МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГИЙ УПРАВЛІННЯ

Лабораторна робота № 2

З курсу «Якість і тестування програмного забезпечення»

«Модульне тестування. Частина 2. Модульне тестування в C#»

Виконав:

Студент групи КН 36-А

Кулик В.В.

Перевірила:

Бабкова Н. В.

ХАРКІВ 2018

**Цель работы:**

Знакомство с методами и техниками модульного тестирования на реальных примерах программного кода, выполненного на C# .

**Код на C#**:

public class

{

private int num;

public Model(int num)

{

this.num = num;

}

public double getNum()

{

return num;

}

public void setNum(int num)

{

this.num = num;

}

public override string ToString()

{

return num;

}

public bool soversh ()

{

if (num <= 3000 && num >= 0) {

for(int i=1;i<=num/2;i++){

if(num%i == 0){

temp += i;

}

}

if (temp == num) {

return true;

} else {

return false;

}

} else if (num > 3000) {

return false;

} else {

return false;

}

}

private static void Main(string[] arg)

{

Solve sol = new Solve(6);

Console.WriteLine(sol.soversh());

}

**Тестовый класс:**

[TestClass]

public class UnitTest1

{

[TestMethod]

public void testShortenFraction()

{

Solve sol = new Solve(6);

Assert.AreEqual(true, sol.soversh());

}

[TestMethod]

public void testShortenFractionSecond()

{

Solve sol = new Solve(0);

Assert.AreEqual(true, sol.soversh());

}

[TestMethod]

public void testShortenFractionThird()

{

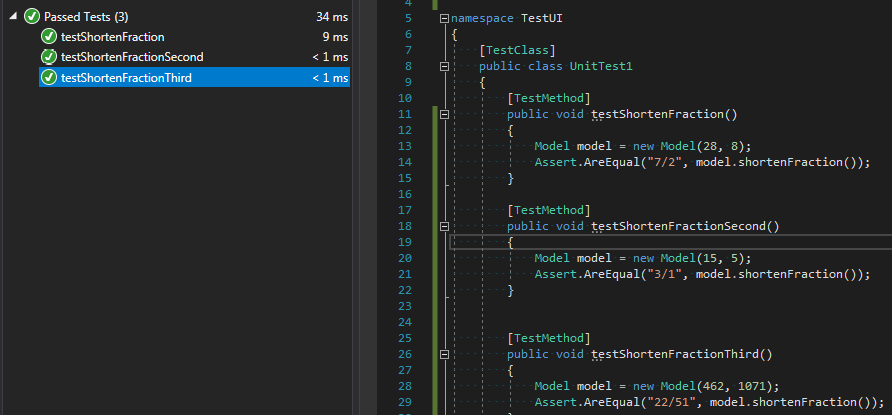
Solve sol = new Solve(7);

Assert.AreEqual(false, sol.soversh());

}

}

**Результаты тестирования:**



**Выводы:**

Выполняя лабораторную работу было совершено знакомство с методами и техниками модульного тестирования на реальных примерах программного кода, выполненного на языке программирования Java.

По результатам тестирования в данной программе не было обнаружено ошибок, все тесты были выполнены удачно.

По моему мнению, и NUnit и использование механизма исключений хорошо себя проявляют в модульном тестировании, но их можно использовать в разных сценариях, Try-Catch использовать как механизм защиты от случайных ошибок, а NUnit для тестирования конкретных модулей и дальнейшего анализа и устранения найденных ошибок.