Московский государственный технический университет
им. Н.Э. Баумана

Факультет информатика и системы управления
Кафедра системы обработки информации и управления

Курс «Парадигмы и конструкции языков программирования» Отчет по рубежному контролю №1
Вариант Б2

Выполнил:
Студент группы ИУ5-32Б
Ковригина Д.М.
Подпись и дата:

Проверил:
Преподаватель кафедры ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Москва, 2025 г.

Постановка задачи

В рамках рубежного контроля №1 по курсу «Парадигмы и конструкции языков программирования» требовалось разработать программу на языке Python.

Цель работы:

1. Создать два класса данных, соответствующие предметной области «Школьник» (Класс 1) и «Класс» (Класс 2) (Вариант №2).
2. Реализовать связь «один-ко-многим» между классами.
3. Создать дополнительный класс для реализации связи «многие-ко-многим».
4. Создать списки объектов классов, содержащие тестовые данные (3-5 записей).
5. Реализовать три запроса в соответствии с Вариантом Б, адаптировав их под выбранную предметную область.
6. При реализации запросов использовать функциональные возможности Python (list/dict comprehensions).

Задачи (Вариант Б):

1. «Класс» и «Школьник» связаны соотношением «один-ко-многим». Вывести список всех связанных классов и школьников, отсортированный по школьникам (по ФИО).
2. «Класс» и «Школьник» связаны соотношением «один-ко-многим». Вывести список классов с количеством школьников в каждом, отсортированный по количеству школьников.
3. «Класс» и «Школьник» связаны соотношением «многие-ко-многим». Вывести список всех школьников, у которых фамилия заканчивается на «ов», и названия их классов.

Текст программы

```python
class Student:
    def __init__(self, id, fio, grade, class_id):
        self.id = id
        self.fio = fio
        self.grade = grade
        self.class_id = class_id

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

class StudentClass:
    def __init__(self, class_id, student_id):
        self.class_id = class_id
        self.student_id = student_id

classes = [
    SchoolClass(1, "7А"),
    SchoolClass(2, "7Б"),
    SchoolClass(3, "8В"),
    SchoolClass(4, "8Г"),
]

students = [
    Student(1, "Иванов", 4.5, 1),
    Student(2, "Петров", 3.8, 2),
    Student(3, "Сидоров", 4.2, 3),
    Student(4, "Кузнец", 4.8, 3),
    Student(5, "Никитин", 3.9, 3),
    Student(6, "Беляев", 4.1, 4),
]

students_classes = [
    StudentClass(1, 1),
    StudentClass(3, 2),
    StudentClass(3, 3),
    StudentClass(3, 4),
    StudentClass(2, 5),
    StudentClass(4, 6),
    StudentClass(4, 2),
    StudentClass(2, 1),
]

def main():
    one_to_many = [[stud.fio, stud.grade, cl.name]
                   for stud in students
                   for cl in classes
                   if stud.class_id == cl.id]

    print("--- Запрос Б1 ---")
```

```
    print("Список всех связанных школьников и классов (1:M), отсортированный по школьникам:")
    arr1 = sorted(one_to_many, key=lambda x: x[0])
    for i in arr1:
        print(f" Школьник: {i[0]}, Оценка: {i[1]}, Класс: {i[2]}")

    print("\n--- Запрос Б2 ---")
    print("Список классов с количеством школьников в каждом (1:M), отсортированный по количеству (по возрастанию):")

    arr2 = []
    for cl in classes:
        studs_in_class = list(filter(lambda x: x[2] == cl.name, one_to_many))
        if len(studs_in_class) > 0:
            arr2.append((cl.name, len(studs_in_class)))

    arr2.sort(key=lambda x: x[1])
    for i in arr2:
        print(f" Класс: {i[0]}, Количество школьников: {i[1]}")

    print("\n--- Запрос Б3 ---")
    print("Список всех школьников, у которых фамилия заканчивается на 'ов', и названия их классов (M:M):")

    many_to_many_first = [[cl.name, sc.class_id, sc.student_id]
                          for cl in classes
                          for sc in students_classes
                          if cl.id == sc.class_id]

    many_to_many = [[stud.fio, class_name]
                    for class_name, class_id, stud_id in many_to_many_first
                    for stud in students
                    if stud.id == stud_id]

    arr3 = []
    for fio, class_name in many_to_many:
        if fio.endswith("ов"):
            arr3.append([fio, class_name])

    arr3.sort(key=lambda x: x[0])
    for i in arr3:
        print(f" Школьник: {i[0]}, Класс: {i[1]}")

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

Скриншот работы программы

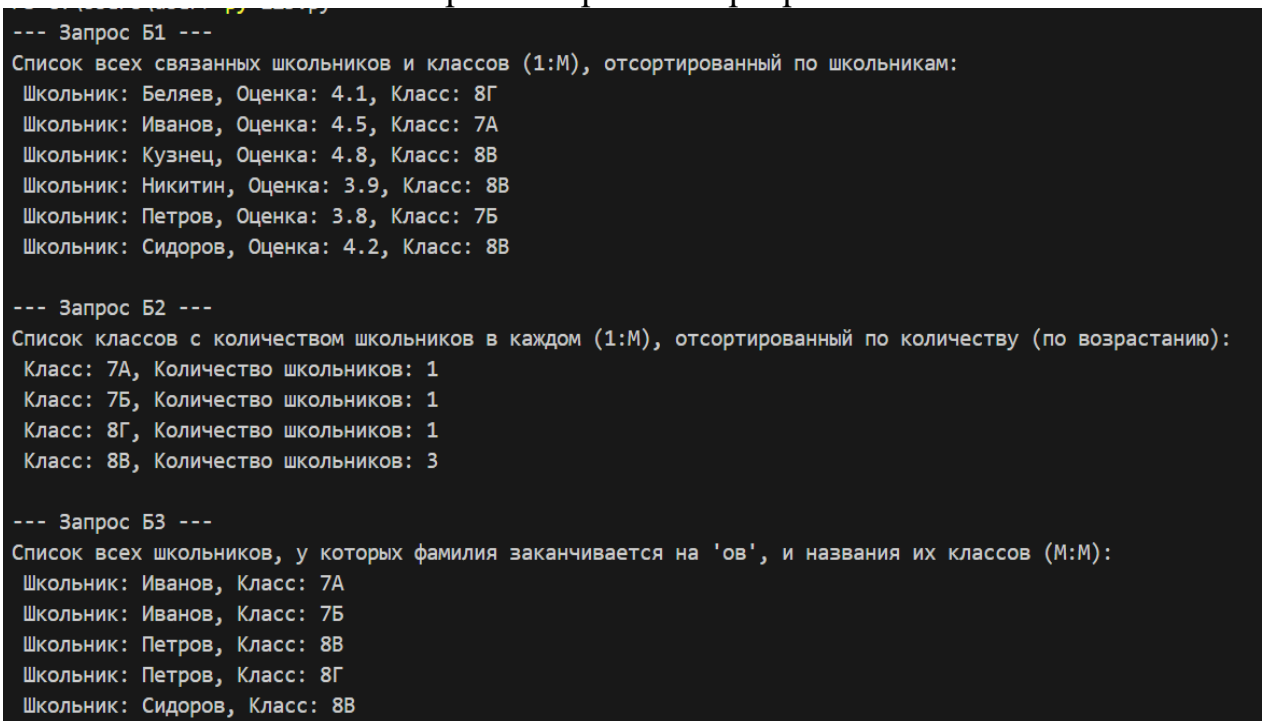

Рис. 1 Вывод результатов программы