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»
Кафедра ИУ5 «Системы обработки информации и управления»

Рубежный контроль №1

по дисциплине «Базовые компоненты интернет-технологий»

Выполнила:
студентка группы ИУ5-33Б
Шаповалова В.В.

Проверил:
Гапанюк Ю.Е

2021 г.

Постановка задания:

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

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

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).

Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Вариант В.

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

Предметная область:

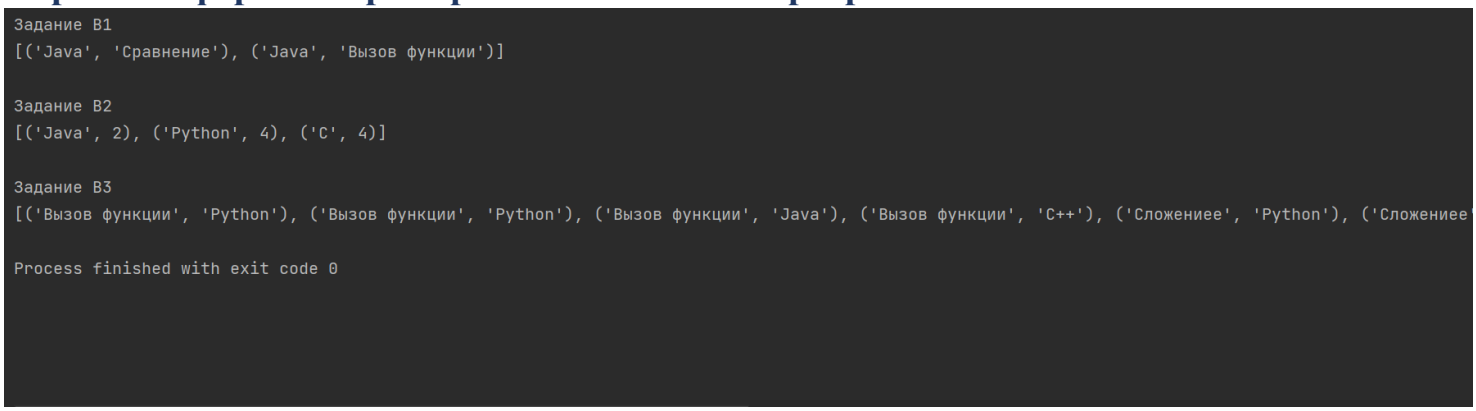
№ варианта	**Класс 1**	**Класс 2**
21	Оператор	Язык программирования

Текст программы:

```
# используется для сортировки
from operator import itemgetter


class Op:
    """Оператор"""

    def __init__(self, id, op_name, memory, ln_id):
        self.id = id
        self.op_name = op_name
        self.memory = memory
        self.ln_id = ln_id


class Lng:
    """Язык программированя"""

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


class OpLng:
    """
    Операторы языков программирования
    """

    def __init__(self, ln_id, op_id):
        self.ln_id = ln_id
        self.op_id = op_id


# Языки программирования
lngs = [
    Lng(1, 'Python'),
    Lng(2, 'C'),
    Lng(3, 'Java'),
    Lng(4, 'C++'),
]

# Операторы
ops = [
    Op(1, 'Сложение', 4, 1),
    Op(2, 'Умножение', 4, 2),
    Op(3, 'Сравнение', 2, 3),
    Op(4, 'Вызов функции', 8, 3),

]


# Языки и операторы, для связи многие-ко-многим
ln_op = [
    OpLng(1, 1),
    OpLng(2, 2),
    OpLng(3, 3),
    OpLng(3, 4),
    OpLng(1, 2),
    OpLng(1, 4),
    OpLng(2, 1),
    OpLng(2, 3),
    OpLng(3, 1),
```

```
    OpLng(3, 2),
    OpLng(1, 4),
    OpLng(4, 4),

]

def main():
    """Основная функция"""
    one_to_many = [(op.op_name, op.memory, ln.name)
                   for ln in lngs
                   for op in ops
                   if op.ln_id == ln.id]

    many_to_many_temp = [(ln.name, lop.ln_id, lop.op_id)
                         for ln in lngs
                         for lop in ln_op
                         if ln.id == lop.ln_id]
    many_to_many = [(op.op_name, op.memory, language)
                    for language, ln_id, op_id in many_to_many_temp
                    for op in ops if op.id == op_id]

    print('Задание B1')
    task1 = []
    for op_name, memory, name in one_to_many:
        if name[0] == "J":
            task1.append((name, op_name))
    print(task1)

    print('\nЗадание B2')
    mas2 = []
    for ln in lngs:
        op_lns = list(filter(lambda i: i[2] == ln.name, one_to_many))
        if len(op_lns) > 0:
            ln_memory = [memory for _, memory, _ in op_lns]
            Memor = min(ln_memory)
            mas2.append((ln.name, Memor))
    task2 = sorted(mas2, key=itemgetter(1))
    print(task2)

    print('\nЗадание B3')
    mas3 = []
    for op_name, memory, name in many_to_many:
        mas3.append((op_name, name))

    task3 = list(sorted(mas3, key=itemgetter(0)))
    print(task3)


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

Экранные формы с примерами выполнения программы:



Задание B1
[('Java', 'Сравнение'), ('Java', 'Вызов функции')]

Задание B2
[('Java', 2), ('Python', 4), ('C', 4)]

Задание B3
[('Вызов функции', 'Python'), ('Вызов функции', 'Python'), ('Вызов функции', 'Java'), ('Вызов функции', 'C++'), ('Сложение', 'Python'), ('Сложение', 'C'), ('Сложение', 'Java'), ('Сравнение', 'C'), ('Сравнение', 'Java'), ('Умножение', 'Python'), ('Умножение', 'C'), ('Умножение', 'Java')]

Process finished with exit code 0