Московский государственный технический университет им. Н.Э. Баумана

Факультет «Парадигмы и конструкции языков программирования»
Кафедра ИУ5 «Системы обработки информации и управления»

РК № 1
Вариант 6(Г)

Курс «Основы информатики»

Отчет по лабораторной работе №2
«Программирование разветвляющихся алгоритмов»

Выполнил:
студент группы ИУ5-34Б:
Каятский Павел Евгеньевич
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2023 г.

Условия рубежного контроля №1 по курсу ПиК ЯП

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.

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).

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

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результаты ее выполнения.

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.

Код программы

```
#Вариант Г, вариант 6( Класс 1 - Дом, Класс 2 - улица)

"""""«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.
«Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список улиц с максимальной высотой дома на каждой улице, отсортированный по максимальной высоте.
«Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех связанных домов и улиц, отсортированный по улицам, сортировка по сотрудникам произвольная.
"""
from operator import itemgetter

class House:
    def __init__(self, id, number, height, street_id):
        self.id = id
        self.number = number
        self.height = height
        self.street_id = street_id

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

class House_Street:
    def __init__(self,house_id, street_id):
        self.house_id = house_id
        self.street_id = street_id


Houses = [
    House(1, 3, 7, 1),
    House(2, 13, 9, 2),
    House(3, 15, 5, 2),
    House(4, 45, 15, 3),
    House(5, 5, 7, 1),
    House(6,88,22,5),
    House(7,54,19,4),
    House(8,15,15,5),

]

Streets = [
```

```
    Street(1, "Zeleniy prospect"),
    Street(2, "Abramovskaya street"),
    Street(3, "Adelmanovskaya shosse"),
    Street(4, "Voskresenskaya street"),
    Street(5, "Babaevskiy proezd"),
    Street(6, "Timura Frunze street"),
    Street(7,"Peshehodniy bulvar"),
]

Houses_Streets = [
    House_Street(1,1),
    House_Street(1,2),
    House_Street(1,3),
    House_Street(2,4),
    House_Street(2,5),
    House_Street(3,7),
    House_Street(3,6),
    House_Street(3,5),
    House_Street(4,4),
    House_Street(4,1),
    House_Street(5,2),


]

def main():
    one_to_many = [(h.number,h.height, s.name)
                   for h in Houses
                   for s in Streets
                   if h.street_id == s.id]
    many_to_many_temp = [(s.name, hs.house_id, hs.street_id)
                         for s in Streets
                         for hs in Houses_Streets
                         if s.id == hs.street_id]
    many_to_many = [(h.number, h.height, street_name)
                    for street_name, street_id, house_id in many_to_many_temp
                    for h in Houses if h.id == house_id]



    print("Task Г1")
    for i in range(0, len(one_to_many)):
        sname_temp = one_to_many[i][2]
        if sname_temp[0] == "A":
            print(one_to_many[i])

    print("Task Г2")
    one_to_many_temp = []
    sname_temp = []
    for i in range(0, len(one_to_many)):
        temp1 = one_to_many[i]
        temp1_sname = one_to_many[i][2]
```

```
        temp1_max_height = one_to_many[i][1]
        for j in range(i + 1, len(one_to_many)):
            if one_to_many[j][2] == temp1_sname and one_to_many[j][1] > temp1_max_height:
                temp1 = one_to_many[j]
        if temp1 not in one_to_many_temp:
            if temp1[2] not in sname_temp:
                one_to_many_temp.append(temp1)
                sname_temp.append(temp1[2])
    print(sorted(one_to_many_temp, key=itemgetter(1)))

    print("Task Г3")
    print(sorted(many_to_many, key=itemgetter(2)))


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

Результат выполнения программы

```
Task Г1
(13, 9, 'Abramovskaya street')
(15, 5, 'Abramovskaya street')
(45, 15, 'Adelmanovskaya shosse')
Task Г2
[(3, 7, 'Zeleniy prospect'), (13, 9, 'Abramovskaya street'), (45, 15, 'Adelmanovskaya shosse'), (54, 19, 'Voskresenskaya street'), (88, 22, 'Babaevskiy proezd')]
Task Г3
[(13, 9, 'Abramovskaya street'), (13, 9, 'Abramovskaya street'), (15, 5, 'Adelmanovskaya shosse'), (5, 7, 'Babaevskiy proezd'), (5, 7, 'Babaevskiy proezd'), (54, 19, 'Peshehodniy bulvar'), (88, 22, 'Timura Frunze street'), (45, 15, 'Voskresenskaya street'), (45, 15, 'Voskresenskaya street'), (3, 7, 'Zeleniy prospect'), (3, 7, 'Zeleniy prospect')]

Process finished with exit code 0
```