Отчет по рубежному контролю

Листинг программы

```
# используется для сортировки
from operator import itemgetter


class Schoolchild:
    """Школьники"""

    def __init__(self, id, fio, avg_res, class_id):
        self.id = id
        self.fio = fio
        self.avg_res = avg_res
        self.class_id = class_id


class Class:
    """Класс"""

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


class SchoolchildClass:
    """
    'Школьники класса' для реализации
    связи многие-ко-многим
    """

    def __init__(self, class_id, schchild_id):
        self.class_id = class_id
        self.schchild_id = schchild_id


# Классы
classes = [
    Class(1, 'отдел Быстров'),
    Class(2, 'Покорители'),
    Class(3, 'отдел Трудягов'),

    Class(11, 'Океанов'),
    Class(22, 'отдел Строителей'),
    Class(33, 'отдел Альпинистов'),
]

# Школьники
schchilds = [
    Schoolchild(1, 'Артамонов', 4.5, 1),
    Schoolchild(2, 'Петров', 4.8, 2),
    Schoolchild(3, 'Иваненко', 3.8, 3),
    Schoolchild(4, 'Иванов', 4.1, 3),
    Schoolchild(5, 'Иванин', 3.3, 3),
]

schchilds_classes = [
    SchoolchildClass(1, 1),
    SchoolchildClass(2, 2),
    SchoolchildClass(3, 3),
    SchoolchildClass(3, 4),
    SchoolchildClass(3, 5),
```

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


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

    # Соединение данных один-ко-многим
    one_to_many = [(e.fio, e.avg_res, d.name_class)
                   for d in classes
                   for e in schchilds
                   if e.class_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.name_class, ed.class_id, ed.schchild_id)
                         for d in classes
                         for ed in schchilds_classes
                         if d.id == ed.class_id]

    many_to_many = [(e.fio, e.fio, dep_name)
                    for dep_name, dep_id, emp_id in many_to_many_temp
                    for e in schchilds if e.id == emp_id]

    print('Задание А1')
    # Сортировка по среднему баллу
    res_11 = sorted(one_to_many, key=itemgetter(1))
    print(res_11)

    print('\nЗадание А2')
    res_12_unsorted = []
    # Перебираем все классы
    for d in classes:
        # Список школьников класса
        d_classes = list(filter(lambda i: i[2] == d.name_class, one_to_many))
        # Если класс не пустой
        if len(d_classes) > 0:
            # Средние баллы школьников класса
            d_sals = [sal for _, sal, _ in d_classes]
            # Суммарный средний балл школьников классов
            d_sals_sum = sum(d_sals)
            res_12_unsorted.append((d.name_class, d_sals_sum))

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

    print('\nЗадание А3')
    res_13 = {}
    # Перебираем все классы
    for d in classes:
        if 'отдел' in d.name_class:
            # Список школьников класса
            d_emps = list(filter(lambda i: i[2] == d.name_class,
many_to_many))
            # Только фамилия школьников класса
            d_emps_names = [x for x, _, _ in d_emps]
            # Добавляем результат в словарь
            # ключ - класс, значение - список фамилий
            res_13[d.name_class] = d_emps_names
```

```
    print(res_13)


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

Результат программы

Задание А1

[('Иванин', 3.3, 'отдел Трудягов'), ('Иваненко', 3.8, 'отдел Трудягов'), ('Иванов', 4.1, 'отдел Трудягов'), ('Артамонов', 4.5, 'отдел Быстров'), ('Петров', 4.8, 'Покорители')]

Задание А2

[('отдел Трудягов', 11.2), ('Покорители', 4.8), ('отдел Быстров', 4.5)]

Задание А3

{'отдел Быстров': ['Артамонов'], 'отдел Трудягов': ['Иваненко', 'Иванов', 'Иванин'], 'отдел Строителей': ['Петров'], 'отдел Альпинистов': ['Иваненко', 'Иванов', 'Иванин']}