Кулешовой Ирины, ИУ5-35Б

Рубежный контроль №2 по курсу БКИТ

Классы: «Браузер» - «Компьютер»

Запросы:

1. «Браузер» и «Компьютер» связаны соотношением один-ко-многим. Выведите список всех браузеров, название компании-производителя которых заканчивается на «х», и названия компьютеров на которых их можно поставить.
2. «Браузер» и «Компьютер» связаны соотношением один-ко-многим. Выведите список компьютеров со средним значением количества баллов по безопасности браузеров на каждом компьютере, отсортированный по среднему значению количества баллов.
3. «Браузер» и «Компьютер» связаны соотношением многие-ко-многим. Выведите список всех компьютеров, у которых название начинается с буквы «S», и список браузеров, которые на них можно поставить.

Текст программы:

main.py

```
from operator import itemgetter


class Browser:
    """Браузер"""
    def __init__(self, id, name, dev, safety, com_id):
        self.id = id
        self.name = name
        self.dev = dev
        self.saf = safety
        self.com_id = com_id



class Computer:
    """Компьютер"""
```

```
    def __init__(self, id, name, dev, display_size, storage, exp_storage):
        self.id = id
        self.name = name
        self.dev = dev
        self.d_size = display_size
        self.stor = storage
        self.exp_stor = exp_storage


class BrowComp:
    """
    'Браузеры компьютера' для реализации
    связи многие-ко-многим
    """
    def __init__(self, com_id, brow_id):
        self.com_id = com_id
        self.brow_id = brow_id


def num1(browsers, computers):
    one_to_many = [(c.name, b.name, b.dev) for b in browsers for c in computers if b.com_id ==
c.id]
    a = [i for i in one_to_many if i[2][len(i[2]) - 1] == 'x']
    return a


def num2(browsers, computers):
    s = 0
    count = 0
    avg = []
    for c in computers:
        for b in browsers:
            if b.com_id == c.id:
```

```
            count = count + 1
            s = s + b.saf
        avg.append((c.name, '%.2f' % (s / count)))
    result2 = sorted(avg, key=itemgetter(1))
    return result2


def manytomany(browsers, computers, brow_comp):
    many_to_many_temp = [(c.name, bc.com_id, bc.brow_id) for c in computers for bc in
brow_comp if c.id == bc.com_id]
    many_to_many = [(com_name, b.name) for com_name, com_id, brow_id in
many_to_many_temp for b in browsers if
                b.id == brow_id]
    print(many_to_many)
    return many_to_many


def main():
    # Компьютеры
    computers = [
        Computer(1, 'Surface', 'Microsoft', 10.6, 64, 'microSD'),
        Computer(2, 'Surface Pro', 'Microsoft', 10.6, 128, 'microSDXC'),
        Computer(3, 'iPad', 'Apple', 9.7, 64, None),
        Computer(4, 'Galaxy Tab 2 10.1', 'Samsung', 10.1, 32, 'microSD'),
        Computer(5, 'Excite 10', 'Toshiba', 10.1, 64, 'microSD'),
        Computer(6, 'Transformer Pad Infinity 700', 'Asus', 10.1, 64, 'microSD'),
        Computer(7, 'Icona Tab A700', 'Acer', 10.1, 64, 'microSD')
    ]


    # Браузеры
    browsers = [
        Browser(1, 'Chrome', 'Google', 16, 1),
        Browser(2, 'Firefox', 'Mozilla', 13, 1),
        Browser(3, 'IE', 'Microsoft', 15, 2),
        Browser(4, 'Epic', 'Hidden Reflex', 16, 4),
```

```
        Browser(5, 'Vivaldi', 'Vivaldi Technologies', 16, 2),
        Browser(6, 'Yandex', 'Yandex', 16, 3),
        Browser(7, 'Opera', 'Opera', 16, 5),
        Browser(8, 'Safari', 'Apple', 15, 3)
    ]

    # Таблицы для связи многие-ко-многим
    brow_comp = [
        BrowComp(2, 5)
    ]

    print("Задание Д1")
    print(num1(browsers, computers))

print("_____
____\n")

    print("Задание Д2")
    print(num2(browsers, computers))

print("_____
____\n")

    print("Задание Д3")
    many_to_many = manytomany(browsers, computers, brow_comp)
    result3 = list(filter(lambda i: i[0][0]=='S', many_to_many))
    comp = result3[0][0]
    print('Название компьютера: ', result3[0][0])
    print('Список браузеров для {}:'.format(comp))
    for i in result3:
        if i[0] == comp:
            print('    -', i[1])
        else:
```

```
        print('Название компьютера: ', i[0])
        comp = i[0]
        print('Список браузеров для {}:'.format(comp))

print("_____
____")


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

tdd-tests.py

