**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Информационные технологии»**

**Тема: Парадигмы программирования**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Пивоев Н. М. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2024

## Цель работы

Ознакомиться с классами в Python и, применив полученные знания на практике, написать программу с их реализацией.

## Задание

**Базовый класс — печатное издание Edition:**

class Edition:

Поля объекта класса Edition:

* название (строка)
* цена (в руб., целое положительное число)
* возрастное ограничение (в годах, целое положительное число)
* стиль(значение может быть одной из строк: c (color), b (black))
* При создании экземпляра класса Edition необходимо убедиться, что переданные в конструктор параметры удовлетворяют требованиям, иначе выбросить исключение ValueError с текстом 'Invalid value'.

##### Книга - Book:

class Book: #Наследуется от класса Edition

Поля объекта класс Book:

* название (строка)
* цена (в руб., целое положительное число)
* возрастное ограничение (в годах, целое положительное число)
* стиль(значение может быть одной из строк: c (color), b (black))
* автор (фамилия, в виде строки)
* твердый переплет (значениями могут быть или True, или False)
* количество страниц (целое положительное число)
* При создании экземпляра класса Book необходимо убедиться, что переданные в конструктор параметры удовлетворяют требованиям, иначе выбросить исключение ValueError с текстом 'Invalid value'.

*В данном классе необходимо реализовать следующие методы:*

Метод \_\_str\_\_():

Преобразование к строке вида: Book: название <название>, цена <цена>, возрастное ограничение <возрастное ограничение>, стиль <стиль>, автор <автор>, твердый переплет <твердый переплет>, количество страниц <количество страниц>.

Метод \_\_eq\_\_():

Метод возвращает True, если два объекта класса равны и False иначе. Два объекта типа Book равны, если равны их название и автор.

##### Газета - Newspaper:

class Newspaper: #Наследуется от класса Edition

Поля объекта класс Newspaper:

* название (строка)
* цена (в руб., целое положительное число)
* возрастное ограничение (в годах, целое положительное число)
* стиль(значение может быть одной из строк: c (color), b (black))
* интернет издание (значениями могут быть или True, или False)
* страна (строка)
* периодичность (период выпуска газеты в днях, целое положительное число)
* При создании экземпляра класса Newspaper необходимо убедиться, что переданные в конструктор параметры удовлетворяют требованиям, иначе выбросить исключение ValueError с текстом 'Invalid value'.

*В данном классе необходимо реализовать следующие методы:*

Метод \_\_str\_\_():

Преобразование к строке вида: Newspaper: название <название>, цена <цена>, возрастное ограничение <возрастное ограничение>, стиль <стиль>, интернет издание <интернет издание>, страна <страна>, периодичность <периодичность>.

Метод \_\_eq\_\_():

Метод возвращает True, если два объекта класса равны и False иначе. Два объекта типа Newspaper равны, если равны их название и страна.

##### **Необходимо определить список list для работы с печатным изданием:** Книги:

class BookList – список книг - наследуется от класса list.

Конструктор:

1. Вызвать конструктор базового класса.
2. Передать в конструктор строку name и присвоить её полю name созданного объекта

*Необходимо реализовать следующие методы:*

Метод append(p\_object): Переопределение метода append() списка. В случае, если p\_object - книга, элемент добавляется в список, иначе выбрасывается исключение TypeError с текстом: Invalid type <тип\_объекта p\_object> (результат вызова функции type)

Метод total\_pages(): Метод возвращает сумму всех страниц всех имеющихся книг.

Метод print\_count(): Вывести количество книг.

##### Газеты:

class NewspaperList – список газет - наследуется от класса list.

Конструктор:

1. Вызвать конструктор базового класса.
2. Передать в конструктор строку name и присвоить её полю name созданного объекта

*Необходимо реализовать следующие методы:*

Метод extend(iterable): Переопределение метода extend() списка. В случае, если элемент iterable - объект класса Newspaper, этот элемент добавляется в список, иначе не добавляется.

Метод print\_age(): Вывести самое низкое возрастное ограничение среди всех газет.

Метод print\_total\_price(): Посчитать и вывести общую цену всех газет.

## Выполнение работы

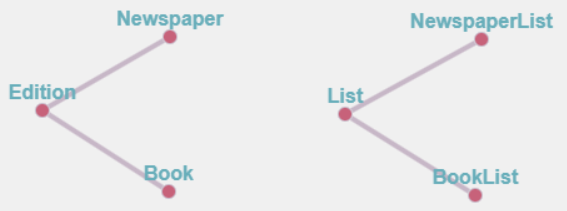


Рисунок 1. Иерархия классов

Методы созданных классов:

1) Методы Edition

*\_\_init\_\_* - конструктор класса, принимающий следующие аргументы *name*, *price*, *age\_limit* и *style*, если их тип совпадает с определённым в задании.

2) Методы Book и Newspaper

­­*\_\_init\_\_* - конструктор класса, наследующий поля класса Edition и добавляющий новые поля *author*, *hardcover* и *pages* в случае класса Book, и методы *online\_edition*, *country* и *frequency*, в случае.

