

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

Факультет «Радиотехнический»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по рубежному контролю № 1

Вариант E19

Выполнил:

студент группы РТ5-51Б
Сысоев Александр

Проверил:

доцент каф. ИУ5
Гапанюк Ю. Е.

Москва, 2021 г.

ПОЛУЧЕННОЕ ЗАДАНИЕ

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

- 1) Необходимо создать два класса данных в соответствии с вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.
- 2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
- 3) Необходимо разработать запросы в соответствии с вариантом. При разработке запросов необходимо по возможности использовать функциональные возможности языка Python (list/dict comprehensions, функции высших порядков). Для реализации запроса №2 необходимо ввести в класс, находящийся на стороне связи «много», произвольный количественный признак.

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результаты ее выполнения.

ИСХОДНЫЕ ДАННЫЕ

Предметная область:

Класс 1	Класс 2
Деталь	Производитель

Запросы:

1. «Производитель» и «Деталь» связаны соотношением один-ко-многим. Выведите список всех производителей, у которых в наименовании присутствует слово «производитель», и список изготовленных ими деталей.
2. «Производитель» и «Деталь» связаны соотношением один-ко-многим. Выведите список производителей со средней ценой деталей, изготовленных каждым производителем и отсортированный по средней цене. Средняя цена детали должна быть округлена до 2 знака после запятой.
3. «Производитель» и «Деталь» связаны соотношением многие-ко-многим. Выведите список всех деталей, у которых название начинается с буквы «Г», и наименования их производителей.

ИСХОДНЫЙ КОД

```
from operator import itemgetter

#Класс "Производитель"
class Manufacturer:
    def __init__(self, id, name):
        self.id = id
        self.name = name

#Класс "Деталь"
class Detail:
    def __init__(self, id, name, cost, man_id):
        self.id = id
        self.name = name
        self.cost = cost
        self.man_id = man_id

#Класс "Детали производителя"
class DetMan:
    def __init__(self, man_id, det_id):
```

```
self.man_id = man_id
self.det_id = det_id
```

#Производители

```
manufacturers = [
    Manufacturer(1, 'Производитель №1'),
    Manufacturer(2, 'Производитель №2'),
    Manufacturer(3, 'Иванов'),
    Manufacturer(4, 'Петров'),
    Manufacturer(5, 'Сидоров')
]
```

#Детали

```
details = [
    Detail(1, 'Гайка', 19.99, 1),
    Detail(2, 'Шуруп', 49.99, 2),
    Detail(3, 'Гвоздь', 9.99, 3),
    Detail(4, 'Винт', 39.99, 1),
    Detail(5, 'Штуцер', 99.99, 3),
    Detail(6, 'Болт', 59.99, 3)
]
```

#Детали производителей

```
detmans = [
    DetMan(1, 1),
    DetMan(1, 2),
    DetMan(2, 3),
    DetMan(3, 3),
    DetMan(3, 5),
    DetMan(4, 1),
    DetMan(5, 3),
    DetMan(5, 4),
    DetMan(5, 6)
]
```

#Основная функция

```
def main():
    #ОДИН-КО-МНОГИМ
    one_to_many = [(d.name, d.cost, m.name)
                    for m in manufacturers
                    for d in details
                    if d.man_id == m.id]

    #МНОГИЕ-КО-МНОГИМ
```

```

many_to_many_temp = [(m.name, dm.man_id, dm.det_id)
                      for m in manufacturers
                      for dm in detmans
                      if m.id == dm.man_id]
many_to_many = [(d.name, d.cost, man_name)
                 for man_name, man_id, det_id in
many_to_many_temp
                 for d in details
                 if d.id == det_id]

#E1
print('Задание E1:')
res_11 = {}
for m in manufacturers:
    if 'производитель' in m.name.lower():
        m_d = [item[0]
                 for item in one_to_many
                 if item[2] == m.name]
        res_11[m.name] = m_d
print(res_11)

#E2
print('\nЗадание E2:')
res_12 = []
for m in manufacturers:
    m_d = list(filter(lambda i: i[2] == m.name,
one_to_many))
    if len(m_d) > 0:
        m_cost = [cost
                   for _, cost, _ in m_d]
        m_cost_avg = sum(m_cost) / len(m_d)
        res_12.append((m.name, round(m_cost_avg, 2)))
print(sorted(res_12, key = itemgetter(1)))

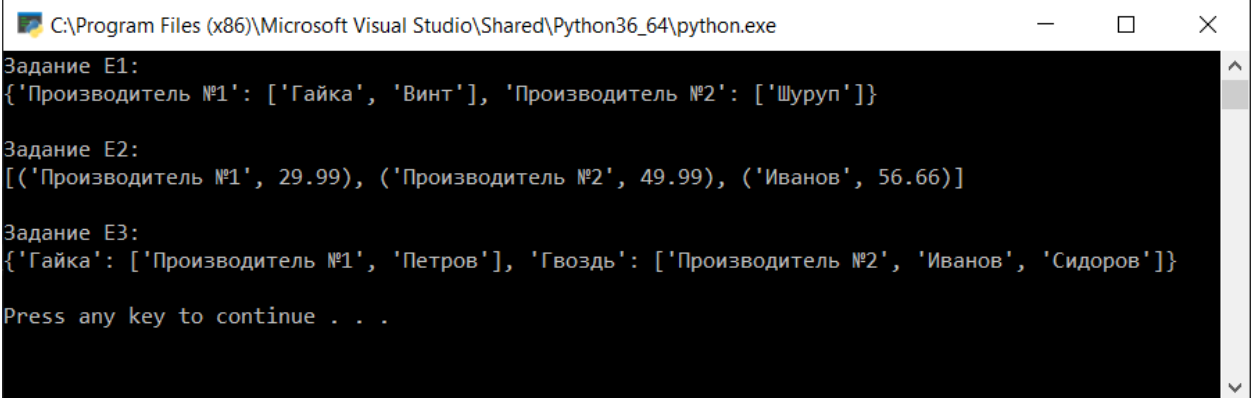
#E3
print('\nЗадание E3:')
res_13 = {}
for d in details:
    if d.name.lower()[0] == 'r':
        d_m = [item[2]
                 for item in many_to_many
                 if d.name == item[0]]
        res_13[d.name] = d_m
print(res_13)

```

```
print()

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

РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

A screenshot of a Python console window titled "C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe". The window has a black background with white text. The output shows three tasks: "Задание E1:" with a dictionary of manufacturers, "Задание E2:" with a list of tuples, and "Задание E3:" with a dictionary of products and their manufacturers. The window ends with the prompt "Press any key to continue . . .".

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe
Задание E1:
{'Производитель №1': ['Гайка', 'Винт'], 'Производитель №2': ['Шуруп']}

Задание E2:
[('Производитель №1', 29.99), ('Производитель №2', 49.99), ('Иванов', 56.66)]

Задание E3:
{'Гайка': ['Производитель №1', 'Петров'], 'Гвоздь': ['Производитель №2', 'Иванов', 'Сидоров']}

Press any key to continue . . .
```