Самбиева Анжела Аюбовна

ИБМ3-34Б

Парадигмы и конструкции языков программирования

**Рубежный Контроль №2**

**Задание:**

Рубежный контроль представляет собой разработку тестов на языке Python.  
1) Проведите рефакторинг текста программы рубежного контроля No1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля No1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

1. Проведем рефакторинг текста программы из РК1, чтобы он был пригоден для модульного тестирования.

class Library:

*def* \_\_init\_\_(self, library\_id, name):

self.library\_id = library\_id

self.name = name

self.books = [] *# Список книг в библиотеке*

*def* add\_book(self, book):

self.books.append(book)

class ProgrammingLanguage:

*def* \_\_init\_\_(self, language\_id, name, book\_count):

self.language\_id = language\_id

self.name = name

self.book\_count = book\_count

self.libraries = [] *# Список библиотек, где используется язык*

*def* add\_library(self, library):

self.libraries.append(library)

class Book:

*def* \_\_init\_\_(self, book\_id, library\_id, language\_id):

self.book\_id = book\_id

self.library\_id = library\_id

self.language\_id = language\_id

class LibrarySystem:

*def* \_\_init\_\_(self):

self.libraries = []

self.programming\_languages = []

*def* add\_library(self, library):

self.libraries.append(library)

*def* add\_programming\_language(self, language):

self.programming\_languages.append(language)

*def* sorted\_libraries(self):

return sorted(self.libraries, key=*lambda* lib: lib.name)

*def* library\_book\_count(self):

library\_book\_count = {lib.name: len(lib.books) for lib in self.libraries}

return sorted(library\_book\_count.items(), key=*lambda* x: x[1], reverse=True)

*def* filter\_libraries\_by\_name(self, keyword):

return [lib for lib in self.libraries if keyword.lower() in lib.name.lower()]

*# Пример использования класса LibrarySystem*

library\_system = LibrarySystem()

*# Создание библиотек*

library1 = Library(1, "Центральная библиотека")

library2 = Library(2, "Городская библиотека")

library3 = Library(3, "Научная библиотека")

*# Создание языков программирования*

python = ProgrammingLanguage(1, "Python", 10)

java = ProgrammingLanguage(2, "Java", 15)

*# Создание книг*

book1 = Book(1, 1, 1)

book2 = Book(2, 1, 2)

book3 = Book(3, 2, 1)

book4 = Book(4, 3, 2)

*# Добавление книг в библиотеки*

library1.add\_book(book1)

library1.add\_book(book2)

library2.add\_book(book3)

library3.add\_book(book4)

*# Добавление библиотек в систему*

library\_system.add\_library(library1)

library\_system.add\_library(library2)

library\_system.add\_library(library3)

*# Добавление языков программирования в систему*

python.add\_library(library1)

python.add\_library(library2)

java.add\_library(library3)

*# Запросы и их вывод*

sorted\_libraries = library\_system.sorted\_libraries()

for lib in sorted\_libraries:

print(f"Библиотека: {lib.name}")

for book in lib.books:

lang = next((pl for pl in library\_system.programming\_languages if pl.language\_id == book.language\_id), None)

print(f" Язык: {lang.name if lang else 'Неизвестный язык'}")

library\_book\_counts = library\_system.library\_book\_count()

for lib, count in library\_book\_counts:

print(f"Библиотека: {lib}, Суммарное количество книг: {count}")

keyword = "библиотека"

filtered\_libraries = library\_system.filter\_libraries\_by\_name(keyword)

for lib in filtered\_libraries:

print(f"Библиотека: {lib.name}")

for book in lib.books:

lang = next((pl for pl in library\_system.programming\_languages if pl.language\_id == book.language\_id), None)

print(f" Язык: {lang.name if lang else 'Неизвестный язык'}")

1. Создаем модульные тесты

import unittest

class TestLibrarySystem(unittest.TestCase):

*def* setUp(self):

self.library\_system = LibrarySystem()

*# Создание библиотек*

self.library1 = Library(1, "Центральная библиотека")

self.library2 = Library(2, "Городская библиотека")

*# Создание языков программирования*

self.python = ProgrammingLanguage(1, "Python", 10)

self.library\_system.add\_library(self.library1)

self.library\_system.add\_library(self.library2)

self.library\_system.add\_programming\_language(self.python)

*def* test\_add\_library(self):

self.assertEqual(len(self.library\_system.libraries), 2)

new\_library = Library(3, "Научная библиотека")

self.library\_system.add\_library(new\_library)

self.assertEqual(len(self.library\_system.libraries), 3)

self.assertEqual(self.library\_system.libraries[-1].name, "Научная библиотека")

