**Московский государственный технический университет им. Н.Э. Баумана**

Факультет «Радиотехнический»   
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчёт по РК 2

Выполнил: Проверил:

студент группы РТ5-31Б преподаватель каф. ИУ5

Петров Егор Гапанюк Юрий

Владимирович Евгеньевич

Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2021 г.

**Описание задания**

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).

**Текст РК1**

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

1. Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Пример классов данных для предметной области Сотрудник-Отдел:

* 1. Класс «Глава», содержащий поля:
     + ID записи о главе;
     + Название главы;
     + Количество страниц в главе (количественный признак);
     + ID записи о книге. (для реализации связи один-ко-многим)
  2. Класс «Книга», содержащий поля:
     + ID записи о книге;
     + Название книги.
  3. (Для реализации связи многие-ко-многим) Класс «Главы книги», содержащий поля:

o ID записи о главе;

o ID записи о книге.

1. Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
2. Необходимо разработать запросы в соответствии с вариантом.
   1. «Книга» и «Глава» связаны соотношением один-ко-многим. Выведите список всех книг, у которых в названии присутствует слово “оригинала”, и список содержащихся в них главы.
   2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате. Средняя зарплата должна быть округлена до 2 знака после запятой.
   3. «Книга» и «Глава» связаны соотношением многие-ко-многим. Выведите список всех книг, у которых в названии начинается с буквы «А», и названия их отделов.

Текст программы:  
**Файл main.py:**

from operator import itemgetter  
class Chapter:  
 *"""Глава"""* def \_\_init\_\_(self, title, id, pages, book\_id):  
 self.title = title  
 self.id = id  
 self.pages = pages  
 self.book\_id = book\_id  
  
  
class Book:  
 *"""Книга"""* def \_\_init\_\_(self, title, id):  
 self.title = title  
 self.id = id  
  
  
class ChapterBook:  
 def \_\_init\_\_(self, chap\_id, books\_id):  
 self.chap\_id = chap\_id  
 self.books\_id = books\_id  
 # Книги  
  
  
books = [  
 Book('Преступление и наказание', 1),  
 Book('1984(на языке оригинала)', 2),  
 Book('Искусство сновидения', 3),  
 Book('Властелин колец(на языке оригинала)', 4),  
 Book('Братья Карамазовы', 5),  
 Book('Великий Гэтсби(на языке оригинала)', 6),  
]  
# Главы  
chapters = [  
 Chapter('(Глава 1)', 1, 20, 1),  
 Chapter('(Глава 2)', 2, 12, 2),  
 Chapter('Глава 3', 3, 37, 3),  
 Chapter('Глава 4', 4, 24, 4),  
 Chapter('Глава 5', 5, 18, 5),  
 Chapter('Глава 6', 6, 40, 6),  
]  
# Главы книги, для связи многие-ко-многим   
chapters\_books = [  
 ChapterBook(1, 1),  
 ChapterBook(2, 5),  
 ChapterBook(3, 2),  
 ChapterBook(4, 4),  
 ChapterBook(5, 3),  
 ChapterBook(6, 6),  
 ChapterBook(11, 5),  
 ChapterBook(12, 4),  
 ChapterBook(13, 3),  
 ChapterBook(13, 2),  
 ChapterBook(13, 1),  
 ChapterBook(13, 3),  
 ]  
  
  
# Соединение данных один-ко-многим  
one\_to\_many = [(c.title, c.pages, b.title)  
 for b in books  
 for c in chapters  
 if b.id == c.book\_id]  
  
# Соединение данных многие-ко-многим  
many\_to\_many\_temp = [(b.title, cb.chap\_id, cb.books\_id)  
 for b in books  
 for cb in chapters\_books  
 if b.id == cb.books\_id]  
  
many\_to\_many = [(c.title, c.pages, b\_title)  
 for b\_title, chapters\_id, b\_id in many\_to\_many\_temp  
 for c in chapters  
 if c.id == chapters\_id]  
  
#print(one\_to\_many)  
def e1():  
 res\_3 = {}  
 # Перебираем все книги  
 for b in books:  
 #print(b.title)  
 # print(b.id)  
 if 'оригинала' in b.title:  
 # Список глав книги  
 b\_chapters = list(filter(lambda i: i[2] == b.title, one\_to\_many))  
 #print(one\_to\_many)  
 #print(b\_chapters)  
 # Только названия глав  
 b\_chapters\_titles = [e for e, \_, \_ in b\_chapters]  
 #print(b\_chapters\_titles)  
 # Добавляем результат в словарь  
 # ключ - книга, значение - список названий  
 res\_3[b.title] = b\_chapters\_titles  
 return(res\_3)  
  
def e2():  
 res\_12 = []  
 for b in books:  
 b\_chapters = list(filter(lambda x: x[2] == b.title, one\_to\_many))  
 if len(b\_chapters) > 0:  
 b\_pages = [pages for \_, pages, \_ in b\_chapters]  
 b\_pages\_min = min(b\_pages)  
 res\_12.append((b.title, b\_pages\_min))  
 res\_12\_sorted = sorted(res\_12, key=itemgetter(1))  
 return(res\_12\_sorted)  
  
  
def e3():  
 res\_3 = []  
 for title, \_, book in many\_to\_many:  
 if title[0] == "Г":  
 res\_3.append((title, book))  
 return(res\_3)  
  
def main():  
 print('\nЗадание E1')  
 print(e1())  
 print('\nЗадание E2')  
 print(e2())  
 print('\nЗадание E3')  
 print(e3())  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Файл test.py:**

import unittest  
import main  
class TDDtestGetRoots(unittest.TestCase):  
 def testGetRoots(self):  
 self.assertEqual(main.e1(), {'1984(на языке оригинала)': ['(Глава 2)'], 'Властелин колец(на языке оригинала)': ['Глава 4'], 'Великий Гэтсби(на языке оригинала)': ['Глава 6']})  
 self.assertEqual(main.e2(), [('1984(на языке оригинала)', 12), ('Братья Карамазовы', 18), ('Преступление и наказание', 20), ('Властелин колец(на языке оригинала)', 24), ('Искусство сновидения', 37), ('Великий Гэтсби(на языке оригинала)', 40)])  
 self.assertEqual(main.e3(), [('Глава 3', '1984(на языке оригинала)'), ('Глава 5', 'Искусство сновидения'), ('Глава 4', 'Властелин колец(на языке оригинала)'), ('Глава 6', 'Великий Гэтсби(на языке оригинала)')])  
if \_\_name\_\_ == "\_\_main\_\_":  
 unittest.main()

**Пример работы программы:**

