Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»
Отчёт по рубежному контролю №2

Выполнил:
студент группы ИУ5-31Б
Смыслов Дмитрий
Олегович

Подпись:_____

Дата:_____

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич

Подпись:_____

Дата:_____

Москва, 2021 г.

Вариант предметной области – 19 (“Деталь” и “Производитель”)
Вариант запросов – А

Условия рубежного контроля №2 по курсу БКИТ

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 программы

Файл main.py

```
from operator import itemgetter


class Detail:
    """Деталь"""

    def __init__(self, id, name, price, manufacturer_id):
        self.id = id
        self.name = name
        self.price = price  # Стоимость детали
        self.manufacturer_id = manufacturer_id


class Manufacturer:
    """Производитель"""

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


class Detail_Manufacturer:
    """
    'Детали производителя' для реализации
    связи многие-ко-многим
    """

    def __init__(self, manufacturer_id, detail_id):
        self.manufacturer_id = manufacturer_id
        self.detail_id = detail_id


# Производители
manufacturers = [
    Manufacturer(1, 'Надёжный производитель'),
    Manufacturer(2, 'Производитель деталей машин'),
    Manufacturer(3, 'Детали будущего для конструкторов и инженеров
```

```
настоящего'),

    Manufacturer(11, 'Надёжный (другой) производитель'),
    Manufacturer(22, 'Производитель (другой) деталей машин'),
    Manufacturer(33, '(другие) детали будущего для конструкторов и инженеров
настоящего'),
]

# Детали
details = [
    Detail(1, 'Штуцер', 1000, 1),
    Detail(2, 'Маховик', 5500, 2),
    Detail(3, 'Вал', 15750, 2),
    Detail(4, 'Поршень', 12250, 3),
    Detail(5, 'Фланец', 2000, 3),
    Detail(6, 'Демпфер', 8999, 3),
]

manufacturer_detail = [
    Detail_Manufacturer(1, 1),
    Detail_Manufacturer(2, 2),
    Detail_Manufacturer(2, 3),
    Detail_Manufacturer(3, 4),
    Detail_Manufacturer(3, 5),
    Detail_Manufacturer(3, 6),

    Detail_Manufacturer(11, 1),
    Detail_Manufacturer(22, 2),
    Detail_Manufacturer(22, 3),
    Detail_Manufacturer(33, 4),
    Detail_Manufacturer(33, 5),
    Detail_Manufacturer(33, 6),
]


# Соединение данных один-ко-многим
def bound_otm(mnfs, dts):
    return [(d.name, d.price, m.name)
            for m in mnfs
            for d in dts
            if d.manufacturer_id == m.id]


# Соединение данных многие-ко-многим
def bound_mtm(mnfs, det, mnfdt):
    many_to_many_temp = [(m.name, md.manufacturer_id, md.detail_id)
                         for m in mnfs
                         for md in mnfdt
                         if m.id == md.manufacturer_id]
    return [(d.name, d.price, m_name)
            for m_name, m_id, d_id in many_to_many_temp
            for d in det if d.id == d_id]


def taskA1(otm):
    return sorted(otm, key=itemgetter(2))


def taskA2(mnfs, otm):
    res_unsorted = []
    for m in mnfs:
        m_details = list(filter(lambda i: i[2] == m.name, otm))
        if len(m_details) > 0:
            m_price = [price for _, price, _ in m_details]
```

```
            m_price_sum = sum(m_price)
            res_unsorted.append((m.name, m_price_sum))
    return sorted(res_unsorted, key=itemgetter(1), reverse=True)


def taskA3(mnfs, mtm):
    res = {}
    for m in mnfs:
        if 'производитель' in m.name.lower():
            m_details = list(filter(lambda i: i[2] == m.name, mtm))
            m_details_names = [x for x, _, _ in m_details]
            res[m.name] = m_details_names
    return res


def main():
    # Соединение данных один-ко-многим
    one_to_many = bound_otm(manufacturers, details)

    # Соединение данных многие-ко-многим
    many_to_many = bound_mtm(manufacturers, details, manufacturer_detail)

    print('Задание А1')
    print(taskA1(one_to_many))

    print('\nЗадание А2')
    print(taskA2(manufacturers, one_to_many))

    print('\nЗадание А3')
    print(taskA3(manufacturers, many_to_many))


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

Файл tests.py

```
from main import *
import unittest


class RKTest(unittest.TestCase):

    def setUp(self) -> None:
        # Производители
        self.manufacturers = [
            Manufacturer(1, 'Производитель деталей и запчастей для роботов'),
            Manufacturer(2, 'Детали студентам на ИнжГрафе'),
            Manufacturer(3, 'Современный производитель надежных деталей'),

