Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Кубанский государственный технологический университет»

(ФГБОУ ВО «КубГТУ»)

Институт компьютерных систем и информационной безопасности

Кафедра информационных систем и программирования

**ОТЧЕТ**

по лабораторной работе №4.

Тема работы: «Тестирование методом белого ящика».

Наименование дисциплины: «Тестирование и отладка программного обеспечения».

Подготовил:

Студент группы 23-КБ-ПР1

Булгаков В. В.

Краснодар

2025

**Цель работы**: изучить подход к тестированию методом белого ящика.

**Задание**

1) Создать класс (в соответствии с вариантом задания из п.5), реализующий обработку с текстовой информации.

2) Построить управляющий граф программы (УГП).

3) Используя знания о структуре программы определить набор тестов для тестирования методом белого ящика.

4) Указать какие пути реализуются (проверяются) тестом (см. таблица 4.1).

5) Указать какие критерии покрываются путем (см. таблица 4.2).

6) Оценить качество полученных тестовых данных с точки зрения покрытия кода тестами.

7) Протестировать метод с использованием средств автоматизации тестирования.

8) Составить отчет о результатах проведенного тестирования.

Вариант задания. 5. Создать класс, реализующий метод для вставки подстроки в произвольную позицию строки.

**Ход работы**:

class Substrings{

    InsertSubstring(str = '', pos, substring) {

        if (pos < 0) {                                           //1

            return ''                                            //2

        }

        if (pos > str.length) {                                  //3

            return ''                                            //4

        }

        let result = '';                                          //5

        for (let i = 0;                                           //6

            i < str.length;                                       //7

            i++)                                                  //8

            {

                if (i === pos) {                                  //9

                    result+=substring;                            //10

                }

                result+= str[i];                                  // 11

            }

        return result;                                            //12

    }

}

module.exports = Substrings;

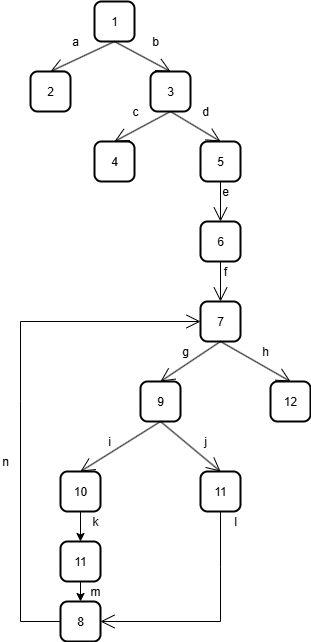


Рисунок 1 - УГП

Пути в УГП:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Путь** | **str** | **pos** | **substring** | **Результат** |
| 1 | (1,2) a | ‘h’ | -2 | ‘x’ | ‘’ |
| 2 | (1,3,4) bc | ‘h’ | 10 | ‘x’ | ‘’ |
| 3 | (1,3,5,6,7,12) bdefh | ‘’ | 0 | ‘x’ | ‘’ |
| 4 | (1,3,5,6,7,9,10,11,8,7,12) bdefgikmnh | ‘h’ | 0 | ‘x’ | ‘xh’ |
| 5 | (1,3,5,6,7,9,11,8,7,9,10,11,8,7,12) bdefgjlngikmnh | ‘hl’ | 1 | ‘x’ | ‘hxl’ |
| 6 | (1,3,5,6,7,9,10,11,8,7,9,11,8,7,12) bdefgikmngjlmnh | ‘hl’ | 0 | ‘x’ | ‘xhl’ |

Соответствие критериев покрытия выбранным путям

|  |  |  |
| --- | --- | --- |
| **№** | **Критерий** | **Пути** |
| 1 | Покрытие операторов | 4, 5 |
| 2 | Покрытие решений | 1, 2, 3, 4, 5 |
| 3 | Покрытие условий | 1, 2, 3, 4, 5 |
| 4 | Покрытие путей | 1, 2, 3, 4, 5, 6 |

Substring.test.js:

const Substrings = require('../src/Substrings')

describe('Substrings', () => {

    describe('Путь 1: pos < 0', () => {

        test('pos = -2, str = "h", substring = "x" → ""', () => {

            const s = new Substrings();

            expect(s.InsertSubstring('h', -2, 'x')).toBe('');

        });

    });

    describe('Путь 2:pos > str.length', () => {

        test('pos = 10, str = "h", substring = "x" → ""', () => {

            const s = new Substrings();

            expect(s.InsertSubstring('h', 10, 'x')).toBe('');

        });

    });

    describe('Путь 3: (1,3,5,6,7,12)', () => {

        test('str = "", pos = 0, substring = "x" → ""', () => {

            const s = new Substrings();

            expect(s.InsertSubstring('', 0, 'x')).toBe('');

        });

