

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю №1
Вариант 20Д

Выполнил:
студент группы ИУ5-35Б
Шакиров Тимур
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич
Подпись и дата:

Постановка задачи

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

```
import sys
import random
import pprint

# Деталь
class detail:
    def __init__(self, id, name, price, provider_id):
        self.id = id
        self.name = name
        self.price = price
        self.provider_id = provider_id

# Поставщик
class provider:
    def __init__(self, id, name):
        self.id = id
        self.name = name

# Детали поставщика
class ProvDet:
    def __init__(self, provider_id, detail_id):
        self.provider_id = provider_id
        self.detail_id = detail_id

details = [
    detail(1, "Винт", 100, 1),
    detail(2, "Гайка", 300, 2),
    detail(3, "Шайба", 250, 3),
    detail(4, "Болт", 50, 4),
    detail(5, "Пружина", 65, 5),
    detail(6, "Плитка", 154, 6),
    detail(7, "Кабель", 39, 7),
    detail(8, "Датчик", 439, 8),
    detail(9, "Кнопка", 120, 9),
    detail(10, "Реле", 30, 10),
    detail(11, "Панель", 13, 11),
    detail(12, "Ручка", 34, 12),
    detail(13, "Штекер", 99, 13),
    detail(14, "Датчик движения", 129, 14),
    detail(15, "Мотор", 500, 15),
    detail(16, "Зубчатое колесо", 210, 16),
    detail(17, "Трубка", 59, 17),
    detail(18, "Пластик", 60, 18),
    detail(19, "Стекло", 70, 19),
    detail(20, "Металл", 80, 20),
```

```
detail(21, "Пластмасса", 90, 10),
detail(22, "Платина", 100, 9),
detail(23, "Батарейка", 90, 9),
detail(24, "Светодиод", 80, 8),
detail(25, "Жгут проводов", 70, 7),
detail(26, "Рамка", 60, 7),
detail(27, "Разъём", 50, 6),
detail(28, "Микроконтроллер", 304, 5),
detail(29, "Дисплей", 230, 5),
detail(30, "Слуховой аппарат", 27, 4),
detail(31, "Колесо", 85, 3),
detail(32, "Зеркало", 69, 3),
detail(33, "Клавиша", 30, 2),
detail(34, "Звезда", 31, 1),
detail(35, "Спичка", 5, 1),
detail(36, "Подшипник", 19, 20),
detail(37, "Транзистор", 20, 19),
detail(38, "Динамик", 30, 19),
detail(39, "Магнит", 34, 18),
detail(40, "Проводник", 39, 17),
detail(41, "Сирена", 96, 17),
detail(42, "Скрепка", 78, 16),
detail(43, "Шестерня", 127, 15),
detail(44, "Колпачок", 149, 15),
detail(45, "Трансформатор", 203, 14),
detail(46, "Зарядное устройство", 459, 13),
detail(47, "Пульт управления", 67, 13),
detail(48, "Антенна", 206, 12),
detail(49, "Светильник", 105, 11),
detail(50, "Вентилятор", 200, 11)
```

