Московский государственный технический университет им. Н.Э. Баумана

Факультет Радиотехнический
Кафедра РТ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю №2
Вариант предметной области №12: язык программирования, средство разработки

Выполнил:
студент группы РТ5-11Б:
Корчагина В.А.
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк
Подпись и дата:

Москва, 2025 г.

Описание задания

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).

Текст программы

refactored_program.py

```
class ProgrammingLanguage:
    """Класс 'Язык программирования'"""
    def __init__(self, id, name, popularity_score, ide_id):
        self.id = id
        self.name = name
        self.popularity_score = popularity_score
        self.ide_id = ide_id

    def __repr__(self):
        return f"ProgrammingLanguage(id={self.id}, name='{self.name}',
popularity={self.popularity_score}, ide_id={self.ide_id})"


class IDE:
    """Класс 'Средство разработки' (Integrated Development Environment)"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

    def __repr__(self):
        return f"IDE(id={self.id}, name='{self.name}')"


class LanguageIDE:
```

```
    """Класс для связи многие-ко-многим между Языком программирования и Средством разработки"""
    def __init__(self, language_id, ide_id):
        self.language_id = language_id
        self.ide_id = ide_id

    def __repr__(self):
        return f"LanguageIDE(language_id={self.language_id}, ide_id={self.ide_id})"


class ProgrammingSystem:
    """Основной класс для работы с данными"""
    def __init__(self, ides, languages, language_ide_links):
        self.ides = ides
        self.languages = languages
        self.language_ide_links = language_ide_links

    def get_test_data(self):
        """Возвращает тестовые данные"""
        return {
            'ides': self.ides,
            'languages': self.languages,
            'language_ide_links': self.language_ide_links
        }

    def task1(self):
        """Задание 1: Список всех средств разработки, у которых в названии присутствует слово 'Studio' или 'VS',
        и список языков программирования, которые с ними связаны"""
        result = {}

        # Находим IDE с нужными названиями
```

```
        target_ides = [ide for ide in self.ides if 'Studio' in ide.name or 'VS' in ide.name]

        for ide in target_ides:
            # Находим языки, связанные с этим IDE (один-ко-многим)
            related_languages = [lang for lang in self.languages if lang.ide_id == ide.id]
            result[ide.name] = [lang.name for lang in related_languages]

        return result

    def task2(self):
        """Задание 2: Список средств разработки со средней популярностью языков в каждом IDE,
        отсортированный по средней популярности"""
        result = []

        # Группируем языки по IDE
        ide_languages = {}
        for lang in self.languages:
            if lang.ide_id not in ide_languages:
                ide_languages[lang.ide_id] = []
            ide_languages[lang.ide_id].append(lang)

        # Вычисляем среднюю популярность для каждого IDE
        for ide_id, lang_list in ide_languages.items():
            ide_name = next(ide.name for ide in self.ides if ide.id == ide_id)
            total_popularity = sum(lang.popularity_score for lang in lang_list)
            avg_popularity = total_popularity / len(lang_list)
            result.append((ide_name, round(avg_popularity, 2)))

        # Сортируем по средней популярности
        result.sort(key=lambda x: x[1])
```

```
        return result

    def task3(self):
        """Задание 3: Список всех языков программирования, у которых название начинается с буквы 'C',
        и названия средств разработки, которые с ними связаны"""
        result = {}

        # Находим языки, начинающиеся на 'C'
        c_languages = [lang for lang in self.languages if lang.name.startswith('C')]

        for lang in c_languages:
            # Находим IDE, связанные с этим языком (многие-ко-многим)
            related_ide_ids = [link.ide_id for link in self.language_ide_links if link.language_id == lang.id]
            related_ides = [ide for ide in self.ides if ide.id in related_ide_ids]

            result[lang.name] = [ide.name for ide in related_ides]

        return result


def create_test_data():
    """Создание тестовых данных"""
    # Список средств разработки
    ides = [
        IDE(1, "Visual Studio"),
        IDE(2, "IntelliJ IDEA"),
        IDE(3, "PyCharm"),
        IDE(4, "Eclipse"),
        IDE(5, "VS Code")
    ]
```

```
    # Список языков программирования (связь один-ко-многим с IDE)
    languages = [
        ProgrammingLanguage(1, "Python", 95, 3),
        ProgrammingLanguage(2, "Java", 88, 2),
        ProgrammingLanguage(3, "C#", 85, 1),
        ProgrammingLanguage(4, "JavaScript", 92, 5),
        ProgrammingLanguage(5, "C++", 78, 1)
    ]

    # Список для связи многие-ко-многим
    language_ide_links = [
        LanguageIDE(1, 3),  # Python поддерживается в PyCharm
        LanguageIDE(1, 5),  # Python - VS Code
        LanguageIDE(2, 2),  # Java - IntelliJ IDEA
        LanguageIDE(2, 4),  # Java - Eclipse
        LanguageIDE(3, 1),  # C# - Visual Studio
        LanguageIDE(4, 5),  # JavaScript - VS Code
        LanguageIDE(5, 1),  # C++ - Visual Studio
        LanguageIDE(5, 5),  # C++ - VS Code
    ]

    return ides, languages, language_ide_links


def main():
    """Основная функция программы"""
    print("РУБЕЖНЫЙ КОНТРОЛЬ - ВАРИАНТ 12")
    print("Классы: Язык программирования - Средство разработки\n")

    # Создаем данные и систему
    ides, languages, links = create_test_data()
```

```
    system = ProgrammingSystem(ides, languages, links)

    # Вывод тестовых данных
    print("ТЕСТОВЫЕ ДАННЫЕ:")
    print("Средства разработки:", ides)
    print("Языки программирования:", languages)
    print("Связи многие-ко-многим:", links)
    print("\n" + "="*50 + "\n")

    # Выполнение заданий
    print("=== ЗАДАНИЕ 1 ===")
    print("Средства разработки, содержащие 'Studio' или 'VS', и связанные языки программирования:")
    task1_result = system.task1()
    for ide_name, langs in task1_result.items():
        print(f"\n{ide_name}:")
        for lang in langs:
            print(f"  - {lang}")

    print("\n=== ЗАДАНИЕ 2 ===")
    print("Средства разработки со средней популярностью языков (отсортировано):")
    task2_result = system.task2()
    for ide_name, avg_pop in task2_result:
        print(f"{ide_name}: {avg_pop}")

    print("\n=== ЗАДАНИЕ 3 ===")
    print("Языки программирования, начинающиеся на 'C', и связанные средства разработки:")
    task3_result = system.task3()
    for lang_name, ides_list in task3_result.items():
        print(f"\n{lang_name}:")
        for ide_name in ides_list:
```

```
            print(f"  - {ide_name}")
if __name__ == "__main__":
    main()
```

