Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

ОТЧЁТ ПО
Рубежному контролю

Удалова Виктория ИУ5-35Б

Парадигмы и конструкции языков программирования

Москва
2023

Задание

Необходимо создать два класса. Предметная область 1 класс - Музыкальное прроизведение, 2 класс – Оркестор, которые связаны отношениями один-ко-многим и многие-ко-многим. Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.

запросы:Вариант Д.

«Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех сотрудников,

у которых фамилия заканчивается на «ов», и названия их отделов.

«Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате (отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений).

«Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А», и список работающих в них сотрудников.

При разработке запросов необходимо по возможности использовать функциональные возможности языка Python

Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Код программы:

```
class Произведение:
    def __init__(self, Название: str, Автор: str, Длительность: int) -> None:
        self.название = Название
        self.длительность = Длительность
        self.автор = Автор

    def __str__(self):
        return f"""Произведение("{self.название}", "{self.автор}",
{self.длительность})"""


class Оркестр:
    def __init__(self, Название: str, Год_создания :str, Произведения) -> None:
        self.Название = Название
        self.Год_создания = Год_создания
        self.Произведения = list(Произведения)
```

```
    def __iter__(self):
        return iter(self.Произведения)

    def __len__(self):
        return len(self.Произведения)

    def append(self, Новое_произведение: Произведение):
        self.Произведения.append(Новое_произведение)

    def __str__(self):
        Вывод = str.format("""Оркестр("{0}", "{1}", [""",
                           self.Название, self.Год_создания)
        for произведение in self:
            Вывод += str(произведение) + ",\n"
        Вывод = Вывод[:-2]
        Вывод += "])\n,"
        return Вывод

def Задание2(Оркестры):
    return map(lambda x: (x.Название, (sum([произведение.длительность for
произведение in x]) / len(x))), Оркестры)

def Задание3(Оркестры):
    return list(filter(lambda x: x.Название[0] == 'A', Оркестры))

def Задание1(Оркестры):
    return map(lambda x: Оркестр(x.Название, x.Год_создания, list(filter(lambda
произв: произв.название[-2:] == 'ов', x.Произведения))), Оркестры)


Оркестры = [Оркестр("Крюков", "LINUX", [Произведение("Большаков", "Большаков",
10000)]),
                Оркестр("Кошелева", "LINUX", [Произведение("Орлов", "Большаков",
10000)]),
                Оркестр("Жданов", "LINUX", [Произведение("Логинов", "Большаков",
10000),
                                            Произведение("Егоров", "Большаков",
10000)]),
                Оркестр("Кузьмин", "LINUX", [Произведение("Соколов",
"Большаков", 10000)]),
                Оркестр("Андреева", "LINUX", [Произведение("Волков",
"Большаков", 10000),
                                              Произведение("Волков",
"Большаков", 10000),
                                              Произведение("Мартынов",
"Большаков", 10000)]),
                Оркестр("Щербакова", "LINUX", [Произведение("Иванов",
"Большаков", 10000),
                                               Произведение("Иванов",
"Большаков", 10000),
                                               Произведение("Черкасов",
"Большаков", 10000),
                                               Произведение("Озеров",
"Большаков", 10000)]),
                Оркестр("Цветков", "LINUX", [Произведение("Комаров",
"Большаков", 10000),
Произведение("Комаров", "Большаков", 10000),
Произведение("Петров", "Большаков", 10000)])
, Оркестр("Петров", "LINUX", [Произведение("Громов", "Большаков", 10000),
Произведение("Громов", "Большаков", 10000),
Произведение("Тихонов", "Большаков", 10000)])
, Оркестр("Алексеев", "LINUX", [Произведение("Спиридонов", "Большаков", 10000),
Произведение("Аксенов", "Большаков", 10000),
Произведение("Кузнецов", "Большаков", 10000)])
```

```
, Оркестр("Рудакова", "LINUX", [Произведение("Антонов", "Большаков", 10000),
Произведение("Ефремов", "Большаков", 10000),
Произведение("Басов", "Большаков", 10000)])]


print(*Задание1(Оркестры), sep='\n')
print(*Задание2(Оркестры), sep='\n')
print(*Задание3(Оркестры), sep='\n')

print('-'*80)
print("""«Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите
список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их
отделов.""")
print('-'*80)
print(*Задание1(Оркестры), sep='\n')
"""for computer in computers:
    print_flag = False
    for disc in computer:
        if disc.name.split()[0][-2:] == 'ов':
            if print_flag == False:
                print(computer.name)
                print_flag = True
            print(disc)"""
print('-'*80)
print("""«Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите
список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный
по средней зарплате
(отдельной функции вычисления среднего значения в Python нет, нужно использовать
комбинацию функций вычисления суммы и количества значений).""")
print('-'*80)
print(*Задание2(Оркестры), sep='\n')
print('-'*80)
print("""«Музыкальное произведение» и «Оркестр» связаны соотношением многие-ко-
многим. Выведите список всех отделов, у которых название начинается с буквы «А»,
и список работающих в них сотрудников.""")
print('-'*80)
print(*Задание3(Оркестры), sep='\n')
```

Вывод:

```
«Музыкальное произведение» и «Оркестор» связаны соотношением один-ко-многим. Выведите список
(отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию ф
--------------------------------------------------------------------------------
('Крюков', 10000.0)
('Кошелева', 10000.0)
('Жданов', 10000.0)
('Кузьмин', 10000.0)
('Андреева', 10000.0)
('Щербакова', 10000.0)
('Цветков', 10000.0)
('Петров', 10000.0)
('Алексеев', 10000.0)
('Рудакова', 10000.0)
--------------------------------------------------------------------------------
«Музыкальное произведение» и «Оркестр» связаны соотношением многие-ко-многим. Выведите список
--------------------------------------------------------------------------------
Оркестр("Андреева", "1900", [Произведение("Волков", "Большаков", 10000),
Произведение("Волков", "Большаков", 10000),
Произведение("Мартынов", "Большаков", 10000)])
,
Оркестр("Алексеев", "1900", [Произведение("Спиридонов", "Большаков", 10000),
Произведение("Аксенов", "Большаков", 10000),
Произведение("Кузнецов", "Большаков", 10000)])
,

Process finished with exit code 0
```