]

```
providers = [
    provider(1, "ООО 'ТехноКомпания'"),
    provider(2, "АО 'Индустрия'"),
    provider(3, "ЗАО 'Производство'"),
    provider(4, "ИП 'МастерТех'"),
    provider(5, "ОАО 'Прогресс'"),
    provider(6, "Техногрупп"),
    provider(7, "МастерМеханика"),
    provider(8, "Инновационные Технологии"),
    provider(9, "СпецПромТехника"),
    provider(10, "Автоэлектрика"),
    provider(11, "Производство и Сервис"),
    provider(12, "ТехноСревисГарант"),
    provider(13, "ПромХолдинг"),
    provider(14, "Инженерные Решения"),
    provider(15, "АгроТехника"),
    provider(16, "МедТехИнжиниринг"),
    provider(17, "ЭнергоСервис"),
    provider(18, "ТехноВидение"),
    provider(19, "СпецТехноГрупп"),
    provider(20, "ГоризонтПроизводство")
]
```

```

providers_details = [
    ProvDet(1, 2),
    ProvDet(2, 4),
    ProvDet(2, 10),
    ProvDet(3, 5),
    ProvDet(4, 1),
    ProvDet(5, 3),
    ProvDet(5, 7),
    ProvDet(5, 8),
    ProvDet(6, 11),
    ProvDet(7, 12),
    ProvDet(8, 14),
    ProvDet(9, 13),
    ProvDet(9, 20),
    ProvDet(9, 21),
    ProvDet(9, 32),
    ProvDet(10, 43),
    ProvDet(11, 15),
    ProvDet(12, 16),
    ProvDet(12, 27),
    ProvDet(13, 28),
    ProvDet(13, 29),
    ProvDet(14, 31),
    ProvDet(15, 33),
    ProvDet(16, 47),
    ProvDet(17, 42),
    ProvDet(18, 50),
    ProvDet(19, 19),
    ProvDet(19, 22),
    ProvDet(19, 23),
    ProvDet(20, 38)
]

def main():
    one_to_many = [(d.name, d.price, p.name)
                    for p in providers
                    for d in details
                    if d.provider_id == p.id]

    many_to_many_temp = [(p.name, pd.provider_id, pd.detail_id)
                          for p in providers
                          for pd in providers_details
                          if p.id == pd.provider_id]

    many_to_many = [(d.name, d.price, provider_name)
                     for provider_name, provider_id, detail_id in many_to_many_temp
                     for d in details if d.id == detail_id]

    print('Задание Д1') # детали, название которых заканчивается на "а" + их стоимость +
    поставщик
    filtered_details = [(name, price, provider)
                        for name, price, provider in one_to_many

```

```

        if name.endswith("a")]
    for i in filtered_details:
        print(i)

    print('\nЗадание Д2') # Список поставщиков со средней стоимостью деталей в каждом
отделе, отсортированный по средней стоимости
    provider_details = {}
    for name, price, provider_name in one_to_many:
        if provider_name not in provider_details:
            provider_details[provider_name] = {"total_price": 0, "num_details": 0}
        provider_details[provider_name]["total_price"] += price
        provider_details[provider_name]["num_details"] += 1
    average_prices = []
    for provider_name, data in provider_details.items():
        average_price = data["total_price"] / data["num_details"]
        average_prices.append((provider_name, average_price))
    sorted_average_prices = sorted(average_prices, key=lambda x: x[1], reverse=True)
    for i in sorted_average_prices:
        print(i)

    print('\nЗадание Д3')
    filtered_providers = {}
    for p in providers:
        if p.name.startswith("A"):
            p_details = list(filter(lambda i: i[2]==p.name, many_to_many))
            p_details_names = [x for x, _, _ in p_details]
            filtered_providers[p.name] = p_details_names
    pprint.pprint(filtered_providers)

if __name__ == '__main__':
    main()

```

test.py

```

from io import StringIO
import unittest
from unittest.mock import patch, mock_open
import main

class TestProgram(unittest.TestCase):

    @patch("builtins.open", mock_open(read_data="test data"))
    def test_filter_details_ending_with_a(self):
        expected_result = [
            ("Звезда", 31, "ООО 'ТехноКомпания'"),
            ("Спичка", 5, "ООО 'ТехноКомпания'"),
            ("Гайка", 300, "АО 'Индустрия'"),
            ("Клавиша", 30, "АО 'Индустрия'"),
            ("Шайба", 250, "ЗАО 'Производство'"),
            ("Пружина", 65, "ОАО 'Прогресс'"),
            ("Плитка", 154, "Техногрупп"),
            ("Рамка", 60, "МастерМеханика"),
            ("Кнопка", 120, "СпецПромТехника"),
            ("Платина", 100, "СпецПромТехника"),
            ("Батарейка", 90, "СпецПромТехника"),
            ("Пластмасса", 90, "Автоэлектрика"),
        ]

