Текст программы (Вариант 4Б).

```python
from operator import itemgetter
from collections import Counter
from typing import Dict, List


class Computer:
    """ Класс компьютер"""

    def __init__(self, id_: int, name: str, cost: int, display_id: int):
        self.id = id_
        self.name = name
        self.cost = cost
        self.display_id = display_id


class Display:
    """Дисплейный класс"""

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


class ComputerDisplay:
    """
    'Компьютеры дисплейного класса' для реализации
    связи многие-ко-многим
    """

    def __init__(self, computer_id: int, display_id: int):
        self.computer_id = computer_id
        self.display_id = display_id


displays = [
    Display(1, 'Первый дисплей'),
    Display(2, 'Второй дисплей'),
    Display(3, 'Третий дисплей'),
]

computers = [
    Computer(1, 'Admin', 100000, 1),
    Computer(2, 'Hero', 60000, 1),
    Computer(3, 'CoolComputer', 75000, 2),
    Computer(4, 'MasterOfComputers', 80000, 2),
    Computer(5, 'PusherForever',167000, 2),
]

computerDisplays = [
    ComputerDisplay(1, 1),
    ComputerDisplay(1, 3),

    ComputerDisplay(2, 1),
```

```
    ComputerDisplay(2, 3),

    ComputerDisplay(3, 2),

    ComputerDisplay(4, 2),
    ComputerDisplay(4, 3),

    ComputerDisplay(5, 2),
]


def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    oneToMany = [(computer.name, computer.cost, display.name)
                 for display in displays
                 for computer in computers
                 if computer.display_id == display.id]

    # Соединение данных многие-ко-многим
    manyToManyTemp = [(display.name, computerDisplay.display_id,
computerDisplay.computer_id)
                      for display in displays
                      for computerDisplay in computerDisplays
                      if display.id == computerDisplay.display_id]

    manyToMany = [(computer.id, computer.name, computer.cost, display_name)
                  for display_name, display_id, computer_id in manyToManyTemp
                  for computer in computers if computer.id == computer_id]

    # «Дисплейный класс» и «Компьютер» связаны соотношением один-ко-многим.
    # Выведите список всех связанных компьютеров и дисплеев, отсортированный по
названию компьютера,
    # сортировка по дисплеям произвольная.
    print('Задание Б1')
    resultA = sorted(oneToMany, key=itemgetter(0))
    print(resultA)

    # «Дисплей» и «Компьютер» связаны соотношением один-ко-многим.
    # Выведите список дисплеев с количеством комьютеров в каждом дисплее,
отсортированный по количеству компьютеров.
    print('\nЗадание Б2')
    displaysCounter = Counter([computer.display_id for computer in computers])
    resultBUnsorted = [(d.name, displaysCounter[d.id]) for d in displays]
    print(sorted(resultBUnsorted, key=itemgetter(1), reverse=True))

    # «Дисплей» и «Компьютер связаны соотношением многие-ко-многим.
    # Выведите список всех компьютеров, у которых название заканчивается на «r», и
названия их дисплеев.
    print('\nЗадание Б3')
    resultC: Dict[str, List[str]] = {}
    for computer in computers:
        if not computer.name.endswith('r'):
```

```
            continue
        displaysWithComputer = [a[3] for a in filter(lambda i: i[0] == computer.id,
manyToMany)]
        resultC[computer.name] = displaysWithComputer
    print(resultC)


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

Результаты выполнения:

Задание Б1

[('Admin', 100000, 'Первый дисплей'), ('CoolComputer', 75000, 'Второй дисплей'), ('Hero', 60000, 'Первый дисплей'), ('MasterOfComputers', 80000, 'Второй дисплей'), ('PusherForever', 167000, 'Второй дисплей')]

Задание Б2

[('Второй дисплей', 3), ('Первый дисплей', 2), ('Третий дисплей', 0)]

Задание Б3

{'CoolComputer': ['Второй дисплей'], 'PusherForever': ['Второй дисплей']}