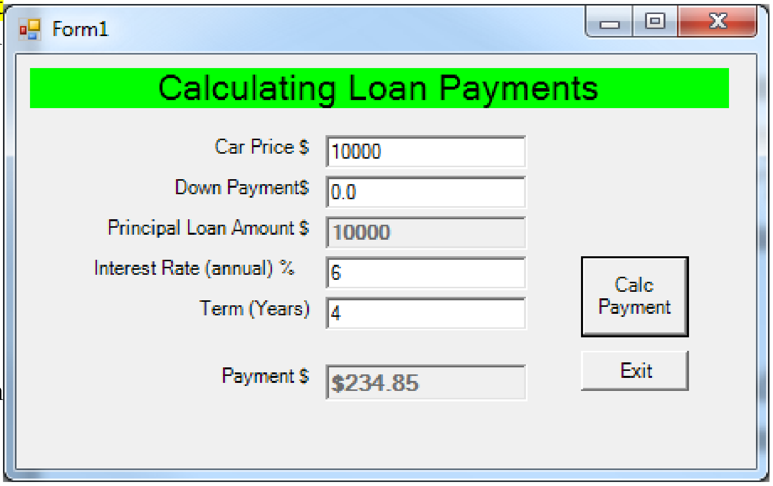
A201/A505 Laboratory Session #9

LAB GOALS

> To learn how to use Built in functions such as Pmt().
>
> In addition, we will review input validation using IsNumeric() function, the IF statement (using AND to create a complex condition), type conversion, handling multiple events using the same event handler, and formatting statement.

Step 1: Create a new project named "Built_in_Functions". Once the main form is displayed, create the following interface. Make sure to set the ReadOnly property of PrincipalLoanAmount and Payment textboxes to "TRUE". This property will ensure that the user cannot change the values in these two textboxes.



Calculating Loan Payments

Car Price $ 10000

Down Payment$ 0.0

Principal Loan Amount $ 10000

Interest Rate (annual) % 6

Term (Years) 4

Payment $ $234.85

Also don't forget to add the following two lines at the beginning of the program:

```
Option Explicit On
Option Strict On
```

Step 2: Learning about Built-in Functions:

To calculate the payment amount for a given loan, we need to learn about the Pmt() function. Pmt is a built-in VB function. This function requires (a minimum of) three parameters: these are the loan amount, the interest rate, and the term. The function will then calculate and return the monthly payment for the loan.

Note: The payment function requires that both the interest rate and the term match. In other word, if we wish to calculate the monthly payment for a $10,000 loan, then the interest rate, and the term, must be expressed in terms of months. See below:

```
Pmt(MonthlyIntRate, NumPeriods, Principal)
Pmt(6%/12,    5 * 12,    1000)
```

To learn more about the Pmt() function, look up the VB help facilities. While looking at help facilities, look up other built-in functions such as FV(), PV(), Abs(), and Rnd().

Step 3: Calculating Payments:

In this step, we would like to write the event handler for the CalcPayment button. Double click the CalcPayment button and enter the code below in the event handler.

```
Dim MonthlyIntRate, NumPeriods, Principal, Payment As Double

MonthlyIntRate = CDbl(txbAnnualIntRate.Text) / 1200           'Take the annual interest rate, convert
                                                              it to percentage (divide by 100) and
                                                              then convert it to monthly rate (divide
                                                              by 12)
NumPeriods = CDbl(txbTerm.Text) * 12                  'Convert the term from annual to monthly
                                                      (multiply by 12)

txbPrincipalLoanAmount.Text = CStr(CDbl(txbCarPrice.Text) - CDbl(txbDownPayment.Text))
Principal = CDbl(txbPrincipalLoanAmount.Text)

Payment = Pmt(MonthlyIntRate, NumPeriods, Principal) * -1     'Call the Pmt() function
txbPayment.Text = Format(Payment, "C")                    ' Convert to Currency
```

Run: Compile and Run your program. If your program does not compile, fix the syntax errors and compile the program again. Once you are able to successfully run the program, check to see if the monthly payment appears to be correct. For example, if you borrow $10,000 to buy a car, with a 6% annual interest rate and a 4 year term your payment should be $234.85

** If you have a house loan or a car loan, try those numbers to see if your monthly payment is correct.

Step 4: Input Validation:

