Put up a form to take in the following information - note that the year should be done with a radio button since there can only be one year. There should be check boxes where you can check off good driving points, accident points, moving violation ticket points and non-moving violation ticket points. A driver can have multiple of these checked so you should be able to handle one checked or all checked. There should also be a place to enter the number of points that should become visible when the boxes are checked. The data on the form should include:

- Vehicle identification number
- Vehicle name
- Year (just do it for the last four years and an other category)
- Value of the car
- Check boxes for good driving points, accident points, moving violation ticket points and non-moving violation ticket points
- Boxes to enter the number of points for each of the categories above
- List box where information is displayed (see more on this below)
- Insurance premium or amount owed (result of your calculation)

The user will key in all information and you need to calculate and display information in the list box and information in the insurance premium/amount owed box. The insurance premium/amount owed is based on the year, the value of the car, and the points. The answer should be formatted.

- For the year 2011, the premium before considering the points is 6% of the value
- For the year 2010, the premium before considering the points is 5% of the value
- For the year 2009, the premium before considering the points is 4.5% of the value
- For the year 2008, the premium before considering the points is 4% of the value
- For the years before that, the premium is 3.5% of the value

The points should be handled this way:

- For good driving points, subtract $35 per point from the insurance premium/amount owed
- For accident points, add $75 per point to the insurance premium/amount owed
- For moving violation ticket points, add $50 per point to the insurance premium/amount owed
- For non-moving violation ticket points, add $25 per point to the insurance premium/amount owed
These programs are under loops under .net examples. This one is under the break and validate group.

Notice that this is txtName_Validating.
Private Sub frmCheck_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    grpSalary.Enabled = False
    grpHourly.Enabled = False
End Sub

Private Sub txtSalary_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs) Handles txtSalary.Validating
    If txtSalary.Text < 25000 Or txtSalary.Text > 200000 Then
        MessageBox.Show("Salary between 25000 and 200000")
        txtSalary.Text ="
        txtSalary.Focus()
    End If
End Sub

Private Sub btrClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btrClear.Click
    txtName.Clear()
    txtDept.Clear()
    txtSalHR.Clear()
    txtSalary.Clear()
    txtHrs.Clear()
    txtPayHR.Clear()
    txtSalPay.Clear()
End Sub

Private Sub txtHrs_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs) Handles txtHrs.Validating
    If txtHrs.Text < 0 Then
        MessageBox.Show("Negative hours")
        txtHrs.Text ="
        txtHrs.Focus()
    End If
End Sub
Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnProcess.Click
    If grpSalary.Enabled Then
        grpHourly.Enabled = False
    Else
        grpHourly.Enabled = True
    End If
End Sub

Private Sub frmCheck_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    grpSalary.Enabled = False
    grpHourly.Enabled = False
End Sub

Private Sub txtSalary_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs) Handles txtSalary.Validating
    If txtSalary.Text < 25000 Or txtSalary.Text > 200000 Then
        MessageBox.Show("Salary between 25000 and 200000", "Error Salary")
    End If
End Sub
I put the group box on the form and then added the contents. When I enable the group box, I enable the content.
Relate to previous two slides.
Private Sub txtPayhr_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs)
    If txtPayhr.Text < 10 Or txtPayhr.Text > 50 Then
        MessageBox.Show("Pay/hour between 10 and 50", "Error Pay/Hr")
        With txtPayhr
            .SelectionStart = 0
            .SelectionLength = txtPayhr.Text.Length
            .Focus()
        End With
    End If
End Sub

Private Sub btnSalCalc_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    txtsalpay.Text = FormatCurrency(CDec(txtsalary.Text) / 52)
End Sub

Private Sub btnHrCalc_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Dim wkPay As Decimal
    If txtHrs.Text > 40 Then
        wkPay = CDec(txtPayhr.Text) * 40 + CDec(txtPayhr.Text) * 1.5 * (CInt(txtHrs.Text) - 40)
    Else
        wkPay = CDec(txtPayhr.Text) * CInt(txtHrs.Text)
    End If
    txtHrPay.Text = FormatCurrency(wkPay)
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
End Sub
If txtSalHr.Text <> "S" Then
    If txtSalHr.Text <> "H" Then
        MessageBox.Show("Enter S or H", "Employee Type Error")
        txtSalHr.Focus()
    End If
End If
End Sub

Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    If txtSalHr.Text = "S" Then
        grpSalary.Enabled = True
        grpHourly.Enabled = False
    Else
        grpSalary.Enabled = False
        grpHourly.Enabled = True
    End If
End Sub

Private Sub frmCheck_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    grpSalary.Enabled = False
    grpHourly.Enabled = False
End Sub

Private Sub txtSalary_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs)
    If txtSalary.Text < 25000 Or txtSalary.Text > 20000 Then
        MessageBox.Show("Salary between 25000 and 20000", "Error Salary")
        txtSalary exercitation = 0
        txtSalary.SelectionStart = 0
        txtSalary.SelectionLength = txtSalary.Text.Length
        txtSalary.Focus()
    End If
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Minor, Intermediate and Major Breaks

Please use speaker notes for additional information!

