|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»

ДИСЦИПЛИНА:

«БКИТ»

**Рубежная контроль № 1**

Студент Распашнов А.А. ИУ5Ц-54Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(И.О. Фамилия) (Группа) (Подпись, дата)

Преподаватель Гапанюк Ю.Е.  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(И.О. Фамилия) (Подпись, дата)

Москва, 2021г.

**Задание дано:**

**Вариант Г.**

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А», и список работающих в них сотрудников.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с максимальной зарплатой сотрудников в каждом отделе, отсортированный по максимальной зарплате.
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по отделам, сортировка по сотрудникам произвольная.

**Вариант:**

|  |  |  |
| --- | --- | --- |
| 21 | Оператор | Язык программирования |

**Текст программы:**

class Oper:

    def \_\_init\_\_(self, id, name, salary, lang\_id):

        self.id = id

        self.name = name

        self.salary = salary

        self.lang\_id = lang\_id

class Lang:

    def \_\_init\_\_(self, id, name):

        self.id = id

        self.name = name

class OperLang:

    def \_\_init\_\_(self, oper\_id, lang\_id):

        self.oper\_id = oper\_id

        self.lang\_id = lang\_id

languages = [

    Lang(1, 'Python'),

    Lang(2, 'Java'),

    Lang(3, 'JavaScript'),

    Lang(4, 'C++'),

    Lang(5, 'C'),

    Lang(6, 'Go'),

    Lang(7, 'Rust')

]

operators = [

    Oper(1, 'Maxim', 1000, 1),

    Oper(2, 'Andrey', 2000, 2),

    Oper(3, 'Daniel', 1500, 1),

    Oper(4, 'Viktor', 3000, 5),

    Oper(5, 'Misha', 2500, 4),

    Oper(6, 'Dima', 500, 7),

    Oper(7, 'Rebekka', 300, 3)

]

opers\_langs = [

    OperLang(1, 1),

    OperLang(1, 5),

    OperLang(1, 3),

    OperLang(2, 2),

    OperLang(2, 3),

    OperLang(3, 1),

    OperLang(4, 2),

    OperLang(4, 3),

    OperLang(6, 1),

    OperLang(6, 5)

]

def main():

    one\_to\_many = [(oper.name, oper.salary, lang.name)

                                 for lang in languages

                                 for oper in operators

                                 if oper.lang\_id == lang.id]

    many\_to\_many\_temp = [(lang.name, ol.oper\_id, ol.lang\_id)

                                             for lang in languages

                                             for ol in opers\_langs

                                             if lang.id == ol.lang\_id]

    many\_to\_many = [(oper.name, oper.salary, lang\_name)

                                    for lang\_name, \_, lang\_id in many\_to\_many\_temp

                                    for oper in operators if oper.lang\_id == lang\_id]

    print('Задание Г1')

    res\_11 = filter(lambda entry: entry[2][0] == 'J', one\_to\_many)

    print([res for res in res\_11])

    print('\nЗадание Г2')

    res = [(lang.name, max([salary for \_, salary, lang\_name in one\_to\_many if lang\_name == lang.name]))

                 for lang in languages if len(list(filter(lambda entry: entry[2] == lang.name, one\_to\_many))) > 0]

    print(sorted(res, key=lambda x: x[1], reverse=True))

    print('\nЗадание Г3')

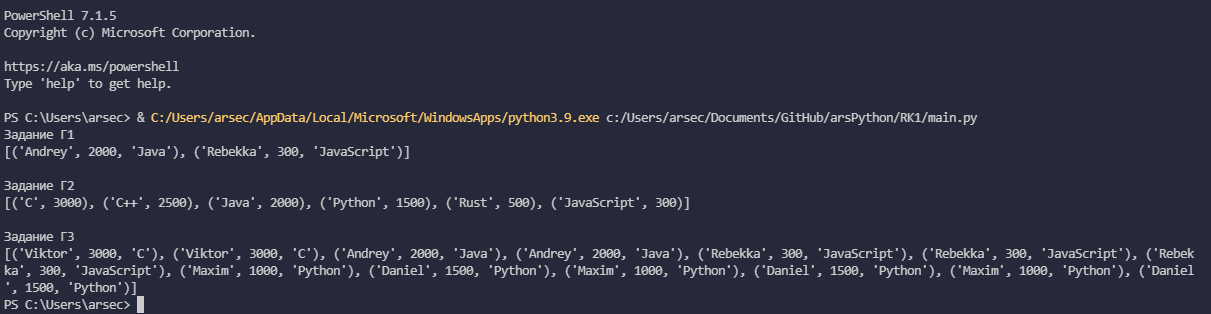
    res = sorted(many\_to\_many, key=lambda entry: entry[2])

    print(res)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**Результат программы:**

****