```
# используется для сортировки
from operator import itemgetter
class Group:
    """Группа"""
    def __init__(self, id, group, page, student_id):
        self.id = id
        self.group = group
        self.page = page
        self.student_id = student_id
class Student:
    """Студент"""
    def __init__(self, id, name):
        self.id = id
        self.name = name
# Студенты
students = [
    Student(11, 'Гриша'),
    Student(22, 'Миша'),
    Student(33, 'Рафик'),
    Student(44, 'Эрнест'),
    Student(55, 'Рафаэль'),
]

# Группы
groups = [
    Group(1, 'СМ', 6, 11),
    Group(2, 'ЭНЕРГО', 35, 22),
    Group(3, 'МТ', 94, 33),
    Group(4, 'ИБМ', 143, 44),
    Group(5, 'АК', 254, 55),
]
def main():
    """Основная функция"""
    # Соединение данных один-ко-многим
    one_to_many = [(c.group, c.page, b.name)
        for b in students
        for c in groups
        if c.student_id == b.id]
    print('Задание 1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)

    print('nЗадание 2')
    res_12_unsorted = []
    # Перебираем всех студентов
    for b in students:
        c_groups = list(filter(lambda i: i[2]==b.name, one_to_many))
        if len(c_groups) > 0:
            # уровень студента
            c_pages = [page for _,page,_ in c_groups]
            # Суммарный уровень
            c_pages_sum = sum(c_pages)
            res_12_unsorted.append((b.name, c_pages_sum))

            # Сортировка по суммарному уровню
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)
if __name__ == '__main__':
    main()
```

```
"""Задание 1
[('СМ', 6, 'Гриша'), ('ЭНЕРГО', 35, 'Миша'), ('АК', 254, 'Рафаэль'), ('МТ', 94,
'Рафик'), ('ИБМ', 143, 'Эрнест')]
nЗадание 2
[('Рафаэль', 254), ('Эрнест', 143), ('Рафик', 94), ('Миша', 35), ('Гриша', 6)]"""
```