**CST 236 Software System Testing**

Lab 3 – Unit Test using Nunit

Due: Monday, 03/05/2012

**Objective:** To understand how to unit test software code using Nunit feature of Visual Studio 2010. You will unit test the code you have written for Lab2.

Write the following test classes for lab2. These classes should throw an exception if the code fails.

* A class to test if the IQ score is > 200, the upper limit on the IQ score
* Class to test “very superior”
* Class to test “average”
* Class to test “Extremely low”
* Class to test CEO profession.

The criteria for IQ classification and profession categorization are shown in the following tables:

Wechler’s IQ scale:

|  |  |  |
| --- | --- | --- |
| IQ Range | Intelligent Classification | Recommended Occupation |
| >= 130 | Very Superior | Scientist |
| 120 – 129 | Superior | Innovators |
| 110 – 119 | High Average | CEOs |
| 90 – 109 | Average | Sorts Person |
| 80 – 89 | Low Average | Politicians, TV Anchor |
| 70 – 79 | Border Line | Used car salesman |
| <= 69 | Extremely Low | Worker Bees |

The following table provides a “not so real - wild guess” occupations:

|  |  |
| --- | --- |
| IQ Range | Recommended Occupation |
| >= 130 | Scientist |
| 120 – 129 | Innovators |
| 110 – 119 | CEOs |
| 90 – 109 | Sorts Person |
| 80 – 89 | Politicians, TV Anchor |
| 70 – 79 | Used car salesman |
| <= 69 | Worker Bees |

The overall IQ range that your program should accept is 20 – 200. The occupation should be based upon the classification.

**Approach:**

Here is a simple example of Nunit setup:

Here is a function that generates the Fibonacci sequence. Fibonacci sequence includes: 0, 1, 1,2,3,5,8 … where the next number is the sum of previous two numbers.

namespace ExtendededMath

{

public static class functions

{

public static int Fibonacci (int Factor)

{

if (factor < 2)

return (factor)

int x = Fibonacci (-- factor)

int y = Fibonacci (-- factor)

return (x+y)

}

}

}

This function can be unit tested by a test class where we call the Fibonacci with a known number and validate the result that you get with the known result. If the result does not match the Nunit will throw an exception.

Create a test project from the test template project. Create file type FunctionsTest.cs. Add the following to it:

using ExtenedMath;

Using Microsoft.VisualStudio.TestTools.UnitTesting; This is the framework for Unit testing.

Add the following code to the class:

[TestMethod]

public void FibonacciTest()

{

const int FACTOR = 8;

const int expected = 21;

int actual = ExtendedMath.Functions.Fibonacci (FACTOR);

Assert.AreEqual (Expected, Actual);

}

This will test your Fibonacci function.

Turning In: Email the program to me after making sure that it runs.