

Московский государственный технический университет им.  
Н.Э. Баумана

Факультет “Радиотехнический”  
Кафедра ИУ5 “Системы обработки информации и управления”

Отчет по РК2 по курсу  
**Базовые компоненты интернет технологий**

Вариант 8

Подготовил:  
Студент группы РТ5-31Б  
Коваленко В.И.

Проверил:  
Доцент кафедры ИУ5  
Гапанюк Ю.Е.

12 Декабря 2021г.

## 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

#### RK2\_BKIT.py

```
from operator import itemgetter
```

```
class HDD:
```

```
    """Жесткий диск"""
```

```
    def __init__(self, id, name_hdd, cap, pc_id):
```

```
        self.id = id
```

```
        self.name_hdd = name_hdd # имя
```

```
        self.cap = cap # вместимость диска (Гигабайт)
```

```
        self.pc_id = pc_id # в каком компьютере
```

```
class PC:
```

```
    """Компьютер"""
```

```
    def __init__(self, id, name):
```

```
        self.id = id
```

```
        self.name = name
```

```
class HddPc:
```

```
    """
```

```
    СВЯЗЬ МНОГИЕ КО МНОГИМ
```

```
    """
```

```
    def __init__(self, pc_id, hdd_id):
```

```
        self.pc_id = pc_id
```

```
self.hdd_id = hdd_id
```

```
# Компьютеры
```

```
Pcs = [  
    PC(1, 'компьютер HP'),  
    PC(2, 'компьютер MSI'),  
    PC(3, 'MacBook Pro'),  
    PC(4, 'компьютер Alienware'),  
    PC(5, 'компьютер Asus'),  
    PC(6, 'MacBook Air'),  
]
```

```
# Жесткие диски
```

```
hdds = [  
    HDD(1, 'Seagate', 5120, 2),  
    HDD(2, 'Samsung', 2048, 1),  
    HDD(3, 'Macintosh', 1024, 3),  
    HDD(4, 'Toshiba', 3072, 4),  
    HDD(5, 'WD blue', 4096, 5),  
    HDD(6, 'WD gold', 2048, 1),  
    HDD(7, 'Macintosh', 512, 6),  
    HDD(8, 'Seagate Baracuda', 4096, 4),  
    HDD(9, 'Toshiba', 4096, 5)  
]
```

```
hdds_pcs = [  
    HddPc(1,1),  
    HddPc(1,2),  
    HddPc(1,4),  
    HddPc(1,8),
```

```
HddPc(2,4),
HddPc(2,5),
HddPc(2,6),
HddPc(2,1),
HddPc(3,3),
HddPc(4,1),
HddPc(4,5),
HddPc(5,4),
HddPc(5,2),
HddPc(6,7),

]
```

```
# Соединение данных один-ко-многим
one_to_many = [(h.name_hdd, h.cap, p.name)
                for p in Pcs
                for h in hdds
                if h.pc_id==p.id]

# Соединение данных многие-ко-многим
many_to_many_temp = [(p.name, ph.pc_id, ph.hdd_id)
                      for p in Pcs
                      for ph in hdds_pcs
                      if p.id==ph.pc_id]

many_to_many = [(h.name_hdd, h.cap, pc_name)
                 for pc_name, pc_id, hdd_id in many_to_many_temp
                 for h in hdds if h.id==hdd_id]

def E1():
```

```

print('Задание E1')
res_E1 = []
for name_hdd, cap, name in one_to_many:
    if 'компьютер' in name: # Ищем компьютеры с ключевым словом
        "компьютер"
        res_E1.append((name, name_hdd))
return res_E1

```

```

def E2():
    print('\nЗадание E2')
    # находим среднюю вместимость жестких дисков
    res_E2_unsorted = []
    # Перебираем все компьютеры
    for p in Pcs:
        # Список жестких дисков компьютера
        list_hdd = list(filter(lambda i: i[2]==p.name, one_to_many))
        # Если в компьютере есть жесткий диск
        if len(list_hdd) > 0:
            # вместимость HDD
            list_cap = [cap for _,cap,_ in list_hdd]
            # средняя вместимость
            avg_sum = sum(list_cap)/len(list_cap)
            res_E2_unsorted.append((p.name, avg_sum))
    res_E2 = sorted(res_E2_unsorted, key=itemgetter(1))
    return res_E2

