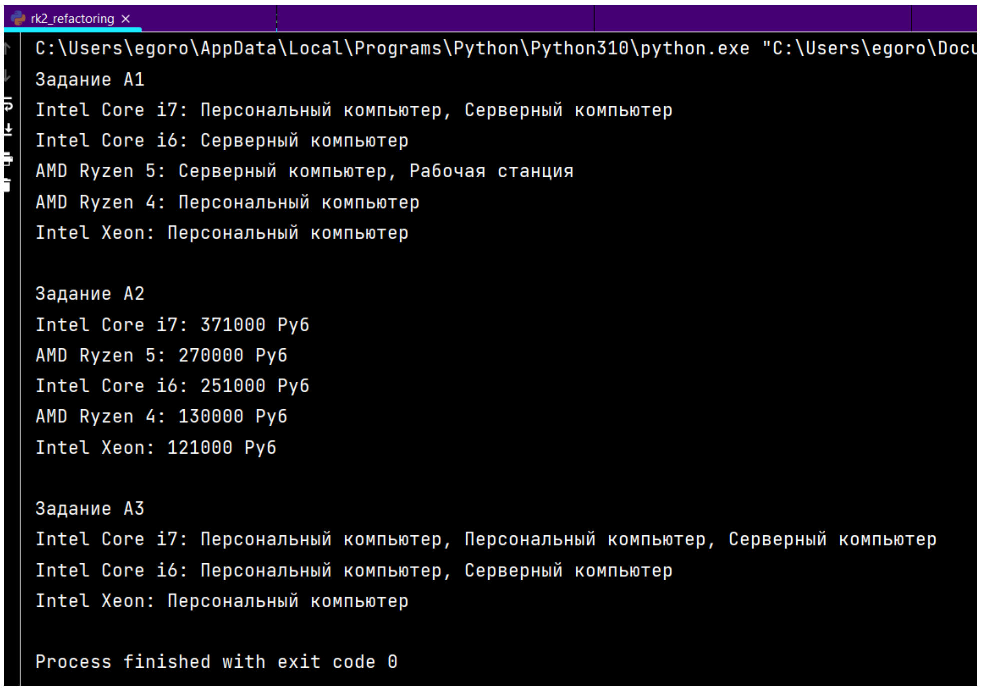
Егоров Дмитрий ИУ5-31Б Вариант 7

Файл rk1_refactoring.py

```python
from operator import itemgetter

class Computer:
    """Класс для представления Компьютера"""
    def __init__(self, id, name, price, micro_id):
        self.id = id
        self.name = name
        self.price = price
        self.micro_id = micro_id

class Microprocessor:
    """Класс для представления Микропроцессора"""
    def __init__(self, id, model):
        self.id = id
        self.model = model

class CompMicro:
    """
    Связь многие-ко-многим между компьютерами и
микропроцессорами
    """
    def __init__(self, comp_id, micro_id):
        self.comp_id = comp_id
        self.micro_id = micro_id


# Список микропроцессоров
microprocessors = [
    Microprocessor(1, 'Intel Core i7'),
    Microprocessor(2, 'Intel Core i6'),
    Microprocessor(3, 'AMD Ryzen 5'),
    Microprocessor(4, 'AMD Ryzen 4'),
    Microprocessor(5, 'Intel Xeon'),
]

# Список компьютеров
computers = [
    Computer(1, 'Персональный компьютер', 120000, 1),
    Computer(2, 'Персональный компьютер', 121000, 5),
    Computer(3, 'Персональный компьютер', 121000, 4),
    Computer(4, 'Серверный компьютер', 150000, 1),
    Computer(5, 'Серверный компьютер', 130000, 2),
    Computer(6, 'Серверный компьютер', 125252, 3),
    Computer(7, 'Рабочая станция', 52, 3),
]

# Связь многие-ко-многим между компьютерами и
микропроцессорами
comp_micros = [
    CompMicro(1, 1),
    CompMicro(2, 1),
    CompMicro(3, 2),
    CompMicro(4, 3),
    CompMicro(5, 2),
    CompMicro(5, 1),
    CompMicro(5, 4),
    CompMicro(2, 5),
    CompMicro(1, 3),
]


def one_to_many_relationship(computers, microprocessors):
    """Создает связь один-ко-многим"""
    return [(m.model, c.name)
            for m in microprocessors
            for c in computers
            if c.micro_id == m.id]


def many_to_many_relationship(computers, microprocessors,
comp_micros):
    """Создает связь многие-ко-многим"""
    return [(m.model, c.name, c.price)
            for m in microprocessors
            for cm in comp_micros
            for c in computers
            if cm.micro_id == m.id and cm.comp_id == c.id]

def task_A1(computers, microprocessors):
    """Задание А1: Список всех компьютеров для каждого
микропроцессора"""
    one_to_many = one_to_many_relationship(computers,
microprocessors)
    result = {}
    for model, comp_name in one_to_many:
        result.setdefault(model, []).append(comp_name)
    return result

def task_A2(computers, microprocessors, comp_micros):
    """Задание А2: Суммарная стоимость всех компьютеров,
использующих каждый микропроцессор"""
    many_to_many = many_to_many_relationship(computers,
microprocessors, comp_micros)
    result = [
        (m.model, sum(comp_price for _, _, comp_price in
filter(lambda x: x[0] == m.model, many_to_many)))
        for m in microprocessors
    ]
    return sorted(result, key=itemgetter(1), reverse=True)

def task_A3(computers, microprocessors, comp_micros,
keyword):
    """Задание А3: Вывод всех процессоров с определенным
```

```
словом и их компьютеров"""
    many_to_many = many_to_many_relationship(computers,
microprocessors, comp_micros)
    result = {}
    for model, comp_name, _ in many_to_many:
        if keyword.lower() in model.lower():
            result.setdefault(model, []).append(comp_name)
    return result

if __name__ == '__main__':
    print('Задание A1')
    for model, comps in task_A1(computers,
microprocessors).items():
        print(f"{model}: {', '.join(comps)}")

    print('\nЗадание A2')
    for model, total in task_A2(computers, microprocessors,
comp_micros):
        print(f"{model}: {total} Руб")

    print('\nЗадание A3')
    for model, comps in task_A3(computers, microprocessors,
comp_micros, 'Intel').items():
        print(f"{model}: {', '.join(comps)}")
```

