Громова Ольга ИУ5-51Б РК№1

Вариант А, вариант 10

1. «Компьютер» и «Жесткий диск» связаны соотношением один-ко-многим. Выведите список всех связанных компьютеров и браузеров, отсортированный по компьютерам, сортировка по браузерам произвольная.
2. «Компьютер» и «Жесткий диск» связаны соотношением один-ко-многим. Выведите список компьютеров с суммарной занятой памятью браузеров в каждом компьютере, отсортированный по суммарной памяти.
3. «Компьютер» и «Жесткий диск» связаны соотношением многие-ко-многим. Выведите список всех компьютеров, у которых в названии присутствует слово «ASUS», и список присутствующих в них браузеров.

Исходный код программы:

```
from operator import itemgetter


class Emp:
    """Жесткий диск"""

    def __init__(self, id, fio, sal, dep_id):
        self.id = id
        self.fio = fio
        self.sal = sal
        self.dep_id = dep_id


class Dep:
    """Компьютер"""

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


class EmpDep:
    """
    'Жесткие диски в компьютерах' для реализации
    связи многие-ко-многим
    """

    def __init__(self, dep_id, emp_id):
        self.dep_id = dep_id
        self.emp_id = emp_id


# Компьютеры
deps = [
    Dep(1, 'ASUS'),
```

```
    Dep(2, 'APPLE'),
    Dep(3, 'DELL'),
]

# Жесткие диски
emps = [
    Emp(1, 'WESTERN', 25000, 1),
    Emp(2, 'DIGITAL', 35000, 2),
    Emp(3, 'EASTERN', 45000, 3),
    Emp(4, 'SOUTHTERN', 35000, 3),
    Emp(5, 'SANYA-HAMLO', 25000, 3),
]

emps_deps = [
    EmpDep(1, 1),
    EmpDep(2, 2),
    EmpDep(3, 3),
    EmpDep(3, 4),
    EmpDep(3, 5),


]


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

    # Соединение данных один-ко-многим
    one_to_many = [(e.fio, e.sal, d.name)
                   for d in deps
                   for e in emps
                   if e.dep_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.name, ed.dep_id, ed.emp_id)
                         for d in deps
                         for ed in emps_deps
                         if d.id == ed.dep_id]

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

    print('Задание А1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)

    print('\nЗадание А2')
    res_12_unsorted = []
    # Перебираем все компьютеры
    for d in deps:
        # Список жестких дисков
        d_emps = list(filter(lambda i: i[2] == d.name, one_to_many))
        # Если есть компьютер
        if len(d_emps) > 0:
            # Стоимость
            d_sals = [sal for _, sal, _ in d_emps]
            # Суммарная стоимость жестких дисков
            d_sals_sum = sum(d_sals)
            res_12_unsorted.append((d.name, d_sals_sum))

    # Сортировка по суммарной стоимости
```

```
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)

    print('\nЗадание А3')
    res_13 = {}
    # Перебираем все компьютеры
    for d in deps:
        if 'ASUS' in d.name:
            # Список жестких дисков
            d_emps = list(filter(lambda i: i[2] == d.name, many_to_many))
            # Только названте жестких дисков
            d_emps_names = [x for x, _, _ in d_emps]
            # Добавляем результат в словарь
            # ключ - компьютер, значение - список названий
            res_13[d.name] = d_emps_names

    print(res_13)


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

Результат программы

Задание А1
[('DIGITAL', 35000, 'APPLE'), ('WESTERN', 25000, 'ASUS'), ('EASTERN', 45000, 'DELL'), ('SOUTHTERN', 35000, 'DELL'), ('SANYA-HAMLO', 25000, 'DELL')]

Задание А2
[('DELL', 105000), ('APPLE', 35000), ('ASUS', 25000)]

Задание А3
{'ASUS': ['WESTERN']}