*Федеральное государственное бюджетное образовательное учреждение* *высшего профессионального образования*

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления (ИУ5)

**Отчет**

**по рубежному контролю №1**

**Дисциплина: Разработка Интернет-Приложений**

Студент гр. ИУ5-53Б  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Терентьев В.О.

(Подпись, дата) (Фамилия И.О.)

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

(Подпись, дата) (Фамилия И.О.)

Москва, 2020

1. **Описание задания**

**Вариант В.**

1. «Поставщик» и «Деталь» связаны соотношением один-ко-многим. Выведите список всех деталей, у которых название начинается с буквы «А», и названия их поставщиков.
2. «Поставщик» и «Деталь» связаны соотношением один-ко-многим. Выведите список поставщиков с минимальной стоимостью деталей у каждого поставщика, отсортированный по минимальной стоимости детали.
3. «Поставщик» и «Деталь» связаны соотношением многие-ко-многим. Выведите список всех связанных деталей и поставщиков, отсортированный по деталям, сортировка по поставщикам произвольная.
4. **Текст программы**

from operator import itemgetter

class Detail:

    """Деталь"""

    def \_\_init\_\_(self, id, name, price, sup\_id):

        self.id = id

        self.name = name

        self.price = price

        self.sup\_id = sup\_id

class Supplier:

    """Поставщик"""

    def \_\_init\_\_(self, id, name):

        self.id = id

        self.name = name

class DetSup:

    """'Детали поставщика' для реализации связи многие-ко-многим"""

    def \_\_init\_\_(self, sup\_id, det\_id):

        self.sup\_id = sup\_id

        self.det\_id = det\_id

# Поставщики

sups = [

    Supplier(1, 'GATES'),

    Supplier(2, 'NORMA'),

    Supplier(3, 'FINWHALE'),

    Supplier(4, 'CTR'),

]

# Детали

dets = [

    Detail(1, 'Ремень ГРМ', 362, 1),

    Detail(2, 'Хомут', 28, 2),

    Detail(3, 'Амортизатор двери', 622, 3),

    Detail(4, 'Тяга переднего стабилизатора', 600, 4),

    Detail(5, 'Колодки передние тормозные', 1193, 4),

]

dets\_sups = [

    DetSup(1,1),

    DetSup(2,2),

    DetSup(3,3),

    DetSup(4,4),

    DetSup(4,5),

    DetSup(4,3)

]

def main():

    """Основная функция"""

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

        for s in sups

        for d in dets

        if d.sup\_id==s.id]

    many\_to\_many\_temp = [(s.name, ds.sup\_id, ds.det\_id)

        for s in sups

        for ds in dets\_sups

        if s.id==ds.sup\_id]

    many\_to\_many = [(d.name, d.price, dep\_name)

        for dep\_name, sup\_id, det\_id in many\_to\_many\_temp

        for d in dets if d.id==det\_id]

    print('Задание В1')

    res\_11 = list(filter(lambda x: x[0].startswith('А'), one\_to\_many))

    print(res\_11)

    print('\nЗадание В2')

    res\_12\_unsorted = []

    for s in sups:

        s\_dets = list(filter(lambda i: i[2]==s.name, one\_to\_many))

        if len(s\_dets) > 0:

            s\_prices = [price for \_,price,\_ in s\_dets]

            s\_prices\_min = min(s\_prices)

            res\_12\_unsorted.append((s.name, s\_prices\_min))

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

    print(res\_12)

    print('\nЗадание В3')

    res\_13 = sorted(many\_to\_many, key=itemgetter(0))

    print(res\_13)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

1. **Результаты работы программы**

