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 ИУ5 «Системы обработки информации и управления»

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

Выполнила:
студентка группы ИУ5-33Б
Юрова Е.О.

Проверил:
Гапанюк Ю.Е.

Описание задачи:

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

```
# coding=utf-8
from operator import itemgetter

class Lib:
    """ Библиотека """

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

class Lang:
    """ Язык программирования"""
    def __init__(self, id, title):
        self.id = id
        self.title = title

class LibLang:
    """ Библиотеки языков программирования для реализации связи многие-ко-
многим"""
    def __init__(self, lang_id, lib_id):
        self.lib_id = lib_id
        self.lang_id = lang_id

# Языки программирования
langs = [
    Lang(1, 'Python'),
    Lang(2, 'JavaScript'),
    Lang(3, 'Java'),
    Lang(5, 'C++'),
    Lang(6, 'C#'),
    Lang(7, 'Delphi')
]
# библиотеки
libs = [
    Lib(1, 'Pandas', 890, 1),
    Lib(2, 'Matplotlib', 950, 2),
    Lib(6, 'Voca', 754, 2),
    Lib(3, 'Maven', 489, 3),
    Lib(4, 'Leaflet', 481, 5),
    Lib(5, 'SFML', 678, 6),

    Lib(7, 'VCL',350 , 6)
]

libs_langs = [
    LibLang(1,1),
    LibLang(2,2),
```

```
    LibLang(2,6),
    LibLang(3,3),
    LibLang(5,4),
    LibLang(6,5),
    LibLang(6,7),
    LibLang(7,1),
    LibLang(7,2),
    LibLang(7, 3),
    LibLang(7, 4)

]

def B1(one_to_many):
    res_11 = []
    for i in one_to_many:
        if i[0][0] == "M":
            res_11.append([i[0], i[2]])
    return res_11

def B2(one_to_many):
    res_22 = []
    res_22 = [[one_to_many[0][2], one_to_many[0][1]]]
    for name, memory, title in one_to_many:
        if title == res_22[len(res_22) - 1][0]:
            if memory < res_22[len(res_22) - 1][1]:
                res_22[len(res_22) - 1][1] = memory
        else:
            res_22.append([title, memory])
        res_22 = sorted(res_22, key = itemgetter(1))
    return res_22

def B3(many_to_many):
    res_33 = []
    for title, memory, lib in many_to_many:
        res_33.append([title, lib])
    res_33 = sorted(res_33, key = itemgetter(0))
    return res_33


def main():
    """Основная функция"""
    # Соединение данных один-ко-многим
    one_to_many = [(l.name, l.memory, la.title)
                   for la in langs
                   for l in libs
                   if l.lang_id == la.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(la.title, lal.lang_id, lal.lib_id)
                         for la in langs
                         for lal in libs_langs
                         if la.id == lal.lang_id]

    many_to_many = [(l.name, l.memory, lang_title)
                    for lang_title, lang_id, lib_id in many_to_many_temp
                    for l in libs if l.id == lib_id]

    print('Задание B1')
    print(B1(one_to_many))

    print('\n Задание B2')
    print(B2(one_to_many))

    print('\n Задание B3')
```

```
    print(B3(many_to_many))


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

Файл tdd.py

```
import unittest
import sys, os

from main import *


class TestMain(unittest.TestCase):
    def test_1(self):
        one_to_many = [(l.name, l.memory, la.title)
                       for la in langs
                       for l in libs
                       if l.lang_id == la.id]
        self.assertEqual(B1(one_to_many),[['Matplotlib', 'JavaScript'],
['Maven', 'Java']])

    def test_2(self):
        one_to_many = [(l.name, l.memory, la.title)
                       for la in langs
                       for l in libs
                       if l.lang_id == la.id]
        self.assertEqual(B2(one_to_many),[['C#', 350], ['C++', 481], ['Java',
489], ['C#', 678], ['JavaScript', 754], ['Python', 890]])

    def test_3(self):
        many_to_many_temp = [(la.title, lal.lang_id, lal.lib_id)
                             for la in langs
                             for lal in libs_langs
                             if la.id == lal.lang_id]

        many_to_many = [(l.name, l.memory, lang_title)
                        for lang_title, lang_id, lib_id in many_to_many_temp
                        for l in libs if l.id == lib_id]
        self.assertEqual(B3(many_to_many),[['Leaflet', 'C++'], ['Leaflet',
'Delphi'],
                                           ['Matplotlib', 'JavaScript'],
['Matplotlib', 'Delphi'],
                                           ['Maven', 'Java'], ['Maven',
'Delphi'], ['Pandas', 'Python'],
                                           ['Pandas', 'Delphi'], ['SFML',
'C#'], ['VCL', 'C#'], ['Voca', 'JavaScript']])
```

Результаты выполнения программы: