Московский государственный технический университет им. Н.Э. Баумана

Факультет Радиотехнический
Кафедра “Системы обработки информации и управления”

Курс «Парадигмы и конструкции языков программирования»

Отчет по Рубежному контролю №1
Вариант №1

Выполнил:
студент группы РТ5-31Б:
Аверьянов Александр
Дмитриевич
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич
Подпись и дата:

Москва, 2025 г.

Постановка задачи

Условия рубежного контроля №1 по курсу ПиК ЯП

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

1) Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Пример классов данных для предметной области Сотрудник-Отдел:

1. Класс «Сотрудник», содержащий поля:
   - ID записи о сотруднике;
   - Фамилия сотрудника;
   - Зарплата (количественный признак);
   - ID записи об отделе. (для реализации связи один-ко-многим)
2. Класс «Отдел», содержащий поля:
   - ID записи об отделе;
   - Наименование отдела.
3. (Для реализации связи многие-ко-многим) Класс «Сотрудники отдела», содержащий поля:
   - ID записи о сотруднике;
   - ID записи об отделе.

2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.

3) Необходимо разработать запросы в соответствии с Вашим вариантом. Запросы сформулированы в терминах классов «Сотрудник» и «Отдел», которые используются в примере. Вам нужно перенести эти требования в Ваш вариант предметной области. При разработке запросов необходимо по возможности использовать функциональные возможности языка Python (list/dict comprehensions, функции высших порядков).

Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результаты ее выполнения.

Вариант Е.

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех отделов, у которых в названии присутствует слово «отдел», и список работающих в них сотрудников.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате. Средняя зарплата должна быть округлена до 2 знака после запятой (*отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений; для округления необходимо использовать функцию https://docs.python.org/3/library/functions.html#round*).
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников, у которых фамилия начинается с буквы «А», и названия их отделов.

