

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ  
им. Н.Э. Баумана

Факультет “Информатика и системы управления”  
Кафедра “Системы обработки информации и управления”



Дисциплина “Парадигмы и конструкции языков программирования”

Отчет по РК №2

**Выполнил:**  
Студент группы ИУ5-34Б  
Невечеря А.Д.  
**Преподаватель:**  
Нардид А.Н.

Москва 2025

## Код программы

```
1  from typing import List, Tuple
2
3  class Teacher:
4      def __init__(self, id: int, surname: str, salary: int, course_id: int):
5          self.id = id
6          self.surname = surname
7          self.salary = salary
8          self.course_id = course_id
9
10     def __repr__(self):
11         return f"Teacher(id={self.id}, surname={self.surname!r}, salary={self.salary}, course_id={self.course_id})"
12
13     class Course:
14         def __init__(self, id: int, name: str):
15             self.id = id
16             self.name = name
17
18         def __repr__(self):
19             return f"Course(id={self.id}, name={self.name!r})"
20
21     class TeacherCourse:
22         def __init__(self, teacher_id: int, course_id: int):
23             self.teacher_id = teacher_id
24             self.course_id = course_id
25
26         def __repr__(self):
27             return f"TeacherCourse(teacher_id={self.teacher_id}, course_id={self.course_id})"
28
29     courses: List[Course] = [
30         Course(1, "Математический анализ"),
31         Course(2, "Физика"),
32         Course(3, "Программирование"),
33     ]
34
35     teachers: List[Teacher] = [
36         Teacher(1, "Абрамов", 50000, 1),
37         Teacher(2, "Антонов", 60000, 1),
38         Teacher(3, "Борисов", 55000, 2),
```

```

38     Teacher(3, "Борисов", 55000, 2),
39     Teacher(4, "Алексеев", 45000, 3),
40     Teacher(5, "Смирнов", 70000, 3),
41 ]
42
43 ✓ teacher_courses: List[TeacherCourse] = [
44     TeacherCourse(1, 1),
45     TeacherCourse(2, 1),
46     TeacherCourse(3, 2),
47     TeacherCourse(4, 3),
48     TeacherCourse(5, 3),
49     TeacherCourse(1, 3),
50 ]
51
52 course_by_id = {c.id: c for c in courses}
53 teacher_by_id = {t.id: t for t in teachers}
54
55 ✓ def query_1_teachers_starting_with_a(teachers: List[Teacher]) -> List[Tuple[str, str]]:
56     pairs = [
57         (t.surname, course_by_id[t.course_id].name)
58         for t in teachers
59         if t.surname.upper().startswith("A")
60     ]
61     return sorted(pairs, key=lambda x: (x[0], x[1]))
62
63 ✓ def query_2_min_salary_per_course(teachers: List[Teacher]) -> List[Tuple[str, int]]:
64     groups = {}
65     for t in teachers:
66         groups.setdefault(t.course_id, []).append(t.salary)
67
68     agg = [(course_by_id[cid].name, min(salaries)) for cid, salaries in groups.items()]
69     return sorted(agg, key=lambda x: x[1])
70
71 def query_3_all_teacher_course_m2m(links: List[TeacherCourse]) -> List[Tuple[str, str]]:
72     pairs = [(teacher_by_id[link.teacher_id].surname, course_by_id[link.course_id].name) for link in links]
73     return sorted(pairs, key=lambda x: x[0])
74
75 ✓ def main():
76     q1 = query_1_teachers_starting_with_a(teachers)
77     q2 = query_2_min_salary_per_course(teachers)
78     q3 = query_3_all_teacher_course_m2m(teacher_courses)
79
80     print("Задание B1")
81     for surname, course_name in q1:
82         print(f"    - {surname} - {course_name}")
83
84     print("Задание B2")
85     for course_name, min_salary in q2:
86         print(f"    - {course_name}: {min_salary} py6.")
87
88     print("Задание B3")
89     for surname, course_name in q3:
90         print(f"    - {surname} - {course_name}")
91
92 if __name__ == "__main__":
93     main()

```

Результат

Задание В1

- Абрамов – Математический анализ
- Алексеев – Программирование
- Антонов – Математический анализ

Задание В2

- Программирование: 45000 руб.
- Математический анализ: 50000 руб.
- Физика: 55000 руб.

Задание В3

- Абрамов – Математический анализ
- Абрамов – Программирование
- Алексеев – Программирование
- Антонов – Математический анализ
- Борисов – Физика
- Смирнов – Программирование