|  |  |
| --- | --- |
| 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 Макеев В. 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:**

|  |  |  |
| --- | --- | --- |
| 20 | Деталь | Поставщик |

**Текст программы:**

# используется для сортировки

from operator import itemgetter

"""Деталь"""

class Detail:

    def \_\_init\_\_(self, id, name1, price, provider\_id):

        self.id=id

        self.name1=name1

        self.price=price

        self.provider\_id=provider\_id

"""Поставщик"""

class Provider:

    def \_\_init\_\_(self, id, name):

        self.id=id

        self.name=name

"""Детали поставщика"""

class DetailProvider:

    def \_\_init\_\_(self, provider\_id, detail\_id):

        self.provider\_id=provider\_id

        self.detail\_id=detail\_id

#поставщики (id поставщика, название поставщика)

providers=[Provider(1,'АвтоСпейс'),

Provider(2,'Favorit-auto'),

Provider(3,'Автотрейд'),

Provider(11,'Next-auto'),

Provider(22,'Гарант-Авто'),

Provider(33,'Forum-Auto'),

]

#детали (id детали, название детали, цена, id поставщика)

details=[Detail(1,'сцепление',4000,1),

Detail(2,'маховик',2000,3),

Detail(3,'поршень',15000,3),

Detail(4,'колодка',1500,2),

Detail(5,'подвеска',14000,1),

]

#детали поставщика (id поставщика,id детали)

details\_providers=[DetailProvider(1,1),

DetailProvider(2,2),

DetailProvider(3,3),

DetailProvider(4,4),

DetailProvider(5,5),

DetailProvider(22,1),

DetailProvider(11,2),

DetailProvider(33,3),

DetailProvider(33,4),

DetailProvider(11,5),

]

def main():

    one\_to\_many=[(d.name1,d.price,p.name)

        for p in providers

        for d in details

        if d.provider\_id == p.id

    ]

    many\_to\_many\_temp=[(p.name, dp.provider\_id,dp.detail\_id)

        for p in providers

        for dp in details\_providers

        if p.id==dp.provider\_id

    ]

    many\_to\_many=[(d.name1,d.price,provider\_name)

        for provider\_name, provider\_id, detail\_id in many\_to\_many\_temp

        for d in details if d.id == detail\_id

    ]

    print('Задание Г1')

    res\_11 = [(p.name,list(name1 for name1,\_,name in one\_to\_many if name == p.name)) for p in providers if p.name[0] == 'А']

    print(res\_11)

    print('\nЗадание Г2')

    res\_12\_unsorted = []

    # Перебираем все дисплейные классы

    for p in providers:

        # Список компьютеров дисплейного класса

        p\_details = list(filter(lambda x: x[2] == p.name, one\_to\_many))

        # Если дисплейный класс не пустой

        if len(p\_details) > 0:

            res\_12\_unsorted.append((p.name, max(p\_details, key=lambda x: x[1])[1]))

    # Сортировка по максимальной стоимости

    res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)

    print(res\_12)

    print('Задание Г3')

    res\_13 = []

    # Перебираем все дисплейные классы

    for detail,\_,provider in many\_to\_many:

        res\_13.append((detail, provider))

    res\_13 = sorted(res\_13, key=itemgetter(1))

    print(res\_13)

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

    main()

**Результат программы:**

