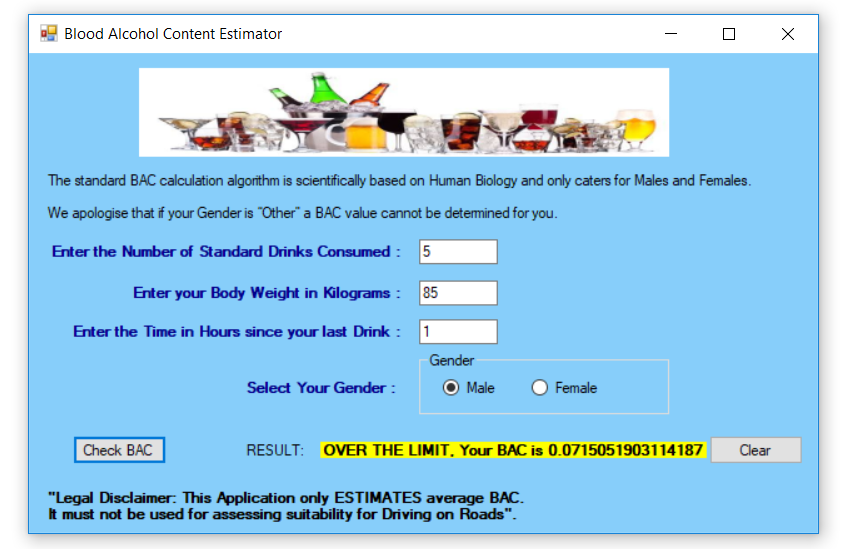
In a Blood Alcohol Calculator program we built, the final calculated result comes out with a very large number of decimal places. This is overly complicated for the user to interpret. *(Eg. Not “Effective”).*



*It would be more “User Friendly” or more “Effective” if we rounded off this result to 4 decimal places and presented the result as “0.0715” .*

We need to round off the “dblBAC” variable value in the section of program which calculates and displays the final result:

'--We now have everything we need and we Calculate the BAC

dblBAC = ((dblAlcoholContent / (dblGramsWeight \* dblResistance) \* 100) - (dblHours \* 0.015))

'--Display the results. In Victoria > 0.05 is over the Legal Limit.

If dblBAC > 0.05 Then

lblBACResult.Text = "OVER THE LIMIT, Your BAC is " & dblBAC

Else

lblBACResult.Text = "Under the Limit, Your BAC is " & dblBAC

End If

We can do this by adding a line of code to do the rounding off for the “dblBAC” immediately after it has been calculated. The line of code we have added in is highlighted below:

'--We now have everything we need and we Calculate the BAC

dblBAC = ((dblAlcoholContent / (dblGramsWeight \* dblResistance) \* 100) - (dblHours \* 0.015))

'-- Round off the calculated dblBAC value to 4 decimal places:

dblBAC = Math.Round(dblBAC, 4, MidpointRounding.AwayFromZero)

'--Display the results. In Victoria > 0.05 is over the Legal Limit.

If dblBAC > 0.05 Then

lblBACResult.Text = "OVER THE LIMIT, Your BAC is " & dblBAC

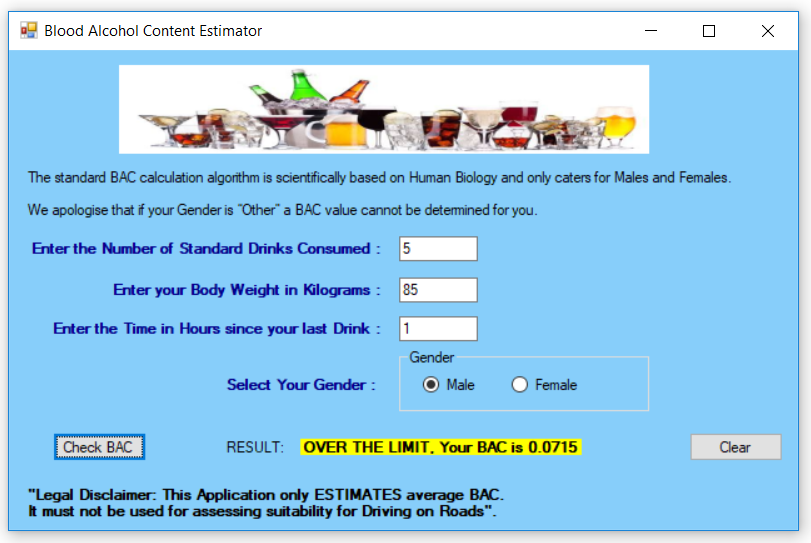
Else

lblBACResult.Text = "Under the Limit, Your BAC is " & dblBAC

End If

We use the inbuilt Visual Basic “Round” function, and set a “4” in the middle of it to round off to 4 decimal places. If we only wanted 2 decimal places we would set a “2”, for 1 decimal place rounding we would set a “1” and so on.

