МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра информационных систем и цифровых технологий

**ОТЧЕТ**

по лабораторной работе №5

на тему: «Автоматизированное тестирование программного обеспечения»

по дисциплине: «Качество и тестирование программного обеспечения»

Вариант 10

Выполнил: Музалевский Н.С.

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 92ПГ

Проверил: Олькина Е.В.

Отметка о зачете:

Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г.

Орел, 2022 г.

**Код программы «Queen's Attack II»:**

namespace \_5

{

public static class QueenAttack

{

public static int QueenAttackCells()

{

if (ReadLine(out int n, out int k))

{

Position queen = new();

if (ReadLine(out queen.r, out queen.c))

{

Position[] barriers = new Position[k];

for (int i = 0; i < k; i++)

{

if (!ReadLine(out barriers[i].r, out barriers[i].c))

{

return -1;

}

}

return CellSearchForAttack(n, queen, barriers);

}

}

return -1;

}

static bool ReadLine(out int arg1, out int arg2)

{

string? str = Console.ReadLine();

if (!string.IsNullOrEmpty(str))

{

string[] arg = str.Split(' ');

if (arg.Length == 2)

{

arg1 = int.Parse(arg[0]);

arg2 = int.Parse(arg[1]);

return true;

}

}

arg1 = 0;

arg2 = 0;

return false;

}

public static int CellSearchForAttack(int n, Position queen, Position[] barriers)

{

int cellsAttack = 0;

for (int i = queen.r - 1; i > 0; i--)

{

if (IsFigure(i, queen.c, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.r + 1; i <= n; i++)

{

if (IsFigure(i, queen.c, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.c - 1; i > 0; i--)

{

if (IsFigure(queen.r, i, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.c + 1; i <= n; i++)

{

if (IsFigure(queen.r, i, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.r - 1, f = queen.c - 1; i > 0 && f > 0; i--, f--)

{

if (IsFigure(i, f, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.r + 1, f = queen.c + 1; i <= n && f <= n; i++, f++)

{

if (IsFigure(i, f, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.r - 1, f = queen.c + 1; i > 0 && f <= n; i--, f++)

{

if (IsFigure(i, f, barriers))

{

break;

}

cellsAttack++;

}

for (int i = queen.r + 1, f = queen.c - 1; i <= n && f > 0; i++, f--)

{

if (IsFigure(i, f, barriers))

{

break;

}

cellsAttack++;

}

return cellsAttack;

}

static bool IsFigure(int r, int c, Position[] barriers)

{

for (int i = 0; i < barriers.Length; i++)

{

if (barriers[i].r == r && barriers[i].c == c)

{

return true;

}

}

return false;

}

}

public struct Position

{

public int r;

public int c;

public Position(int r, int c)

{

this.r = r;

this.c = c;

}

}

}

**Код Unit-тестов для программы «Queen's Attack II»:**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Position = \_5.Position;

using QueenAttack = \_5.QueenAttack;

namespace UnitTest

{

[TestClass]

public class UnitTestQueenAttack

{

[TestMethod]

public void FirstTest()

{

int n = 4;

Position queen = new(4, 4);

Position[] barriers = System.Array.Empty<Position>();

int countCells = QueenAttack.CellSearchForAttack(n, queen, barriers);

Assert.IsTrue(countCells == 9);

}

[TestMethod]

public void SecondTest()

{

int n = 5;

Position queen = new(4, 3);

Position[] barriers = new Position[3];

barriers[0] = new(5, 5);

barriers[1] = new(4, 2);

barriers[2] = new(2, 3);

int countCells = QueenAttack.CellSearchForAttack(n, queen, barriers);

Assert.IsTrue(countCells == 10);

}

[TestMethod]

public void ThirdTest()

{

int n = 1;

Position queen = new(1, 1);

Position[] barriers = System.Array.Empty<Position>();

int countCells = QueenAttack.CellSearchForAttack(n, queen, barriers);

Assert.IsTrue(countCells == 0);

}

}

}

Результат Unit-тестов для программы «Queen's Attack II» изображён на рисунке 1.

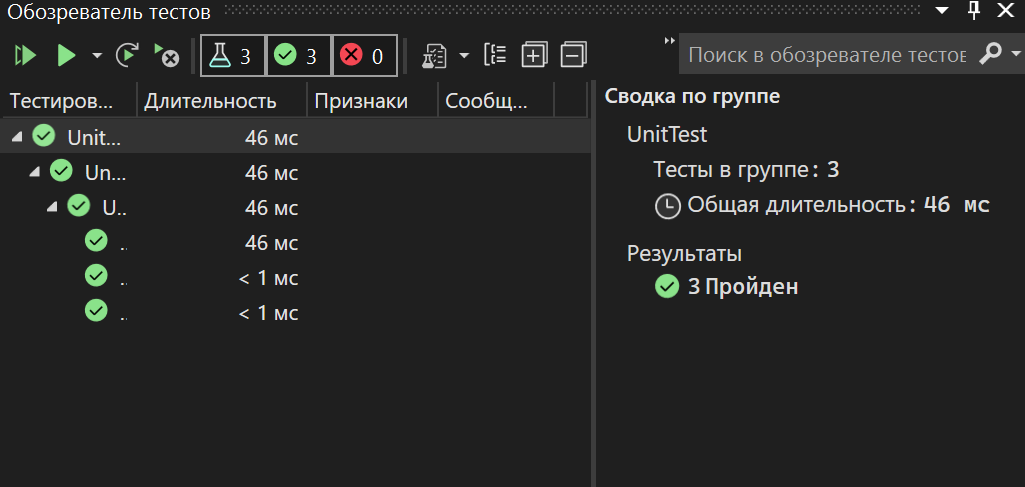


Рисунок 1 – Результат Unit-тестов