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 Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

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

Вариант № 25
(Вариант Г, класс 1 – Раздел, класс 2 – Документ)

ИСПОЛНИТЕЛЬ:

студент группы ИУ5Ц-54Б _____
Перевощиков Н.Д.

(подпись)

Преподаватель
Гапанюк Ю.Е. "__"_____2022 г.

Москва, МГТУ - 2022

СОДЕРЖАНИЕ ОТЧЕТА

1. 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).

2. Листинг программы

2.1.RK_2.py (Основной файл)

```
from operator import itemgetter

# Класс Раздел
class Section:

    def __init__(self, id, name, page, document_id):
        # Номер раздела
        self.id = id
        # Наименование раздела
        self.name = name
        # Количество страниц раздела
        self.page = page
        # Номер документа
        self.document_id = document_id

# Класс Документ
class Document:

    def __init__(self, id, name):
        # Номер документа
        self.id = id
        # Наименование документа
        self.name = name

# Класс Раздел Документ
class SectionDocument:

    def __init__(self, document_id, section_id):
        self.document_id = document_id
        self.section_id = section_id

# Документы
documents = [
    Document(1, 'Положение'),
    Document(2, 'Агентский договор'),
    Document(3, 'Правила'),
    Document(4, 'Инструкция'),
    Document(5, 'Регламент'),
]

# Разделы
sections = [
    Section(1, 'Содержание', 10, 1),
    Section(2, 'Введение', 20, 2),
    Section(3, 'Определение', 31, 5),
    Section(4, 'Нормативные ссылки', 8, 2),
    Section(5, 'Заключение', 44, 3),
    Section(6, 'Список использованных источников', 28, 4),
    Section(7, 'Общие положения', 61, 1),
    Section(8, 'Права', 11, 5),
    Section(9, 'Функции', 66, 3),
    Section(10, 'Ответственность', 94, 4),
]

# Разделы документов
sections_documents = [
    SectionDocument(2, 1),
    SectionDocument(3, 5),
    SectionDocument(4, 3),
    SectionDocument(5, 6),
    SectionDocument(5, 2),
```

```
    SectionDocument(1, 4),
    SectionDocument(4, 7),
    SectionDocument(3, 6),
    SectionDocument(1, 1),
    SectionDocument(5, 3),
]

def one_to_many(documents, sections):
    return[(sec.name, sec.page, document.name)
                   for document in documents
                   for sec in sections
                   if sec.document_id == document.id]

def many_to_many_temp(documents, sections_documents):
    return [(document.name, sec_documents.document_id,
sec_documents.section_id)
                          for document in documents
                          for sec_documents in sections_documents
                          if document.id == sec_documents.document_id]

def many_to_many(documents, sections):
    return [(sec.name, sec.page, document_name)
                     for document_name, document_id, sec_id in
many_to_many_temp(documents, sections_documents)
                     for sec in sections if sec.id == sec_id]

def task_1(documents, sections):
    mas_dict = {}
    for folder_name, i, document_name in one_to_many(documents, sections):
        if document_name[0] == 'A':
            if document_name in mas_dict:
                mas_dict[document_name].append(folder_name)
            else:
                mas_dict[document_name] = [folder_name]
    return mas_dict.items()

def task_2(documents, sections):
    mas_dict_1 = {}
    for i, poisk_max, document_name in one_to_many(documents, sections):
        if document_name in mas_dict_1:
            mas_dict_1[document_name] = max(mas_dict_1[document_name],
poisk_max)
        else:
            mas_dict_1[document_name] = poisk_max
        mas_dict_1 = {key: meaning for key, meaning in
sorted(mas_dict_1.items(), key=lambda item: item[1])}
    return mas_dict_1.items()

def task_3(documents, sections):
    mas_list = []
    for folder_name, i, document_name in many_to_many(documents,
sections):
        mas_list.append((document_name, folder_name))
    mas_list = sorted(mas_list, key=lambda item: item[0])
    return mas_list

def main():
    print('Задание Г1')
    print(*task_1(documents, sections))

    print('\nЗадание Г2')
    print(*task_2(documents, sections))

    print('\nЗадание Г3')
```

```
    print(*task_3(documents, sections))

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

2.2.Modul_test.py (Файл модульного теста)

```
import unittest

from RK_2 import Section, Document, SectionDocument, task_1, task_2,
task_3

modul_document = [
    Document(1, 'Отчет по БКИТ'),
    Document(2, 'Акт'),
    Document(3, 'Практическое задание')
]
modul_section = [
    Section(1, 'Список литературы', 3, 2),
    Section(2, 'Задания', 4, 3),
    Section(3, 'Результат работы', 25, 1)
]
modul_sections_documents = [
    SectionDocument(2, 1),
    SectionDocument(1, 1),
    SectionDocument(3, 2)
]

class test_section(unittest.TestCase):
    def test_class_section_zero_parameters(self):
        with self.assertRaises(TypeError) as context:
            Section()
        self.assertEqual(
            "__init__() missing 4 required positional arguments: 'id',
'name', 'page', and 'document_id'",
            str(context.exception)
        )

    def test_class_section_zero_meaning(self):
        test_class_section = Section(None, None, None, None)
        self.assertEqual(test_class_section.id, None)
        self.assertEqual(test_class_section.name, None)
        self.assertEqual(test_class_section.page, None)
        self.assertEqual(test_class_section.document_id, None)

    def test_class_section_meaning(self):
        test_class_section = Section(1, 'Технические основы разработки',
10, 3)
        self.assertEqual(test_class_section.id, 1)
        self.assertEqual(test_class_section.name, 'Технические основы
разработки')
        self.assertEqual(test_class_section.page, 10)
        self.assertEqual(test_class_section.document_id, 3)

    def test_class_document_meaning(self):
        test_class_document = Document(1, 'Отчет по МД')
        self.assertEqual(test_class_document.id, 1)
        self.assertEqual(test_class_document.name, 'Отчет по МД')

    def test_class_section_document_meaning(self):
        test_class_sections_documents = SectionDocument(3, 1)
        self.assertEqual(test_class_sections_documents.document_id, 3)
        self.assertEqual(test_class_sections_documents.section_id, 1)
```

