Защищено:
Гапанюк Ю.Е.

"__"_____**2025** г.

Демонстрация:
Константинов А.А.

"__"_____**2025** г.

Отчет по рубежному контролю № 1 по курсу Парадигмы и конструкции языков программирования

Вариант №26

5
(количество листов)

студент группы ИУ5Ц-53Б _____
(подпись)

Константинов А.А. "__"_____**2025** г.

1. **Описание задания**

Вариант предметной области:

26	Студенческая группа	Учебный курс

Запросы под мою предметную область (Д):

1. «Учебный курс» и «Студенческая группа» связаны соотношением один-ко-многим (один курс может быть назначен нескольким группам). Выведите список всех групп, у которых название заканчивается на «-01», и названия их учебных курсов.

2. «Учебный курс» и «Студенческая группа» связаны соотношением один-ко-многим. Выведите список курсов со средним количеством студентов в группах, которые на него записаны, отсортированный по этому среднему количеству (отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений).

3. «Студенческая группа» и «Учебный курс» связаны соотношением многие-ко-многим (студенты из одной группы могут записываться на разные курсы, и на одном курсе могут быть студенты из разных групп). Выведите список всех курсов, у которых название начинается с «Математика», и список групп, которые на них записаны.

2. **Листинг программы**

```
class StudyCourse:
    def __init__(self, course_id, name):
        self.id = course_id
        self.name = name

class StudentGroup:
    def __init__(self, group_id, name, student_count, course_id):
        self.id = group_id
        self.name = name
        self.student_count = student_count
        self.course_id = course_id

class GroupCourse:
    def __init__(self, group_id, course_id):
        self.group_id = group_id
        self.course_id = course_id

# Создаем объекты классов с тестовыми данными
courses = [
    StudyCourse(1, "Математика для инженеров"),
    StudyCourse(2, "Физика"),
```

```
    StudyCourse(3, "Математический анализ"),
    StudyCourse(4, "Программирование"),
    StudyCourse(5, "Математика для физиков")
]

groups = [
    StudentGroup(1, "ИУ-101", 25, 1),
    StudentGroup(2, "ФИ-201", 30, 2),
    StudentGroup(3, "ИУ-301", 20, 4),
    StudentGroup(4, "МТ-102", 28, 1),
    StudentGroup(5, "ИБМ-202", 32, 2),
    StudentGroup(6, "МТ-103-01", 22, 3),
    StudentGroup(7, "РК-302-01", 18, 4),
    StudentGroup(8, "ИБМ-203-01", 26, 5)
]

group_courses = [
    GroupCourse(1, 1),
    GroupCourse(2, 2),
    GroupCourse(3, 4),
    GroupCourse(4, 1),
    GroupCourse(5, 2),
    GroupCourse(6, 3),
    GroupCourse(7, 4),
    GroupCourse(8, 5),
    # Дополнительные связи для многих-ко-многим
    GroupCourse(1, 3),
    GroupCourse(2, 5),
    GroupCourse(3, 1)
]

# Запрос 1: Вывести список всех групп, у которых название заканчивается на «-01»,
# и названия их учебных курсов
print("Запрос 1:")
groups_ending_with_01 = [group for group in groups if group.name.endswith("-01")]
for group in groups_ending_with_01:
    course = next((course for course in courses if course.id == group.course_id),
None)
    if course:
        print(f"Группа: {group.name}, Курс: {course.name}")
print("")

# Запрос 2: Вывести список курсов со средним количеством студентов в группах,
# которые на него записаны, отсортированный по этому среднему количеству
print("Запрос 2:")
course_stats = []

for course in courses:
    # Находим все группы, связанные с этим курсом
    course_groups = [group for group in groups if group.course_id == course.id]

    if course_groups:
        total_students = sum(group.student_count for group in course_groups)
```

```
        group_count = len(course_groups)
        average_students = total_students / group_count
        course_stats.append((course, average_students, total_students, group_count))

# Сортируем по среднему количеству студентов
course_stats.sort(key=lambda x: x[1])

for course, avg_students, total_students, group_count in course_stats:
    print(f"Курс: {course.name}")
    print(f"  Среднее количество студентов: {avg_students:.2f}")
    print(f"  Всего студентов: {total_students}")
    print(f"  Количество групп: {group_count}")
print("")

# Запрос 3: Вывести список всех курсов, у которых название начинается с
«Математика»,
# и список групп, которые на них записаны (связь многие-ко-многим)
print("Запрос 3:")
math_courses = [course for course in courses if
course.name.startswith("Математика")]

for course in math_courses:
    print(f"Курс: {course.name}")

    # Находим все связи для этого курса
    course_relations = [rel for rel in group_courses if rel.course_id == course.id]

    if course_relations:
        # Находим все группы, связанные с этим курсом
        related_groups = [group for group in groups
                          if group.id in [rel.group_id for rel in course_relations]]

        for group in related_groups:
            print(f"  Группа: {group.name}, Количество студентов:
{group.student_count}")
    else:
        print("  На этот курс не записаны группы")
    print("")
```

3. **Результаты работы программы**

Запрос 1:
Группа: МТ-103-01, Курс: Математический анализ
Группа: РК-302-01, Курс: Программирование
Группа: ИБМ-203-01, Курс: Математика для физиков

Запрос 2:
Курс: Программирование
  Среднее количество студентов: 19.00
  Всего студентов: 38
  Количество групп: 2
Курс: Математический анализ
  Среднее количество студентов: 22.00
  Всего студентов: 22
  Количество групп: 1
Курс: Математика для физиков
  Среднее количество студентов: 26.00
  Всего студентов: 26
  Количество групп: 1
Курс: Математика для инженеров
  Среднее количество студентов: 26.50
  Всего студентов: 53
  Количество групп: 2
Курс: Физика
  Среднее количество студентов: 31.00
  Всего студентов: 62
  Количество групп: 2

Запрос 3:
Курс: Математика для инженеров
  Группа: ИУ-101, Количество студентов: 25
  Группа: ИУ-301, Количество студентов: 20
  Группа: МТ-102, Количество студентов: 28

Курс: Математика для физиков
  Группа: ФИ-201, Количество студентов: 30
  Группа: ИБМ-203-01, Количество студентов: 26