These slides will go over the logic for minor, intermediate and major breaks in a program.
Report produced by major.cbl

This is the output report from the program major.cbl.
This is the output report from the program major.cbl.
The first seven input records:

<table>
<thead>
<tr>
<th>DEPT</th>
<th>ITEM NUMBER</th>
<th>ITEM NAME</th>
<th>ON HAND</th>
<th>PRICE</th>
<th>INVENTORY VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>101021234</td>
<td>HOE</td>
<td>12</td>
<td>$18.00</td>
<td>$216.00</td>
</tr>
<tr>
<td>01</td>
<td>101021235</td>
<td>SKEWER</td>
<td>20</td>
<td>$2.00</td>
<td>$40.00</td>
</tr>
<tr>
<td>01</td>
<td>101015111</td>
<td>SPOON</td>
<td>30</td>
<td>$3.00</td>
<td>$90.00</td>
</tr>
<tr>
<td>01</td>
<td>101015112</td>
<td>SPATULA</td>
<td>15</td>
<td>$1.00</td>
<td>$15.00</td>
</tr>
<tr>
<td>01</td>
<td>102017100</td>
<td>LILY</td>
<td>10</td>
<td>$2.00</td>
<td>$20.00</td>
</tr>
<tr>
<td>01</td>
<td>121212223</td>
<td>DANSY</td>
<td>5</td>
<td>$3.00</td>
<td>$15.00</td>
</tr>
<tr>
<td>02</td>
<td>131313131</td>
<td>BRANCH</td>
<td>125</td>
<td>$3.00</td>
<td>$375.00</td>
</tr>
<tr>
<td>02</td>
<td>141414141</td>
<td>BRANCH</td>
<td>125</td>
<td>$3.00</td>
<td>$375.00</td>
</tr>
</tbody>
</table>

This shows the first seven records and the resulting report.

Note when the branch changes from 10 to 20, I need to write a department total for the last department and then a branch total for the branch before I process the first record in branch 20.

Note that when the department changes from 12 to 15, a total is written for department 12 before department 15 is processed.
This is the logic flowchart - page 3.
This is the logic flowchart - page 4.
This is the logic flowchart - page 5. It simply handles writing headers and the wrapup which closes the files.
<table>
<thead>
<tr>
<th>Logic including Pseudocode and flowcharts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Separate speaker notes to accompany introduction to arrays - logic</td>
</tr>
<tr>
<td>Using arrays to accumulate totals based on input - logic</td>
</tr>
<tr>
<td>Separate speaker notes to accompany using arrays to accumulate totals based on input - logic</td>
</tr>
<tr>
<td>Two dimensional array - logic</td>
</tr>
<tr>
<td>Separate speaker notes to accompany two dimensional array - logic</td>
</tr>
<tr>
<td>Searching an array using an indirect subscript - logic</td>
</tr>
<tr>
<td>Separate speaker notes to accompany searching an array using an indirect subscript - logic</td>
</tr>
<tr>
<td>Validating/editing data - logic</td>
</tr>
<tr>
<td>Separate speaker notes to accompany validating/editing data - logic</td>
</tr>
<tr>
<td>Presentation on sequential update processing</td>
</tr>
<tr>
<td>Separate speaker notes to accompany sequential update processing</td>
</tr>
<tr>
<td>Presentation on random update processing</td>
</tr>
<tr>
<td>Separate speaker notes to accompany random update processing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Programming - examples mainly with COBOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation on programming structures</td>
</tr>
<tr>
<td>Separate speaker notes to accompany programming structures</td>
</tr>
<tr>
<td>Presentation on looping - flowcharts and COBOL code</td>
</tr>
<tr>
<td>Separate speaker notes to accompany presentation on looping</td>
</tr>
<tr>
<td>Presentation on looping - flowcharts and COBOL code (includes pseudocode)</td>
</tr>
<tr>
<td>Separate speaker notes to accompany presentation on looping (includes pseudocode)</td>
</tr>
<tr>
<td>Minor break logic and processing</td>
</tr>
<tr>
<td>Separate speaker notes to accompany Minor break logic and processing</td>
</tr>
<tr>
<td>Minor, Intermediate and Major break logic and processing</td>
</tr>
<tr>
<td>Separate speaker notes to accompany Minor, Intermediate, and Major break logic and processing</td>
</tr>
<tr>
<td>Introduction to arrays or tables in COBOL</td>
</tr>
<tr>
<td>Separate speaker notes to accompany introduction to arrays or tables</td>
</tr>
<tr>
<td>Presentation on table search</td>
</tr>
<tr>
<td>Speaker notes to accompany Presentation on table search</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logo (StarLogo)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation on using StarLogo</td>
</tr>
<tr>
<td>Separate speaker notes to accompany using StarLogo</td>
</tr>
</tbody>
</table>
The data file MIM.txt is in the bin of my project.
Public Class frmMIM
Inherits System.Windows.Forms.Form

