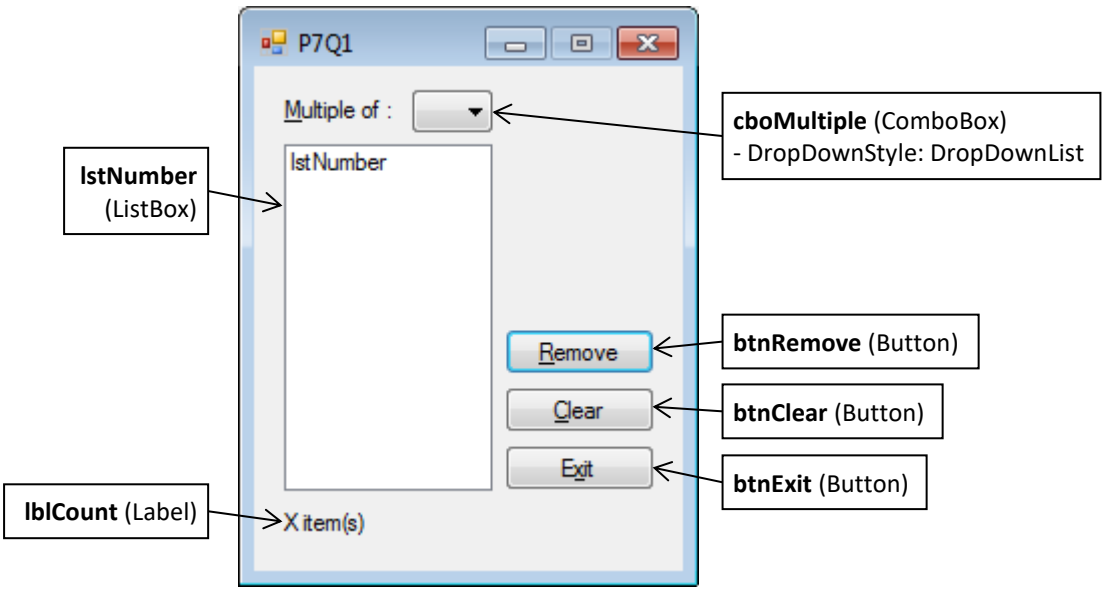
Practical 7: List Box, Combo Box and Printing

Q1.

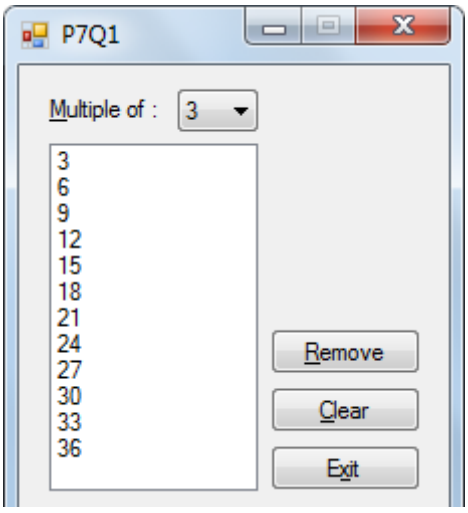
- Create a form named **FrmP7Q1** with the following layout:



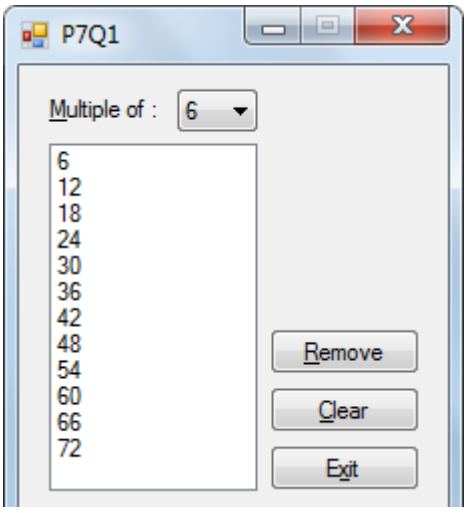
- Configure the form with appropriate usability features: access keys, tab order, accept button, cancel button, etc.

- Program the **Load** event of the form. Perform the following tasks:
  - Clear all items from **cboMultiple**.
  - Add integers 1 - 9 to **cboMultiple** by using a **For** loop.
  - Select the first item of **cboMultiple**.

