Нестерова Екатерина, ИУ5-33Б

РК2 по ПиКЯП

Вариант:

**Вариант В.**

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех сотрудников, у которых фамилия начинается с буквы «А», и названия их отделов.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с минимальной зарплатой сотрудников в каждом отделе, отсортированный по минимальной зарплате.
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по сотрудникам, сортировка по отделам произвольная.



**Текст программы:**

import unittest  
  
class HardDisk:  
 *"""Жесткий диск"""* def \_\_init\_\_(self, id, capacity, computer\_id):  
 self.id = id  
 self.capacity = capacity  
 self.computer\_id = computer\_id  
  
  
class Computer:  
 *"""Компьютер"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
class HardDiskComputer:  
 *"""  
 'Жесткие диски в компьютере' для реализации  
 связи многие-ко-многим  
 """* def \_\_init\_\_(self, computer\_id, hard\_disk\_id, count):  
 self.computer\_id = computer\_id  
 self.hard\_disk\_id = hard\_disk\_id  
 self.count = count  
  
  
computers = [  
 Computer(1, 'Рабочая станция'),  
 Computer(2, 'Сервер'),  
 Computer(3, 'Ноутбук'),  
 Computer(4, 'Планшет'),  
 Computer(5, 'Смартфон'),  
 Computer(6, 'Игровая консоль'),  
]  
  
# Жесткие диски  
hard\_disks = [  
 HardDisk(1, '1 ТБ', 1),  
 HardDisk(2, '2 ТБ', 1),  
 HardDisk(3, '500 ГБ', 2),  
 HardDisk(4, '1 ТБ', 2),  
 HardDisk(5, '128 ГБ', 3),  
 HardDisk(6, '256 ГБ', 3),  
 HardDisk(7, '64 ГБ', 4),  
 HardDisk(8, '128 ГБ', 4),  
 HardDisk(9, '16 ГБ', 5),  
 HardDisk(10, '32 ГБ', 5),  
]  
  
# Связь жестких дисков и компьютеров  
hard\_disks\_computers = [  
 HardDiskComputer(1, 1, 2),  
 HardDiskComputer(1, 2, 1),  
 HardDiskComputer(2, 3, 1),  
 HardDiskComputer(2, 4, 2),  
 HardDiskComputer(3, 5, 1),  
 HardDiskComputer(3, 6, 1),  
 HardDiskComputer(4, 7, 1),  
 HardDiskComputer(4, 8, 1),  
 HardDiskComputer(5, 9, 1),  
 HardDiskComputer(5, 10, 1),  
 HardDiskComputer(2, 1, 3),  
 HardDiskComputer(3, 2, 2),  
 HardDiskComputer(4, 3, 1),  
 HardDiskComputer(5, 4, 1),  
 HardDiskComputer(6, 5, 2),  
 HardDiskComputer(6, 6, 3),  
 HardDiskComputer(1, 7, 2),  
 HardDiskComputer(3, 8, 1),  
 HardDiskComputer(2, 9, 1),  
 HardDiskComputer(1, 10, 3),  
]  
  
class TestHardDisk(unittest.TestCase):  
 def test\_hard\_disk\_init(self):  
 hard\_disk = HardDisk(1, "1TB", 1)  
 self.assertEqual(hard\_disk.id, 1)  
 self.assertEqual(hard\_disk.capacity, "1TB")  
 self.assertEqual(hard\_disk.computer\_id, 1)  
class TestComputer(unittest.TestCase):  
 def test\_computer\_init(self):  
 computer = Computer(1, "Workstation")  
 self.assertEqual(computer.id, 1)  
 self.assertEqual(computer.name, "Workstation")  
  
class TestHardDiskComputer(unittest.TestCase):  
 def test\_hard\_disk\_computer\_init(self):  
 hard\_disk\_computer = HardDiskComputer(1, 1, 2)  
 self.assertEqual(hard\_disk\_computer.computer\_id, 1)  
 self.assertEqual(hard\_disk\_computer.hard\_disk\_id, 1)  
 self.assertEqual(hard\_disk\_computer.count, 2)  
  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

