МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Информационные технологии»

Тема: Парадигмы программирования

Студентка гр. 3388 Басик В. В.

Преподаватель Иванов Д. В.

Санкт-Петербург

2024

Цель работы

Изучить основные парадигмы программирования. Реализовать на практике программу с использованием объектно-ориентированого программирования.

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. В отчете укажите:

Изображение иерархии описанных вами классов.

Методы, которые вы переопределили (в том числе методы класса

object).

В каких случаях будут использованы методы __str__() и __eq__().

Будут ли работать переопределенные методы класса list для BookList и.

Выполнение работы

Методы разработанных классов

1) Методы классов-наследников Edition

__init__ – конструктор класса, принимающий следующие параметры: name, price, age_limit, style – соотвествующие имени, цене, возрастному ограничению и цветовому стилю издания; author, hardcover, pages – дополнительные поля класса Book: автор, тип переплёта, количество страниц; Newspaper же принимает дополнительные параметры online_edition, country, frequency – тип издания, страна и частота выпусков газеты. При инициализации идёт проверка всех типов данных на соответствие требованиям задания, в случае несовпадения поднимается исключение Value Error.

__str__ – метод, возвращающий строку данных о книге или газете, в соответствии с требованиями текста задания

__eq__ – метод, переопределяющий оператор сравнения между объектами типов book и newspaper, на основе сравнения значений заданных полей (т. е. например книги считаются равными, в случае равенства имени автора и названия)

2) Методы классов-наследников List

__init__ – конструктор класса, принимающий параметр name, с инициализацией соотвествующего поля и последующим вызовом родительского конструктора

append(p_object) – метод, реаизующий добавление одного элемента в Booklist, с предварительной проверкой типа на соответствие типу Book с поднятием исключения TypeError в случае несоответствия.

extend(iterable) – метод, аналогичный списковому методу extend метод класса NewspaperList, с игнорированием иных типов кроме Newspaper.

total_pages() – метод класса BookList возвращающий суммы числа всех страниц входыщих в список

print_count() – метод класса BookList, выводящий в консоль количество элеметов списка

print_age() – метод класса NewspaperList, выводящий минимальное возрастное ограничение, содержащееся в списке

print_total_price() – метод класса NewspaperList, возвращающий суммарную цену всех газет списка

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

Да, будут работать, потому что они используют функцию super() для вызова соотвествующих методов родительского класса, поэтому дополнительные проверки не мешают основной работе метода.

Примером может послужить метод append у BookList – в коде программы реализован лишь один дополнительный if для установления соответсвия типу и вызову ошибки в случае несовпадния, далее тут же следует функция super(), через которую фактически осуществляется работа метода append как у list.
Разработанный программный код см. в приложении А.

Выводы

Было изучены основные парадигмы программирования и их реализация в языке Python. Помимо объектно-ориентированного программирования были применены элементы функционального программирования.

Был разработан код, в котором были описаны пять классов – Edition, наследники которого – Book и Newspaper; BookList и NewspaperList, родителем которых выступил стандартный класс List. Были написаны методы для работы с экземплярами этих классов, причём важной их объединяющей особенностью служил контроль типов с помощью функции isinstance, с вызовом ошибки с помощью raise при необходимости. Так же были учтены особенности наследования и при перепределении метода родительского класса использовалась функция super() для избежания дублирования кода.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
    allStr = lambda *iterable: all(map(lambda x: isinstance(x, str), iterable))
    allBool = lambda *iterable: all(map(lambda x: isinstance(x, bool), iterable))
    allNatural = lambda *iterable: all(map(lambda x: isinstance(x, int) and (x > 0), iterable))

    class Edition:
        def init(self, name, price, age_limit, style):
           if allStr(name) and allNatural(price, age_limit) and (style in ['c','b']):
                self.name = name
                self.price = price
                self.age_limit = age_limit
                self.style = style
            else:
                raise ValueError('Invalid value')

    class Book(Edition):
          def init(self, name, price, age_limit, style, author, hardcover, pages):
            super().init(name, price, age_limit, style)
                  if allStr(author) and allBool(hardcover) and allNatural(pages):
                self.author = author
                self.hardcover = hardcover
                self.pages = pages
            else:
                 raise ValueError('Invalid value')
        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 allBool(online_edition) and allStr(country) and allNatural(frequency):
                self.online_edition = online_edition
                self.country = country
                self.frequency = frequency
            else:
                raise ValueError('Invalid value')
        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):
            return sum(map(lambda book: book.pages, 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(filter(lambda x: isinstance(x, Newspaper),
iterable))
        def print_age(self):
                print(min(map(lambda newspaper: newspaper.age_limit,
self)))
        def print_total_price(self):
            print(sum(map(lambda newspaper: newspaper.price, self)))
```