МГТУ им. Н.Э. Баумана

Отчёт по рубежному контролю №2

по курсу «Базовые компоненты и интернет-технологии»

Вариант 13.

Руководитель Гапанюк Ю.Е. 15.12.2022

Студент группы ИУ5-34Б Мкртчян Д.А.

15.12.2022

2022 г.

Полученное задание:

Рубежный контроль представляет собой разработку тестов на языке Python.

1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

Предметная область: класс\_1 – Книга, класс\_2 – Библиотека, вариант запросов: Г.

Запросы:

1. «Библиотека» и «Книга» связаны соотношением один-ко-многим. Выведите список всех библиотек, у которых название начинается с буквы «А», и список содержащихся в них книг.

2. «Библиотека» и «Книга» связаны соотношением один-ко-многим. Выведите список библиотек с максимальным размером книг в каждой библиотеке, отсортированный по максимальному количеству страниц.

3. «Библиотека» и «Книга» связаны соотношением многие-ко-многим. Выведите список всех связанных книг и библиотек, отсортированный по библиотекам, сортировка по книгам произвольная.

Текст программы:

1. Файл main.py:

# используется для сортировки from operator import itemgetter

from operator import itemgetter

class Book:

"""Книга"""

def \_\_init\_\_(self, id, name, pg, lib\_id):

self.id = id

self.name = name

self.pg = pg

self.lib\_id = lib\_id

class Lib:

"""Библиотека"""

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

self.id = id

self.name = name

class BookLib:

"""

'Книги библиотек' для реализации связи многие-ко-многим

"""

def \_\_init\_\_(self, lib\_id, book\_id):

self.lib\_id = lib\_id

self.book\_id = book\_id

# Библиотеки

libs = [

Lib(1, 'аниме'),

Lib(2, 'драма'),

Lib(3, 'комедии'),

Lib(4, 'биография'),

Lib(5, 'романтика'),

Lib(6, 'прочее'),

]

# Книги

books = [

Book(1, 'Дракула', 500, 1),

Book(2, 'Террор', 350, 2),

Book(3, 'Манюня', 600, 3),

Book(4, 'Дюна', 250, 1),

Book(5, 'Компромисс', 750, 3),

]

books\_libs = [BookLib(1, 1),

BookLib(2, 2),

BookLib(3, 3),

BookLib(3, 4),

BookLib(3, 5),

BookLib(4, 1),

BookLib(5, 2),

BookLib(6, 3),

BookLib(4, 4),

BookLib(3, 5),

]

# Соединение данных один-ко-многим

one\_to\_many = [(b.name, b.pg, l.name)

for l in libs for b in books

if b.lib\_id == l.id]

# Соединение данных многие-ко-многим

many\_to\_many\_temp = [(l.name, lb.lib\_id, lb.book\_id)

for l in libs

for lb in books\_libs if l.id == lb.lib\_id]

many\_to\_many = [(b.name, b.pg, lib\_name)

for lib\_name, lib\_id, book\_id in many\_to\_many\_temp

for b in books if b.id == book\_id]

def func1(one\_to\_many):

res\_11 = {}

for l in libs:

if l.name[0] == 'а':

l\_books = list(filter(lambda i: i[2] == l.name, one\_to\_many))

l\_books\_names = [x for x, \_, \_ in l\_books]

res\_11[l.name] = l\_books\_names

return res\_11

def func2(one\_to\_many):

res\_12\_unsorted = []

for l in libs:

l\_books = list(filter(lambda i: i[2] == l.name, one\_to\_many))

if len(l\_books) > 0:

l\_members = [mem for \_, mem, \_ in l\_books]

l\_members\_max = max(l\_members)

res\_12\_unsorted.append((l.name, l\_members\_max))

res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)

return res\_12

def func3(many\_to\_many):

res\_13 = sorted(many\_to\_many, key=itemgetter(2))

return res\_13

1. Файл test.py:

import unittest

from main import \*

class Test(unittest.TestCase):

def test\_1(self):

res = func1(one\_to\_many)

exp = {'аниме': ['Дракула', 'Дюна']}

self.assertEqual(res, exp)

def test\_2(self):

res = func2(one\_to\_many)

exp = [('комедии', 750), ('аниме', 500), ('драма', 350)]

self.assertEqual(res, exp)

def test\_3(self):

res = func3(many\_to\_many)

exp = [('Дракула', 500, 'аниме'),

('Дракула', 500, 'биография'),

('Дюна', 250, 'биография'),

('Террор', 350, 'драма'),

('Манюня', 600, 'комедии'),

('Дюна', 250, 'комедии'),

('Компромисс', 750, 'комедии'),

('Компромисс', 750, 'комедии'),

('Манюня', 600, 'прочее'),

('Террор', 350, 'романтика')]

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

Результаты выполнения:

