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 ИУ5-32Б

Милевич Артём Андреевич

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Подпись и дата:

Москва, 2021 г.

Описание задания

(Вариант предметной области - 12, вариант запросов - Б)

1. Провести рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.
2. Для текста программы рубежного контроля №1 создать модульные тесты с применением TDD - фреймворка (3 теста).

Текст программы

Файл **main.py**:

```
from operator import itemgetter

class Language:
    """Язык программирования"""
    def __init__(self, id, title, creator, tool_id):
        self.id = id
        self.title = title
        self.creator = creator
        self.tool_id = tool_id

class Tool:
    """Средство разработки"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

class LangTool:
    """
    'Язык программирования' для реализации
    связи многие-ко-многим
    """
    def __init__(self, tool_id, lang_id):
        self.tool_id = tool_id
        self.lang_id = lang_id

def task1(languages, tools):
    one_to_many = [(l.title, l.creator, t.name)
        for t in tools
        for l in languages
        if l.tool_id == t.id]
    return sorted(one_to_many, key=itemgetter(2))

def task2(languages, tools):
```

```
    res = list()
    one_to_many = [(l.title, l.creator, t.name)
        for t in tools
        for l in languages
        if l.tool_id == t.id]
    for t in tools:
        # Список языков в средствах разработки
        t_leng = list(filter(lambda i: i[2] == t.name, one_to_many))
        # Если средства разработки не пустые
        if len(t_leng) > 0:
            res.append((t.name, len(t_leng)))
    return sorted(res, key=itemgetter(1), reverse=True)

def task3(languages, tools, lang_tool):
    res = dict()
    many_to_many = [(l.title, l.creator, t.name)
        for t in tools
        for l in languages
        for relation in lang_tool
        if t.id == relation.tool_id and l.id == relation.lang_id]
    for l in languages:
        if l.title.endswith("ов"):
            # Ищем средства разработки конкретного языка
            l_tools = list(filter(lambda x: x[0] == l.title, many_to_many))
            # Получаем их названия
            l_tools_names = [x[2] for x in l_tools]
            res[l.title] = l_tools_names
    return res


def main():

    tools = [
    Tool(1, 'Visual Studio'),
    Tool(2, 'XCode'),
    Tool(3, 'Visual Studio Code'),
    Tool(4, 'Netbeans'),
]

    # Языки программирования
    languages = [
        Language(1, 'Python', 'Гвидо ван Россум', 1),
        Language(2, 'C++_ов', 'Бьёрн Страуструп', 2),
        Language(3, 'C', 'Деннис Ритчи', 3),
        Language(4, 'Pascal_ов', 'Никлаус Вирт', 3),
        Language(5, 'Java', 'Джеймс Гослинг', 4),
    ]

    lang_tool = [
        LangTool(1,1),
```

```
        LangTool(1,2),
        LangTool(1,3),
        LangTool(2,2),
        LangTool(2,4),
        LangTool(3,1),
        LangTool(3,3),
        LangTool(3,4),
    ]

    print('Задание Б1')
    res_1 = task1(languages, tools)
    for res in res_1:
        print(res)

    print('\nЗадание Б2')
    res_2 = task2(languages, tools)
    [print(el) for el in res_2]

    print("\nЗадание Б3")
    res_3 = task3(languages, tools, lang_tool)
    [print(k, v) for k, v in res_3.items()]

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

Файл **test.py**:

```
import unittest
from main import Tool, Language, LangTool, task2, task1, task3

class Test(unittest.TestCase):

    def setUp(self):

        self.tools = [
        Tool(1, 'Visual Studio'),
        Tool(2, 'XCode'),
        Tool(3, 'Visual Studio Code'),
        Tool(4, 'Netbeans'),
        ]

        self.languages = [
            Language(1, 'Python', 'Гвидо ван Россум', 1),
            Language(2, 'C++_ов', 'Бьёрн Страуструп', 2),
            Language(3, 'C', 'Деннис Ритчи', 3),
            Language(4, 'Pascal_ов', 'Никлаус Вирт', 3),
            Language(5, 'Java', 'Джеймс Гослинг', 4),
        ]
```

```
        self.lang_tool = [
            LangTool(1,1),
            LangTool(1,2),
            LangTool(1,3),
            LangTool(2,2),
            LangTool(2,4),
            LangTool(3,1),
            LangTool(3,3),
            LangTool(3,4),
        ]

    def test_task1(self):
        keys = [
            ('Java', 'Джеймс Гослинг', 'Netbeans'),
            ('Python', 'Гвидо ван Россум', 'Visual Studio'),
            ('C', 'Деннис Ритчи', 'Visual Studio Code'),
            ('Pascal_ов', 'Никлаус Вирт', 'Visual Studio Code'),
            ('C++_ов', 'Бьёрн Страуструп', 'XCode'),
            ]
        self.assertEqual(task1(self.languages, self.tools), keys)

    def test_task2(self):
        keys = [
            ('Visual Studio Code', 2),
            ('Visual Studio', 1),
            ('XCode', 1),
            ('Netbeans', 1),
            ]
        self.assertEqual(task2(self.languages, self.tools), keys)

    def test_task3(self):
        keys = {
            "C++_ов": ['Visual Studio', 'XCode'],
            "Pascal_ов": ['XCode', 'Visual Studio Code'],
        }
        self.assertEqual(task3(self.languages, self.tools, self.lang_tool), keys)

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

Результат выполнения программы

```
PS C:\Microsoft VS Code\python\rk2\code> & C:/Users/artem/AppData/Local/Microsoft/WindowsApps/python3.9.exe "c:/Microsoft VS Code/python/rk2/code/test.py"
...
----------------------------------------------------------------------
Ran 3 tests in 0.000s

OK
```