*\_\_str\_\_* - метод класса, выводящий информацию о созданном объекте.

*\_\_eq\_\_* - метод класса, проводящий сравнение двух объектов по имени и автору для класса Book, по имени и стране для класса Newspaper.

3) Методы BookList

*\_\_init\_\_* - конструктор класса, заполняющий поле *name*.

*append* – метод класса, добавляющий объект класса Book в конец списка. При несовпадении типа объекта вызывает ошибку.

*total\_pages* – метод класса, возвращающий число страниц во всех книгах списка.

*print\_count* – метод класса, возвращающий длину списка.

4) Методы NewspaperList

*\_\_init\_\_* - конструктор класса, заполняющий поле *name*.

*extend* – метод класса, добавляющий несколько элементов в список, в случае совпадения типа объекта с классом Newspaper.

*print\_age* – метод класса, выводящий минимальный возраст среди объектов в списке.

*print\_total\_price* – метод класса, выводящий суммарную цену всех объектов списка.

Вопросы:

1) В каких случаях будут использованы методы *\_\_str\_\_()* и *\_\_eq\_\_()*.

Метод *\_\_str\_\_* можно использовать, чтобы получить полную информацию об объекте класса. Метод *\_\_eq\_\_* можно использовать для сравнения, например, во время сортировки.

2) Будет ли работать переопределенные методы класса list для BookList и NewspaperList? Объясните почему и приведите примеры.

Методы будут работать, потому что в них используются функция *super()*, вызывающая методы класса-родителя.

В качестве примера можно рассмотреть метод *extend*, в котором через функцию *super()* осуществляется работа метода *extend*, как у класса-родителя *list*.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Входные данные | Выходные данные | Комментарий |
|  | a = Book("Book", 10, 16, 'c', "Writer", False, 30)  b = Newspaper("Newspaper", 10, 16, 'c', True, "Russia", 10)  print(a.\_\_str\_\_())  print(b.\_\_str\_\_()) | Book: название Book, цена 10, возрастное ограничение 16, стиль c, автор Writer, твердый переплет False, количество страниц 30.  Newspaper: название Newspaper, цена 10, возрастное ограничение 16, стиль c, интернет издание True, страна Russia, периодичность 10. | Вывод соответствует ожиданию. |

## Выводы

В ходе выполнения лабораторной работы была написана программа с использованием классов, изучены особенности их реализации на языке Python.

# Приложение А Исходный код программы

Название файла: main.py

class Edition:

def \_\_init\_\_(self, name, price, age\_limit, style):

if type(name) != str or type(price) != int or type(age\_limit) != int or price <= 0 or age\_limit <= 0 or (style != 'c' and style != 'b'):

raise ValueError('Invalid value')

self.name = name

self.price = price

self.age\_limit = age\_limit

self.style = style

class Book(Edition):

def \_\_init\_\_(self, name, price, age\_limit, style, author, hardcover, pages):

if type(author) != str or type(hardcover) != bool or type(pages) != int or pages <= 0:

raise ValueError('Invalid value')

super().\_\_init\_\_(name, price, age\_limit, style)

self.author = author

self.hardcover = hardcover

self.pages = pages

def \_\_str\_\_(self):

return f'Book: название {self.name}, цена {self.price}, возрастное ограничение {self.age\_limit}, стиль {self.style}, автор {self.author}, твердый переплет {self.hardcover}, количество страниц {self.pages}.'

def \_\_eq\_\_(self, other):

return (self.name == other.name) and (self.author == other.author)

class Newspaper(Edition):

def \_\_init\_\_(self, name, price, age\_limit, style, online\_edition, country, frequency):

if type(online\_edition) != bool or type(country) != str or type(frequency) != int or frequency <= 0:

raise ValueError('Invalid value')

super().\_\_init\_\_(name, price, age\_limit, style)

self.online\_edition = online\_edition

self.country = country

self.frequency = frequency

def \_\_str\_\_(self):

return f'Newspaper: название {self.name}, цена {self.price}, возрастное ограничение {self.age\_limit}, стиль {self.style}, интернет издание {self.online\_edition}, страна {self.country}, периодичность {self.frequency}.'

def \_\_eq\_\_(self, other):

return (self.name == other.name) and (self.country == other.country)

class BookList(list):

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

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

self.name = name

def append(self, p\_object):

if isinstance(p\_object,Book):

super().append(p\_object)

else:

raise TypeError('Invalid type <тип\_объекта p\_object> (результат вызова функции type)')

def total\_pages(self):

return sum([book.pages for book in list(self)])

def print\_count(self):

print(len(list(self)))

class NewspaperList(list):

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

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

self.name = name

def extend(self, iterable):

for i in iterable:

if isinstance(i, Newspaper):

super().append(i)

def print\_age(self):

print(min([newspaper.age\_limit for newspaper in list(self)]))

def print\_total\_price(self):

print(sum([newspaper.price for newspaper in list(self)]))