    Dim wkMinTot As Integer
    Dim wkInterTot As Integer
    Dim wkMajorTot As Integer
    Dim wkFinTot As Integer
    Dim wkHoldDept As String
    Dim wkHoldBr As String
    Dim wkHoldDiv As String

    Windows Form Designer generated code

    Private Sub frmMIM_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        FileOpen(1, "MIM.txt", OpenMode.Input)
        wkHoldDept = ""
        wkHoldBr = ""
        wkHoldDiv = ""
    End Sub

    Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnProcess.Click
        Dim wkDivNo As String, wkBrNo As String
        If Not EOF(1) Then
            Input(1, wkDivNo)
            Input(1, wkBrNo)
            Input(1, wkDeptNo)
            Input(1, wkAnt)
            If wkHoldDiv = "" Then
                wkHoldDiv = wkDivNo
                wkHoldBr = wkBrNo
                wkHoldDept = wkDeptNo
            End If
            If wkHoldDiv <> wkDivNo Then
                lstMIM.Items.Add("Dept Total:"
                lstMIM.Items.Add(wkHoldDept)
                lstMIM.Items.Add(wkHoldBr)
                lstMIM.Items.Add(wkHoldDiv)
                lstMIM.Items.Add(wkAnt)
                lstMIM.Items.Add(wkDivNo)
                lstMIM.Items.Add(wkBrNo)
                lstMIM.Items.Add(wkDeptNo)
                lstMIM.Items.Add(wkHoldDept)
                lstMIM.Items.Add(wkHoldBr)
                lstMIM.Items.Add(wkHoldDiv)
            Else
                lstMIM.Items.Add("")
            End If
        End If
    End Sub
Public Class frmMIM
    Inherits System.Windows.Forms.Form
    Dim wkMinTot As Integer
    Dim wkInTot As Integer
    Dim wkIntTot As Integer
    Dim wkMajorTot As Integer
    Dim wkFinTot As Integer
    Dim wkHldDept As String
    Dim wkHolder As String
    Dim wkHoldDiv As String
End Class

Private Sub frmMIM_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    FileOpen(1, "MIM.txt", OpenMode.Input)
    wkHldDept = ""
    wkHolder = ""
    wkHoldDiv = ""
End Sub

Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnProcess.Click
    Dim wkDivNo As String, wkBrNo As String, wkDeptNo As String, wkAmt As String
    If Not EOF(1) Then
        Input(1, wkDivNo)
        Input(1, wkBrNo)
        Input(1, wkDeptNo)
        Input(1, wkAmt)
        If wkHoldDiv = "" Then
            wkHoldDiv = wkDivNo
            wkHldDept = wkBrNo
            wkHolder = wkDeptNo
        End If
        If wkHoldDiv <> wkDivNo Then
            lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
        End If
    End If
    FileClose(1)
End Sub
Input(1, wkDvlvNo)
Input(1, wkBrNo)
Input(1, wkDeptNo)
Input(1, wkInd)
If wkHoldDiv = "" Then
    wkHoldDiv = wkDvlvNo
    wkHoldBr = wkBrNo
    wkHoldDept = wkDeptNo
End If
If wkHoldDiv <> wkDvlvNo Then
    lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
    lstMIM.Items.Add("Branch Total: " & FormatCurrency(wkInterTot))
    lstMIM.Items.Add("Division Total: " & FormatCurrency(wkMajorTot))
    wkHoldDept = wkDeptNo
    wkHoldBr = wkBrNo
    wkHoldDiv = wkDvlvNo
    wkMinTot = 0
    wkInterTot = 0
Else
    If wkHoldBr <> wkBrNo Then
        lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
        lstMIM.Items.Add("Branch Total: " & FormatCurrency(wkInterTot))
        wkHoldDept = wkDeptNo
        wkHoldBr = wkBrNo
        wkMinTot = 0
        wkInterTot = 0
    Else
        If wkHoldDept <> wkDeptNo Then
            lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
            wkHoldDept = wkDeptNo
            wkMinTot = 0
        End If
Else If wkHoldBr <> wkBrNo Then
    lstNIM.Items.Add(Dept Total: " & FormatCurrency(wkMinTot))
    lstNIM.Items.Add(Branch Total: " & FormatCurrency(wkIntTot))
    wkHoldDept = wkDeptNo
    wkHoldBr = wkBrNo
    wkMinTot = 0
    wkInterTot = 0
Else
    If wkHoldDept <> wkDeptNo Then
        lstNIM.Items.Add(Dept Total: " & FormatCurrency(wkMinTot))
        wkHoldDept = wkDeptNo
        wkMinTot = 0
    End If
End If
Else
    lstNIM.Items.Add(wkDivNo & " " & wkBrNo & " " & wkDeptNo & " " & wkFinTot & " " & FormatCurrency(wkMinTot) + wkAmt
    wkInterTot = wkInterTot + wkAmt
    wkMajorTot = wkMajorTot + wkAmt
    wkFinTot = wkFinTot + wkAmt
Else
    MsgBox("End of File", vbOKOnly, "EOF")
End If
End Sub
Sub btnProcess_Click