test_programming_system.py

```
import unittest
from refactored_program import ProgrammingSystem, create_test_data, IDE,
ProgrammingLanguage, LanguageIDE


class TestProgrammingSystem(unittest.TestCase):
    """Тесты для системы программирования"""

    def setUp(self):
        """Настройка тестовых данных перед каждым тестом"""
        ides, languages, links = create_test_data()
        self.system = ProgrammingSystem(ides, languages, links)

    def test_task1_studio_vs_ides_and_languages(self):
        """Тест задания 1: проверка IDE с 'Studio' или 'VS' и связанных языков"""
        result = self.system.task1()

        print("\n[Тест 1] Проверка IDE с 'Studio' или 'VS' и связанных языков")
        print(f"Найдено IDE: {list(result.keys())}")

        # Проверяем, что найдены правильные IDE
        expected_ides = {"Visual Studio", "VS Code"}
        self.assertEqual(set(result.keys()), expected_ides)

        # Проверяем языки для Visual Studio
        self.assertEqual(sorted(result["Visual Studio"]), ["C#", "C++"])
        print("  ✓ Visual Studio содержит C# и C++")

        # Проверяем языки для VS Code
```

```
    self.assertEqual(sorted(result["VS Code"]), ["JavaScript"])
    print("  ✓ VS Code содержит JavaScript")

def test_task2_average_popularity_sorted(self):
    """Тест задания 2: проверка средней популярности и сортировки"""
    result = self.system.task2()

    print("\n[Тест 2] Проверка средней популярности языков по IDE")
    print(f"Результат: {result}")

    # Проверяем структуру результата
    # Должно быть 4 записи (Eclipse не имеет связанных языков в связи один-ко-многим)
    self.assertEqual(len(result), 4)
    print(f"  ✓ Найдено {len(result)} IDE с языками")

    # Проверяем, что результат отсортирован по возрастанию средней популярности
    for i in range(len(result) - 1):
        self.assertLessEqual(result[i][1], result[i + 1][1])
    print("  ✓ Результат корректно отсортирован")

    # Проверяем конкретные значения
    result_dict = dict(result)
    self.assertEqual(result_dict["Visual Studio"], 81.5)  # (85 + 78) / 2
    print("  ✓ Visual Studio: средняя популярность 81.5")

    self.assertEqual(result_dict["IntelliJ IDEA"], 88.0)
    print("  ✓ IntelliJ IDEA: средняя популярность 88.0")

    self.assertEqual(result_dict["VS Code"], 92.0)
    print("  ✓ VS Code: средняя популярность 92.0")
```

```
        self.assertEqual(result_dict["PyCharm"], 95.0)
        print("  ✓ PyCharm: средняя популярность 95.0")

    def test_task3_c_languages_and_ides(self):
        """Тест задания 3: проверка языков на 'C' и связанных IDE"""
        result = self.system.task3()

        print("\n[Тест 3] Проверка языков на 'C' и связанных IDE")
        print(f"Найдено языков: {list(result.keys())}")

        # Проверяем, что найдены правильные языки
        expected_languages = {"C#", "C++"}
        self.assertEqual(set(result.keys()), expected_languages)

        # Проверяем IDE для C#
        self.assertEqual(result["C#"], ["Visual Studio"])
        print("  ✓ C# поддерживается в Visual Studio")

        # Проверяем IDE для C++
        self.assertEqual(sorted(result["C++"]), ["VS Code", "Visual Studio"])
        print("  ✓ C++ поддерживается в VS Code и Visual Studio")

class TestEdgeCases(unittest.TestCase):
    """Тесты граничных случаев"""

    def test_empty_data(self):
        """Тест с пустыми данными"""
        print("\n[Тест 4] Проверка работы с пустыми данными")
        system = ProgrammingSystem([], [], [])

        # Все задачи должны возвращать пустые результаты
```

```
        self.assertEqual(system.task1(), {})
        print("  ✓ Задание 1 возвращает пустой словарь")

        self.assertEqual(system.task2(), [])
        print("  ✓ Задание 2 возвращает пустой список")

        self.assertEqual(system.task3(), {})
        print("  ✓ Задание 3 возвращает пустой словарь")

    def test_custom_data(self):
        """Тест с пользовательскими данными"""
        print("\n[Тест 5] Проверка с пользовательскими данными")
        # Создаем тестовые данные
        ides = [
            IDE(1, "Custom Studio"),
            IDE(2, "Test IDE")
        ]

        languages = [
            ProgrammingLanguage(1, "Cobra", 90, 1),
            ProgrammingLanguage(2, "Crystal", 85, 2)
        ]

        links = [
            LanguageIDE(1, 1),
            LanguageIDE(2, 2)
        ]

        system = ProgrammingSystem(ides, languages, links)

        # Проверяем задание 1
```

```
        task1_result = system.task1()
        self.assertEqual(list(task1_result.keys()), ["Custom Studio"])
        print("  ✓ Custom Studio найден в задании 1")

        # Проверяем задание 3
        task3_result = system.task3()
        self.assertEqual(sorted(task3_result.keys()), ["Cobra", "Crystal"])
        print("  ✓ Языки Cobra и Crystal найдены в задании 3")

if __name__ == "__main__":
    print("=" * 60)
    print("МОДУЛЬНОЕ ТЕСТИРОВАНИЕ - ВАРИАНТ 12")
    print("=" * 60)

    # Запускаем тесты с минимальным выводом unittest
    runner = unittest.TextTestRunner(verbosity=0)
    suite = unittest.TestLoader().loadTestsFromModule(__import__(__name__))

    # Запускаем тесты и сохраняем результат
    result = runner.run(suite)

    print("\n" + "=" * 60)
    print("ИТОГИ ТЕСТИРОВАНИЯ:")
    print(f"Всего тестов: {result.testsRun}")
    print(f"Успешно: {result.testsRun - len(result.failures) - len(result.errors)}")
    print(f"Провалено: {len(result.failures)}")
    print(f"Ошибок: {len(result.errors)}")

    if result.failures:
        print("\nПроваленные тесты:")
        for test, traceback in result.failures:
            print(f"  - {test}")
```

```
    if result.errors:
        print("\nТесты с ошибками:")
        for test, traceback in result.errors:
            print(f"  - {test}")

    print("=" * 60)
```