    });

    describe('Путь 4: (1,3,5,6,7,9,10,11,8,7,12)', () => {

        test('str = "h", pos = 0, substring = "x" → "xh"', () => {

            const s = new Substrings();

            expect(s.InsertSubstring('h', 0, 'x')).toBe('xh');

        });

    });

    describe('Путь 5: (1,3,5,6,7,9,11,8,7,9,10,11,8,7,12)', () => {

        test('str = "hl", pos = 1, substring = "x" → "hxl"', () => {

            const s = new Substrings();

            expect(s.InsertSubstring('hl', 1, 'x')).toBe('hxl');

        });

    });

    describe('Путь 6: (1,3,5,6,7,9,10,11,8,7,9,11,8,7,12)', () => {

        test('str = "hl", pos = 0, substring = "x" → "xhl"', () => {

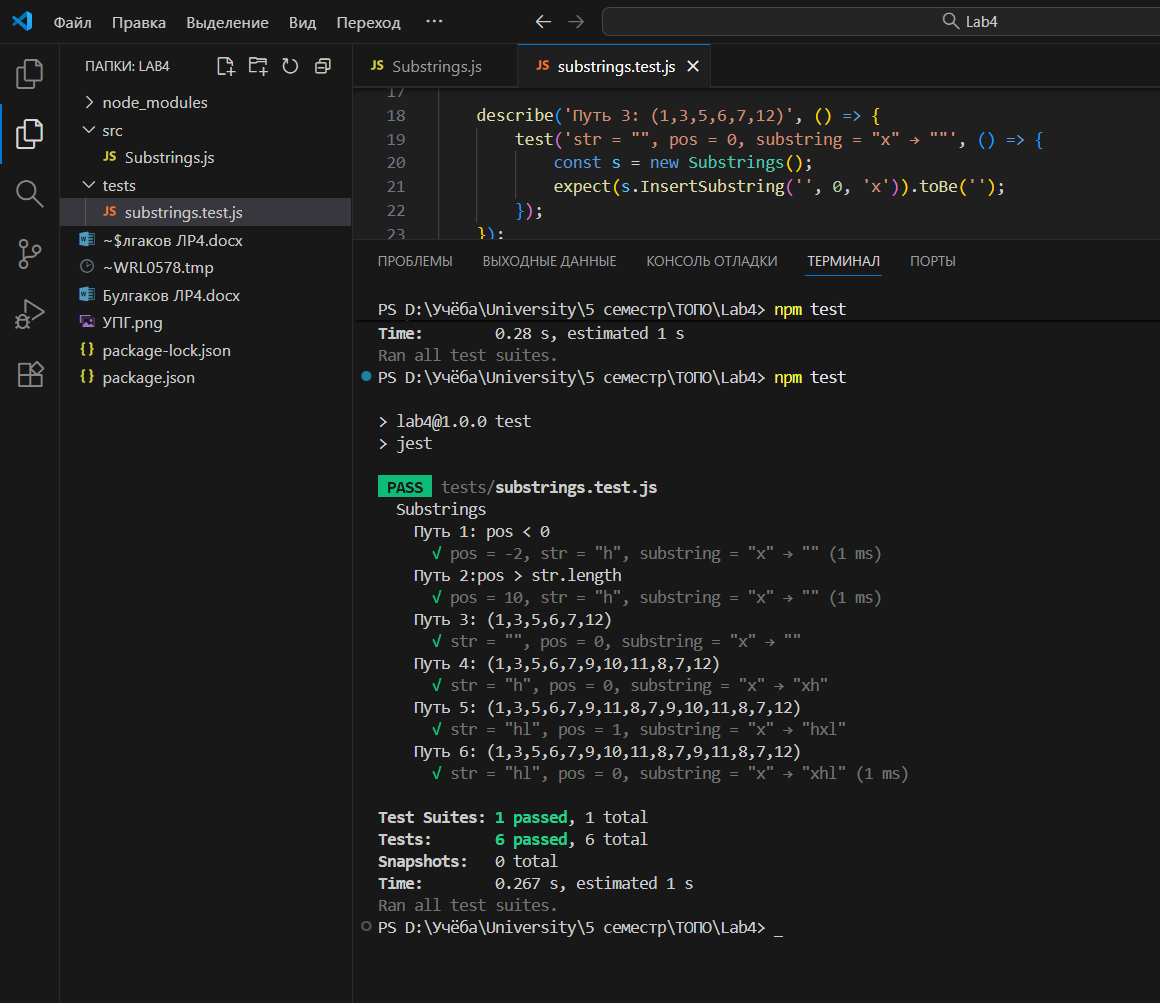
            const s = new Substrings();

            expect(s.InsertSubstring('hl', 0, 'x')).toBe('xhl');

        });

    });

});



Можно сделать вывод о достаточности набор тестовых данных (если каждый из критериев порывается хотя бы один из путей).

Вывод: изучен подход к тестированию методом белого ящика.