Текст программы

```
# используется для сортировки
from operator import itemgetter


# Класс "Школьник"
class Pupil:
    def __init__(self, id, fio, skip, class_id):
        self.id = id
        self.fio = fio
        # Колличество пропущенных учебных дней
        self.skip = skip
        self.class_id = class_id


# Класс "Учебный класс"
class Sch_class:
    def __init__(self, id, name):
        self.id = id
        self.name = name



class PupilClass:
    def __init__(self, class_id, pupil_id):
        self.class_id = class_id
        self.pupil_id = pupil_id


# Классы
classes = [
    Sch_class(1, '1А'),
    Sch_class(2, '1Б'),
    Sch_class(3, '1В'),

    Sch_class(11, '2А'),
    Sch_class(22, '3Б'),
    Sch_class(33, '4Г'),
]


# Школьники
pupils = [
    Pupil(1, 'Аксенов', 3, 1),
    Pupil(2, 'Соломахин', 4, 1),
    Pupil(3, 'Лемзиков', 6, 2),
    Pupil(4, 'Семенов', 0, 2),
    Pupil(5, 'Уваров', 10, 3),
]


pupils_classes = [
    PupilClass(1,1),
    PupilClass(1,2),
    PupilClass(2,3),
```

```
    PupilClass(2,4),
    PupilClass(3,5),

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


def main():

    # Соединение данных один-ко-многим
    one_to_many = [(p.fio, p.skip, c.name)
        for c in classes
        for p in pupils
        if p.class_id==c.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(c.name, pc.class_id, pc.pupil_id)
        for c in classes
        for pc in pupils_classes
        if c.id==pc.class_id]

    many_to_many = [(p.fio, p.skip, class_name)
        for class_name, class_id,pupil_id in many_to_many_temp
        for p in pupils if p.id==pupil_id]



    print('Задание А1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)


    print('\nЗадание А2')
    res_12_unsorted = []
    # Перебираем все классы
    for c in classes:
        # Список учеников класса
        c_pupils = list(filter(lambda i: i[2]==c.name, one_to_many))
        # Если класс не пустой
        if len(c_pupils) > 0:
            # Пропучки учебных дней учеников класса
            c_skips = [skip for _,skip,_ in c_pupils]
            # Суммарное количество пропусков учеников класса
            c_skips_sum = sum(c_skips)
            res_12_unsorted.append((c.name, c_skips_sum))
    # Сортировка по суммарному пропуску
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)


    print('\nЗадание А3')
    res_13 = {}
```

```
    # Перебираем все классы
    for c in classes:
        if 'А' in c.name:
            # Список учеников класса
            c_pupils = list(filter(lambda i: i[2]==c.name, many_to_many))
            # Только ФИО учеников
            c_pupils_names = [x for x,_,_ in c_pupils]
            # Добавляем результат в словарь
            # ключ - класс, значение - список фамилий
            res_13[c.name] = c_pupils_names
    print(res_13)


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

Результаты выполнения:

Задание А1

[('Аксенов', 3, '1А'), ('Соломахин', 4, '1А'), ('Лемзиков', 6, '1Б'), ('Семенов', 0, '1Б'), ('Уваров', 10, '1В')]

Задание А2

[('1В', 10), ('1А', 7), ('1Б', 6)]

Задание А3

{'1А': ['Аксенов', 'Соломахин'], '2А': ['Аксенов', 'Соломахин']}