```

```

def E3():
    print('\nЗадание E3')
    # находим жесткие диски, начинающиеся с "S" и выводим их компьютеры
    res_E3 = []

```

```
for name_hdd, cap, name in many_to_many:
    if name_hdd.find("S") == 0:
        res_E3.append((name_hdd, name))
return res_E3
```

### **module\_test.py**

```
import unittest
```

```
import RK2_BKIT
```

```
class RK_Test(unittest.TestCase):
```

```
    def test_E1(self):
```

```
        self.assertEqual(RK2_BKIT.E1(),
                           [('компьютер HP', 'Samsung'), ('компьютер HP', 'WD gold'),
                            ('компьютер MSI', 'Seagate'), ('компьютер Alienware', 'Toshiba'),
                            ('компьютер Alienware', 'Seagate Baracuda'), ('компьютер Asus',
                            'WD blue'), ('компьютер Asus', 'Toshiba')])
```

```
    def test_E2(self):
```

```
        self.assertEqual(RK2_BKIT.E2(),
                           [('MacBook Air', 512.0), ('MacBook Pro', 1024.0), ('компьютер HP',
                           2048.0), ('компьютер Alienware', 3584.0), ('компьютер Asus',
                           4096.0), ('компьютер MSI', 5120.0)])
```

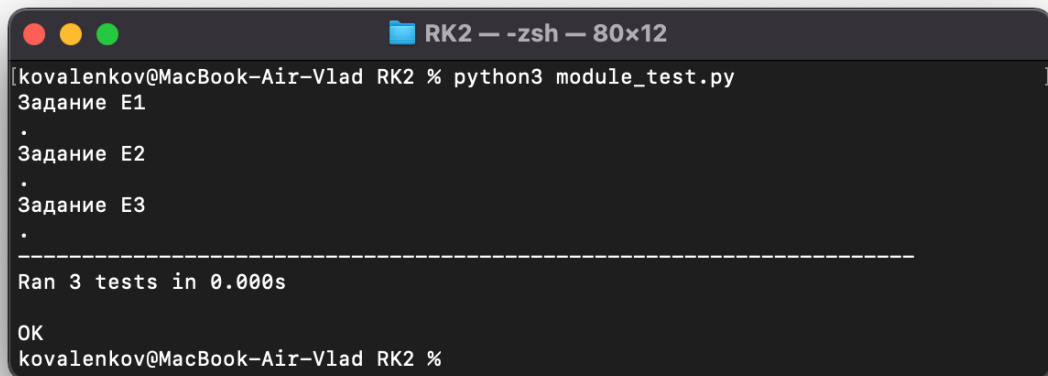
```
    def test_E3(self):
```

```
        self.assertEqual(RK2_BKIT.E3(),
                           [('Seagate', 'компьютер HP'), ('Samsung', 'компьютер HP'), ('Seagate
                           Baracuda', 'компьютер HP'), ('Seagate', 'компьютер MSI'), ('Seagate',
                           'компьютер Alienware'), ('Samsung', 'компьютер Asus')])
```

```
if __name__ == '__main__':
```

```
    unittest.main()
```

## Результат выполнения программы

A terminal window with a dark background and light gray text. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, followed by a folder icon and the text 'RK2 — -zsh — 80x12'. The terminal content shows a command prompt where a Python script was executed. The output consists of three lines, each starting with 'Задание' followed by 'E1', 'E2', and 'E3' respectively, each with a period on the next line. A dashed line separates this from the summary 'Ran 3 tests in 0.000s'. The prompt ends with 'OK' and the user's name and host information.

```
[kovalenkov@MacBook-Air-Vlad RK2 % python3 module_test.py  
Задание E1  
.  
Задание E2  
.  
Задание E3  
.  
-----  
Ran 3 tests in 0.000s  
  
OK  
kovalenkov@MacBook-Air-Vlad RK2 %
```