Московский государственный технический университет имени Н.Э.Баумана

Факультет РТ Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

Отчет по рубежному контролю №2 по курсу Базовые компоненты интернет-технологий

Вариант № **13**

Исполнитель

студент группы РТ5-21б _____ Малахов И.Д.

“___” _____ 2021 г.

Проверил

Преподаватель кафедры ИУ5 _____ Гапанюк Ю.Е.

“___” _____ 2021 г.

Москва - 2021

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).

Вариант Е.

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех отделов, у которых в названии присутствует слово «отдел», и список работающих в них сотрудников.
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 (*отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений; для округления необходимо использовать функцию https://docs.python.org/3/library/functions.html#round*).
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников, у которых фамилия начинается с буквы «А», и названия их отделов.

Группа	**ФИО**	**Вариант запросов РК1**	**Вариант предметной области РК1**
РТ5-31Б	Малахов Иван Дмитриевич	Е	13

№ варианта	**Класс 1**	**Класс 2**
13	Книга	Библиотека

Текст программы

main.py

```
class Book:
  def __init__(self, _id, name, num_of_pages, lib_id):
    self.id = _id
    self.name = name
    self.num_of_pages = num_of_pages
    self.lib_id = lib_id


class Library:
  """Отдел"""

  def __init__(self, _id, name):
    self.id = _id
    self.name = name


class BookLibrary:
  """
  'Сотрудники отдела' для реализации
  связи многие-ко-многим
  """

  def __init__(self, lib_id, book_id):
    self.lib_id = lib_id
    self.book_id = book_id


# Отделы
libraries = [
  Library(1, '№101'),
  Library(2, '№106'),
  Library(3, '№1182 (детская)'),
  Library(4, '№437'),
  Library(5, '№206 (детская)'),
  Library(6, '№549')
]

# Сотрудники
books = [
  Book(1, 'Улисс', 992, 2),
  Book(2, 'Над пропастью во ржи', 213, 5),
  Book(3, 'Преступление и наказание', 672, 1),
  Book(4, 'Басни Крылова', 100, 5),
  Book(5, 'Война и Мир', 1300, 4),
  Book(6, 'Хоббит, или Туда и Обратно', 268, 3),
  Book(7, 'Бойцовский клуб', 256, 6),
  Book(8, 'Дети синего фламинго', 224, 3),
  Book(9, 'Богатый папа, бедный папа', 224, 6),
```

```
  Book(10, '20000 лье под водой', 448, 5)
]
# связь 'многие ко многим'
books_libraries = [
  BookLibrary(1, 1),
  BookLibrary(1, 3),
  BookLibrary(1, 5),
  BookLibrary(1, 7),
  BookLibrary(1, 9),
  BookLibrary(2, 2),
  BookLibrary(2, 3),
  BookLibrary(2, 6),
  BookLibrary(2, 10),
  BookLibrary(3, 4),
  BookLibrary(3, 6),
  BookLibrary(3, 8),
  BookLibrary(4, 1),
  BookLibrary(4, 2),
  BookLibrary(4, 3),
  BookLibrary(4, 7),
  BookLibrary(5, 3),
  BookLibrary(5, 5),
  BookLibrary(5, 6),
  BookLibrary(5, 10),
  BookLibrary(6, 7),
  BookLibrary(6, 9)
]

# Соединение данных один-ко-многим
one_to_many = [(book.name, book.num_of_pages, lib.name)
        for lib in libraries
        for book in books
        if book.lib_id == lib.id]

# Соединение данных многие-ко-многим
many_to_many_temp = [(lib.name, bl.lib_id, bl.book_id)
            for lib in libraries
            for bl in books_libraries
            if lib.id == bl.lib_id]

many_to_many = [(book.name, book.num_of_pages, lib_name)
        for lib_name, lib_id, book_id in many_to_many_temp
        for book in books if book.id == book_id]


def task1():
  res1 = []
  for book_name, book_num, lib_name in one_to_many:
    if '(детская)' in lib_name:
      res1.append((lib_name, book_name))
  return res1
```

```
def task2():
  dict2 = {}
  for lib in libraries:
    dict2[lib.name] = [0, 0]
  # перебираем библиотеки
  for lib in libraries:
    # перебираем записи соответствия книги и библиотеки
    for i in one_to_many:
      if i[2] == lib.name:
        # добавляем количество страниц книги в общую сумму страниц в библиотеке
        dict2[lib.name][0] += i[1]
        # увеличиваем количество книг в библиотеке
        dict2[lib.name][1] += 1
  res2 = []
  # делаем список библиотек и среднего числа страниц в книге в библиотеке
  for key, value in dict2.items():
    res2.append((key, round(value[0] / value[1], 2)))
  # сортируем по числу страниц
  res2 = sorted(res2, key=lambda av: av[1], reverse=True)
  return res2


def task3():
  res3 = {}
  # создаем список для библиотек для книг с первой "Б"
  for book in books:
    if book.name[0] == 'Б':
      res3[book.name] = []

  # перебираем записи соответствия книг и библиотек
  for i in many_to_many:
    # если книга подходит
    if i[0][0] == 'Б':
      # записываем библиотеку в список библиотек, в которых есть эта книга
      res3[i[0]].append(i[2])
  return res3


def main():
  """Основная функция"""
  print('Задание А1')
  print(task1())
  print('\nЗадание А2')
  print(task2())
  print('\nЗадание А3')
  print(task3())


if __name__ == '__main__':
  main()
```

test.py

```
import unittest
import main

class MainTests(unittest.TestCase):

  def test_equal_1(self):
    self.assertEqual(main.task1(), [('№1182 (детская)', 'Хоббит, или Туда и Обратно'),
                    ('№1182 (детская)', 'Дети синего фламинго'),
                    ('№206 (детская)', 'Над пропастью во ржи'), ('№206 (детская)', 'Басни Крылова'),
                    ('№206 (детская)', '20000 лье под водой')])

  def test_equal_2(self):
    self.assertEqual(main.task2(), [('№437', 1300.0), ('№106', 992.0), ('№101', 672.0), ('№206 (детская)', 253.67),
                    ('№1182 (детская)', 246.0), ('№549', 240.0)])

  def test_equal_3(self):
    self.assertEqual(main.task3(),
             {'Басни Крылова': ['№1182 (детская)'], 'Бойцовский клуб': ['№101', '№437', '№549'],
              'Богатый папа, бедный папа': ['№101', '№549']})


if __name__ == "__main__":
  unittest.main(verbosity=2)
```

Результат работы программы

```
C:\Python\python.exe D:/учеба/прога/PyCharm/BKIT/BKIT/RK2/test.py
test_equal_1 (__main__.MainTests) ... ok
test_equal_2 (__main__.MainTests) ... ok
test_equal_3 (__main__.MainTests) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.000s

OK

Process finished with exit code 0
```