**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 №2

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.

**Условия рубежного контроля №2 по курсу ПиКЯП:**

Рубежный контроль представляет собой разработку тестов на языке Python.

1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

# **Код программы main.py**

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

def find\_students\_starting\_with\_A(student\_group):

    return [(student.name, group.name) for student, group in student\_group if student.name.startswith('А')]

def find\_group\_min\_ratings(student\_group):

    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

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

def list\_students\_groups(student\_group):

    return [(student.name, group.name) for student, group in student\_group]

def main():

    # 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)

    ]

    # Задание В1

    print("Задание В1")

    for name, group\_name in find\_students\_starting\_with\_A(student\_group):

        print(f"{name} - {group\_name}")

    # Задание В2

    print("\nЗадание В2")

    for group, min\_rating in find\_group\_min\_ratings(student\_group):

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

    # Задание В3

    print("\nЗадание В3")

    for name, group\_name in list\_students\_groups(student\_group):

        print(f"{name} - {group\_name}")

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

    main()

# **Код программы test\_program.py**

from main import Student, Group, find\_students\_starting\_with\_A, find\_group\_min\_ratings, list\_students\_groups

import unittest

import sys

import os

# Добавляем путь к папке RK2 в PYTHONPATH

sys.path.insert(0, os.path.abspath(

    os.path.join(os.path.dirname(\_\_file\_\_), '..')))

class TestProgram(unittest.TestCase):

    def setUp(self):

        self.group1 = Group(1, "Группа A")

        self.group2 = Group(2, "Группа B")

        self.student\_group = [

            (Student(1, "Андреев", 1, 90), self.group1),

            (Student(2, "Борисов", 1, 85), self.group1),

            (Student(3, "Абрамов", 2, 92), self.group2)

        ]

    def test\_find\_students\_starting\_with\_A(self):

        result = find\_students\_starting\_with\_A(self.student\_group)

        self.assertEqual(len(result), 2)

        self.assertIn(("Андреев", "Группа A"), result)

    def test\_find\_group\_min\_ratings(self):

        result = find\_group\_min\_ratings(self.student\_group)

        self.assertEqual(result, [("Группа A", 85), ("Группа B", 92)])

    def test\_list\_students\_groups(self):

        result = list\_students\_groups(self.student\_group)

        self.assertEqual(len(result), 3)

        self.assertIn(("Борисов", "Группа A"), result)

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

    unittest.main()

# **Результат**

# 