = speed

def go(self):

print("bus go")

@classmethod

def abstract\_class\_method(cls):

pass

То в нем обязательно нужно переопределить абстрактные методы go и abstract\_class\_method класса Transport. Иначе, объект класса Bus не будет создан (возникнет исключение TypeError).

Используя эту информацию, объявите базовый класс **Model** (модель), в котором нужно объявить один абстрактный метод с сигнатурой:

def get\_pk(self): ...

и один обычный метод:

def get\_info(self): ...

который бы возвращал строку "Базовый класс Model".

На основе класса Model объявите дочерний класс **ModelForm**, объекты которого создаются командой:

form = ModelForm(login, password)

где login - заголовок перед полем ввода логина (строка); password - заголовок перед полем ввода пароля (строка). В каждом объекте класса ModelForm должны формироваться локальные атрибуты с именами \_login и \_password, а также автоматически появляться локальный атрибут \_id с уникальным целочисленным значением для каждого объекта класса ModelForm.

В классе ModelForm переопределите метод:

def get\_pk(self): ...

который должен возвращать значение атрибута \_id.

Пример использования классов (эти строчки в программе писать не нужно):

form = ModelForm("Логин", "Пароль")

print(form.get\_pk())

P.S. В программе требуется объявить только классы. На экран выводить ничего не нужно.

from abc import ABC, abstractmethod

# здесь объявляйте классы