**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Информатика»**

**Тема: Парадигмы программирования**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3344 |  | Сербиновский Ю.М. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2024

## Цель работы

Изучение основ парадигм ООП в языке программирования Python.

## 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:

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:

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: Вызвать конструктор базового класса. 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: Вызвать конструктор базового класса. 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

1. Иерархия классов представлена на рисунке 1.

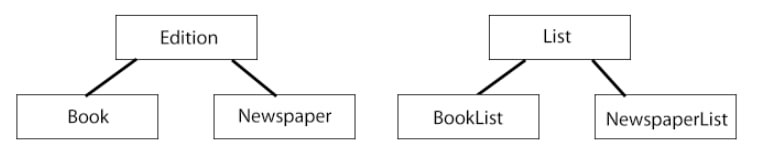


Рисунок 1 – Иерархия классов

1. Переопределенные методы:

*\_\_init\_\_()* – Метод, который переопределен для всех классов. Инициализирует поля класса.

*\_\_str\_\_()* – Метод, который используется для строчного представления информации о полях объекта. Реализован в Book и Newspaper.

*\_\_eq\_\_()* – Метод для сравнения двух объектвов класса. Реализован в Book и Newspaper.

*append()*– Метод, который переопределен у класса *BookList*. Если объект класса Book, то элемент добавляется в список.

*extend()* – Метод, который переопределен у класса Newspaper*List*. Метод перебирает iterable объект на наличие объектов типа Newspaper и добавляет их в NewspaperList.

1. Переопределенные методы класса *list* для BookList и *NewspaperList* будут работать, так как наследуются от класса *list*. То есть BookList и NewspaperList имеют доступ ко всем методам list, также могут менять их логику для работы со своими объектами, но не родительскими.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Тест | Выходные данные | Комментарии |
|  | a = Newspaper('a', 11, 8, 'c', True, 'aslfdk', 1) b = Newspaper('a', 10, 10, 'b', True, 'aslfdk', 1) print(a.\_\_eq\_\_(b)) print(a.\_\_str\_\_())  c = Book('a', 12, 18, 'c', 'sdf', True, 1)  d = Book('a', 12, 18, 'c', 'sdf', True, 100)  l1 = BookList('man') l2 = BookList('meow') l1.append(c) l1.append(d) print(l1.total\_pages()) l1.print\_count() print(c.\_\_eq\_\_(d)) n1 = NewspaperList('sleeper') n2 = NewspaperList('123') n1.append(a) n1.append(b) n2.append(a) n2.append('2') n1.extend(n2) n1.print\_total\_price() n1.print\_age() print(n1) m = 0 l = NewspaperList('name') l.extend([a, b, 1]) print(l) | True  Newspaper: название a, цена 11, возрастное ограничение 8, стиль c, интернет издание True, страна aslfdk, периодичность 1.  101  2  True  32  8  [<\_\_main\_\_.Newspaper object at 0x7e99a839a290>, <\_\_main\_\_.Newspaper object at 0x7e99a839a260>, <\_\_main\_\_.Newspaper object at 0x7e99a839a290>]  [<\_\_main\_\_.Newspaper object at 0x7e99a839a290>, <\_\_main\_\_.Newspaper object at 0x7e99a839a260>] | Данные обработаны корректно |
|  | try:  a = Newspaper('a', 11, 8, 'w', True, 'aslfdk', 1) except (ValueError):  print('incorrect value')  try:  BookList('man').append(1) except(TypeError):  print('incorrect object') | incorrect value  incorrect object | Данные обработаны корректно |

## Выводы

Были получены базовые навыки работы с ООП на языке программирования Python. Была написана программа, в которой реализованы ключевые возможности классов.

# Приложение А Исходный код программы

Название файла: main.py

class Edition:

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

if not (isinstance(name, str) and isinstance(price, int) and price > 0 and

isinstance(age\_limit, int) and age\_limit > 0 and (style == 'c' or 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):

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

if not (isinstance(author, str) and isinstance(hardcover, bool) and isinstance(pages, int) and pages > 0):

raise ValueError('Invalid value')

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 isinstance(other, Book) and self.name == other.name and self.author == other.author

class Newspaper(Edition):

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

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

if not (isinstance(online\_edition, bool) and isinstance(country, str) and isinstance(frequency, int) and frequency > 0):

raise ValueError('Invalid value')

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 isinstance(other, Newspaper) and 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(f"Invalid type <тип\_объекта {type(object)}>")

def total\_pages(self):

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

def print\_count(self):

print(len(self))

class NewspaperList(list):

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

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

self.name = name

def extend(self, iterable):

super().extend([news for news in iterable if isinstance(news, Newspaper)])

def print\_age(self):

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

def print\_total\_price(self):

print(sum([news.price for news in self]))