Таблица 1. Варианты предметной област

№ варианта	Класс 1	Класс 2
1	Студент	Группа

Текст программы

```
class Сотрудник:
    def __init__(self, id_сотрудника, фамилия, зарплата, id_отдела):
        self.id_сотрудника = id_сотрудника
        self.фамилия = фамилия
        self.зарплата = зарплата
        self.id_отдела = id_отдела  # Для связи один-ко-многим

class Отдел:
    def __init__(self, id_отдела, наименование):
        self.id_отдела = id_отдела
        self.наименование = наименование

class СотрудникиОтдела:
    def __init__(self, id_сотрудника, id_отдела):
        self.id_сотрудника = id_сотрудника
        self.id_отдела = id_отдела
```

```
# Тестовые данные для Отделов
отделы = [
    Отдел(1, "Отдел разработки"),
    Отдел(2, "Отдел маркетинга"),
    Отдел(3, "Отдел продаж"),
    Отдел(4, "Отдел поддержки"),
    Отдел(5, "Отдел аналитики")
]

# Тестовые данные для Сотрудников
сотрудники = [
    Сотрудник(1, "Иванов", 75000.0, 1),
    Сотрудник(2, "Петров", 82000.0, 1),
    Сотрудник(3, "Сидоров", 68000.0, 2),
    Сотрудник(4, "Алексеев", 71000.0, 3),
    Сотрудник(5, "Андреева", 79000.0, 4),
    Сотрудник(6, "Белов", 65000.0, 5),
    Сотрудник(7, "Афанасьев", 85000.0, 1),
    Сотрудник(8, "Григорьева", 73000.0, 3)
]

сотрудники_отдела = [
    СотрудникиОтдела(1, 1),
    СотрудникиОтдела(1, 3),
    СотрудникиОтдела(2, 1),
    СотрудникиОтдела(3, 2),
    СотрудникиОтдела(4, 3),
    СотрудникиОтдела(5, 4),
    СотрудникиОтдела(6, 5),
    СотрудникиОтдела(7, 1),
    СотрудникиОтдела(8, 3),
    СотрудникиОтдела(8, 5)
]

print("=== Запрос 1 ===")
```

```
отделы_с_словом_отдел = [отдел for отдел in отделы if "отдел" in
отдел.наименование.lower()]
print("Отделы, содержащие слово 'отдел':")
for отдел in отделы_с_словом_отдел:
    print(f"  - {отдел.наименование}")

print("\nСотрудники, работающие в этих отделах:")
for отдел in отделы_с_словом_отдел:
    сотрудники_в_отделе = [сотр for сотр in сотрудники if сотр.id_отдела ==
отдел.id_отдела]
    print(f"  {отдел.наименование}:")
    for сотр in сотрудники_в_отделе:
        print(f"    {сотр.фамилия} (ID: {сотр.id_сотрудника}, Зарплата:
{сотр.зарплата})")

print("\n=== Запрос 2 ===")
# Группировка сотрудников по отделам и вычисление средней зарплаты
средняя_зарплата_по_отделам = []
for отдел in отделы:
    зарплаты_в_отделе = [сотр.зарплата for сотр in сотрудники if
сотр.id_отдела == отдел.id_отдела]
    if зарплаты_в_отделе:  # Проверка, что в отделе есть сотрудники
        средняя_зарплата = sum(зарплаты_в_отделе) / len(зарплаты_в_отделе)
        средняя_зарплата = round(средняя_зарплата, 2)  # Округление до 2
знаков
        средняя_зарплата_по_отделам.append((отдел.наименование,
средняя_зарплата))

# Сортировка по средней зарплате (по убыванию)
средняя_зарплата_по_отделам.sort(key=lambda x: x[1], reverse=True)

print("Отделы, отсортированные по средней зарплате (убывание):")
for название, средняя in средняя_зарплата_по_отделам:
    print(f"  {название}: {средняя}")

print("\n=== Запрос 3 ===")
```

```
сотрудники_с_а = [сотр for сотр in сотрудники if
сотр.фамилия.startswith("А")]
print("Сотрудники, чья фамилия начинается с 'А':")
for сотр in сотрудники_с_а:
    отдел = next((отд for отд in отделы if отд.id_отдела == сотр.id_отдела),
None)
    отдел_название = отдел.наименование if отдел else "Неизвестный отдел"
    print(f"  {сотр.фамилия} (Отдел: {отдел_название})")
```

Анализ результатов

Входные данные:

- Отображены в терминале

Выходные данные:

```
E:\3sem\Парадигмы\РК1\.venv\Scripts\python.exe E:\3sem\Парадигмы\РК1\рк1.py
=== Запрос 1 ===
Отделы, содержащие слово 'отдел':
  - Отдел разработки
  - Отдел маркетинга
  - Отдел продаж
  - Отдел поддержки
  - Отдел аналитики

Сотрудники, работающие в этих отделах:
  Отдел разработки:
    Иванов (ID: 1, Зарплата: 75000.0)
    Петров (ID: 2, Зарплата: 82000.0)
    Афанасьев (ID: 7, Зарплата: 85000.0)
  Отдел маркетинга:
    Сидоров (ID: 3, Зарплата: 68000.0)
  Отдел продаж:
    Алексеев (ID: 4, Зарплата: 71000.0)
    Григорьева (ID: 8, Зарплата: 73000.0)
  Отдел поддержки:
    Андреева (ID: 5, Зарплата: 79000.0)
  Отдел аналитики:
    Белов (ID: 6, Зарплата: 65000.0)

=== Запрос 2 ===
Отделы, отсортированные по средней зарплате (убывание):
  Отдел разработки: 80666.67
  Отдел поддержки: 79000.0
  Отдел продаж: 72000.0
  Отдел маркетинга: 68000.0
  Отдел аналитики: 65000.0

=== Запрос 3 ===
Сотрудники, чья фамилия начинается с 'А':
  Алексеев (Отдел: Отдел продаж)
  Андреева (Отдел: Отдел поддержки)
  Афанасьев (Отдел: Отдел разработки)

Process finished with exit code 0
```