Московский государственный технический

университет им. Н. Э. Баумана

Факультет «Радиотехнический»

Кафедра РТ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю №2

Выполнил:

студент группы РТ5-31Б

Суровец Е. А.

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Москва, 2024 г.

Текст программы

```
from operator import itemgetter
import unittest


class Detail:
    def __init__(self, id, name, weight, price, id_prov):
        self.id = id
        self.name = name
        self.weight = weight
        self.price = price
        self.id_prov = id_prov


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


class DetProv:
    def __init__(self, det_id, prov_id):
        self.det_id = det_id
        self.prov_id = prov_id


providers = [
    Provider(1, 'АртАвто'),
    Provider(2, 'РусТрейдСервис'),
    Provider(3, 'Кетекс'),
    Provider(4, 'Партгрейд'),
    Provider(5, 'Комтранс')
]

details = [
    Detail(1, 'Катушка', 2000, 12000, 3),
    Detail(2, 'Свеча зажигания', 1400, 7000, 1),
    Detail(3, 'Фара', 430, 9600, 1),
    Detail(4, 'Термостат', 1150, 13500, 4),
    Detail(5, 'Аккумулятор', 1330, 8750, 5)
]

dets_provs = [
    DetProv(1, 1),
    DetProv(2, 2),
    DetProv(3, 3),
    DetProv(3, 4),
    DetProv(3, 5),
    DetProv(4, 1),
    DetProv(4, 2),
    DetProv(5, 3),
    DetProv(5, 4),
```

```
    DetProv(5, 5),
]

one_to_many = [(d.name, d.price, p.name)
               for d in details
               for p in providers
               if d.id_prov == p.id]

many_to_many_temp = [(p.name, dp.prov_id, dp.det_id)
                     for p in providers
                     for dp in dets_provs
                     if p.id == dp.prov_id]

many_to_many = [(d.name, d.price, prov_name)
                for prov_name, prov_id, det_id in
many_to_many_temp
                for d in details if d.id == det_id]


def findByWord(data, word):
    res_1 = [el for el in data if word in el[2]]
    print(res_1)
    return res_1


def getAvgEachProvider(data):
    res_2_unsorted = []
    for prov in providers:

        prov_det = [(el[0], el[1]) for el in data if prov.name
in el]
        if len(prov_det) > 0:
            print("prov_det=", prov_det)
            sred_price = sum([el[1] for el in prov_det]) /
len(prov_det)
            print(sred_price)
            res_2_unsorted.append((prov.name, sred_price))

    res_2 = sorted(res_2_unsorted, key=itemgetter(1),
reverse=True)
    print(res_2)
    return res_2


def getDetailByLetter(data, letter):
    res_3 = []
    for el in data:
        if el[0][0] == letter:
            res_3.append((el[0], el[2]))
    print(res_3)
    return res_3
```

```
class TestByModule(unittest.TestCase):
    def test_find_by_word(self):
        result = findByWord(one_to_many, 'Авто')
        expected = [('Свеча зажигания', 7000, 'АртАвто'),
('Фара', 9600, 'АртАвто')]
        self.assertEqual(result, expected)

    def test_get_avg_each_provider(self):
        result = getAvgEachProvider(one_to_many)
        expected = [('Партгрейд', 13500.0), ('Кетекс', 12000.0),
('Комтранс', 8750.0), ('АртАвто', 8300.0)]
        self.assertEqual(result, expected)

    def test_get_detail_by_letter(self):
        result = getDetailByLetter(many_to_many, 'Т')
        expected = [('Термостат', 'АртАвто'), ('Термостат',
'РусТрейдСервис')]
        self.assertEqual(result, expected)


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

Результат выполнения

```
Ran 3 tests in 0.003s

OK
prov_det= [('Свеча зажигания', 7000), ('Фара', 9600)]
8300.0
prov_det= [('Катушка', 12000)]
12000.0
prov_det= [('Термостат', 13500)]
13500.0
prov_det= [('Аккумулятор', 8750)]
8750.0
[('Партгрейд', 13500.0), ('Кетекс', 12000.0), ('Комтранс', 8750.0), ('АртАвто', 8300.0)]
[('Термостат', 'АртАвто'), ('Термостат', 'РусТрейдСервис')]

Process finished with exit code 0
```