Вариант 26Б

Текст программы:

```
# Класс сущности "Студенческая группа"
class Group:

    def __init__(self, id, name, number_of_students):
        self.id = id  # id группы
        self.name = name  # Название группы
        self.number_of_students = number_of_students  # Кол-во студентов


# Класс сущности "Учебный курс"
class Course:

    def __init__(self, id, name):
        self.id = id  # id курса
        self.name = name  # Название курса


# Класс сущности "Студенческая группа - Учебный курс" (Для связи многие ко
многим)
class GroupCourse:

    def __init__(self, course_id, group_id):
        self.course_id = course_id  # id курса
        self.group_id = group_id  # id группы


# Студенческие группы
groups = [
    Group(1, 'IU1-12A', 26),
    Group(2, 'IU1-11B', 28),
    Group(3, 'IU3-21C', 20),
    Group(4, 'IU4-42D', 24),
    Group(5, 'IU5-44B', 19),
]

# Учебные курсы
courses = [
    Course(1, 'First'),
    Course(2, 'Second'),
    Course(3, 'Third'),
    Course(4, 'Fourth'),
]

# Студенческая группа - Учебный курс
groups_courses = [
    GroupCourse(1, 1),
    GroupCourse(1, 2),
    GroupCourse(2, 3),
    GroupCourse(4, 4),
    GroupCourse(4, 5),
    GroupCourse(3, 5),
]
```

```
def main():
    # Связь один-ко-многим
    """
    Генерируем связи один к многим путем равнозначимой
    привязки id курса к id группы,
    плюс одна связь пятой группы (IU5-44B) с четвертым курсом
    """
    one_to_many = [(g.name, g.number_of_students, c.name)
                   for c in courses
                   for g in groups
                   if g.id == c.id]
    one_to_many.append((groups[5 - 1].name, groups[5 - 1].number_of_students,
courses[4 - 1].name))

    print('Запрос Б1')
    b1 = sorted(one_to_many, key=lambda x: x[0])
    print(b1)

    print('\nЗапрос Б2')
    b2 = []
    for cource in courses:
        cource_groups = list(filter(lambda i: i[2] == cource.name,
one_to_many))
        if len(cource_groups) > 0:
            cource_students = [students for _, students, _ in cource_groups]
            cource_students_sum = sum(cource_students)
            b2.append((cource.name, cource_students_sum))

    b2 = sorted(b2, key=lambda x: x[1], reverse=True)
    print(b2)

    """
    Пояснение к запросу Б3:
    Вместо окончания "ов" в фамилии сотрудников,
    я взял окончание "B" в имени группы
    """
    print('\nЗапрос Б3')
    # Связь многие-ко-многим
    many_to_many_group = [(c.name, gc.course_id, gc.group_id)
                          for c in courses
                          for gc in groups_courses
                          if c.id == gc.course_id]

    many_to_many = [(g.name, g.number_of_students, name)
                    for name, _, group_id in many_to_many_group
                    for g in groups if g.id == group_id]

    b3 = {}
    for g in groups:
        if 'B' in g.name:
            course_groups = list(filter(lambda i: i[0] == g.name,
many_to_many))
            course_groups_names = [x for _, _, x in course_groups]
            b3[g.name] = course_groups_names

    print(b3)
```

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

Результат выполнения:

Запрос Б1

[('IU1-11B', 28, 'Second'), ('IU1-12A', 26, 'First'), ('IU3-21C', 20, 'Third'), ('IU4-42D', 24, 'Fourth'), ('IU5-44B', 19, 'Fourth')]

Запрос Б2

[('Fourth', 43), ('Second', 28), ('First', 26), ('Third', 20)]

Запрос Б3

{'IU1-11B': ['First'], 'IU5-44B': ['Third', 'Fourth']}