```
import unittest
from operator import itemgetter
from main import num1, num2, manytomany, Computer, Browser, BrowComp


class TestBrowComp(unittest.TestCase):
    def test_num1(self):
        br1 = [
            Browser(1, 'Firefox', 'Mozilla', 13, 1),
            Browser(2, 'IE', 'Microsoft', 15, 2),
            Browser(3, 'Vivaldi', 'Vivaldi Technologies', 16, 2),
            Browser(4, 'Yandex', 'Yandex', 16, 3),
            Browser(5, 'Safari', 'Apple', 15, 3)
        ]
        cm = [
            Computer(1, 'Surface', 'Microsoft', 10.6, 64, 'microSD'),
            Computer(2, 'Surface Pro', 'Microsoft', 10.6, 128, 'microSDXC'),
            Computer(3, 'iPad', 'Apple', 9.7, 64, None),
```

```
            Computer(4, 'Galaxy Tab 2 10.1', 'Samsung', 10.1, 32, 'microSD'),
            Computer(5, 'Excite 10', 'Toshiba', 10.1, 64, 'microSD'),
            Computer(6, 'Transformer Pad Infinity 700', 'Asus', 10.1, 64, 'microSD'),
            Computer(7, 'Icona Tab A700', 'Acer', 10.1, 64, 'microSD')
        ]
        self.assertEqual(num1(br1, cm), [('iPad', 'Yandex', 'Yandex')])

        br2 = [
            Browser(1, 'Firefox', 'Mozilla', 13, 1),
            Browser(2, 'IE', 'Microsoft', 15, 2),
            Browser(3, 'Vivaldi', 'Vivaldi Technologies', 16, 2),
            Browser(4, 'Safari', 'Apple', 15, 3)
        ]
        self.assertEqual(num1(br2, cm), [])

        br3 = [
            Browser(1, 'Chrome', 'Google', 16, 1),
            Browser(2, 'Firefox', 'Mozilla', 13, 1),
            Browser(3, 'IE', 'Microsoft', 15, 2),
            Browser(4, 'Epic', 'Hidden Reflex', 16, 4),
            Browser(5, 'Vivaldi', 'Vivaldi Technologies', 16, 2),
            Browser(6, 'Yandex', 'Yandex', 16, 3),
            Browser(7, 'Opera', 'Opera', 16, 5),
            Browser(8, 'Safari', 'Apple', 15, 3)
        ]
        self.assertEqual(num1(br3, cm), [('Galaxy Tab 2 10.1', 'Epic', 'Hidden
Reflex'), ('iPad', 'Yandex', 'Yandex')])


    def test_num2(self):
```

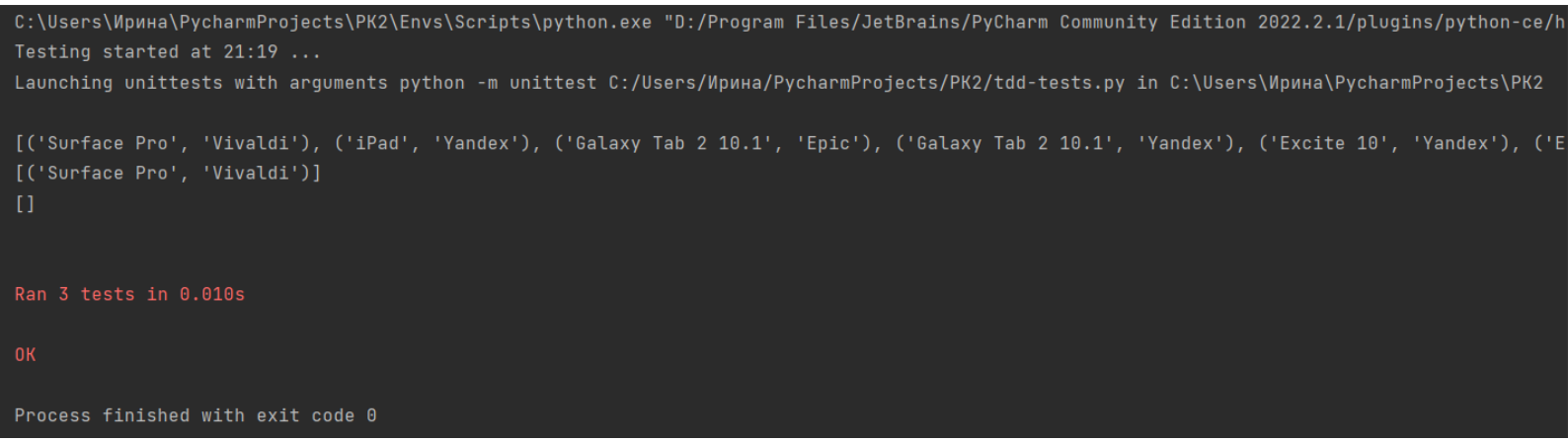
```
cm1 = [
    Computer(1, 'Surface', 'Microsoft', 10.6, 64, 'microSD'),
    Computer(2, 'Surface Pro', 'Microsoft', 10.6, 128, 'microSDXC')
]
br1 = [
    Browser(1, 'Chrome', 'Google', 16, 1),
    Browser(2, 'Firefox', 'Mozilla', 13, 1),
    Browser(3, 'IE', 'Microsoft', 15, 1)
]
self.assertEqual(num2(br1, cm1), [('Surface', '14.67'), ('Surface Pro', '14.67')])

