Группа ИУ5-32Б

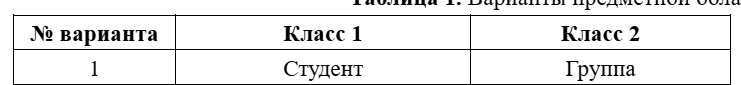
Студент Астахов Иван

**Рубежный контроль 1**

**Полученное задание:**

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).

**Вариант Б.**

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по сотрудникам, сортировка по отделам произвольная.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с количеством сотрудников в каждом отделе, отсортированный по количеству сотрудников.
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их отделов

**Программа:**

from operator import itemgetter  
"""Студент"""  
class Student:  
 def \_\_init\_\_(self, id, name, age, id\_group):  
 self.id = id  
 self.name = name  
 self.age = age  
 self.id\_group = id\_group  
"""Группа"""  
class Group:  
 def \_\_init\_\_(self, id, name, course):  
 self.id = id  
 self.name = name  
 self.course = course  
class Student\_Group:  
 def \_\_init\_\_(self, student\_id, group\_id):  
 self.student\_id = student\_id  
 self.group\_id = group\_id  
students = [  
 Student(1, "Emelyanov D.B.", 20, 2),  
 Student(2, "Semenov E.Y", 22, 6),  
 Student(3, "Dmitriev S.A", 21, 17),  
 Student(4, "Pavlenko T.D", 23, 13),  
 Student(5, "Ivanov P.A.", 20, 3),  
 Student(6, "Petrov I.N.", 21, 1),  
 Student(7, "Sidorov O.K.", 22, 6),  
 Student(8, "Orlov V.M.", 20, 2)  
]  
groups = [  
 Group(1, "Group A", 1),  
 Group(2, "Group B", 2),  
 Group(3, "Group C", 3),  
 Group(4, "Group D", 4),  
 Group(5, "Group E", 5)  
]  
groups\_students = [  
 Student\_Group(1, 1),  
 Student\_Group(2, 2),  
 Student\_Group(3, 3),  
 Student\_Group(3, 2),  
 Student\_Group(4, 1),  
 Student\_Group(5, 3),  
 Student\_Group(6, 4),  
 Student\_Group(7, 5),  
 Student\_Group(8, 2)  
]  
def first\_task(one\_to\_many):  
 res\_1 = sorted(one\_to\_many, key=itemgetter(0))  
 return res\_1  
def second\_task(one\_to\_many):  
 temp\_dict = {}  
 for student\_name, student\_id\_group, group\_name in one\_to\_many:  
 if group\_name in temp\_dict:  
 temp\_dict[group\_name] += 1  
 else:  
 temp\_dict[group\_name] = 1  
  
 res\_2 = [(group\_name, count) for group\_name, count in temp\_dict.items()]  
 res\_2.sort(key=itemgetter(1), reverse=True)  
 return res\_2  
def third\_task(many\_to\_many, end\_ch):  
 res\_3 = [(student\_name, group\_name) for student\_name, student\_id, group\_name in many\_to\_many if  
 student\_name.split()[0].endswith(end\_ch)]  
 return res\_3  
def main():  
 one\_to\_many = [(st.name, st.id\_group, gr.name)  
 for st in students  
 for gr in groups  
 if st.id\_group == gr.id]  
  
 many\_to\_many\_temp = [(st.name, sg.student\_id, sg.group\_id)  
 for st in students  
 for sg in groups\_students  
 if sg.student\_id == st.id]  
  
 many\_to\_many = [(student\_name, student\_id, gr.name)  
 for student\_name, student\_id, group\_id in many\_to\_many\_temp  
 for gr in groups if gr.id == group\_id]  
  
 print('Задание 1')  
 print(first\_task(one\_to\_many))  
  
 print("\nЗадание 2")  
 print(second\_task(one\_to\_many))  
  
 print("\nЗадание 3")  
 print(third\_task(many\_to\_many, 'ov'))  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Вывод программы:**  
  
  
  
