**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»

# Отчет по рубежному контролю №2

Выполнил:
студент группы ИУ5-32Б
Зайцев А.Д.

Москва, 2021 г.

Описание задания:
(Вариант предметной области – 7, вариант запросов – Б)

1. «Компьютер» и «Микропроцессор» связаны соотношением один-ко-многим. Выведите список всех связанных микропроцессоров и компьютеров, отсортированный по микропроцессорам, сортировка по компьютерам произвольная.
2. «Компьютер» и «Микропроцессор» связаны соотношением один-ко-многим. Выведите список компьютеров с количеством микропроцессоров в каждом компьютере, отсортированный по количеству микропроцессоров.
3. «Компьютер» и «Микропроцессор» связаны соотношением многие-ко-многим. Выведите список всех микропроцессоров, у которых имя заканчивается на «1», и названия их компьютеров.

## Условия рубежного контроля №2 по курсу БКИТ

4. Рубежный контроль представляет собой разработку тестов на языке Python.
5. 1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.
6. 2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

Текст программы:

Файл RK2.py

```
from operator import itemgetter


class Microprocessor:
    def __init__(self, id, name, freq, computer_id):
        self.id = id
        self.name = name
        self.freq = freq
        self.computer_id = computer_id


class Computer:
    def __init__(self, id, name):
        self.id = id
        self.name = name
```

```python
class MicCom:
    def __init__(self, proc_id, comp_id):
        self.proc_id = proc_id
        self.comp_id = comp_id


micros = [
    Microprocessor(1, "intel1", 2300, 1),
    Microprocessor(2, "ryzen1", 2400, 3),
    Microprocessor(3, "ryzen3", 3500, 2),
    Microprocessor(4, "ryzen7", 4200, 4),
    Microprocessor(5, "ryzen5", 6500, 3),
    Microprocessor(6, "pentium1", 5643, 1),
    Microprocessor(7, "xenon9", 3456, 2),
    Microprocessor(8, "intel3", 2950, 1),
    Microprocessor(9, "intel9", 4053, 3),
]
comps = [
    Computer(1, "RussianButcher"),
    Computer(2, "Buster"),
    Computer(3, "Evelon"),
    Computer(4, "TORONTOTOKYO"),
]

miccomps = [
    MicCom(1, 1),
    MicCom(2, 3),
    MicCom(3, 2),
    MicCom(4, 4),
    MicCom(5, 3),
    MicCom(6, 1),
    MicCom(2, 1),
    MicCom(8, 1),
    MicCom(9, 3),
    MicCom(3, 1),
    MicCom(8, 3),
    MicCom(6, 4),
    MicCom(7, 1),
    MicCom(7, 3),
    MicCom(1, 2),
    MicCom(7, 3),
    MicCom(2, 2),
    MicCom(1, 4),
    MicCom(9, 4),
]

def sortingB1(one_to_many):
    one_to_many = sorted(one_to_many, key=lambda x: x[0])
    return one_to_many
def sortingB2(one_to_many):
    res_12_unsorted = []
```

```
    for c in comps:

        c_micros = list(filter(lambda i: i[2] == c.name, one_to_many))

        if len(c_micros) > 0:
            res_12_unsorted.append((c.name, len(c_micros)))
    res12 = sorted(res_12_unsorted, key=lambda x: x[1], reverse=True)
    return res12
def sortingB3(many_to_many):
    res_13 = {}
    for m in micros:
        if m.name.endswith("1"):
            m_comps = list(filter(lambda i: i[0] == m.name, many_to_many))
            m_comps_name = [x[2] for x in m_comps]
            res_13[m.name] = m_comps_name
    return res_13
def main():

    one_to_many = [
        (m.name, m.freq, c.name) for c in comps for m in micros if m.computer_id
== c.id
    ]

    many_to_many = [
        (m.name, m.freq, c.name)
        for c in comps
        for m in micros
        for r in miccomps
        if c.id == r.comp_id and m.id == r.proc_id
    ]

    print("Задание Б1")

    res1 = sortingB1(one_to_many)
    [print(x) for x in res1]

    print("\nЗадание Б2")
    res12 = sortingB2(one_to_many)
    [print(x) for x in res12]

    print("\nЗадание Б3")
    res_13 = sortingB3(many_to_many)
    for k, v in res_13.items():
        print(k, v)


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

Файл test_RK2.py:

```
import unittest
from RK2 import *

class test_RK2(unittest.TestCase):
    def test_sortingB1(self):
        one_to_many = [(m.name, m.freq, c.name) for c in comps for m in micros if
m.computer_id == c.id]
        res = sortingB1(one_to_many)
        self.assertEqual(res,[('intel1', 2300, 'RussianButcher'),('intel3', 2950,
'RussianButcher'),('intel9', 4053, 'Evelon'),('pentium1', 5643,
'RussianButcher'),('ryzen1', 2400, 'Evelon'),('ryzen3', 3500,
'Buster'),('ryzen5', 6500, 'Evelon'),('ryzen7', 4200, 'TORONTOTOKYO'),('xenon9',
3456, 'Buster')])
    def test_sortingB2(self):
        one_to_many = [(m.name, m.freq, c.name) for c in comps for m in micros if
m.computer_id == c.id]
        res = sortingB2(one_to_many)
        self.assertEqual(res,[('RussianButcher', 3),('Evelon', 3),('Buster',
2),('TORONTOTOKYO', 1)])
    def test_sortingB3(self):
        many_to_many = [
        (m.name, m.freq, c.name)
        for c in comps
        for m in micros
        for r in miccomps
        if c.id == r.comp_id and m.id == r.proc_id
        ]
        res = sortingB3(many_to_many)
        self.assertEqual(res,{"intel1" : ['RussianButcher', 'Buster',
'TORONTOTOKYO'],"ryzen1" : ['RussianButcher', 'Buster', 'Evelon'],"pentium1" :
['RussianButcher', 'TORONTOTOKYO']})




if __name__ == "__name__":
    unittest.main()
    #python -m unittest test_RK2.py
```

Результат выполнения программы:

```
PS C:\Users\Scare> & c:/Users/Scare/AppData/Local/Programs/Python/Python39/python.exe c:/Users/Scare/Documents/BKIT1/RK-2_BKIT/RK2.py
Задание Б1
('intel1', 2300, 'RussianButcher')
('intel3', 2950, 'RussianButcher')
('intel9', 4053, 'Evelon')
('pentium1', 5643, 'RussianButcher')
('ryzen1', 2400, 'Evelon')
('ryzen3', 3500, 'Buster')
('ryzen5', 6500, 'Evelon')
('ryzen7', 4200, 'TORONTOTOKYO')
('xenon9', 3456, 'Buster')

Задание Б2
('RussianButcher', 3)
('Evelon', 3)
('Buster', 2)
('TORONTOTOKYO', 1)

Задание Б3
intel1 ['RussianButcher', 'Buster', 'TORONTOTOKYO']
ryzen1 ['RussianButcher', 'Buster', 'Evelon']
pentium1 ['RussianButcher', 'TORONTOTOKYO']
PS C:\Users\Scare>
```

Результат выполнения тестов:

```
PS C:\Users\Scare> cd C:\Users\Scare\Documents\BKIT1\RK-2_BKIT
PS C:\Users\Scare\Documents\BKIT1\RK-2_BKIT> python -m unittest test_RK2.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
PS C:\Users\Scare\Documents\BKIT1\RK-2_BKIT>
```