If we rerun our program with this modified code, we see that we now get our final result rounded off correctly to 4 decimal places:



*Read on for a full discussion of Rounding Off and the related but different process of “Truncating”.*

**Rounding Off to 2 Decimal Places**

Rounding Off to 2 Decimal Places means that VB will look at the 3rd Decimal place, and if it is 5 or greater it will round it up, (add 1), but if the 3rd decimal place is 0 to 4 then it will just discard everything after the second decimal place.   
Eg. It works exactly the same way that it does in mathematics.

**2.55**7 will round up to **2.56**

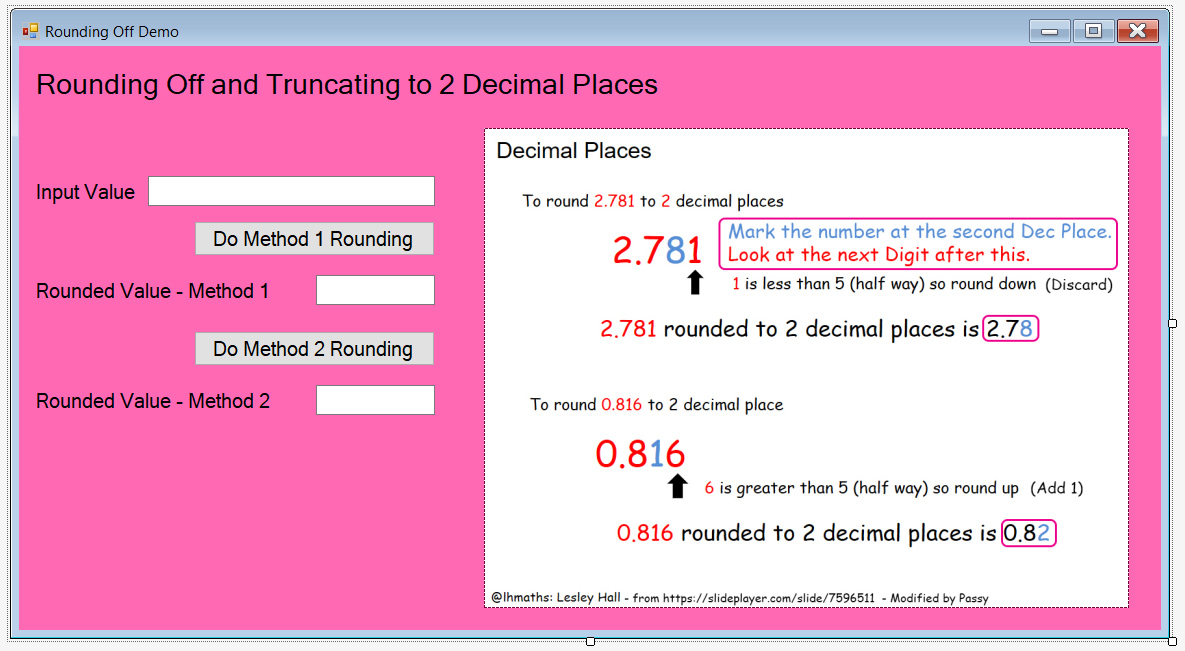
**2.55**74321 will round up to **2.56**

**2.55**3 will round off to **2.55**

**2.55**378912 will round off to **2.55**

**Here is a VB Program that shows two methods of Rounding Off.**

The first method is the easiest and quickest one to use.



