

Код Программы

```
from typing import List, Tuple

class Orchestra:
    def __init__(self, id: int, name: str):
        self.id = id
        self.name = name

    def __repr__(self):
        return f"Orchestra(id={self.id}, name={self.name!r})"

class Musician:
    def __init__(self, id: int, surname: str, salary: int, orchestra_id: int):
        self.id = id
        self.surname = surname
        self.salary = salary
        self.orchestra_id = orchestra_id

    def __repr__(self):
        return (f"Musician(id={self.id}, surname={self.surname!r}, "
                f"salary={self.salary}, orchestra_id={self.orchestra_id})")

class MusicianInOrchestra:
    def __init__(self, musician_id: int, orchestra_id: int):
        self.musician_id = musician_id
        self.orchestra_id = orchestra_id

    def __repr__(self):
        return f"MusicianInOrchestra(musician_id={self.musician_id}, orchestra_id={self.orchestra_id})"

orchestras: List[Orchestra] = [
    Orchestra(1, "Симфонический оркестр"),
    Orchestra(2, "Джазовый оркестр"),
    Orchestra(3, "Камерный оркестр"),
]

musicians: List[Musician] = [
    Musician(1, "Абрамов", 50000, 1),
    Musician(2, "Антонов", 60000, 1),
    Musician(3, "Борисов", 55000, 2),
    Musician(4, "Алексеев", 45000, 3),
    Musician(5, "Смирнов", 70000, 3),
]

musicians_in_orchestras: List[MusicianInOrchestra] = [
    MusicianInOrchestra(1, 1),
    MusicianInOrchestra(2, 1),
    MusicianInOrchestra(3, 2),
]
```

```

    MusicianInOrchestra(4, 3),
    MusicianInOrchestra(5, 3),
    MusicianInOrchestra(1, 3),
]
orchestra_by_id = {o.id: o for o in orchestras}
musician_by_id = {m.id: m for m in musicians}
def query_1_musicians_starting_with_a(musicians: List[Musician]) -> List[Tuple[str, str]]:
    pairs = [
        (m.surname, orchestra_by_id[m.orchestra_id].name)
        for m in musicians
        if m.surname.upper().startswith("A")
    ]
    return sorted(pairs, key=lambda x: (x[0], x[1]))

def query_2_min_salary_per_orchestra(musicians: List[Musician]) -> List[Tuple[str, int]]:
    groups = {}
    for m in musicians:
        groups.setdefault(m.orchestra_id, []).append(m.salary)

    agg = [(orchestra_by_id[oid].name, min(salaries)) for oid, salaries in groups.items()]
    return sorted(agg, key=lambda x: x[1])

def query_3_all_musician_orchestra_m2m(links: List[MusicianInOrchestra]) -> List[Tuple[str, str]]:
    pairs = [(musician_by_id[link.musician_id].surname, orchestra_by_id[link.orchestra_id].name) for link in links]
    return sorted(pairs, key=lambda x: x[0])

def main():
    q1 = query_1_musicians_starting_with_a(musicians)
    q2 = query_2_min_salary_per_orchestra(musicians)
    q3 = query_3_all_musician_orchestra_m2m(musicians_in_orchestras)
    print("Задание B1")
    for surname, orchestra_name in q1:
        print(f"    - {surname} – {orchestra_name}")
    print("Задание B2")
    for orchestra_name, min_salary in q2:
        print(f"    - {orchestra_name}: {min_salary} руб.")
    print("Задание B3")
    for surname, orchestra_name in q3:
        print(f"    - {surname} – {orchestra_name}")
if __name__ == "__main__":
    main()

```

Результаты выполнения

Задание B1

- Абрамов – Симфонический оркестр
- Алексеев – Камерный оркестр
- Антонов – Симфонический оркестр

Задание B2

- Камерный оркестр: 45000 руб.
- Симфонический оркестр: 50000 руб.
- Джазовый оркестр: 55000 руб.

Задание B3

- Абрамов – Симфонический оркестр
- Абрамов – Камерный оркестр
- Алексеев – Камерный оркестр
- Антонов – Симфонический оркестр
- Борисов – Джазовый оркестр
- Смирнов – Камерный оркестр