**Подвиг 5.** В программе объявлены два класса:

class ShopItem:

ID\_SHOP\_ITEM = 0

def \_\_init\_\_(self):

super().\_\_init\_\_()

ShopItem.ID\_SHOP\_ITEM += 1

self.\_id = ShopItem.ID\_SHOP\_ITEM

def get\_pk(self):

return self.\_id

class Book(ShopItem):

def \_\_init\_\_(self, title, author, year):

super().\_\_init\_\_()

self.\_title = title

self.\_author = author

self.\_year = year

Затем, создается объект класса Book (книга) и отображается в консоль:

book = Book("Python ООП", "Балакирев", 2022)

print(book)

В результате, на экране увидим что то вроде:

<\_\_main\_\_.Book object at 0x0000015FBA4B3D00>

Но нам требуется, чтобы здесь отображались локальные атрибуты объекта с их значениями в формате:

<атрибут\_1>: <значение\_1>  
<атрибут\_2>: <значение\_2>  
...  
<атрибут\_N>: <значение\_N>

Для этого вам дают задание разработать два класса:

**ShopGenericView** - для отображения всех локальных атрибутов объектов любых дочерних классов (не только Book);  
**ShopUserView** - для отображения всех локальных атрибутов, кроме атрибута \_id, объектов любых дочерних классов (не только Book).

То есть, в этих классах нужно переопределить два магических метода: \_\_str\_\_() и \_\_repr\_\_().

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

class Book(ShopItem, ShopGenericView): ...

book = Book("Python ООП", "Балакирев", 2022)

print(book)

# на экране увидим строчки:

# \_id: 1

# \_title: Python ООП

# \_author: Балакирев

# \_year: 2022

Другой вариант использования классов:

class Book(ShopItem, ShopUserView): ...

book = Book("Python ООП", "Балакирев", 2022)

print(book)

# на экране увидим строчки:

# \_title: Python ООП

# \_author: Балакирев

# \_year: 2022

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

class ShopItem:

ID\_SHOP\_ITEM = 0

def \_\_init\_\_(self):

super().\_\_init\_\_()

ShopItem.ID\_SHOP\_ITEM += 1

self.\_id = ShopItem.ID\_SHOP\_ITEM

def get\_pk(self):

return self.\_id

# здесь объявляйте классы ShopGenericView и ShopUserView

class Book(ShopItem):

def \_\_init\_\_(self, title, author, year):

super().\_\_init\_\_()

self.\_title = title

self.\_author = author

self.\_year = year