|  |  |
| --- | --- |
|  | Министерство науки и высшего образования Российской Федерации  Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»  (МГТУ им. Н.Э. Баумана) |
| ФАКУЛЬТЕТ «ИНЖЕНЕРНЫЙ БИЗНЕС И МЕНЕДЖМЕНТ»  КАФЕДРА «ПРОМЫШЛЕННАЯ ЛОГИСТИКА» (ИБМ-3)  Рубежный контроль номер 2  «Парадигмы и конструкции языков программирования»  38.03.05. Бизнес-информатика, Маркетинг цифровых технологий (уровень бакалавриата)  Студент ИБМ3- 34Б  \_\_\_\_\_\_\_\_\_\_ К.В.Разлада  (Подпись, дата)  Руководитель Ю.Е. Гапанюк  (Подпись, дата)  2024 г.  **1)Рефакторинг кода**  from typing import List, Tuple  class Detail:  def \_\_init\_\_(self, id\_detail: int, name: str, description: str, id\_supplier: int):  self.id\_detail = id\_detail  self.name = name  self.description = description  self.id\_supplier = id\_supplier  class Supplier:  def \_\_init\_\_(self, id\_supplier: int, name: str, address: str):  self.id\_supplier = id\_supplier  self.name = name  self.address = address  class DetailsSupplier:  def \_\_init\_\_(self, id\_detail: int, id\_supplier: int, price: float):  self.id\_detail = id\_detail  self.id\_supplier = id\_supplier  self.price = price  # Данные details = [  Detail(1, "Шайба", "Шайба М6", 1),  Detail(2, "Гайка", "Гайка М8", 2),  Detail(3, "Болт", "Болт М10", 3),  Detail(4, "Винт", "Винт саморез", 1),  Detail(5, "Шпилька", "Шпилька М12", 2) ]  suppliers = [  Supplier(1, "Поставщик 1", "ул. Ленина, 1"),  Supplier(2, "Поставщик 2", "ул. Мира, 2"),  Supplier(3, "Поставщик 3", "ул. Победы, 3") ]  details\_supplier = [  DetailsSupplier(1, 1, 10),  DetailsSupplier(2, 2, 20),  DetailsSupplier(3, 3, 30),  DetailsSupplier(4, 1, 40),  DetailsSupplier(5, 2, 50) ]  # Функции запросов def find\_details\_by\_supplier(supplier\_id: int) -> List[Detail]:  return [detail for detail in details if detail.id\_supplier == supplier\_id]  def find\_details\_and\_prices\_by\_supplier\_name(supplier\_name: str) -> List[Tuple[str, float]]:  return [  (detail.name, ds.price)  for detail in details  for ds in details\_supplier  if detail.id\_detail == ds.id\_detail  for supplier in suppliers  if ds.id\_supplier == supplier.id\_supplier and supplier.name == supplier\_name  ]  def find\_suppliers\_by\_detail\_name(detail\_name: str) -> List[Supplier]:  return [  supplier  for supplier in suppliers  for detail in details  for ds in details\_supplier  if supplier.id\_supplier == ds.id\_supplier and ds.id\_detail == detail.id\_detail and detail.name == detail\_name  ]  **2)** **Модульные тесты (с использованием** unittest**)**  import unittest from details\_suppliers import Detail, Supplier, DetailsSupplier, find\_details\_by\_supplier, find\_details\_and\_prices\_by\_supplier\_name, find\_suppliers\_by\_detail\_name  class TestDetailsSuppliers(unittest.TestCase):  def test\_find\_details\_by\_supplier(self):  result = find\_details\_by\_supplier(1)  self.assertEqual(len(result), 2) # Поставщик 1 поставляет 2 детали  self.assertEqual(result[0].name, "Шайба")  self.assertEqual(result[1].name, "Винт")   def test\_find\_details\_and\_prices\_by\_supplier\_name(self):  result = find\_details\_and\_prices\_by\_supplier\_name("Поставщик 2")  self.assertEqual(len(result), 2) # Поставщик 2 поставляет 2 детали  self.assertIn(("Гайка", 20), result)  self.assertIn(("Шпилька", 50), result)   def test\_find\_suppliers\_by\_detail\_name(self):  result = find\_suppliers\_by\_detail\_name("Шайба")  self.assertEqual(len(result), 1) # Только 1 поставщик для детали "Шайба"  self.assertEqual(result[0].name, "Поставщик 1")  if \_\_name\_\_ == "\_\_main\_\_":  unittest.main()  **3)Результаты тестов** | |