**PROG8170-23W-SEC3**

**Software Quality Assurance Tech**

**Assignment 2 – Unit Testing**

**Name:** Rutvik Gandhi

**Student Number:** 8809972

**Name:** Ricky Satyam

**Student Number:** 8826138

**Created On-** 02/09/2023

**Program.cs –**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assignment\_2

{

class Program

{

static void Main(string[] args)

{

int circleRadius;

Circle circle;

Console.WriteLine("Enter Raduis of the circle: ");

while (!int.TryParse(Console.ReadLine(), out circleRadius) || circleRadius <= 0)

{

Console.WriteLine("Wrong Input!! Enter valid Radius ");

}

circle = new Circle(circleRadius);

int select;

do

{

Console.WriteLine("1) Get Circle Radius");

Console.WriteLine("2) Change Circle Radius");

Console.WriteLine("3) Get Circle Circumference");

Console.WriteLine("4) Get Circle Area");

Console.WriteLine("5) Exit");

while (!int.TryParse(Console.ReadLine(), out select) || select < 1 || select > 5)

{

Console.WriteLine("You have entered wrong an wrong input !! please select again ");

}

switch (select)

{

case 1:

Console.WriteLine("Radius of circle is " + circle.GetRadius());

break;

case 2:

Console.WriteLine("Enter new radius ");

while (!int.TryParse(Console.ReadLine(), out circleRadius) || circleRadius <= 0)

{

Console.WriteLine("You have entered wrong an wrong input !!");

}

circle.SetRadius(circleRadius);

Console.WriteLine("The new Radius is " + circle.GetRadius());

break;

case 3:

Console.WriteLine("The Circumference is " + circle.GetCircumference());

break;

case 4:

Console.WriteLine("The Area is " + circle.GetArea());

break;

case 5:

Console.WriteLine("The Application is ended.");

break;

}

} while (select != 5);

}

}

}

**Circle.cs –**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assignment\_2

{

public class Circle

{

private int radius;

public Circle()

{

radius = 1;

}

public Circle(int radius)

{

this.radius = radius;

}

public int GetRadius()

{

return radius;

}

public void SetRadius(int radius)

{

this.radius = radius;

}

public double GetCircumference()

{

return 2 \* Math.PI \* radius;

}

public double GetArea()

{

return Math.PI \* Math.Pow(radius, 2);

}

}

}

**Circletest.cs –**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using NUnit.Framework;

using Assignment\_2;

namespace CircleTest

{

public class Circletest

{

[TestFixture]

public class Class1

{

public class CircleTests

{

[Test]

public void TestGetRadiusForFirstPositive()

{

Circle circle = new Circle(5);

int expected = 5;

int actual = circle.GetRadius();

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSetRadiusForSecondPositive()

{

Circle circle = new Circle(5);

circle.SetRadius(10);

int expected = 10;

int actual = circle.GetRadius();

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSetRadiusForNegative()

{

Circle circle = new Circle(10);

circle.SetRadius(-5);

int expected = -5;

int actual = circle.GetRadius();

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSetRadiusForZero()

{

Circle circle = new Circle(5);

circle.SetRadius(0);

int expected = 0;

int actual = circle.GetRadius();

Assert.AreEqual(expected, actual);

}

[Test]

public void TestRadiusAfterConstructor()

{

Circle circle = new Circle();

int expected = 1;

int actual = circle.GetRadius();

Assert.AreEqual(expected, actual);

}

[Test]

public void TestGetCircumferenceForPositiveRadius()

{

Circle circle = new Circle(5);

double expected = 31.42;

double actual = circle.GetCircumference();

Assert.AreEqual(expected, actual, 0.01);

}

[Test]

public void TestGetCircumferenceForRadiusZero()

{

Circle circle = new Circle(0);

double expected = 0;

double actual = circle.GetCircumference();

Assert.AreEqual(expected, actual, 0.01);

}

[Test]

public void TestGetCircumferenceForNegativeRadius()

{

Circle circle = new Circle(-5);

double expected = -31.42;

double actual = circle.GetCircumference();

Assert.AreEqual(expected, actual, 0.01);

}

[Test]

public void TestGetAreaForPositiveRadius()

{

Circle circle = new Circle(5);

double expected = 78.54;

double actual = circle.GetArea();

Assert.AreEqual(expected, actual, 0.01);

}

[Test]

public void TestGetAreaForRadiusZero()

{

Circle circle = new Circle(0);

double expected = 0;

double actual = circle.GetArea();

Assert.AreEqual(expected, actual, 0.01);

}

[Test]

public void TestGetAreaForNegativeRadius()

{

Circle circle = new Circle(-5);

double expected = 78.54;

double actual = circle.GetArea();

Assert.AreEqual(expected, actual, 0.01);

}

}

}

}

}

**Unit Test:**

**Graphical user interface, text, application

Description automatically generated**

**Git Repository log:**

**Text

Description automatically generated**