The Text Boxes have been named as: **txtInputValue**, **txtRoundedMethod1**, **txtRoundedMethod2**

The Buttons are named as: **btnMethod1** and **btnMethod2**

*The Picture Box is not needed, and is just an Image of how to round off in maths to 2 decimal places.*

The Program Code for the two different Method Buttons is as follows:

Public Class Form1

Private Sub btnMethod1\_Click(sender As Object, e As EventArgs) Handles btnMethod1.Click

'--Simplest method is to take the Input Textbox and Round its value to 2 decimal places

'--and put the answer into the Form's Output text box which is called txtRoundedMethod1

txtRoundedMethod1.Text = Math.Round(Val(txtInputValue.Text), 2)

'--If you want to round to just 1 decimal place, change the ,2 on the end to ,1

End Sub

Private Sub btnMethod2\_Click(sender As Object, e As EventArgs) Handles btnMethod2.Click

'--Method 2 for Rounding Off is from Stack Overflow and takes more coding

'--Get Input Value (No Validation - needs to be Decimal or Whole Number)

Dim dblInputNumber As Double

dblInputNumber = CDbl(txtInputValue.Text)

Dim dblRoundedOutput As Double

'Round Off and put the rounded off result into the dblRoundedOutput variable

dblRoundedOutput = Math.Round(dblInputNumber, 2, MidpointRounding.AwayFromZero)

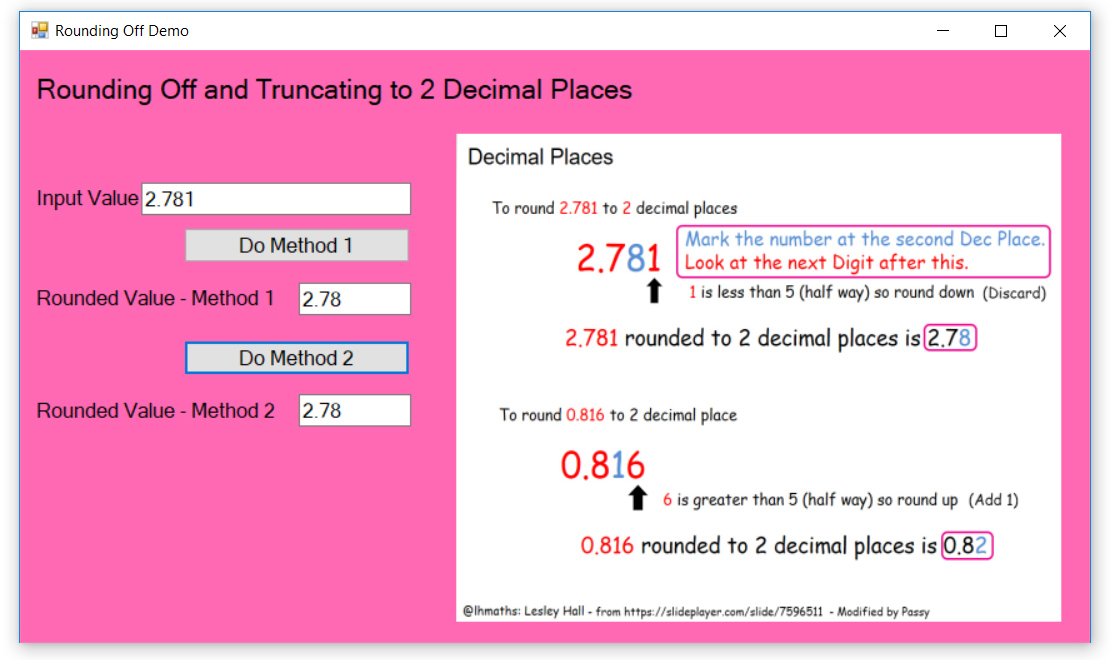
'--Move the Rounded off answer into the method 2 text box that is on the form