*def* test\_add\_book\_to\_library(self):

book = Book(1, 1, 1)

self.library1.add\_book(book)

self.assertEqual(len(self.library1.books), 1)

self.assertEqual(self.library1.books[0].book\_id, 1)

*def* test\_sorted\_libraries(self):

self.library1 = Library(1, "Библиотека А")

self.library2 = Library(2, "Библиотека Б")

self.library\_system.add\_library(self.library1)

self.library\_system.add\_library(self.library2)

sorted\_libs = self.library\_system.sorted\_libraries()

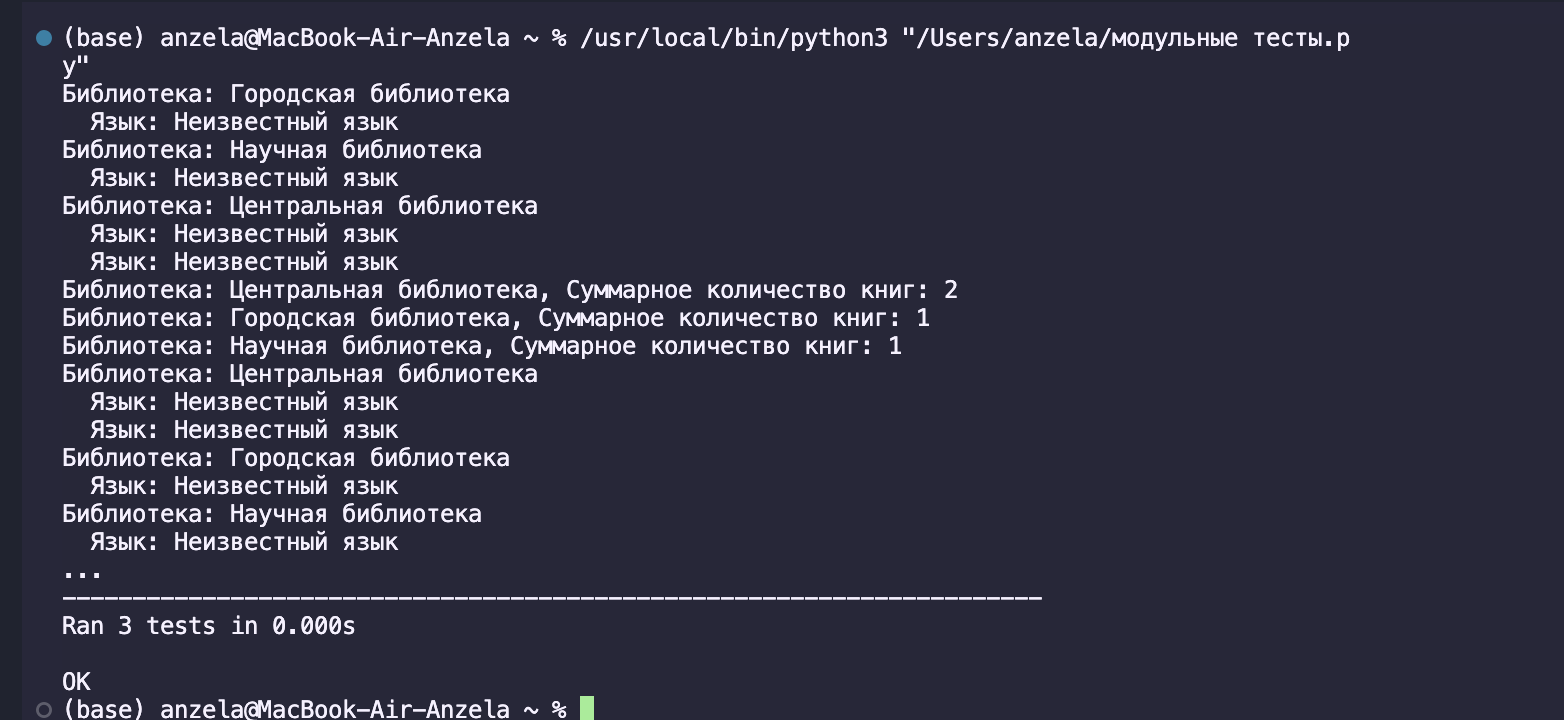
self.assertEqual(sorted\_libs[0].name, "Библиотека А")

self.assertEqual(sorted\_libs[1].name, "Библиотека Б")

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**Результат:**

****