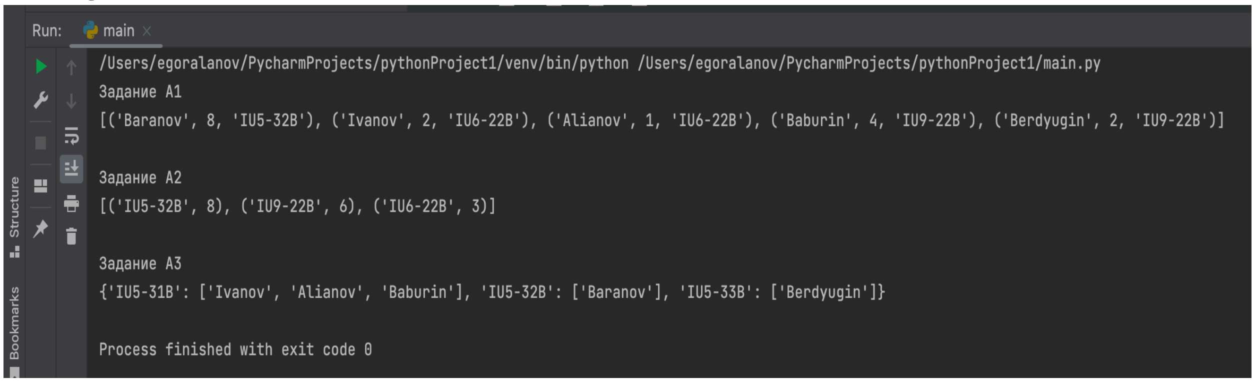
Текст программы:

```python
# используется для сортировки
from operator import itemgetter


class Emp:
    """Студент"""

    def __init__(self, id, name, debts, dep_id):
        self.id = id
        self.name = name
        self.debts = debts
        self.dep_id = dep_id


class Dep:
    """Группа"""

    def __init__(self, id, index):
        self.id = id
        self.index = index


class EmpDep:
    """
    'Студенты группы' для реализации
    связи многие-ко-многим
    """

    def __init__(self, dep_id, emp_id):
        self.dep_id = dep_id
        self.emp_id = emp_id


# Группы
deps = [
    Dep(1, 'IU5-31B'),
    Dep(2, 'IU5-32B'),
    Dep(3, 'IU5-33B'),

    Dep(11, 'IU6-22B'),
    Dep(22, 'IU7-22B'),
    Dep(33, 'IU9-22B'),
]

# Студенты
emps = [
    Emp(1, 'Ivanov', 2, 11),
    Emp(2, 'Alianov', 1, 11),
    Emp(3, 'Baburin', 4, 33),
    Emp(4, 'Berdyugin', 2, 33),
    Emp(5, 'Baranov', 8, 2),
]

emps_deps = [
    EmpDep(1, 1),
    EmpDep(1, 2),
    EmpDep(1, 3),
    EmpDep(3, 4),
    EmpDep(2, 5),

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


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

    # Соединение данных один-ко-многим
    one_to_many = [(e.name, e.debts, d.index)
                   for d in deps
                   for e in emps
                   if e.dep_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.index, ed.dep_id, ed.emp_id)
                         for d in deps
                         for ed in emps_deps
                         if d.id == ed.dep_id]

    many_to_many = [(e.name, e.debts, dep_name)
                    for dep_name, dep_id, emp_id in many_to_many_temp
                    for e in emps if e.id == emp_id]

    print('Задание A1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)

    print('\nЗадание A2')
    res_12_unsorted = []
    # Перебираем все группы
    for d in deps:
        # Список студентов групп
        d_emps = list(filter(lambda i: i[2] == d.index, one_to_many))
        # Если в группе есть студенты
        if len(d_emps) > 0:
            # Кол-во долгов студента группы
            d_sals = [sal for _, sal, _ in d_emps]
            # Сумма долгов студента группы
            d_sals_sum = sum(d_sals)
            res_12_unsorted.append((d.index, d_sals_sum))

    # Сортировка по сумме долгов
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)

    print('\nЗадание A3')
    res_13 = {}
    # Перебираем все группы
    for d in deps:
        if 'IU5' in d.index:
            # Список студентов групп
            d_emps = list(filter(lambda i: i[2] == d.index, many_to_many))
            # Только фамилия студента
            d_emps_names = [x for x, _, _ in d_emps]
            # Добавляем результат в словарь
            # ключ - группа, значение - список фамилий студентов
            res_13[d.index] = d_emps_names

    print(res_13)
```

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

Результаты выполнения:



```
/Users/egoralanov/PycharmProjects/pythonProject1/venv/bin/python /Users/egoralanov/PycharmProjects/pythonProject1/main.py
Задание А1
[('Baranov', 8, 'IU5-32B'), ('Ivanov', 2, 'IU6-22B'), ('Alianov', 1, 'IU6-22B'), ('Baburin', 4, 'IU9-22B'), ('Berdyugin', 2, 'IU9-22B')]

Задание А2
[('IU5-32B', 8), ('IU9-22B', 6), ('IU6-22B', 3)]

Задание А3
{'IU5-31B': ['Ivanov', 'Alianov', 'Baburin'], 'IU5-32B': ['Baranov'], 'IU5-33B': ['Berdyugin']}

Process finished with exit code 0
```