- Program the **SelectedIndexChanged** event of **cboMultiple**. Perform the following tasks:
  - Read the selected integer from **cboMultiple**.
  - Clear all items from **lstNumber**.
  - Add the first 12 multiples of the selected integer to **lstNumber** by using a **For** loop.



The first 12 multiples of 3 (as selected in **cboMultiple**) are added to **lstNumber**.

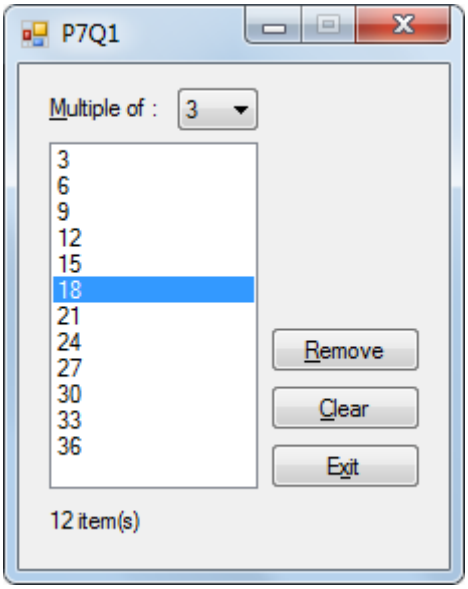


The first 12 multiple of 6 (as selected in **cboMultiple**) are added to **lstNumber**.

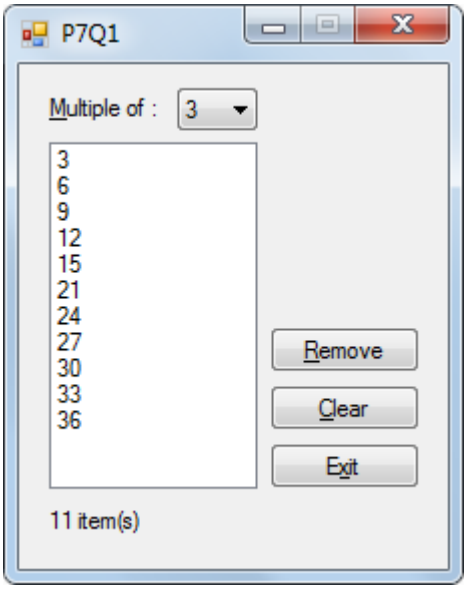
- When **btnRemove** is clicked, remove the selected item from **lstNumber**.
- When **btnClear** is clicked, clear all the items from **lstNumber**.
- When **btnExit** is clicked, close the form.
- Program the **DoubleClick** event of **lstNumber**. Remove the selected item from **lstNumber**.
- Add the following sub procedure to the form (which updates the text of **lblCount** with the item count of **lstNumber**):

```
Private Sub UpdateCount()
    lblCount.Text = lstNumber.Items.Count.ToString("0 item(s)")
End Sub
```

- Call the **UpdateCount()** sub procedure in the appropriate event handlers, so that the text of **lblCount** is updated when items are added, removed or cleared from **lstNumber**.



