

## РК1, Вариант 17, вариант запроса В

## Код

#17 В

```
from operator import itemgetter
```

```
class Conductor:
```

```
    def __init__(self, id, fio, salary, orchestra_id):
```

```
        self.id = id
```

```
        self.fio = fio
```

```
        self.salary = salary
```

```
        self.orchestra_id = orchestra_id
```

```
class Orchestra:
```

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

```
        self.id = id
```

```
        self.name = name
```

```
class ConductorOrchestra:
```

```
    def __init__(self, orchestra_id, conductor_id):
```

```
        self.orchestra_id = orchestra_id
```

```
        self.conductor_id = conductor_id
```

```
orchestras = [
```

```
    Orchestra(1, "Симфонический оркестр"),
```

```
    Orchestra(2, "Филармонический оркестр"),
```

```
    Orchestra(3, "Камерный оркестр"),
```

```
    Orchestra(4, "Академический оркестр"),
```

```
]
```

```
conductors = [
```

```
    Conductor(1, "Антонов", 50000, 1),
```

```
    Conductor(2, "Петров", 60000, 2),
```

```
    Conductor(3, "Алексеев", 55000, 1),
```

```
    Conductor(4, "Сидоров", 70000, 3),
```

```
    Conductor(5, "Андреев", 45000, 2),
```

```
]
```

```
conductors_orchestras = [
```

```
    ConductorOrchestra(1, 1),
```

```
    ConductorOrchestra(2, 2),
```

```
    ConductorOrchestra(1, 3),
```

```
    ConductorOrchestra(3, 4),
```

```
    ConductorOrchestra(2, 5),
```

```
    ConductorOrchestra(4, 1),
```

```
ConductorOrchestra(4, 4),  
]
```

```
def main():
```

```
    one_to_many = [(c.fio, c.salary, o.name)  
                    for o in orchestras  
                    for c in conductors  
                    if c.orchestra_id == o.id]
```

```
    many_to_many_temp = [(o.name, co.orchestra_id, co.conductor_id)  
                           for o in orchestras  
                           for co in conductors_orchestras  
                           if o.id == co.orchestra_id]
```

```
    many_to_many = [(c.fio, c.salary, orchestra_name)  
                     for orchestra_name, orchestra_id, conductor_id in many_to_many_temp  
                     for c in conductors if c.id == conductor_id]
```

```
#Фильтрация дирижеров с фамилией на «А» из связи один-ко-многим  
print("Задание B1:")  
res_1 = list(filter(lambda i: i[0].startswith('A'), one_to_many))  
print(res_1)
```

```
#Расчет минимальной зарплаты для каждого оркестра и сортировка  
print("\nЗадание B2:")  
res_2_unsorted = []  
for o in orchestras:
```

```
    o_salaries = list(filter(lambda i: i[2] == o.name, one_to_many))  
    if len(o_salaries) > 0:  
        min_salary = min([sal for _, sal, _ in o_salaries])  
        res_2_unsorted.append((o.name, min_salary))  
res_2 = sorted(res_2_unsorted, key=itemgetter(1))  
print(res_2)
```

```
#Сортировка всех связей многие-ко-многим по фамилиям дирижеров.  
print("\nЗадание B3:")  
res_3 = sorted(many_to_many, key=itemgetter(0))  
print(res_3)
```

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

## Результат выполнения

```
C:\Users\varva\PycharmProjects\rk1\.venv\Scripts\python.exe C:\Users\varva\PycharmProjects\rk1\rk1.py
Задание B1:
[('Антонов', 50000, 'Симфонический оркестр'), ('Алексеев', 55000, 'Симфонический оркестр'), ('Андреев', 45000, 'Филармонический оркестр')]

Задание B2:
[('Филармонический оркестр', 45000), ('Симфонический оркестр', 50000), ('Камерный оркестр', 70000)]

Задание B3:
[('Алексеев', 55000, 'Симфонический оркестр'), ('Андреев', 45000, 'Филармонический оркестр'), ('Антонов', 50000, 'Симфонический оркестр'), ('Антонов', 50000, 'Академический оркестр'), ('Петров', 60000, 'Филармонический оркестр'), ('Сидоров', 70000, 'Камерный оркестр'), ('Сидоров', 70000, 'Академический оркестр')]
```