**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Радиотехнический»

Кафедра «Информатика и вычислительная техника»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю №2

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы РТ5-31: |  | преподаватель каф. ИУ5 |
| Слкуни Г.Г. |  | Гапанюк Ю.Е.. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2023

**Текст программы:**

import unittest

from unittest.mock import patch

from main import get\_books\_by\_name, get\_sorted\_books, get\_filtered\_data, Book, BookChapter, Chapter

class TestProgram(unittest.TestCase):

    def setUp(self):

        self.books = [

            Book(1, 'Книга A', 'Автор A'),

            Book(2, 'Учебник B', 'Автор B'),

            Book(3, 'Книга C', 'Автор C'),

            Book(4, 'Учебник D', 'Автор D'),

            Book(5, 'Книга E', 'Автор E'),

            Book(6, 'Учебник F', 'Автор F'),

        ]

        self.chapters = [

            Chapter(1, 'Глава A', 1, 10),

            Chapter(2, 'Глава B', 2, 40),

            Chapter(3, 'Авторская Глава C', 3, 20),

            Chapter(4, 'Глава D', 4, 5),

            Chapter(5, 'Глава E', 5, 11),

            Chapter(6, 'Глава F', 6, 30),

            Chapter(7, 'Авторская Глава G', 1, 40),

            Chapter(8, 'Глава H', 2, 8),

            Chapter(9, 'Глава I', 3, 10),

            Chapter(10, 'Глава J', 4, 33),

            Chapter(11, 'Авторская Глава K', 5, 44),

            Chapter(12, 'Глава L', 6, 55),

        ]

        self.book\_chapters = [

            BookChapter(1, 1),

            BookChapter(1, 7),

            BookChapter(2, 2),

            BookChapter(2, 8),

            BookChapter(3, 3),

            BookChapter(3, 9),

            BookChapter(4, 4),

            BookChapter(4, 10),

            BookChapter(5, 5),

            BookChapter(5, 11),

            BookChapter(6, 6),

            BookChapter(6, 12),

        ]

    def test\_get\_books\_by\_name(self):

        result = get\_books\_by\_name('Книга', self.books, self.chapters)

        self.assertEqual(result, [

            (self.books[0].name, self.books[0].author, self.chapters[0].name, self.chapters[0].pages),

            (self.books[0].name, self.books[0].author, self.chapters[6].name, self.chapters[6].pages),

            (self.books[2].name, self.books[2].author, self.chapters[2].name, self.chapters[2].pages),

            (self.books[2].name, self.books[2].author, self.chapters[8].name, self.chapters[8].pages),

            (self.books[4].name, self.books[4].author, self.chapters[4].name, self.chapters[4].pages),

            (self.books[4].name, self.books[4].author, self.chapters[10].name, self.chapters[10].pages),

        ])

    def test\_get\_sorted\_books(self):

        result = get\_sorted\_books(self.books, self.chapters)

        self.assertEqual(result, [

            (self.books[5].name, 42.5),

            (self.books[4].name, 27.5),

            (self.books[0].name, 25.0),

            (self.books[1].name, 24.0),

            (self.books[3].name, 19.0),

            (self.books[2].name, 15.0),

        ])

    def test\_get\_filtered\_data(self):

        result = get\_filtered\_data('А', self.books, self.chapters, self.book\_chapters)

        self.assertEqual(result, [

            (self.chapters[6].name, self.books[0].name),

            (self.chapters[2].name, self.books[2].name),

            (self.chapters[10].name, self.books[4].name),

        ])

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

    unittest.main()

**Результат выполнения:**

----------------------------------------------------------------------

Ran 3 tests in 0.001s

OK