cm2 = [
    Computer(1, 'Surface', 'Microsoft', 10.6, 64, 'microSD'),
    Computer(2, 'Surface Pro', 'Microsoft', 10.6, 128, 'microSDXC'),
    Computer(3, 'iPad', 'Apple', 9.7, 64, None),
    Computer(4, 'Galaxy Tab 2 10.1', 'Samsung', 10.1, 32, 'microSD'),
    Computer(5, 'Excite 10', 'Toshiba', 10.1, 64, 'microSD'),
    Computer(6, 'Transformer Pad Infinity 700', 'Asus', 10.1, 64, 'microSD'),
    Computer(7, 'Icona Tab A700', 'Acer', 10.1, 64, 'microSD')
]
br2 = [
    Browser(1, 'Chrome', 'Google', 16, 1),
    Browser(2, 'Firefox', 'Mozilla', 13, 1),
    Browser(3, 'IE', 'Microsoft', 15, 2),
    Browser(4, 'Yandex', 'Yandex', 16, 3),
    Browser(5, 'Opera', 'Opera', 16, 5),
    Browser(6, 'Safari', 'Apple', 15, 3)
]
```

```
        self.assertEqual(num2(br2, cm2), [('Surface', '14.50'), ('Surface Pro', '14.67'),
('iPad', '15.00'),
                                          ('Galaxy Tab 2 10.1', '15.00'), ('Excite 10', '15.17'),
                                          ('Transformer Pad Infinity 700', '15.17'), ('Icona Tab
A700', '15.17')])


        cm3 = []
        self.assertEqual(num2(br2, cm3), [])


    def test_manytomany(self):
        cm1 = [
            Computer(2, 'Surface Pro', 'Microsoft', 10.6, 128, 'microSDXC'),
            Computer(3, 'iPad', 'Apple', 9.7, 64, None),
            Computer(4, 'Galaxy Tab 2 10.1', 'Samsung', 10.1, 32, 'microSD'),
            Computer(5, 'Excite 10', 'Toshiba', 10.1, 64, 'microSD'),
            Computer(6, 'Transformer Pad Infinity 700', 'Asus', 10.1, 64, 'microSD'),
            Computer(7, 'Icona Tab A700', 'Acer', 10.1, 64, 'microSD')
        ]
        br1 = [
            Browser(3, 'IE', 'Microsoft', 15, 2),
            Browser(4, 'Epic', 'Hidden Reflex', 16, 4),
            Browser(5, 'Vivaldi', 'Vivaldi Technologies', 16, 2),
            Browser(6, 'Yandex', 'Yandex', 16, 3),
            Browser(7, 'Opera', 'Opera', 16, 5),
            Browser(8, 'Safari', 'Apple', 15, 3)
        ]
        bc1 = [
            BrowComp(2, 5),
            BrowComp(3, 6),
```

```
            BrowComp(4, 4),
            BrowComp(4, 6),
            BrowComp(5, 6),
            BrowComp(5, 7),
            BrowComp(6, 6),
            BrowComp(6, 8),
            BrowComp(7, 7)
        ]
        self.assertEqual(manytomany(br1, cm1, bc1),
                         [('Surface Pro', 'Vivaldi'), ('iPad', 'Yandex'), ('Galaxy Tab 2 10.1',
'Epic'),
                          ('Galaxy Tab 2 10.1', 'Yandex'), ('Excite 10', 'Yandex'), ('Excite 10',
'Opera'),
                          ('Transformer Pad Infinity 700', 'Yandex'), ('Transformer Pad
Infinity 700', 'Safari'),
                          ('Icona Tab A700', 'Opera')])


        bc2 = [
            BrowComp(2, 5)
        ]
        self.assertEqual(manytomany(br1, cm1, bc2), [('Surface Pro', 'Vivaldi')])


        bc3 = []
        self.assertEqual(manytomany(br1, cm1, bc3), [])



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

Результат выполнения tdd-тестов:

Скриншот:



В текстовом виде:

Testing started at 21:19 ...

Launching unittests with arguments python -m unittest C:/Users/Ирина/PycharmProjects/PK2/tdd-tests.py in C:\Users\Ирина\PycharmProjects\PK2

[('Surface Pro', 'Vivaldi'), ('iPad', 'Yandex'), ('Galaxy Tab 2 10.1', 'Epic'), ('Galaxy Tab 2 10.1', 'Yandex'), ('Excite 10', 'Yandex'), ('Excite 10', 'Opera'), ('Transformer Pad Infinity 700', 'Yandex'), ('Transformer Pad Infinity 700', 'Safari'), ('Icona Tab A700', 'Opera')]

[('Surface Pro', 'Vivaldi')]

[]

Ran 3 tests in 0.010s

OK

Process finished with exit code 0