

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

Факультет «Информатика и управление»
Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»
Отчет по рубежному контролю №2

Выполнил:

студент группы ИУ5-32Б

Казицин Алексей

Проверил:

преподаватель кафедры

Гапанюк Ю.Е.

Подпись и дата:

Подпись и дата:

Москва, 2022 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

```
from operator import itemgetter

class OS:
    def __init__(self, id, name, bit, interface, computer_id):
        self.id = id
        self.name = name
        self.bit = bit
        self.interface = interface
        self.computer_id = computer_id

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

class OSComputer:
    def __init__(self, computer_id, os_id):
        self.computer_id = computer_id
        self.os_id = os_id

computers = [
    Computer(1, 'Vasya1'),
    Computer(2, 'Anton-385-WH'),
    Computer(3, 'VladVlad123'),
    Computer(4, 'MoiComputer'),
    Computer(5, 'Netac3405'),
    Computer(6, 'JORDAN404-404')
]

OSs = [
    OS(1, 'Windows10', 64, 'Graphic', 2),
    OS(2, 'MacOS', 32, 'Graphic', 2),
    OS(3, 'Linux', 64, 'Graphic', 4),
```

```

OS(4, 'DOS', 16, 'Text', 4),
OS(5, 'WindowsXP', 32, 'Graphic', 5),
OS(6, 'UNIX', 64, 'CMD', 4),
OS(7, 'Windows7', 64, 'Graphic', 6),
]

OSs_computers = [
    OSComputer(2, 1),
    OSComputer(2, 2),
    OSComputer(4, 3),
    OSComputer(4, 4),
    OSComputer(5, 5),
    OSComputer(4, 6),
    OSComputer(6, 7),

    OSComputer(3, 1),
    OSComputer(3, 2),
    OSComputer(3, 3),
    OSComputer(1, 4),
    OSComputer(1, 5),
    OSComputer(1, 6),
    OSComputer(1, 7)
]

# Соединение данных один-ко-многим
one_to_many = [(o.name, o.bit, o.interface, c.name)
                for c in computers
                for o in OSs
                if o.computer_id == c.id]

# Соединение данных многие-ко-многим
many_to_many_temp = [(c.name, osc.computer_id, osc.os_id)
                      for c in computers
                      for osc in OSs_computers
                      if c.id == osc.computer_id]

many_to_many = [(o.name, o.bit, o.interface, c_name)
                 for c_name, c_id, os_id in many_to_many_temp
                 for o in OSs if o.id == os_id]

def task1(computers, OSs):
    return sorted(one_to_many, key=itemgetter(0))

def task2(computers, OSs):
    second_task = []

    for c in computers:
        c_data = {}
        c_oss = list(filter(lambda i: i[3] == c.name, one_to_many))
        c_data["Name"] = c.name
        c_data["Number of OSs"] = len(c_oss)
        second_task.append(c_data)
    return sorted(second_task, key=itemgetter("Number of OSs"), reverse=True)

def task3(computers, OSs, OSs_computers):
    third_task = []

    for o in OSs:
        o_data = {}
        if o.bit == 32:

```

```

        o_data["Name"] = o.name
        os_comp = [mtmt[0] for mtmt in many_to_many_temp if mtmt[1] ==
o.computer_id]
        o_data["Computer"] = list(set(os_comp))
        o_data["bit"] = 32
        third_task.append(o_data)
    return third_task

def main():
    print('Задание №1')
    print(
        "Выведите список всех связанных операционных систем и компьютеров,
отсортированный по операционным системам, сортировка по компьютерам
произвольная.")
    first_task = task1(computers, OSs)
    print(first_task)

    print('\nЗадание №2')
    print(
        "Выведите список компьютеров с количеством установленных операционных
систем на каждом компьютере, отсортированный по количеству операционных
систем.")
    second_task = task2(computers, OSs)
    for i in second_task:
        print(i)

    print('\nЗадание №3')
    print("Выведите список всех 32-битных операционных систем, и названия
компьютеров, на которых они установлены.")
    third_task = task3(computers, OSs, OSs_computers)

    for i in third_task:
        print(i, end="\n")

if __name__ == '__main__':
    main()

```

Tests.py

```

import unittest
from main import task1, task2, task3, OS, Computer, OSComputer

class TestOsComp(unittest.TestCase):
    def setUp(self):
        self.computers = [
            Computer(1, 'Vasya1'),
            Computer(2, 'Anton-385-WH'),
            Computer(3, 'VladVlad123'),
            Computer(4, 'MoiComputer'),
            Computer(5, 'Netac3405'),
            Computer(6, 'JORDAN404-404')
        ]

        self.OSs = [
            OS(1, 'Windows10', 64, 'Graphic', 2),
            OS(2, 'MacOS', 32, 'Graphic', 2),
            OS(3, 'Linux', 64, 'Graphic', 4),
            OS(4, 'DOS', 16, 'Text', 4),

