Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГТУ»)**

**Институт компьютерных систем и информационной безопасности**

**Кафедра информационных систем и программирования**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**Дисциплина: тестирование и отладка программного обеспечения**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Я.С Давков

Направление подготовки: 09.03.04 Программная инженерия

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Г. Волик

**1 Цель работы**

Изучить подход к тестированию методом черного ящика.

**2 Краткая теория**

Тестирование программного кода – процесс выполнения программного кода, направленный на выявление существующих в нем дефектов. Под дефектом здесь понимается участок программного кода, выполнение которого при определенных условиях приводит к неожиданному поведению системы (т.е. поведению, не соответствующему требованиям).

**3 Задание**

1) Создать класс (в соответствии с вариантом задания из п.5), реализующий проверку принадлежности точки различным областям плоскости, задаваемых пересечением фигур. 2) Выделить классы эквивалентности. 3) Проверить граничные значения для каждого из классов эквивалентности. 4) Составить наборы тестовых данных для созданного метода. 5) Протестировать метод на основе тестового набора с использованием программных отладочных средств. 6) Составить отчет о результатах проведенного тестирования.

**4 Выполнение работы**

Мой вариант – 4, полукруг и треугольник. Создадим требуемый класс.

Листинг программы класса 1:

public class AreaChecker  
{  
 public double R { get; set; }  
  
 public AreaChecker(double radius)  
 {  
 if (radius <= 0)  
 throw new ArgumentException("Радиус должен быть положительным числом");  
 R = radius;  
 }  
  
 public int TestPoint(double x, double y)  
 {  
 if (y >= 0 && (x \* x + y \* y <= R \* R))  
 {  
 return 1;  
 }  
  
 if (x <= 0 && y <= 0 && (Math.Abs(x) + Math.Abs(y) <= R))  
 {  
 return 2;  
 }  
  
 return 0;  
 }  
}

Выделим классы эквивалентности:

1. Точка внутри полукруга: y >= 0 и x^2+y^2 <= R^2
2. Точка внутри треугольника: x <= 0 и y <= 0 и |x| + |y| <= R
3. Точки вне всех областей: не 1 и не 2 эквивалентность

Напишим тесты для метода:

Листин тестов:

[TestFixture]  
public class AreaCheckerTests  
{  
 private AreaChecker checker;  
  
 [SetUp]  
 public void Setup()  
 {  
 checker = new AreaChecker(4.0);  
 }  
  
 [Test]  
 [TestCase(0, 0, 1, TestName = "Начало координат")]  
 [TestCase(0, 4, 1, TestName = "Верхняя точка полукруга")]  
 [TestCase(4, 0, 1, TestName = "Правая граница полукруга")]  
 [TestCase(-4, 0, 1, TestName = "Левая граница полукруга")]  
 [TestCase(2.82842712, 2.82842712, 1, TestName = "Диагональ 45° Точно")]  
 [TestCase(2, 2, 1, TestName = "Внутри полукруга I квадрант")]  
 [TestCase(-2, 2, 1, TestName = "Внутри полукруга II квадрант")]  
 [TestCase(3, 0, 1, TestName = "На оси X внутри радиуса")]  
 [TestCase(-3, 0, 1, TestName = "На оси X слева")]  
 [TestCase(0, 3, 1, TestName = "На оси Y внутри радиуса")]  
 public void TestPoint\_PointInSemicircle\_ReturnsOne(double x, double y, int expected)  
 {  
 int result = checker.TestPoint(x, y);  
 Assert.That(result, Is.EqualTo(expected));  
 }  
  
 [Test]  
 [TestCase(0, -4, 2, TestName = "Вершина треугольника ось Y")]  
 [TestCase(0, -2, 2, TestName = "Правый катет")]  
 [TestCase(-2, -2, 2, TestName = "Гипотенуза")]  
 [TestCase(-2, 0, 1, TestName = "Верхний катет")]  
 [TestCase(-1, -1, 2, TestName = "Внутри треугольника")]  
 [TestCase(-3, 0, 1, TestName = "На верхнем катете")]  
 [TestCase(-0.5, -0.5, 2, TestName = "Близко к началу координат")]  
 public void TestPoint\_PointInTriangle\_ReturnsTwo(double x, double y, int expected)  
 {  
 int result = checker.TestPoint(x, y);  
 Assert.That(result, Is.EqualTo(expected));  
 }  
  
 [Test]  
 [TestCase(5, 0, 0, TestName = "I квадрант вне")]  
 [TestCase(-5, 5, 0, TestName = "II квадрант вне")]  
 [TestCase(3, 4, 0, TestName = "Вне полукруга сверху")]  
 [TestCase(-5, -1, 0, TestName = "III квадрант вне треугольника")]  
 [TestCase(4.1, 0, 0, TestName = "За границей справа")]  
 [TestCase(2, -2, 0, TestName = "IV квадрант положительный X")]  
 public void TestPoint\_PointOutsideAreas\_ReturnsZero(double x, double y, int expected)  
 {  
 int result = checker.TestPoint(x, y);  
 Assert.That(result, Is.EqualTo(expected));  
 }  
  
 [Test]  
 public void Constructor\_NegativeRadius\_ThrowsArgumentException()  
 {  
 Assert.Throws<ArgumentException>(() => new AreaChecker(-1));  
 }  
  
 [Test]  
 public void Constructor\_ZeroRadius\_ThrowsArgumentException()  
 {  
 Assert.Throws<ArgumentException>(() => new AreaChecker(0));  
 }  
}

A screenshot of a computer

AI-generated content may be incorrect.

Рисунок 1 – результат выполнения тестов