

**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

Волохов М.

Преподаватель

Иванов Д.В.

Санкт-Петербург

2024

## **Цель работы**

Получить представление о работе ООП в языке 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: #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



Рисунок 1 - Иерархия классов

### 1. Методы унаследованные у Edition:

`__init__()` – Принимает параметры и проверяет их на корректность

`__str__()` – Преобразует данные в строку и возвращает её

`__eq__()` – Сравнивает два объекта данного типа, проверяет на одинаковость

### 2. Методы унаследованные у list:

`__init__()` – Принимает параметры и проверяет их на корректность

`append()` – Добавляет объект в список

`total_pages()` – Возвращает сумму страниц из всех книг

`print_count()` – Выводит количество книг

`extend()` – Расширяет список элементами класса Newspaper

`print_age()` – Выводит минимальное возрастное ограничение среди всех газет

`print_total_price()` – Выводит сумму стоимости всех газет

### 3. Метод `__str__()` будет применяться при вызове `str(<Edition>)`

Метод `__eq__()` будет применяться при сравнении изданий одного класса

### 4. Переопределённые методы будут рабочими, так как функции методов не изменились, только добавляется проверка на входные данные.

## **Выводы**

Был получен навык работы с ООП. Изучены особенности реализации данной методологии программирования в языке Python.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Volokhov\_Mikhail\_lb1.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 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):
        if isinstance(other, Newspaper):
            return self.name == other.name and self.country ==
other.country
        return False

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(p_object)}')

    def total_pages(self):
        return sum(x.pages for x 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(x for x in __iterable if isinstance(x, Newspaper))

    def print_age(self):
        print(min(x.age_limit for x in self))

    def print_total_price(self):
        print(sum(x.price for x in self))

```