Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Рубежный контроль №2

по дисциплине «Базовые компоненты интернет-технологий»

Выполнила:
студентка группы ИУ5-33Б
Жамнова М. С.

Проверил:
Гапанюк Ю. Е.

2021 г.

Условия рубежного контроля №2 по курсу БКИТ

Рубежный контроль представляет собой разработку тестов на языке Python.

1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

Текст программы

main.py

```
# используется для сортировки
from operator import itemgetter


class Comp:
    """Компьютер"""
    def __init__(self, id, brand, price, disp_cls_id):
        self.id = id
        self.brand = brand
        self.price = price
        self.disp_cls_id = disp_cls_id


class Disp_cls:
    """Дисплейный класс"""
    def __init__(self, id, name):
        self.id = id
        self.name = name


class CompDisp_cls:
    """
    'Компьютеры дисплейного класса' для реализации
    связи многие-ко-многим
    """
    def __init__(self, disp_cls_id, comp_id):
        self.disp_cls_id = disp_cls_id
        self.comp_id = comp_id


# Дисплейные классы
disp_classes = [
    Disp_cls(1, 'А-класс'),
    Disp_cls(2, 'Б-класс'),
    Disp_cls(3, 'В-класс'),
    Disp_cls(11, 'Г-класс'),
    Disp_cls(22, 'Д-класс'),
    Disp_cls(33, 'Е-класс'),

]

# Компьютеры
comps = [
    Comp(1, 'HP', 1168390, 1),
    Comp(2, 'Asus', 55390, 1),
```

```
    Comp(3, 'ThinkPad',78940, 3),
    Comp(4, 'Acer', 97450, 2),
    Comp(5, 'Lenovo', 86980, 3),
]

# Компьютеры и Дисплейные классы для связи многие-ко-многим
comps_disp_classs = [
    CompDisp_cls(1, 1),
    CompDisp_cls(1, 2),
    CompDisp_cls(3, 3),
    CompDisp_cls(2, 4),
    CompDisp_cls(3, 5),

    CompDisp_cls(33, 1),
    CompDisp_cls(22, 2),
    CompDisp_cls(11, 3),
    CompDisp_cls(33, 4),
    CompDisp_cls(22, 5),
]

def first_task(one_to_many):
    task1 = []
    for brand, price, name in one_to_many:
        if brand[0] == "A":
            task1.append((brand, name))
    return task1


def second_task(one_to_many):
    task2_uns = []
    for c in disp_classes:
        # все компьютеры
        d_disp_cls = list(filter(lambda i: i[2] == c.name, one_to_many))
        if len(d_disp_cls) > 0:
            c_price = [price for _, price, _ in d_disp_cls]
            c_minPrice = min(c_price)
            task2_uns.append((c.name, c_minPrice))
    task2 = sorted(task2_uns, key=itemgetter(1))
    return task2


def third_task(many_to_many):
    task3_uns = []
    for brand, price, name in many_to_many:
        task3_uns.append((brand, name))

    task3 = list(sorted(task3_uns, key=itemgetter(0)))
    return task3


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

    # Соединение данных один-ко-многим
    one_to_many = [(c.brand, c.price, d.name)
                   for d in disp classes
                   for c in comps
                   if c.disp_cls_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.name, dc.disp_cls_id, dc.comp_id)
                         for d in disp_classes
                         for dc in comps_disp_classs
                         if d.id == dc.disp_cls_id]
```

```
    many_to_many = [(c.brand, c.price, disp_class_name)
                    for disp_class_name, disp_cls_id, comp_id in
many_to_many_temp
                    for c in comps if c.id == comp_id]

    print('\nЗадание 1 \n', first_task(one_to_many))
    print('\nЗадание 2 \n', second_task(one_to_many))
    print('\nЗадание 3 \n', third_task(one_to_many))

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

test_tdd.py

```
import unittest
import sys, os

sys.path.append(os.getcwd())
from main import *


class TddTest(unittest.TestCase):
    def test_first_task(self):
        one_to_many = [(c.brand, c.price, d.name)
                       for d in disp_classes
                       for c in comps
                       if c.disp_cls_id == d.id]
        self.assertEqual(first_task(one_to_many), [('Asus', 'А-класс'),
('Acer', 'Б-класс')])

    def test_2(self):
        one_to_many = [(c.brand, c.price, d.name)
                       for d in disp_classes
                       for c in comps
                       if c.disp_cls_id == d.id]
        self.assertEqual(second_task(one_to_many), [('А-класс', 55390), ('В-
класс', 78940), ('Б-класс', 97450)])

    def test_3(self):
        many_to_many_temp = [(d.name, dc.disp_cls_id, dc.comp_id)
                             for d in disp_classes
                             for dc in comps_disp_classs
                             if d.id == dc.disp_cls_id]
        many_to_many = [(c.brand, c.price, disp_class_name)
                        for disp_class_name, disp_cls_id, comp_id in
many_to_many_temp
                        for c in comps if c.id == comp_id]
        self.assertEqual(third_task(many_to_many),
                         [('Acer', 'Б-класс'), ('Acer', 'Е-класс'), ('Asus',
'А-класс'), ('Asus', 'Д-класс'),
                          ('HP', 'А-класс'), ('HP', 'Е-класс'), ('Lenovo',
'В-класс'), ('Lenovo', 'Д-класс'),
                          ('ThinkPad', 'В-класс'), ('ThinkPad', 'Г-класс')])
```

Скрины результата работы: