Текст программы

PKmain.py

```
# используется для сортировки
from operator import itemgetter

class Brauser:
    """Браузер"""

    def __init__(self, id, name, share, brause_id):
        self.id = id
        self.name = name
        self.market_share = share
        self.brause_id = brause_id

class Dep:
    """Компьютер"""

    def __init__(self, id, model):
        self.id = id
        self.model = model

class Brauser_Dep:
    """
    'Браузеры компьютера' для реализации связи многие-ко-многим
    """

    def __init__(self, brause_id, emp_id):
        self.brause_id = brause_id
        self.emp_id = emp_id


# Компьютеры
deps = [
    Dep(1, 'MacBook Air'),
    Dep(2, 'MacBook Pro'),
    Dep(3, 'Xiaomi RedmiBook'),

    Dep(11, 'Asus ViviBook Pro'),
    Dep(22, 'Lenovo ThinkPad'),
    Dep(33, 'Honor 2020'),
]

# Браузеры
brausers = [
    Brauser(1, 'Google', 112.52, 11),
    Brauser(2, 'Opera', 21.1, 11),
    Brauser(3, 'Safari', 78.3, 33),
    Brauser(4, 'Yandex', 53.8, 33),
    Brauser(5, 'Firefox', 11.23, 2),
]

brausers_deps = [
    Brauser_Dep(1, 1),
    Brauser_Dep(1, 2),
    Brauser_Dep(1, 3),
    Brauser_Dep(3, 4),
    Brauser_Dep(2, 5),

    Brauser_Dep(11, 1),
    Brauser_Dep(22, 2),
    Brauser_Dep(33, 3),
```

```
    Brauser_Dep(33, 4),
    Brauser_Dep(33, 5),
]


def a1_solution(one_to_many):
    res_a1 = sorted(one_to_many, key=itemgetter(2))
    return res_a1

def a2_solution(one_to_many):
    res_a2_unsorted = []
    # Перебираем все компьютеры
    for i in deps:
        # Список браузеров, установленных компьютерами
        d_brausers = list(filter(lambda k: k[2] == i.model, one_to_many))
        # Если хотя бы один браузеров установлен
        if len(d_brausers) > 0:
            # Доли рынка каждого браузера
            braus_market_shares = [share for _, share, _ in d_brausers]
            # Общая доля рынка установленных браузеров
            braus_ms_sum = sum(braus_market_shares)
            res_a2_unsorted.append((i.model, braus_ms_sum))

    # Сортировка по суммарной доле рынка
    res_a2 = sorted(res_a2_unsorted, key=itemgetter(1), reverse=True)
    return res_a2

def a3_solution(many_to_many):
    res_a3 = {}
    # Перебираем все компьютеры
    for d in deps:
        if 'Mac' in d.model:
            # Список браузеров компьютеров
            d_brausers = list(filter(lambda i: i[2] == d.model, many_to_many))
            # Название браузеров
            d_brausers_names = [x for x, _, _ in d_brausers]
            # Добавляем результат в словарь
            # ключ - компьютер, значение - список названий браузеров
            res_a3[d.model] = d_brausers_names

    return res_a3

def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(e.name, e.market_share, d.model)
                   for d in deps
                   for e in brausers
                   if e.brause_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.model, ed.brause_id, ed.emp_id)
                         for d in deps
                         for ed in brausers_deps
                         if d.id == ed.brause_id]

    many_to_many = [(e.name, e.market_share, dep_name)
                    for dep_name, brause_id, emp_id in many_to_many_temp
                    for e in brausers if e.id == emp_id]

    print('Задание A1')
    print(a1_solution(one_to_many))

    print('\nЗадание A2')
    print(a2_solution(one_to_many))
```

```
    print('\nЗадание А3')
    print(a3_solution(many_to_many))

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

PKtddmain.py

```
import unittest
from PKmain import *


class TestRK2(unittest.TestCase):
    # Компьютеры
    deps = [
        Dep(1, 'MacBook Air'),
        Dep(2, 'MacBook Pro'),
        Dep(3, 'Xiaomi RedmiBook'),

        Dep(11, 'Asus ViviBook Pro'),
        Dep(22, 'Lenovo ThinkPad'),
        Dep(33, 'Honor 2020'),
    ]

    # Браузеры
    brausers = [
        Brauser(1, 'Google', 112.52, 11),
        Brauser(2, 'Opera', 21.1, 11),
        Brauser(3, 'Safari', 78.3, 33),
        Brauser(4, 'Yandex', 53.8, 33),
        Brauser(5, 'Firefox', 11.23, 2),
    ]

    def test_A1(self):
        one_to_many = [(e.name, e.market_share, d.model)
                       for d in deps
                       for e in brausers
                       if e.brause_id == d.id]
        self.assertEqual(a1_solution(one_to_many),
                         [('Google', 112.52, 'Asus ViviBook Pro'), ('Opera',
21.1, 'Asus ViviBook Pro'),
                          ('Safari', 78.3, 'Honor 2020'), ('Yandex', 53.8,
'Honor 2020'), ('Firefox', 11.23, 'MacBook Pro')]
)

    def test_A2(self):
        one_to_many = [(e.name, e.market_share, d.model)
                       for d in deps
                       for e in brausers
                       if e.brause_id == d.id]
        self.assertEqual(a2_solution(one_to_many),
                         [('Asus ViviBook Pro', 133.62), ('Honor 2020', 132.1),
('MacBook Pro', 11.23)])

    def test_A3(self):
        many_to_many_temp = [(d.model, ed.brause_id, ed.emp_id)
                             for d in deps
                             for ed in brausers_deps
                             if d.id == ed.brause_id]

        many_to_many = [(e.name, e.market_share, dep_name)
                        for dep_name, brause_id, emp_id in many_to_many_temp
```

```
                        for e in brausers if e.id == emp_id]
        self.assertEqual(a3_solution(many_to_many),
                         {'MacBook Air': ['Google', 'Opera', 'Safari'], 'MacBook
Pro': ['Firefox']})


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

Результаты выполнения

Пример успешного прохождения тестов
Ran 3 tests in 0.002s

OK

Пример неудачного прохождения тестов
Ran 3 tests in 0.010s

FAILED (failures=1)