Результат работы программы:

```
PS C:\Users\Victoria\Korchagina_PCPL\rk2> & C:/ProgramData/anaconda3/python.exe c:/Users/Victoria/Korcha
ina_PCPL/rk2/test_programming_system.py
============================================================
МОДУЛЬНОЕ ТЕСТИРОВАНИЕ - ВАРИАНТ 12
============================================================

[Тест 5] Проверка с пользовательскими данными
  ✓ Custom Studio найден в задании 1
  ✓ Языки Cobra и Crystal найдены в задании 3

[Тест 4] Проверка работы с пустыми данными
  ✓ Задание 1 возвращает пустой словарь
  ✓ Задание 2 возвращает пустой список
  ✓ Задание 3 возвращает пустой словарь

[Тест 1] Проверка IDE с 'Studio' или 'VS' и связанных языков
Найдено IDE: ['Visual Studio', 'VS Code']
  ✓ Visual Studio содержит C# и C++
  ✓ VS Code содержит JavaScript

[Тест 2] Проверка средней популярности языков по IDE
Результат: [('Visual Studio', 81.5), ('IntelliJ IDEA', 88.0), ('VS Code', 92.0), ('PyCharm', 95.0)]
  ✓ Найдено 4 IDE с языками
  ✓ Результат корректно отсортирован
  ✓ Visual Studio: средняя популярность 81.5
  ✓ IntelliJ IDEA: средняя популярность 88.0
  ✓ VS Code: средняя популярность 92.0
  ✓ PyCharm: средняя популярность 95.0
```

```
[Тест 3] Проверка языков на 'C' и связанных IDE
Найдено языков: ['C#', 'C++']
  ✓ C# поддерживается в Visual Studio
  ✓ C++ поддерживается в VS Code и Visual Studio
----------------------------------------------------------------------
Ran 5 tests in 0.007s

OK
```

```
============================================================
ИТОГИ ТЕСТИРОВАНИЯ:
Всего тестов: 5
Успешно: 5
Провалено: 0
Ошибок: 0
============================================================
PS C:\Users\Victoria\Korchagina_PCPL\rk2>
```