            Manufacturer(11, 'Производитель (другой) деталей и запчастей для
роботов'),
            Manufacturer(22, 'Детали (другие) студентам на ИнжГрафе'),
            Manufacturer(33, 'Современный производитель (другой) надежных
деталей'),
        ]

        # Детали
        self.details = [
            Detail(1, 'Механическая рука', 24999, 1),
            Detail(2, 'Фланец', 1000, 2),
            Detail(3, 'Вал', 2500, 2),
            Detail(4, 'Гайка', 100, 3),
            Detail(5, 'Коленчатый вал', 5000, 3)
```

```
        ]

        # Производители и их детали
        self.manufacturer_detail = [
            Detail_Manufacturer(1, 1),
            Detail_Manufacturer(2, 2),
            Detail_Manufacturer(2, 3),
            Detail_Manufacturer(3, 4),
            Detail_Manufacturer(3, 5),

            Detail_Manufacturer(11, 1),
            Detail_Manufacturer(22, 2),
            Detail_Manufacturer(22, 3),
            Detail_Manufacturer(33, 4),
            Detail_Manufacturer(33, 5),
        ]

        self.one_to_many = bound_otm(self.manufacturers, self.details)
        self.many_to_many = bound_mtm(self.manufacturers, self.details,
self.manufacturer_detail)


    def test_taskA1(self):
        result = taskA1(self.one_to_many)
        needed_res = [('Фланец', 1000, 'Детали студентам на ИнжГрафе'),
                      ('Вал', 2500, 'Детали студентам на ИнжГрафе'),
                      ('Механическая рука', 24999, 'Производитель деталей и
запчастей для роботов'),
                      ('Гайка', 100, 'Современный производитель надежных
деталей'),
                      ('Коленчатый вал', 5000, 'Современный производитель
надежных деталей')]
        self.assertEqual(result, needed_res)


    def test_taskA2(self):
        result = taskA2(self.manufacturers, self.one_to_many)
        needed_res = [('Производитель деталей и запчастей для роботов',
24999),
                      ('Современный производитель надежных деталей', 5100),
                      ('Детали студентам на ИнжГрафе', 3500)]
        self.assertEqual(result, needed_res)


    def test_taskA3(self):
        result = taskA3(self.manufacturers, self.many_to_many)
        needed_res = {'Производитель деталей и запчастей для роботов':
['Механическая рука'],
                      'Современный производитель надежных деталей': ['Гайка',
'Коленчатый вал'],
                      'Производитель (другой) деталей и запчастей для
роботов': ['Механическая рука'],
                      'Современный производитель (другой) надежных деталей':
['Гайка', 'Коленчатый вал']}
        self.assertEqual(result, needed_res)


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

Результат выполнения программы

```
(venv) PS D:\JetBrains\PyCharm 2021.2.2\PycharmProjects\BKIT_RK> python -m main
Задание A1
[('Поршень', 12250, 'Детали будущего для конструкторов и инженеров настоящего'), ('Флане
ц', 2000, 'Детали будущего для конструкторов и инженеров настоящего'), ('Демпфер', 8999,
 'Детали будущего для конструкторов и инженеров настоящего'), ('Штуцер', 1000, 'Надёжный
 производитель'), ('Маховик', 5500, 'Производитель деталей машин'), ('Вал', 15750, 'Прои
зводитель деталей машин')]

Задание A2
[('Детали будущего для конструкторов и инженеров настоящего', 23249), ('Производитель де
талей машин', 21250), ('Надёжный производитель', 1000)]

Задание A3
{'Надёжный производитель': ['Штуцер'], 'Производитель деталей машин': ['Маховик', 'Вал']
, 'Надёжный (другой) производитель': ['Штуцер'], 'Производитель (другой) деталей машин':
 ['Маховик', 'Вал']}
(venv) PS D:\JetBrains\PyCharm 2021.2.2\PycharmProjects\BKIT_RK>
```

TODO Problems Terminal Python Packages Python Console Event Log

```
(venv) PS D:\JetBrains\PyCharm 2021.2.2\PycharmProjects\BKIT_RK> python -m tests -v
test_taskA1 (__main__.RKTest) ... ok
test_taskA2 (__main__.RKTest) ... ok
test_taskA3 (__main__.RKTest) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
(venv) PS D:\JetBrains\PyCharm 2021.2.2\PycharmProjects\BKIT_RK>
```

TODO Problems Terminal Python Packages Python Console Event Log