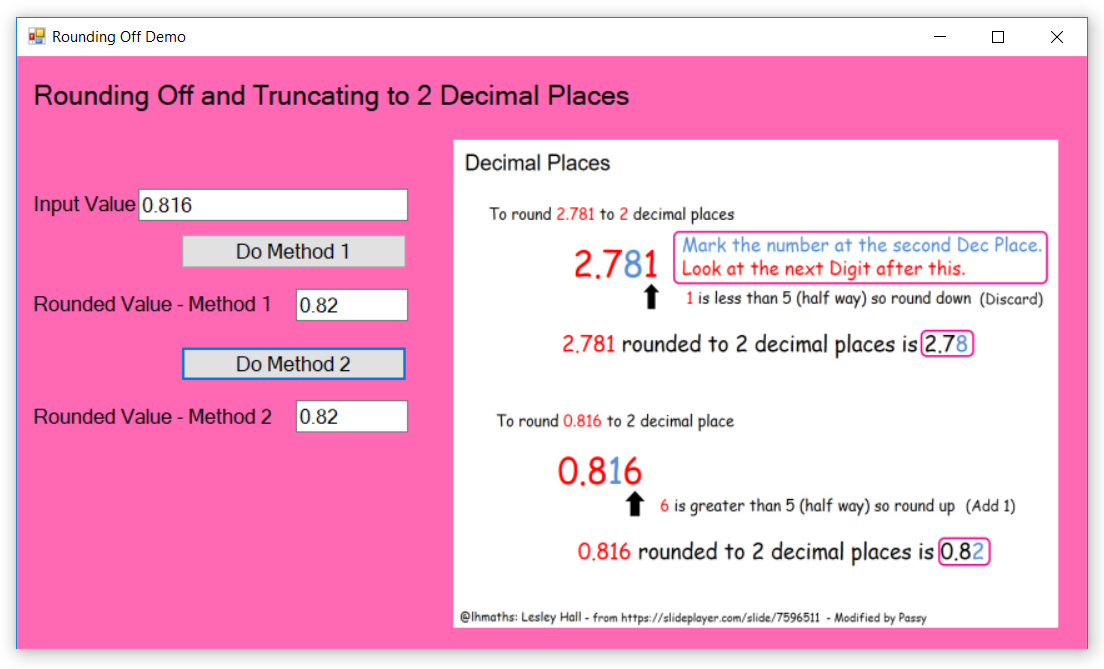
txtRoundedMethod2.Text = CStr(dblRoundedOutput)

End Sub

End Class

**Both of the Methods produce the same correct Rounding results when the Buttons are clicked:**





In the above test run, 0.816 has been correctly rounded up to 2 decimal places to become 0.82

***“Truncation” is another way to take a value down to be only two decimal places,   
but we generally use rounding off and not truncation.***

**Truncation**

*Truncating just removes decimal digits, it does not do any 5 or above Rounding Up.*

**2.54**7 TRUNCATED to two decimal places is **2.54** - the 7 is simply thrown away.

**2.54**74321 TRUNCATED to two decimal places is **2.54** - the 74321 is simply thrown away.

We can upgrade our Program to also do truncation by adding a “Truncate to 2 dp” Button, and a third output text Box called “txtTruncatedValue”.

When we click the “Truncate 2dp” Button on the Form the following code is executed:

Private Sub btnTruncate\_Click(sender As Object, e As EventArgs) Handles btnTruncate.Click

'--Truncate to 2 decimal places which just discards decimal numbers and does not Round

'--values up or down. Truncate is not usually used in programs.

Dim dblInputNbrForTruncation As Double

dblInputNbrForTruncation = CDbl(txtInputValue.Text)

Dim dblTruncatedOutput As Double

'--Truncate the Input Number And Put the Result into the Output Nbr variable

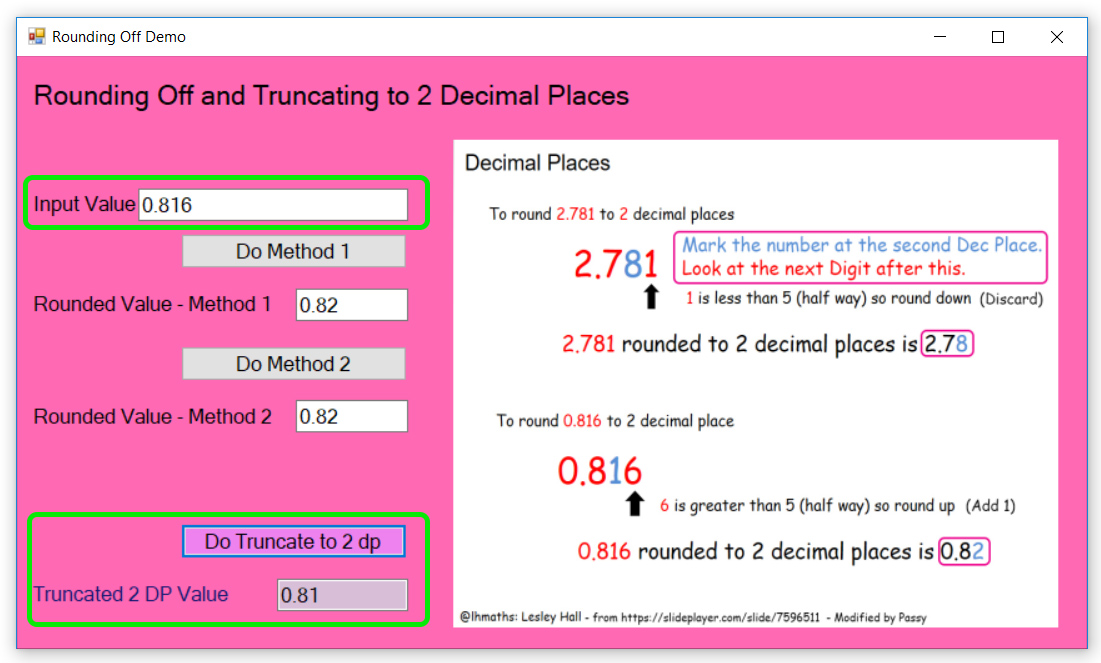
dblTruncatedOutput = dblInputNbrForTruncation - (dblInputNbrForTruncation Mod 0.01)

