**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 предметной области: 8

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-33Б |  | преподаватель каф. ИУ5 |
| Горенков А.А. |  | Гапанюк Ю. Е. |
|  |  |  |

Москва, 2023 г.

**Условия рубежного контроля №2 по курсу ПиК ЯП**

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**

class HardDrive:  
 def \_\_init\_\_(self, hd\_id, capacity\_gb, computer\_id):  
 self.hd\_id = hd\_id  
 self.capacity\_gb = capacity\_gb  
 self.computer\_id = computer\_id  
  
  
class Computer:  
 def \_\_init\_\_(self, computer\_id, brand, model):  
 self.computer\_id = computer\_id  
 self.brand = brand  
 self.model = model  
 self.hard\_drives = []  
  
 def add\_hard\_drive(self, hard\_drive):  
 self.hard\_drives.append(hard\_drive)  
  
 def get\_brand\_starting\_with(self, letter):  
 return self.brand.startswith(letter)  
  
  
def filter\_computers\_by\_brand\_starting\_with(computers\_and\_hard\_drives, letter):  
 filtered\_computers = []  
 for computer, hard\_drive in computers\_and\_hard\_drives:  
 if computer.get\_brand\_starting\_with(letter):  
 filtered\_computers.append((computer, hard\_drive))  
 return filtered\_computers  
  
  
def get\_computers\_with\_minimum\_capacity(computers\_and\_hard\_drives):  
 min\_capacity\_by\_computer = {}  
 for computer, hard\_drive in computers\_and\_hard\_drives:  
 if computer.computer\_id in min\_capacity\_by\_computer:  
 if hard\_drive.capacity\_gb < min\_capacity\_by\_computer[computer.computer\_id]:  
 min\_capacity\_by\_computer[computer.computer\_id] = hard\_drive.capacity\_gb  
 else:  
 min\_capacity\_by\_computer[computer.computer\_id] = hard\_drive.capacity\_gb  
  
 sorted\_computers = sorted(min\_capacity\_by\_computer.items(), key=lambda x: x[1])  
 return sorted\_computers  
  
  
def sort\_computers\_and\_hard\_drives(computers\_and\_hard\_drives):  
 return sorted(computers\_and\_hard\_drives, key=lambda x: (x[0].brand, x[0].model))  
  
  
# Модульные тесты  
import unittest  
  
class TestComputerMethods(unittest.TestCase):  
 def setUp(self):  
 self.computer1 = Computer(1, "Dell", "XPS 13")  
 self.computer2 = Computer(2, "HP", "Pavilion")  
 self.hard\_drive1 = HardDrive(1, 512, 1)  
 self.hard\_drive2 = HardDrive(2, 256, 1)  
 self.hard\_drive3 = HardDrive(3, 1000, 2)  
 self.computers\_and\_hard\_drives = [  
 (self.computer1, self.hard\_drive1),  
 (self.computer1, self.hard\_drive2),  
 (self.computer2, self.hard\_drive3)  
 ]  
  
 def test\_get\_brand\_starting\_with(self):  
 self.assertTrue(self.computer1.get\_brand\_starting\_with("D"))  
 self.assertFalse(self.computer2.get\_brand\_starting\_with("D"))  
  
 def test\_filter\_computers\_by\_brand\_starting\_with(self):  
 filtered = filter\_computers\_by\_brand\_starting\_with(self.computers\_and\_hard\_drives, "D")  
 self.assertEqual(len(filtered), 2)  
 self.assertEqual(filtered[0][0].brand, "Dell")  
 self.assertEqual(filtered[1][0].brand, "Dell")  
  
 def test\_get\_computers\_with\_minimum\_capacity(self):  
 min\_capacity\_computers = get\_computers\_with\_minimum\_capacity(self.computers\_and\_hard\_drives)  
 self.assertEqual(len(min\_capacity\_computers), 2)  
 self.assertEqual(min\_capacity\_computers[0][1], 256)  
 self.assertEqual(min\_capacity\_computers[1][1], 1000)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

# **Результат**

# 