**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 «Радиотехнический»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Отчет по РК1
«Парадигмы и конструкции языков программирования»

Выполнил:
студент группы РТ5-31Б
Иванченко Д.А.

Проверил:
Гапанюк Ю. Е

2023 г.

## Задание

1) Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Пример классов данных для предметной области Сотрудник-Отдел:

1. Класс «Сотрудник», содержащий поля:
   - ID записи о сотруднике;
   - Фамилия сотрудника;
   - Зарплата (количественный признак);
   - ID записи об отделе. (для реализации связи один-ко-многим)
2. Класс «Отдел», содержащий поля:
   - ID записи об отделе;
   - Наименование отдела.
3. (Для реализации связи многие-ко-многим) Класс «Сотрудники отдела», содержащий поля:
   - ID записи о сотруднике;
   - ID записи об отделе.

2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.

3) Необходимо разработать запросы в соответствии с Вашим вариантом. Запросы сформулированы в терминах классов «Сотрудник» и «Отдел», которые используются в примере. Вам нужно перенести эти требования в Ваш вариант предметной области. При разработке запросов необходимо по возможности использовать функциональные возможности языка Python (list/dict comprehensions, функции высших порядков).

1. «Компьютер» и «Операционная система» связаны соотношением один-ко-многим. Выведите список всех компьютеров, у которых в названии процессора присутствует слово «Ryzen», и список работающих их операционных систем.

2. «Компьютер» и «Операционная система» связаны соотношением один-ко-многим. Выведите список операционных систем со средним объемом памяти компьютеров, использующих эту операционную систему, отсортированный по объемом памяти компьютера. Средний объем памяти должен быть округлен до 2 знака после запятой.

3. «Компьютер» и «Операционная система» связаны соотношением многие-ко-многим. Выведите список всех операционных систем, у которых название начинается с буквы «W» и список компьютеров на которых они установлены

## Текст программы

```
# используется для сортировки
from operator import itemgetter

"""
1. «Компьютер» и «Операционная система» связаны соотношением один-ко-многим.
Выведите список всех компьютеров,
у которых в названии процессора присутствует слово «Ryzen», и список
работающих их операционных систем.

2. «Компьютер» и «Операционная система» связаны соотношением один-ко-многим.
Выведите список операционных
систем со средним объемом памяти компьютеров, использующих эту операционную
систему,
отсортированный по объемом памяти компьютера. Средний объем памяти должен
быть округлен до 2 знака после запятой.

3. «Компьютер» и «Операционная система» связаны соотношением многие-ко-
многим. Выведите список всех операционных
систем, у которых название начинается с буквы «W» и список компьютеров на
которых они установлены.
"""


class PC:
    """Компьютер"""

    def __init__(self, id, user, firm, cpu, memory, os_id):
        self.id = id
        self.user = user
        self.firm = firm
        self.cpu = cpu
        self.memory = memory
        self.os_id = os_id


class OS:
    """Операционная система"""

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


class PC_OS:
    def __init__(self, os_id, pc_id):
        self.os_id = os_id
        self.pc_id = pc_id


# Операционные системы
OSs = [
    OS(1, "Windows 10", 2015),
    OS(2, "Linux", 1991),
    OS(3, "Windows 11", 2021),
    OS(4, "Windows 7", 2009),
    OS(5, "FreeDOS", 2006),
]

# Компьютеры
PCs = [
```

```
    PC(1, "Андрейкин", "Lenovo", "Intel Core i3", 1024, 1),
    PC(2, "Карлсон", "ASUS", "Intel Core i5", 2048, 1),
    PC(3, "Грищук", "Dell", "Intel Core i7", 1536, 2),
    PC(4, "Каспаров", "hp", "Intel Core i9", 4096, 3),
    PC(5, "Каруана", "Dell", "AMD Ryzen 3", 512, 2),
    PC(6, "Накамура", "ASUS", "AMD Ryzen 7", 1024, 4),
    PC(7, "Нородинский", "Lenovo", "AMD Ryzen 9", 2048, 4),
    PC(8, "Омариев", "Colorful", "AMD Ryzen 5", 3072, 5)
]

PC_OSs = [
    PC_OS(1, 1),
    PC_OS(1, 6),
    PC_OS(2, 2),
    PC_OS(2, 7),
    PC_OS(3, 3),
    PC_OS(3, 4),
    PC_OS(3, 6),
    PC_OS(3, 8),
    PC_OS(4, 5),
    PC_OS(5, 2),
    PC_OS(5, 3),
    PC_OS(5, 8)
]


def sred_r(mas):  # среднее округл. значение
    return round(sum(mas) / len(mas), 2)


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

    # Задание 1
    task1 = [(pc.firm, os.name)
             for pc in PCs
             for os in OSs
             if "Ryzen" in pc.cpu and pc.os_id == os.id]
    task1 = sorted(task1)

    print('Задание E1')
    print(task1)

    # Задание 2
    task2 = [(os.name, sred_r([pc.memory for pc in PCs if pc.os_id ==
os.id])) for os in OSs]
    task2.sort(key=lambda x: x[1])

    print('\nЗадание E2')
    print(task2)

    # Задание 3
    def pcsearch(pc_id):
        for pc in PCs:
            if pc.id == pc_id:
                return pc.firm

    task3 = {os.name:
                 [pcsearch(pc_os.pc_id) for pc_os in PC_OSs if pc_os.os_id ==
os.id]
             for os in OSs if os.name[0]
             if os.name[0] == 'W'}

    print('\nЗадание E3')
```

```
    print(task3)


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

**Вывод**

```
Задание E1
[('ASUS', 'Windows 7'), ('Colorful', 'FreeDOS'), ('Dell', 'Linux'), ('Lenovo', 'Windows 7')]

Задание E2
[('Linux', 1024.0), ('Windows 10', 1536.0), ('Windows 7', 1536.0), ('FreeDOS', 3072.0), ('Windows 11', 4096.0)]

Задание E3
{'Windows 10': ['Lenovo', 'ASUS'], 'Windows 11': ['Dell', 'hp', 'ASUS', 'Colorful'], 'Windows 7': ['Dell']}
```