```

```

        OS(5, 'WindowsXP', 32, 'Graphic', 5),
        OS(6, 'UNIX', 64, 'CMD', 4),
        OS(7, 'Windows7', 64, 'Graphic', 6),

    ]

    self.OSs_computers = [
        OSComputer(2, 1),
        OSComputer(2, 2),
        OSComputer(4, 3),
        OSComputer(4, 4),
        OSComputer(5, 5),
        OSComputer(4, 6),
        OSComputer(6, 7),

        OSComputer(3, 1),
        OSComputer(3, 2),
        OSComputer(3, 3),
        OSComputer(1, 4),
        OSComputer(1, 5),
        OSComputer(1, 6),
        OSComputer(1, 7)
    ]

    def test_task1(self):
        res = task1(self.computers, self.OSs)
        self.assertEqual(res, [('DOS', 16, 'Text', 'MoiComputer'),
                                ('Linux', 64, 'Graphic', 'MoiComputer'),
                                ('MacOS', 32, 'Graphic', 'Anton-385-WH'),
                                ('UNIX', 64, 'CMD', 'MoiComputer'),
                                ('Windows10', 64, 'Graphic', 'Anton-385-WH'),
                                ('Windows7', 64, 'Graphic', 'JORDAN404-404'),
                                ('WindowsXP', 32, 'Graphic', 'Netac3405')])

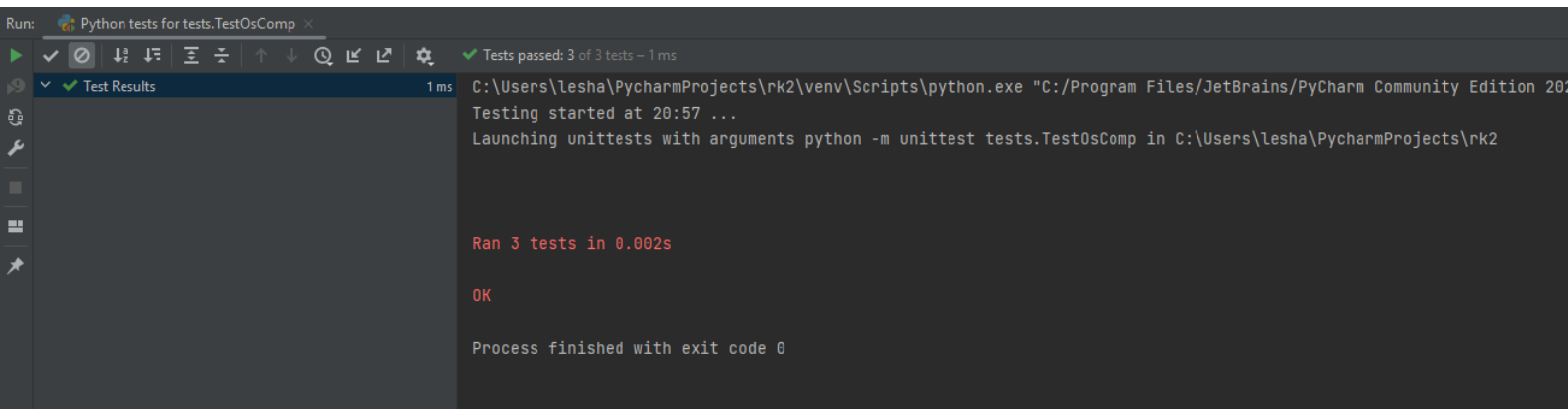
    def test_task2(self):
        res = task2(self.computers, self.OSs)
        self.assertEqual(res, [{'Name': 'MoiComputer', 'Number of OSs': 3},
                                {'Name': 'Anton-385-WH', 'Number of OSs': 2},
                                {'Name': 'Netac3405', 'Number of OSs': 1},
                                {'Name': 'JORDAN404-404', 'Number of OSs': 1},
                                {'Name': 'Vasya1', 'Number of OSs': 0},
                                {'Name': 'VladVlad123', 'Number of OSs': 0}])

    def test_task3(self):
        res = task3(self.computers, self.OSs, self.OSs_computers)
        self.assertEqual(res, [
            {'Name': 'MacOS', 'Computer': ['Anton-385-WH'], 'bit': 32},
            {'Name': 'WindowsXP', 'Computer': ['Netac3405'], 'bit': 32}])

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

```

Результат работы программы



The screenshot shows the PyCharm Run window for a test named 'tests.TestOsComp'. The window has a toolbar with icons for running, debugging, and other actions. The status bar at the top indicates 'Tests passed: 3 of 3 tests - 1 ms'. The main area is divided into two panes: 'Test Results' on the left and a console output on the right. The 'Test Results' pane shows a green checkmark and the text 'Test Results' with a duration of '1 ms'. The console output on the right shows the command used to launch the tests and the results.

```
Run: Python tests for tests.TestOsComp x
✓ Tests passed: 3 of 3 tests - 1 ms
Test Results 1 ms
C:\Users\lesha\PycharmProjects\rk2\venv\Scripts\python.exe "C:/Program Files/JetBrains/PyCharm Community Edition 202
Testing started at 20:57 ...
Launching unittests with arguments python -m unittest tests.TestOsComp in C:\Users\lesha\PycharmProjects\rk2

Ran 3 tests in 0.002s

OK

Process finished with exit code 0
```