Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «ГУИМЦ»

Кафедра ИУ5 «Системы обработки информации и управления»

Дисциплина «Базовые компоненты ИТ»

РУБЕЖНЫЙ КОНТРОЛЬ №2

Студент: Близнева А.Е., группа ИУ5Ц-51Б

Преподаватель: Гапанюк Ю.Е.

2021г.

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).

main.py

```
"""РК№1, Близнева Ангелина, группа ИУ5Ц-51Б
Вариант А, вариант предметной области 25"""

# используется для сортировки
from operator import itemgetter


class Doc:
    """Документ"""

    def __init__(self, id, name, number, sect_id):
        self.id = id
        self.name = name
        self.number = number
        self.doc_id = sect_id


class Sect:
    """Раздел"""

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


class SectDoc:
    """
    'Документы раздела' для реализации
    связи многие-ко-многим
    """

    def __init__(self, sect_id, doc_id):
        self.sect_id = sect_id
        self.doc_id = doc_id


# Разделы
sects = [
    Sect(1, 'Приказы'),
    Sect(2, 'Распоряжения'),
    Sect(3, 'Протоколы'),

    Sect(11, 'Приказы (филиал компании)'),
    Sect(22, 'Распоряжения (филиал компании)'),
    Sect(33, 'Протоколы (филиал компании)'),
]

# Документы
docs = [
```

```
    Doc(1, 'Приказ о ежегодном отпуске', 333, 1),
    Doc(2, 'Распоряжение для выполнения задания', 1231, 2),
    Doc(3, 'Протокол совещания', 56, 3),
    Doc(4, 'Протокол общего собрания учредителей', 34, 3),
    Doc(5, 'Протокол заседания экспертной комиссии', 25, 3),
]

sects_docs = [
    SectDoc(1, 1),
    SectDoc(2, 2),
    SectDoc(3, 3),
    SectDoc(3, 4),
    SectDoc(3, 5),

    SectDoc(11, 1),
    SectDoc(22, 2),
    SectDoc(33, 3),
    SectDoc(33, 4),
]

def sort_name(doc):
    return sorted(doc, key=itemgetter(2))

def sort_number(doc, sects):
    res_12_unsorted = []
    # Перебираем все разделы
    for s in sects:
        # Список документов раздела
        d_docs = list(filter(lambda i: i[2] == s.name, doc))
        # Если раздел не пустой
        if len(d_docs) > 0:
            # Кол-во документов раздела
            d_nums = [sal for _, sal, _ in d_docs]
            # Суммарное кол-во документов раздела
            d_nums_sum = sum(d_nums)
            res_12_unsorted.append((s.name, d_nums_sum))
    # Сортировка по суммарному кол-ву документов
    return sorted(res_12_unsorted, key=itemgetter(1), reverse=True)

def sort_sect(doc, sects):
    res_13 = {}
    # Перебираем все разделы
    for s in sects:
        if 'Протокол' in s.name:
            # Список документов раздела
            d_docs = list(filter(lambda i: i[2] == s.name, doc))
            # Только наименование документа
            d_docs_names = [x for x, _, _ in d_docs]
            # Добавляем результат в словарь
            # ключ - раздел, значение - список документов
            res_13[s.name] = d_docs_names
    return res_13

def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(d.name, d.number, s.name)
                   for s in sects
                   for d in docs
                   if d.doc_id == s.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(s.name, sd.sect_id, sd.doc_id)
```

```
                        for s in sects
                        for sd in sects_docs
                        if s.id == sd.sect_id]

    many_to_many = [(d.name, d.number, sect_name)
                    for sect_name, sect_id, emp_id in many_to_many_temp
                    for d in docs if d.id == emp_id]

    print('Задание A1\n',sort_name(one_to_many))
    print('\nЗадание A2\n', sort_number(one_to_many, sects))
    print('\nЗадание A3\n', sort_sect(many_to_many, sects))



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

tests.py

```
import unittest
import sys, os

sys.path.append(os.getcwd())
from main import *

one_to_many = [(d.name, d.number, s.name)
               for s in sects
               for d in docs
               if d.doc_id == s.id]

many_to_many_temp = [(s.name, sd.sect_id, sd.doc_id)
                     for s in sects
                     for sd in sects_docs
                     if s.id == sd.sect_id]

many_to_many = [(d.name, d.number, sect_name)
                for sect_name, sect_id, emp_id in many_to_many_temp
                for d in docs if d.id == emp_id]

class TestCost(unittest.TestCase):
    def test_sort_name(self):
        self.assertEqual(sort_name(one_to_many),[('Приказ о ежегодном
отпуске', 333, 'Приказы'),
                                                 ('Протокол совещания', 56,
'Протоколы'),
                                                 ('Протокол общего собрания
учредителей', 34, 'Протоколы'),
                                                 ('Протокол заседания
экспертной комиссии', 25, 'Протоколы'),
                                                 ('Распоряжение для
выполнения задания', 1231, 'Распоряжения')]
)

    def test_sort_number(self):
        self.assertEqual(sort_number(one_to_many, sects), [('Распоряжения',
1231), ('Приказы', 333), ('Протоколы', 115)])

    def test_sort_sect(self):
        self.assertEqual(sort_sect(many_to_many, sects), {'Протоколы':
['Протокол совещания', 'Протокол общего собрания учредителей',
                                                          'Протокол
заседания экспертной комиссии'], 'Протоколы (филиал компании)':
            ['Протокол совещания', 'Протокол общего собрания учредителей']})
```

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

Результат выполнения программы:

```
Testing started at 1:44 ...
Launching unittests with arguments python -m unittest



Ran 3 tests in 0.002s

OK
```