Код программы:

```
# используется для сортировки
from operator import itemgetter


class Student:
    """Студент"""
    def __init__(self, id, fio, score, group_id):
        self.id = id
        self.fio = fio
        self.score = score # количество накопленных баллов
        self.group_id = group_id

class Group:
    """Группа"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

class StudentGroup:
    """Студенты группы' для реализации связи многие-ко-многим"""
    def __init__(self, group_id, student_id):
        self.group_id = group_id
        self.student_id = student_id

# Группы
groups = [
    Group(1, 'ИУ5-31Б'),
    Group(2, 'ИБМ2-12Б'),
    Group(3, 'ИУ7-31Б'),

    Group(11, 'ИУ9-12Б'),
    Group(22, 'ИУ8-31'),
    Group(33, 'Э2-22'),
]

# Студенты
students = [
    Student(1, 'Алешечкина', 4.9, 1),
    Student(2, 'Громов', 4.8, 2),
    Student(3, 'Цыплакова', 4.3, 3),
    Student(4, 'Сапунов', 3.9, 3),
    Student(5, 'Шилина', 4.5, 3),
]

student_groups = [
    StudentGroup(1, 1),
    StudentGroup(2, 2),
    StudentGroup(3, 3),
```

```
    StudentGroup(3, 4),
    StudentGroup(3, 5),

    StudentGroup(11, 1),
    StudentGroup(22, 2),
    StudentGroup(33, 3),
    StudentGroup(33, 4),
    StudentGroup(33, 5),
]


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

    one_to_many = [(s.fio, s.score, g.name)
                   for s in students
                   for g in groups
                   if s.group_id == g.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(g.name, sg.group_id, sg.student_id)
                         for g in groups
                         for sg in student_groups
                         if g.id == sg.group_id]

    many_to_many = [(s.fio, s.score, group_name)
                    for group_name, group_id, student_id in many_to_many_temp
                    for s in students if s.id == student_id]

    print('Задание А1')
    res_11 = sorted(one_to_many, key=itemgetter(1)) # сортировка по количеству баллов
    print(res_11)

    print('\nЗадание А2')
    res_12_unsorted = []
    # Перебираем все группы
    for g in groups:
        # Список студентов группы
        g_students = list(filter(lambda x: x[2] == g.name, one_to_many))
        # Если группа не пустая
        if len(g_students) > 0:
            # Баллы студентов группы
            g_scores = [score for _, score, _ in g_students]
            # Суммарное количество баллов студентов группы
            g_scores_sum = sum(g_scores)
            res_12_unsorted.append((g.name, g_scores_sum))

    # Сортировка по суммарному баллу
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)
```

```
    print('\nЗадание А3')
    res_13 = {}
    # Перебираем всех бакалавров
    for g in groups:
        if 'Б' in g.name:
            # Список студентов группы
            g_stud = list(filter(lambda x: x[2] == g.name, many_to_many))
            # Только ФИО студентов
            g_stud_names = [n for n, _, _ in g_stud]
            # Добавляем результат в словарь: ключ - группа, значение - список фамилий
            res_13[g.name] = g_stud_names
    print(res_13)


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

Вывод:

Задание А1

[('Сапунов', 3.9, 'ИУ7-31Б'), ('Цыплакова', 4.3, 'ИУ7-31Б'), ('Шилина', 4.5, 'ИУ7-31Б'), ('Громов', 4.8, 'ИБМ2-12Б'), ('Алешечкина', 4.9, 'ИУ5-31Б')]

Задание А2

[('ИУ7-31Б', 12.7), ('ИУ5-31Б', 4.9), ('ИБМ2-12Б', 4.8)]

Задание А3

{'ИУ5-31Б': ['Алешечкина'], 'ИБМ2-12Б': ['Громов'], 'ИУ7-31Б': ['Цыплакова', 'Сапунов', 'Шилина'], 'ИУ9-12Б': ['Алешечкина']}