**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по РК №1

Вариант запросов: В

Вариант предметной области: 1

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-33Б |  | преподаватель каф. ИУ5 |
| Абитов М. Р. |  | Гапанюк Ю. Е. |
|  |  |  |

Москва, 2023 г.

**Вариант запросов B.**

# «Студент» и «Группа» связаны соотношением один-ко-многим. Выведите список всех деталей, которые начинаются с «Тормоз» и названия их производителей.

# «Студент» и «Группа» связаны соотношением один-ко-многим. Выведите список производителей с минимальной стоимостью деталей, отсортированный их по минимальной стоимости.

# «Студент» и «Группа» связаны соотношением многие-ко-многим. Выведите список всех связанных деталей и производителей, отсортированный по производителям, сортировка деталей по цене.

# **Код программы**

class Student:

    def \_\_init\_\_(self, student\_id, name, group\_id, academic\_rating):

        self.student\_id = student\_id

        self.name = name

        self.group\_id = group\_id

        self.academic\_rating = academic\_rating

class Group:

    def \_\_init\_\_(self, group\_id, name):

        self.group\_id = group\_id

        self.name = name

# Создаем объекты класса Group

group1 = Group(1, "Группа A")

group2 = Group(2, "Группа B")

group3 = Group(3, "Группа C")

# Создаем объекты класса Student

student1 = Student(1, "Андреев", 1, 90)

student2 = Student(2, "Борисов", 1, 85)

student3 = Student(3, "Абрамов", 2, 92)

student4 = Student(4, "Григорьев", 2, 88)

student5 = Student(5, "Иванов", 3, 94)

# Создаем список "Студенты и их группы" для связи один-ко-многим

student\_group = [

    (student1, group1),

    (student2, group1),

    (student3, group2),

    (student4, group2),

    (student5, group3)

]

def main():

    # Задание В1

    print("Задание В1")

    for student, group in student\_group:

        if student.name.startswith('А'):

            print(f"{student.name} - {group.name}")

    # Задание В2

    print("\nЗадание В2")

    group\_min\_ratings = {}

    for student, group in student\_group:

        if group.name in group\_min\_ratings:

            if student.academic\_rating < group\_min\_ratings[group.name]:

                group\_min\_ratings[group.name] = student.academic\_rating

        else:

            group\_min\_ratings[group.name] = student.academic\_rating

    sorted\_groups = sorted(group\_min\_ratings.items(), key=lambda x: x[1])

    for group, min\_rating in sorted\_groups:

        print(f"{group} - Минимальный рейтинг: {min\_rating}")

    # Задание В3

    print("\nЗадание В3")

    for student, group in student\_group:

        print(f"{student.name} - {group.name}")

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# **Результат**

# 