```

```

        ("Ручка", 34, "ТехноСревисГарант"),
        ("Антенна", 206, "ТехноСревисГарант"),
        ("Скрепка", 78, "МедТехИнжиниринг"),
        ("Трубка", 59, "ЭнергоСервис"),
        ("Сирена", 96, "ЭнергоСервис")]
    with patch("sys.stdout", new_callable=StringIO) as mock_stdout:
        main.main()
        actual_output = mock_stdout.getvalue()
        self.assertTrue(all(str(detail) in actual_output for detail in
expected_result))

@patch("builtins.open", mock_open(read_data="test data"))
def test_average_prices_by_provider(self):
    expected_result = [("Инновационные Технологии", 259.5),
        ("АгроТехника", 258.6666666666667),
        ("ПромХолдинг", 208.33333333333334),
        ("ОАО 'Прогресс'", 199.66666666666666),
        ("Инженерные Решения", 166.0),
        ("АО 'Индустрия'", 165.0),
        ("МедТехИнжиниринг", 144.0),
        ("ЗАО 'Производство'", 134.66666666666666),
        ("ТехноСревисГарант", 120.0),
        ("Производство и Сервис", 106.0),
        ("СпецПромТехника", 103.33333333333333),
        ("Техногрупп", 102.0),
        ("ЭнергоСервис", 64.66666666666667),
        ("Автоэлектрика", 60.0),
        ("МастерМеханика", 56.333333333333336),
        ("ГоризонтПроизводство", 49.5),
        ("ТехноВидение", 47.0),
        ("ООО 'ТехноКомпания'", 45.333333333333336),
        ("СпецТехноГрупп", 40.0),
        ("ИП 'МастерТех'", 38.5)]
    with patch("sys.stdout", new_callable=StringIO) as mock_stdout:
        main.main()
        actual_output = mock_stdout.getvalue()
        self.assertTrue(all(str(provider) in actual_output for provider in
expected_result))

@patch("builtins.open", mock_open(read_data="test data"))
def test_filter_providers_starting_with_a(self):
    expected_result = {"АО 'Индустрия'": ["Болт", "Реле"],
        "Автоэлектрика": ["Шестерня"],
        "АгроТехника": ["Клавиша"]}
    with patch("sys.stdout", new_callable=StringIO) as mock_stdout:
        main.main()
        actual_output = mock_stdout.getvalue()
        self.assertTrue(all(provider in actual_output for provider in
expected_result.keys()))
        self.assertTrue(all(all(detail in actual_output for detail in details) for
provider, details in expected_result.items()))

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

```

Анализ результатов

Задание Д1

```
('Звезда', 31, "ООО 'ТехноКомпания'")
('Спичка', 5, "ООО 'ТехноКомпания'")
('Гайка', 300, "АО 'Индустрия'")
('Клавиша', 30, "АО 'Индустрия'")
('Шайба', 250, "ЗАО 'Производство'")
('Пружина', 65, "ОАО 'Прогресс'")
('Плитка', 154, 'Техногрупп')
('Рамка', 60, 'МастерМеханика')
('Кнопка', 120, 'СпецПромТехника')
('Платина', 100, 'СпецПромТехника')
('Батарейка', 90, 'СпецПромТехника')
('Пластмасса', 90, 'Автоэлектрика')
('Ручка', 34, 'ТехноСревисГарант')
('Антенна', 206, 'ТехноСревисГарант')
('Скрепка', 78, 'МедТехИнжиниринг')
('Трубка', 59, 'ЭнергоСервис')
('Сирена', 96, 'ЭнергоСервис')
```

Задание Д2

```
('Инновационные Технологии', 259.5)
('АгроТехника', 258.6666666666667)
('ПромХолдинг', 208.33333333333334)
("ОАО 'Прогресс'", 199.66666666666666)
('Инженерные Решения', 166.0)
("АО 'Индустрия'", 165.0)
('МедТехИнжиниринг', 144.0)
("ЗАО 'Производство'", 134.66666666666666)
('ТехноСревисГарант', 120.0)
('Производство и Сервис', 106.0)
('СпецПромТехника', 103.33333333333333)
('Техногрупп', 102.0)
('ЭнергоСервис', 64.66666666666667)
('Автоэлектрика', 60.0)
('МастерМеханика', 56.333333333333336)
('ГоризонтПроизводство', 49.5)
('ТехноВидение', 47.0)
("ООО 'ТехноКомпания'", 45.333333333333336)
('СпецТехноГрупп', 40.0)
('ИП 'МастерТех'', 38.5)
```

Задание Д3

```
{"АО 'Индустрия'": ['Болт', 'Реле'],
 'Автоэлектрика': ['Шестерня'],
 'АгроТехника': ['Клавиша']}
```

...

Ran 3 tests in 0.004s

OK