

**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 рубежному контролю №2

Вариант 22, Б

Выполнил:
студент группы ИУ5-33Б
Буров Р. Ю.

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.

Москва, 2025 г.

main.py

```
class Library:
```

```
    def __init__(self, id, name, lang_id):
```

```
        self.id = id
```

```
        self.name = name
```

```
        self.lang_id = lang_id
```

```
class Language:
```

```
    def __init__(self, id, title):
```

```
        self.id = id
```

```
        self.title = title
```

```
class LibraryLanguage:
```

```
    def __init__(self, lib_id, lang_id):
```

```
        self.lib_id = lib_id
```

```
        self.lang_id = lang_id
```

```
class ProgrammingData:
```

```
    def __init__(self):
```

```
        self.langs = [
```

```
            Language(1, "JavaScript"),
```

```
            Language(2, "Python"),
```

```
            Language(3, "TypeScript"),
```

```
        ]
```

```
        self.libs = [
```

```
            Library(1, "React", 1),
```

```
            Library(2, "Vue", 1),
```

```
            Library(3, "NumPy", 2),
```

```
            Library(4, "Django", 2),
```

```
            Library(5, "NestJS", 3),
```

```
]
```

```
self.libs_langs = [  
    LibraryLanguage(1, 1),  
    LibraryLanguage(1, 3),  
    LibraryLanguage(2, 1),  
    LibraryLanguage(3, 2),  
    LibraryLanguage(4, 2),  
    LibraryLanguage(5, 3),  
    LibraryLanguage(5, 1),  
]
```

```
def get_one_to_many(self):  
    """Создает связь один-ко-многим"""  
    return [(lib.name, lang.title)  
            for lang in self.langs  
            for lib in self.libs  
            if lib.lang_id == lang.id]
```

```
def get_many_to_many(self):  
    """Создает связь многие-ко-многим"""  
    many_to_many_temp = [(ll.lib_id, ll.lang_id) for ll in self.libs_langs]  
    return [(lib.name, lang.title)  
            for lib_id, lang_id in many_to_many_temp  
            for lib in self.libs if lib.id == lib_id  
            for lang in self.langs if lang.id == lang_id]
```

```
def func1(one_to_many):  
    """Сортирует библиотеки по названию языка программирования"""  
    return sorted(one_to_many, key=lambda x: x[1])
```

```
def func2(one_to_many, langs):
    """Сортирует языки по количеству библиотек"""
    res_b2_unsorted = []
    for lang in langs:
        libs_count = len([x for x in one_to_many if x[1] == lang.title])
        if libs_count > 0:
            res_b2_unsorted.append((lang.title, libs_count))
    return sorted(res_b2_unsorted, key=lambda x: x[1], reverse=True)
```

```
def func3(many_to_many):
    """Находит библиотеки, оканчивающиеся на 'JS'"""
    return [(lib_name, lang_title)
            for lib_name, lang_title in many_to_many
            if lib_name.endswith("JS")]
```

```
def main():
    data = ProgrammingData()
    one_to_many = data.get_one_to_many()
    many_to_many = data.get_many_to_many()

    print("Задание Б1: Отсортировать библиотеки по названию языка программирования")
    for lib in func1(one_to_many):
        print(f" {lib[0]} - {lib[1]}")

    print("\nЗадание Б2: Отсортировать языки по количеству библиотек")
    for lang, count in func2(one_to_many, data.langs):
        print(f" {lang}: {count} ")

    print("\nЗадание Б3: Найти библиотеки, оканчивающиеся на 'JS'")
    js_libs = func3(many_to_many)
    for lib_name, lang_title in js_libs:
```

```
print(f" {lib_name} - {lang_title}")
```

```
if __name__ == '__main__':  
    main()
```

