**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. 3343 |  | Жучков О.Д. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2024

## Цель работы

Целью работы является изучение основ объектно-ориентированного программирования. На языке Python необходимо изучить работу с классами, создание методов для классов, наследование, переопределение методов.

**Задание**

Вариант 4

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:

* 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

Класс Edition содержит такие поля, как название, цена, возрастное ограничение и стиль печатного издания. От него наследуются классы Book и Newspaper. Класс Book имеет поля: автор, количество страниц, жесткость переплета. Класс Newspaper имеет поля: интернет издание газеты, страна, периодичность выпуска. При создании объектов данных классов происходит проверка переданных в конструктор аргументов, если значение для некоторого поля не соответствует требованиям, то выводится исключение ValueError. Класс Book имеет методы для вывода информации об объекте в виде строки и для сравнения двух книг по автору и названию. Класс Newspaper имеет методы для вывода информации об объекте, сравнения двух газет по названию и стране.

Методы \_\_str\_\_ и \_\_eq\_\_, определенные в классах Book и Newspaper, являются специальными методами. Метод \_\_str\_\_ возвращает строковое представление объекта и вызывается при использовании таких функций, как print() или str(). Метод \_\_eq\_\_ используется для определения равенства двух объектов, при вызове выражения a == b для объекта a вызывается метод \_\_eq\_\_ с объектом b в качестве аргумента.

Класс BookList наследуется от list. Переопределён метод append(), в список можно добавить только книгу. Включены методы для счета количества страниц во всех книгах списка и для вывода количества книг в списке. Класс NewspaperList также наследуется от list. Переопределён метод extend(): если элемент итерируемого объекта – газета, то он добавляется в список. Есть методы для вывода наименьшего возрастного ограничения из газет списка, для вывода общей цены всех газет.

В методах append и extend, переопределённых в классах BookList и NewspaperList, вызываются соответствующие методы класса list с помощью функции super(), что обеспечивает их корректную работу.

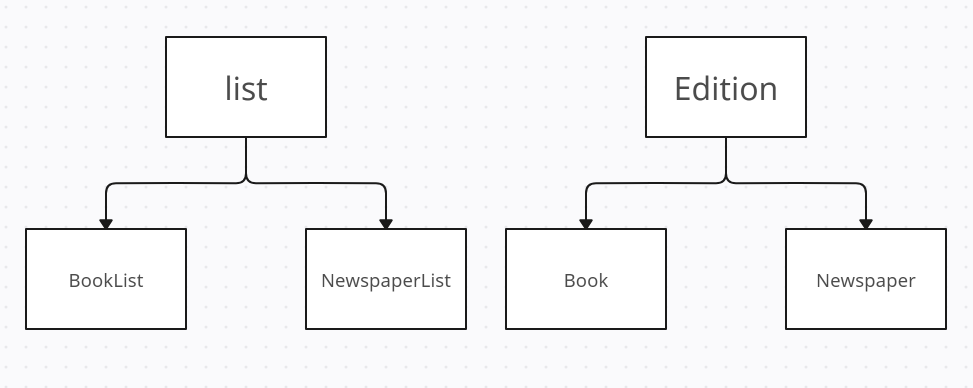


Рисунок 1 – Иерархия созданных классов

## Выводы

В ходе выполнения работы была изучены основы объектно-ориентированного программирования на языке Python. Изучены создание классов, создание функций и методов для классов, работа с классами, наследование классов, переопределение методов.

# Приложение А Исходный код программы

Название файла: main.py

class Edition:

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

if not isinstance(name, str):

raise ValueError("Invalid value")

if not isinstance(price, int):

raise ValueError("Invalid value")

if price <= 0:

raise ValueError("Invalid value")

if not isinstance(age\_limit,int):

raise ValueError("Invalid value")

if age\_limit <= 0:

raise ValueError("Invalid value")

if not isinstance(style,str):

raise ValueError("Invalid value")

if style not in ["b", "c"]:

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):

raise ValueError("Invalid value")

if not isinstance(hardcover, bool):

raise ValueError("Invalid value")

if not isinstance(pages, int):

raise ValueError("Invalid value")

if 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 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):

raise ValueError("Invalid value")

if not isinstance(country, str):

raise ValueError("Invalid value")

if not isinstance(frequency, int):

raise ValueError("Invalid value")

if 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 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 {type(p\_object)}")

def total\_pages(self):

cnt = 0

for book in self:

cnt += book.pages

return cnt

def print\_count(self):

print(len(self))

class NewspaperList(list):

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

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

self.name = name

def extend(self, iterable):

for paper in iterable:

if isinstance(paper, Newspaper):

super().append(paper)

def print\_age(self):

age\_limits = [paper.age\_limit for paper in self]

min\_age = min(age\_limits)

print(min\_age)

def print\_total\_price(self):

cost = 0

for paper in self:

cost += paper.price

print(cost)