Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Кубанский государственный технологический университет»

(ФГБОУ ВО «КубГТУ»)

Факультет информационных технологий и кибербезопасности

Кафедра информационных систем и программирования

**ОТЧЕТ**

по лабораторной работе №9.

Тема работы: «Автоматизация создания тестовых двойников (Test Doubles)».

Наименование дисциплины: «Тестирование и отладка программного обеспечения».

Подготовил:

Студент группы 23-КБ-ПР1

Юданов М. Д.

Краснодар

2025

**1 Цель работы**

Цель работы – изучить подход к автоматизации создания тестовых двойников (test doubles) при помощи библиотеки Mockito для языка Java.

**2 Краткое теоретическое описание**

Тестовые двойники используются для изоляции тестируемого кода от внешних зависимостей. Библиотека NSubstitute (Mockito для Java) автоматически создаёт такие двойники для интерфейсов, делегатов и классов. С её помощью можно задавать возвращаемые значения методов и свойств, проверять, вызывались ли определённые методы, и имитировать исключения. Это позволяет тестировать поведение кода без участия реальных объектов или ресурсов.

**3 Задание**

1) На основе класса из предыдущей лабораторной работы (вариант задания оставить прежним) создать тестирующий проект, создающий тестовых двойников при помощи Mockito.

2) Протестировать класс.

3) Оформить отчёт.

1. **Ход работы**

1-3) Создал новый тестирующий проект для кода прошлой работы, используя Mockito, протестировал класс Graph и оформил отчёт.

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class MockitoGraphTest {

@Test

void readGraphTest() throws IOException {

IGraphReader graphReader = Mockito.mock(IGraphReader.class);

IGraphWriter graphWriter = Mockito.mock(IGraphWriter.class);

Mockito.when(graphReader.loadEdgeList(Mockito.any())).thenAnswer(\_ -> {

List<Edge> list = new ArrayList<>();

list.add(new Edge(1, 2, true));

list.add(new Edge(2, 4, true));

list.add(new Edge(4, 3, true));

list.add(new Edge(3, 1, true));

return list;

});

Graph graph = new Graph(graphReader, graphWriter);

graph.readFromFile(new StringReader(""));

graph.writeToFile(new StringWriter());

Assertions.assertEquals(4, graph.incidenceMatrix.length);

}

@Test

void isConnected() throws IOException {

IGraphReader graphReader = Mockito.mock(IGraphReader.class);

IGraphWriter graphWriter = Mockito.mock(IGraphWriter.class);

Mockito.when(graphReader.loadEdgeList(Mockito.any())).thenAnswer(\_ -> {

List<Edge> list = new ArrayList<>();

list.add(new Edge(1, 2, true));

list.add(new Edge(2, 4, true));

list.add(new Edge(4, 3, true));

list.add(new Edge(3, 1, true));

return list;

});

Graph graph = new Graph(graphReader, graphWriter);

graph.readFromFile(new StringReader(""));

Assertions.assertTrue(graph.isConnected());

}

@Test

void isNotConnected() throws IOException {

IGraphReader graphReader = Mockito.mock(IGraphReader.class);

IGraphWriter graphWriter = Mockito.mock(IGraphWriter.class);

Mockito.when(graphReader.loadEdgeList(Mockito.any())).thenAnswer(\_ -> {

List<Edge> list = new ArrayList<>();

list.add(new Edge(1, 2, true));

list.add(new Edge(2, 4, true));

list.add(new Edge(4, 3, true));

list.add(new Edge(1, 3, true));

return list;

});

Graph graph = new Graph(graphReader, graphWriter);

graph.readFromFile(new StringReader(""));

Assertions.assertFalse(graph.isConnected());

}

}

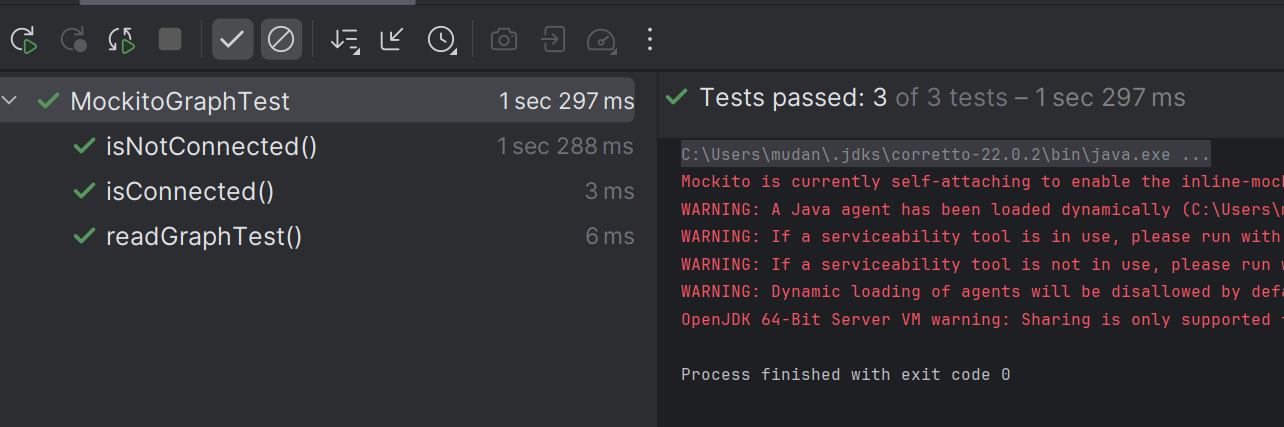


Рисунок 1 – Результат выполнения тестов.

**5 Вывод**

В ходе выполнения данной лабораторной работы удалось изучить подход к автоматизации создания тестовых двойников (test doubles) при помощи библиотеки Mockito для языка Java.