Результаты



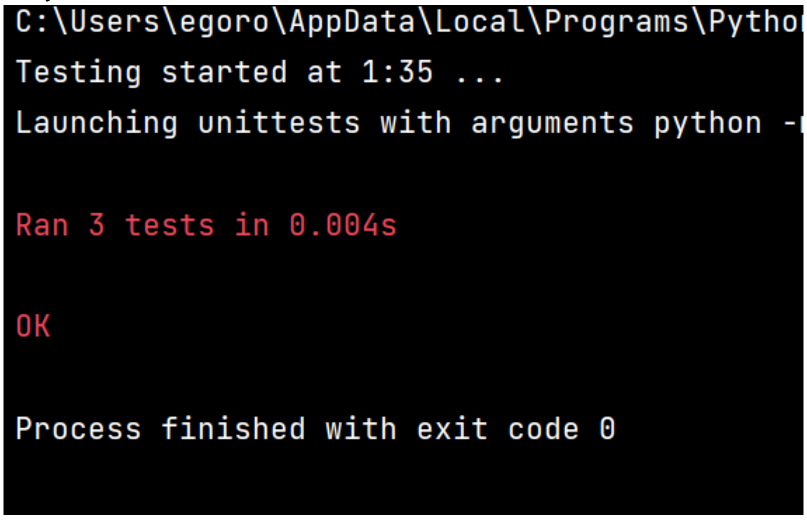
```
C:\Users\egoro\AppData\Local\Programs\Python\Python310\python.exe "C:\Users\egoro\Docu
Задание A1
Intel Core i7: Персональный компьютер, Серверный компьютер
Intel Core i6: Серверный компьютер
AMD Ryzen 5: Серверный компьютер, Рабочая станция
AMD Ryzen 4: Персональный компьютер
Intel Xeon: Персональный компьютер

Задание A2
Intel Core i7: 371000 Руб
AMD Ryzen 5: 270000 Руб
Intel Core i6: 251000 Руб
AMD Ryzen 4: 130000 Руб
Intel Xeon: 121000 Руб

Задание A3
Intel Core i7: Персональный компьютер, Персональный компьютер, Серверный компьютер
Intel Core i6: Персональный компьютер, Серверный компьютер
Intel Xeon: Персональный компьютер

Process finished with exit code 0
```

Файл rk1_tests.py

```
import unittest
from rk2_refactoring import *

class TestTasks(unittest.TestCase):
    def setUp(self):
```

```python
        self.microprocessors = [
            Microprocessor(1, 'Intel Core i7'),
            Microprocessor(2, 'AMD Ryzen 5'),
            Microprocessor(3, 'Intel Xeon'),
        ]
        self.computers = [
            Computer(1, 'Персональный компьютер', 120000,
1),
            Computer(2, 'Серверный компьютер', 150000, 1),
            Computer(3, 'Рабочая станция', 80000, 2),
        ]
        self.comp_micros = [
            CompMicro(1, 1),
            CompMicro(2, 1),
            CompMicro(3, 2),
        ]

    def test_task_A1(self):
        result = task_A1(self.computers,
self.microprocessors)
        expected = {
            'Intel Core i7': ['Персональный компьютер',
'Серверный компьютер'],
            'AMD Ryzen 5': ['Рабочая станция'],
        }
        self.assertEqual(result, expected)

    def test_task_A2(self):
        result = task_A2(self.computers,
self.microprocessors, self.comp_micros)
        expected = [
            ('Intel Core i7', 270000),
            ('AMD Ryzen 5', 80000),
            ('Intel Xeon', 0),
        ]
        self.assertEqual(result, expected)

    def test_task_A3(self):
        result = task_A3(self.computers,
self.microprocessors, self.comp_micros, 'Intel')
        expected = {
            'Intel Core i7': ['Персональный компьютер',
'Серверный компьютер'],
        }
        self.assertEqual(result, expected)


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

Результаты:



```
C:\Users\egoro\AppData\Local\Programs\Pytho
Testing started at 1:35 ...
Launching unittests with arguments python -

Ran 3 tests in 0.004s

OK

Process finished with exit code 0
```