    wkInterTot = 0
    wkMajorTot = 0
    Else
        If wkHoldBr <> wkBrNo Then
            lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
            lstMIM.Items.Add("Branch Total: " & FormatCurrency(wkInterTot))
            wkHoldDept = wkDeptNo
            wkHoldBr = wkBrNo
            wkMinTot = 0
            wkInterTot = 0
        Else
            If wkHoldDept <> wkDeptNo Then
                lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
                wkHoldDept = wkDeptNo
                wkMinTot = 0
            End If
        End If
        End If
    End If
End Sub
Record that caused the break. It caused it because the dept changed.
wkInterTot = 0
wkMajorTot = 0
Else
  If wkHoldBr <> wkBrNo Then
    lstNM.Item.Add("Dept Total: " & FormatCurrency(wkMinTot))
    lstNM.Item.Add("Branch Total: " & FormatCurrency(wkInterTot))
    wkHoldDept = wkDeptNo
    wkHoldBr = wkBrNo
    wkInTot = 0
    wkInterTot = 0
  Else
    If wkHoldDept <> wkDeptNo Then
      lstNM.Item.Add("Dept Total: " & FormatCurrency(wkMinTot))
      wkHoldDept = wkDeptNo
      wkInTot = 0
    End If
  End If
End If
Else
  lstNM.Item.Add(wkDivNo & " & wkBrNo & " & wkDeptNo & " & FormatCurrency(wkInTot))
  wkInTot = wkInterTot + wkAmt
  wkMajorTot = wkMajorTot + wkAmt
  wkFinTot = wkFinTot + wkAmt
Else
  MsgBox("End of File", vbOKOnly, "EOF")
End If
End Sub
if wkHoldDept <> wkDeptNo Then
    lstIMM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
    wkHoldDept = wkDeptNo
    wkMinTot = 0
End If
End If
End If
End Sub

else
    MsgBox("End of File", vbOKOnly, "EOF")
    lstIMM.Items.Add("Dept Total: " & FormatCurrency(wkMinTot))
    lstIMM.Items.Add("Branch Total: " & FormatCurrency(wkInterTot))
    lstIMM.Items.Add("Division Total: " & FormatCurrency(wkMajorTot))
    lstIMM.Items.Add("Final Total: " & FormatCurrency(wkFinTot))
End If
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExit.Click
    End
End Sub

End Class
Private Sub frmMIM_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    FileOpen(1, "MIM.txt", OpenMode.Input)
    wkHoldDept = 
    wkHoldBr = 
    wkHoldDiv = 
End Sub

Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnProcess.Click
    Dim wkDivNo As String, wkBrNo As String, wkDeptNo As String, wkAmt As String
    If Not EOF(1) Then
        Input(1, wkDivNo)
        Input(1, wkBrNo)
        Input(1, wkDeptNo)
        Input(1, wkAmt)
        If wkHoldDiv = "" Then
            wkHoldDiv = wkDivNo
            wkHoldBr = wkBrNo
            wkHoldDept = wkDeptNo
        End If
        If wkHoldDiv <> wkDivNo Then
            lstMIM.Items.Add("Dept Total: " & FormatCurrency(wkHoldTot))
            lstMIM.Items.Add("Branch Total: " & FormatCurrency(wkInterTot))
            lstMIM.Items.Add("Division Total: " & FormatCurrency(wkMajorTot))
            wkHoldDept = wkDeptNo
            wkHoldBr = wkBrNo
            wkHoldDiv = wkDivNo
            wkHoldTot = 0
        End If
    End If
End Sub