```
    # Тестирование задания 1
    def test_task_1(self):
        self.assertEqual(dict(task_1(modul_document, modul_section)),
{'Акт': ['Список литературы']})

    # Тестирование задания 2
    def test_task_2(self):
        self.assertEqual(dict(task_2(modul_document, modul_section)),
{'Отчет по БКИТ': 25, 'Практическое задание': 4, 'Акт': 3})

    # Тестирование задания 3
    def test_task_3(self):
        self.assertEqual(task_3(modul_document, modul_section), [('Акт',
'Список литературы'), ('Отчет по БКИТ', 'Список литературы')])

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

3. Результат работы программы

3.1.RK_2.py (Основной файл)

3.1.1. PyCharm

```
Задание Г1
('Агентский договор', ['Введение', 'Нормативные ссылки'])

Задание Г2
('Агентский договор', 20) ('Регламент', 31) ('Положение', 61) ('Правила', 66) ('Инструкция', 94)

Задание Г3
('Агентский договор', 'Содержание') ('Инструкция', 'Определение') ('Инструкция', 'Общие положения') ('Положение', 'Нормативные

Process finished with exit code 0
```

Задание Г1
('Агентский договор', ['Введение', 'Нормативные ссылки'])

Задание Г2
('Агентский договор', 20) ('Регламент', 31) ('Положение', 61) ('Правила', 66) ('Инструкция', 94)

Задание Г3
('Агентский договор', 'Содержание') ('Инструкция', 'Определение') ('Инструкция', 'Общие положения') ('Положение', 'Нормативные ссылки') ('Положение', 'Содержание') ('Правила', 'Заключение') ('Правила', 'Список использованных источников') ('Регламент', 'Список использованных источников') ('Регламент', 'Введение') ('Регламент', 'Определение')

Process finished with exit code 0

3.1.2. cmd (Командная строка)

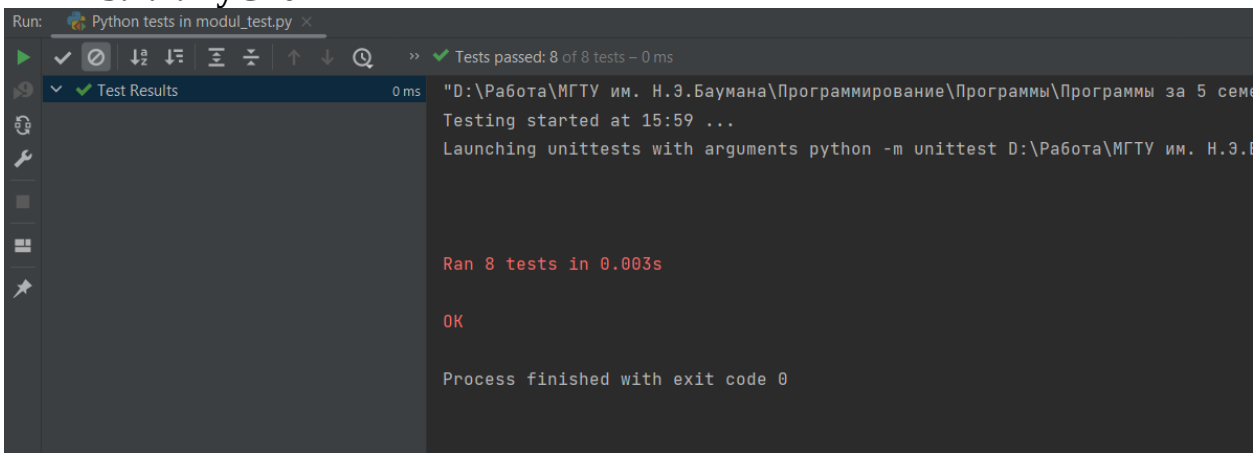
```
D:\Работа\МГТУ им. Н.Э.Баумана\Программирование\Программы\Программы за 5 семестр\RK_2>python RK_2.py
Задание Г1
('Агентский договор', ['Введение', 'Нормативные ссылки'])

Задание Г2
('Агентский договор', 20) ('Регламент', 31) ('Положение', 61) ('Правила', 66) ('Инструкция', 94)

Задание Г3
('Агентский договор', 'Содержание') ('Инструкция', 'Определение') ('Инструкция', 'Общие положения') ('По
ложение', 'Нормативные ссылки') ('Положение', 'Содержание') ('Правила', 'Заключение') ('Правила', 'Списо
к использованных источников') ('Регламент', 'Список использованных источников') ('Регламент', 'Введение'
) ('Регламент', 'Определение')
```

3.2.Modul_test.py (Файл модульного теста)

3.2.1. PyCharm



3.2.2. cmd (Командная строка)

```
D:\Работа\МГТУ им. Н.Э.Баумана\Программирование\Программы\Программы за 5 семестр\RK_2>python modul_test.py
........
----------------------------------------------------------------------
Ran 8 tests in 0.001s

OK
```