'--Move the Truncated answer into the method 2 text box that is on the form

txtTruncatedOutput.Text = CStr(dblTruncatedOutput)

End Sub

**Note in the Output run of the Program that when we Truncate, rounding up never occurs:**



**Truncating is generally not used, we do normal Mathematical rounding off instead.**

**Appendix A - How to find out How To do things like this yourself :**

*You MUST learn how to Google, and the Stack Overflow forums always have the best answers:*

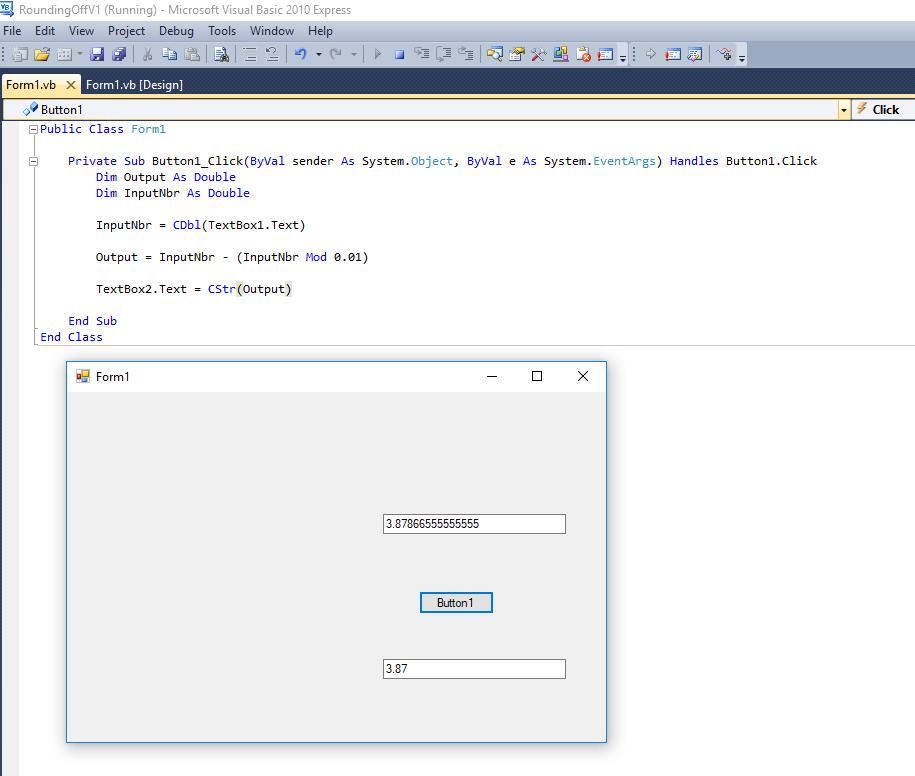
We found out how to round off in Visual Basic by Googling for :

“round off result in visual basic”

And went to the Stack Overflow Answers:

<https://stackoverflow.com/questions/1228469/rounding-a-number-down-in-visual-basic>

*Once you have the Stack Overflow Answer - Write yourself a small VB program to try out the code:*

