**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Вычислительные средства АСОИУ»

Курс «Разработка интернет-приложений»

Отчет по Рубежному контролю № 1

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-51Б |  | преподаватель каф. ИУ5 |
| Андреев А. В. |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

**Вариант 3 А**

1. «Автопарк» и «Водитель» связаны соотношением один-ко-многим. Выведите список всех связанных Водителей и Автопарков, отсортированный по Автопаркам, сортировка по Водителям произвольная.

2. «Автопарк» и «Водитель» связаны соотношением один-ко-многим. Выведите список Автопарков с суммарной зарплатой Водителей в каждом Автопарке, отсортированный по суммарной зарплате.

3. «Автопарк» и «Водитель» связаны соотношением многие-ко-многим. Выведите список всех Автопарков, у которых в названии присутствует слово «Автопарк», и список работающих в них Водителей.

**Текст программы**

rom operator import itemgetter  
  
  
class Driver:  
 *"""Водитель"""* def \_\_init\_\_(self, id, fio, sal, Avtopark\_id):  
 self.id = id  
 self.fio = fio  
 self.sal = sal  
 self.Avtopark\_id = Avtopark\_id  
  
  
class Avtopark:  
 *"""Автопарк"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
class DriverAvtopark:  
 *"""Связи многие-ко-многим"""* def \_\_init\_\_(self, Avtopark\_id, Driver\_id):  
 self.Avtopark\_id = Avtopark\_id  
 self.Driver\_id = Driver\_id  
  
  
# Автопарк  
Avtoparks = [  
 Avtopark(1, 'Автопарк Лада'),  
 Avtopark(2, 'Автопарк Ford'),  
 Avtopark(3, 'Автопарк BMW'),  
]  
  
# Водитель  
Drivers = [  
 Driver(1, 'Андреев', 60000, 3),  
 Driver(2, 'Пахомкин', 35000, 1),  
 Driver(3, 'Толпаров', 45000, 2),  
 Driver(4, 'Шевчук', 35000, 1),  
 Driver(5, 'Рогозин', 45000, 2),  
]  
  
Drivers\_Avtoparks = [  
 DriverAvtopark(1, 2),  
 DriverAvtopark(2, 3),  
 DriverAvtopark(1, 4),  
 DriverAvtopark(2, 5),  
 DriverAvtopark(3, 1),  
]  
  
  
def main():  
 *"""Основная функция"""* # Соединение данных один-ко-многим  
 one\_to\_many = [(e.fio, e.sal, d.name)  
 for d in Avtoparks  
 for e in Drivers  
 if e.Avtopark\_id == d.id]  
  
 # Соединение данных многие-ко-многим  
 many\_to\_many\_tDriver = [(d.name, ed.Avtopark\_id, ed.Driver\_id)  
 for d in Avtoparks  
 for ed in Drivers\_Avtoparks  
 if d.id == ed.Avtopark\_id]  
  
 many\_to\_many = [(e.fio, e.sal, Avtopark\_name)  
 for Avtopark\_name, Avtopark\_id, Driver\_id in many\_to\_many\_tDriver  
 for e in Drivers if e.id == Driver\_id]  
  
 print('Задание А1')  
 res\_11 = sorted(one\_to\_many, key=itemgetter(2))  
 print(res\_11)  
  
 print('\nЗадание А2')  
 res\_12\_unsorted = []  
 for d in Avtoparks:  
 d\_Drivers = list(filter(lambda i: i[2] == d.name, one\_to\_many))  
 if len(d\_Drivers) > 0:  
 d\_sals = [sal for \_, sal, \_ in d\_Drivers]  
 d\_sals\_sum = sum(d\_sals)  
 res\_12\_unsorted.append((d.name, d\_sals\_sum))  
  
 res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)  
 print(res\_12)  
  
 print('\nЗадание А3')  
 res\_13 = {}  
 for d in Avtoparks:  
 if 'Автопарк' in d.name:  
 d\_Drivers = list(filter(lambda i: i[2] == d.name, many\_to\_many))  
 d\_Drivers\_names = [x for x, \_, \_ in d\_Drivers]  
 res\_13[d.name] = d\_Drivers\_names  
  
 print(res\_13)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Результат работы программы**

Задание А1

[('Андреев', 60000, 'Автопарк BMW'), ('Толпаров', 45000, 'Автопарк Ford'), ('Рогозин', 45000, 'Автопарк Ford'), ('Пахомкин', 35000, 'Автопарк Лада'), ('Шевчук', 35000, 'Автопарк Лада')]

Задание А2

[('Автопарк Ford', 90000), ('Автопарк Лада', 70000), ('Автопарк BMW', 60000)]

Задание А3

{'Автопарк Лада': ['Пахомкин', 'Шевчук'], 'Автопарк Ford': ['Толпаров', 'Рогозин'], 'Автопарк BMW': ['Андреев']}