test.py

```
import unittest
```

```
from main import ProgrammingData, func1, func2, func3
```

```
class TestProgrammingData(unittest.TestCase):
```

```
    def setUp(self):
```

```
        """Подготовка тестовых данных перед каждым тестом"""
```

```
        self.data = ProgrammingData()
```

```
        self.one_to_many = self.data.get_one_to_many()
```

```
        self.many_to_many = self.data.get_many_to_many()
```

```
    def test_func1_sort_libraries_by_language(self):
```

```
        """Тест 1: Проверка сортировки библиотек по языку программирования"""
```

```
        sorted_libs = func1(self.one_to_many)
```

```
        languages = [lang for _, lang in sorted_libs]
```

```
        self.assertEqual(languages, sorted(languages))
```

```
        expected_first = ("React", "JavaScript")
```

```
        expected_last = ("NestJS", "TypeScript")
```

```
        self.assertEqual(sorted_libs[0], expected_first)
```

```
        self.assertEqual(sorted_libs[-1], expected_last)
```

```
        self.assertEqual(len(sorted_libs), len(self.one_to_many))
```

```
    def test_func2_count_libraries_per_language(self):
```

```
        """Тест 2: Проверка подсчета библиотек для каждого языка"""
```

```
        lang_counts = func2(self.one_to_many, self.data.langs)
```

```
        counts = [count for _, count in lang_counts]
```

```

self.assertEqual(counts, sorted(counts, reverse=True))

expected_counts = [
    ("JavaScript", 2),
    ("Python", 2),
    ("TypeScript", 1),
]

self.assertEqual(lang_counts, expected_counts)

language_names = [lang for lang, _ in lang_counts]
self.assertIn("JavaScript", language_names)
self.assertIn("Python", language_names)
self.assertIn("TypeScript", language_names)

def test_func3_find_js_libraries(self):
    """Тест 3: Поиск библиотек, оканчивающихся на 'JS'"""

    js_libs = func3(self.many_to_many)

    for lib_name, _ in js_libs:
        self.assertTrue(lib_name.endswith('JS'))

    expected_libs = [("NestJS", "TypeScript"), ("NestJS", "JavaScript")]
    self.assertEqual(len(js_libs), len(expected_libs))

    for expected in expected_libs:
        self.assertIn(expected, js_libs)

    non_js_libs = [("Vue", "JavaScript"), ("NumPy", "Python"), ("Django", "Python")]
    for non_js in non_js_libs:
        self.assertNotIn(non_js, js_libs)

```

```
def test_programming_data_structure(self):
    """Дополнительный тест: Проверка структуры данных"""
    self.assertEqual(len(self.data.langs), 3)
    self.assertEqual(len(self.data.libs), 5)
    self.assertEqual(len(self.data.libs_langs), 7)
    self.assertEqual(len(self.one_to_many), 5)
    self.assertEqual(len(self.many_to_many), 7)

if __name__ == '__main__':
    unittest.main()
```

Скриншоты работы приложения

```
● artorias@LAPTOP-2ESPT302:~/IU5/PCPL/RK/RK2$ python3 main.py
Задание Б1: Отсортировать библиотеки по названию языка программирования
  React - JavaScript
  Vue - JavaScript
  NumPy - Python
  Django - Python
  NestJS - TypeScript

Задание Б2: Отсортировать языки по количеству библиотек
  JavaScript: 2
  Python: 2
  TypeScript: 1

Задание Б3: Найти библиотеки, оканчивающиеся на 'JS'
  NestJS - TypeScript
  NestJS - JavaScript

● artorias@LAPTOP-2ESPT302:~/IU5/PCPL/RK/RK2$ python3 -m unittest test.py -v
test_func1_sort_libraries_by_language (test.TestProgrammingData.test_func1_sort_libraries_by_language)
Тест 1: Проверка сортировки библиотек по языку программирования ... ok
test_func2_count_libraries_per_language (test.TestProgrammingData.test_func2_count_libraries_per_language)
Тест 2: Проверка подсчета библиотек для каждого языка ... ok
test_func3_find_js_libraries (test.TestProgrammingData.test_func3_find_js_libraries)
Тест 3: Поиск библиотек, оканчивающихся на 'JS' ... ok
test_programming_data_structure (test.TestProgrammingData.test_programming_data_structure)
Дополнительный тест: Проверка структуры данных ... ok

-----
Ran 4 tests in 0.001s

OK
```