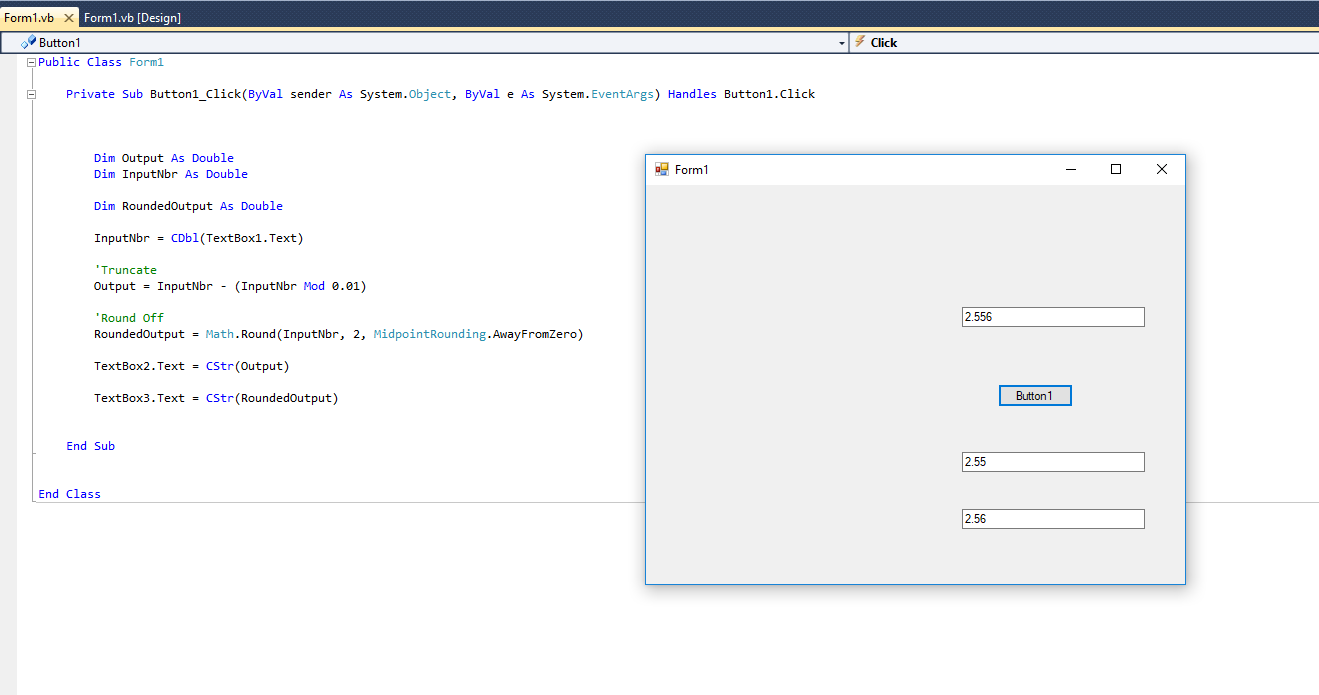


Note that this method Truncates and removes everything after 2 decimal places but does not look at the 3rd decimal place and see if it should round up or down. This was not the answer we were looking for.

**Proper Rounding Off is done via a Math function:**

Stack Overflow

<https://stackoverflow.com/questions/12113096/need-help-rounding-to-2-decimal-places>



Public Class Form1

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim Output As Double

Dim InputNbr As Double

Dim RoundedOutput As Double

InputNbr = CDbl(TextBox1.Text)

'Truncate

Output = InputNbr - (InputNbr Mod 0.01)

'Round Off

RoundedOutput = Math.Round(InputNbr, 2, MidpointRounding.AwayFromZero)

TextBox2.Text = CStr(Output)

TextBox3.Text = CStr(RoundedOutput)

End Sub

End Class

**The easiest way we found to round off to Two decimal Places using similar coding is :**

TextBox2.Text = Math.Round(Val(TextBox2.Text),2)

We do this after our final answer for a calculated formula result like “Blood Alcohol Level”, or “Body Mass Index”, etc has first been moved into Textbox2 complete with its many decimal places.

The VB code shown above will round off a long decimal places answer to only be 2 decimal places long.

**Both Methods of Rounding off shown in this lesson give the exact same answers, as proven by the program we wrote and tested. It is up to you which method you choose to use.**

**Why do we most often choose rounding off to 2 decimal places ?**

Because Money or Currency is always to 2 decimal places, and so it is a format that people are very familiar with and easily understand.

End of Document