In this step, we would like to introduce some input validation and error checking into our code. Note, that in the previous step, if you forget to provide all the needed parameters in the corresponding text boxes (for example leaving the down payment textbox empty), your program will simply crash. In order to prevent the program from crashing we must make sure all the pertinent data items are present before we begin to perform any calculations. We will use the built-in Boolean function IsNumeric(parameter) which will return TRUE if the parameter can be converted to a legitimate number and FALSE if the parameter cannot be converted to a number.   See the use of IsNumeric() function below:

```
Dim MonthlyIntRate, NumPeriods, Principal, Payment As Double

If IsNumeric(txbPrincipalLoanAmount.Text) _
       And IsNumeric(txbAnnualIntRate.Text) _
       And IsNumeric(txbTerm.Text) Then

    MonthlyIntRate = CDbl(txbAnnualIntRate.Text) / 1200          'Take the annual interest rate, convert
                                                                 it to percentage (divide by 100) and
                                                                 then convert it to monthly rate (divide
                                                                 by 12)
    NumPeriods = CDbl(txbTerm.Text) * 12                         'Convert the term from annual to monthly
                                                                 (multiply by 12)

    txbPrincipalLoanAmount.Text = CStr(CDbl(txbCarPrice.Text) - CDbl(txbDownPayment.Text))
    Principal = CDbl(txbPrincipalLoanAmount.Text)

    Payment = Pmt(MonthlyIntRate, NumPeriods, Principal) * -1     'Call the Pmt() function
    txbPayment.Text = Format(Payment, "C")  ' Convert to Currency

Else

    MessageBox.Show("Make sure the PRICE, INTEREST RATE, TERM, etc. are properly specified", _
                      "Calculation Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End If
```

Step 5: Handling Events:

In this step, we would like to develop an event handler for both the Car Price as well as the Down Payment textboxes. Our goal is to design the interface in such a way that when the user enters a number in the Car Price textbox, that number also appears in the PrincipalLoanAmount textbox. Then, when the user begins to enter a number for the DownPayment textbox, that value is automatically subtracted from the CarPrice and the result is shown in the PrincipalLoanAmount textbox.

In order to do this, we need to develop an event handler for the CarPrice and the DownPayment textboxes. Double click each textbox and enter the following highlighted code in the event handler.

```
Dim Principal, Downpayment As Double

If (IsNumeric(txbCarPrice.Text)) Then      'If the CarPrice textbox holds a legitimate value
    Principal = CDbl(txbCarPrice.Text)
Else
    txbCarPrice.Text = "0.0"
    Principal = 0.0
End If

If (IsNumeric(txbDownPayment.Text)) Then     'If the DownPayment textbox holds a legitimate value
    Downpayment = CDbl(txbDownPayment.Text)
Else
    txbDownPayment.Text = "0.0"
    Downpayment = 0.0
End If

txbPrincipalLoanAmount.Text = CStr(Principal - Downpayment)
```

Discussion: (Improving our code)

Handling Multiple Events:

Note that the above code (step 5) would have to be repeated for both CarPrice and DownPayment textbox event handlers. This is not desirable!

Fortunately we have two options available to us. In the first option, we may take the above code and put it in a user defined sub procedure. For example we can create a procedure called CalculatePrincipalLoanAmount(), then call that procedure from each of the event handlers.

The second option is to handle both events using the same event handler. In this case, we simply develop the first event handler, then manually tell the event handler to watch for the other event as well. For example the event handler below was developed to handle the DownPayment text box. Once the code was created and tested, we simply added the highlighted line to the “Handles” area of this handler. The additional information tells the event handler to look out for events generated by the CarPrice textbox as well.

```
Private Sub txbDownPayment_TextChanged(ByVal sender As System.Object, _
                                  ByVal e As System.EventArgs) _
                                     Handles txbDownPayment.TextChanged, txbCarPrice.TextChanged

    Dim Principal, Downpayment As Double

    If (IsNumeric(txbCarPrice.Text)) Then      'If the CarPrice textbox holds a legitimate value
        Principal = CDbl(txbCarPrice.Text)
    Else
        txbCarPrice.Text = "0.0"
        Principal = 0.0
    End If

    If (IsNumeric(txbDownPayment.Text)) Then    'If the DownPayment textbox holds a legitimate value
        Downpayment = CDbl(txbDownPayment.Text)
    Else
        txbDownPayment.Text = "0.0"
        Downpayment = 0.0
    End If

    txbPrincipalLoanAmount.Text = CStr(Principal - Downpayment)

End Sub
```