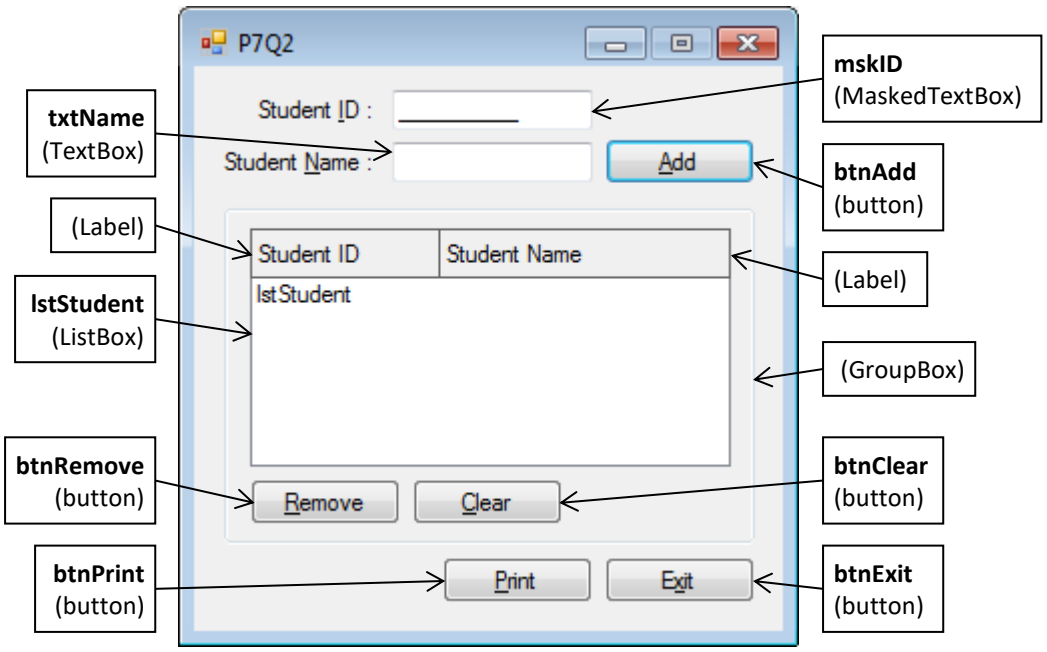
Click on **btnRemove**, or double-click on **lstNumber** to remove the selected item.



Update **lblCount** whenever items are added, removed or cleared from **lstNumber**.

Q2.(a)

- Create a form named **FrmP7Q2** with the following layout:



- Configure the form with appropriate usability features: access keys, tab order, accept button, cancel button, etc.

- Set the **Mask** property of **mskID** (MaskedTextBox) to **00>LLL00000**. This ensures the Student ID starts with 2 digits, follows by 3 uppercase alphabets, and ends with 5 digits.

  http://msdn.microsoft.com/en-us/library/system.windows.forms.maskedtextbox.mask.aspx

- Add the following function to the form (which checks if the given Student ID has been added to **lstStudent**):

```
Function IsDuplicatedID(id As String) As Boolean
    For Each item In lstStudent.Items
        If CStr(item).StartsWith(id) Then Return True
    Next
    Return False
End Function
```

Return **True** if item text starts with the Student ID

Return **False** if none of the items meet the criteria

- When **btnAdd** is clicked, validate the Student ID and Name:

Student ID	• Must follow the right format • Must not duplicate with existing Student ID
Student Name	• Must not empty

If there is input error, show the error message. Otherwise, add the Student ID and Name (joining both texts with a **TAB**) to **lstStudent**.

Complete the **btnAdd_Click** event handler with the following codes:

```
Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click
    ' (1) For validation purpose
    Dim err As New StringBuilder()
    Dim ctr As Control = Nothing

    ' (2) Read inputs
    Dim id As String = If(mskID.MaskCompleted, mskID.Text, "")
    Dim name As String = txtName.Text.Trim()

    ' (3) Validate Student ID
    If id = "" Then
        err.AppendLine("- Invalid Student ID")
        ctr = If(ctr, mskID)
    ElseIf IsDuplicatedID(id) Then
        err.AppendLine("- Student ID duplicated")
        ctr = If(ctr, mskID)
    End If

    ' (4) Validate Student Name
    If name = "" Then
        err.AppendLine("- Student Name empty")
        ctr = If(ctr, txtName)
    End If

    ' (5) Check if there is input error
    If err.Length > 0 Then
        MessageBox.Show(err.ToString(), "Input Error")
        ctr.Focus()
        Return
    End If

    ' (6) Add Student ID and Name to list box
    lstStudent.Items.Add(id & vbTab & name)
    mskID.Text = ""
    txtName.Text = ""
    mskID.Focus()
End Sub
```

Use to remember which control to focus on if there is input error

If the format is valid, take the input text, otherwise take an empty string

Remove white spaces at the front and end of text

If first argument is **Nothing**, the second argument is returned. Otherwise, first argument is returned

Use this technique to remember which control to focus on if there is an input error

The **StringBuilder** object is not empty. Hence there is input error.

Focus on the first control with input error

Return immediately if there is input error

Join Student ID and Name together with a **TAB** and add the text to **lstStudent**

NOTE: To use **StringBuilder** class, you need to import the **System.Text** namespace at the top of the class file:

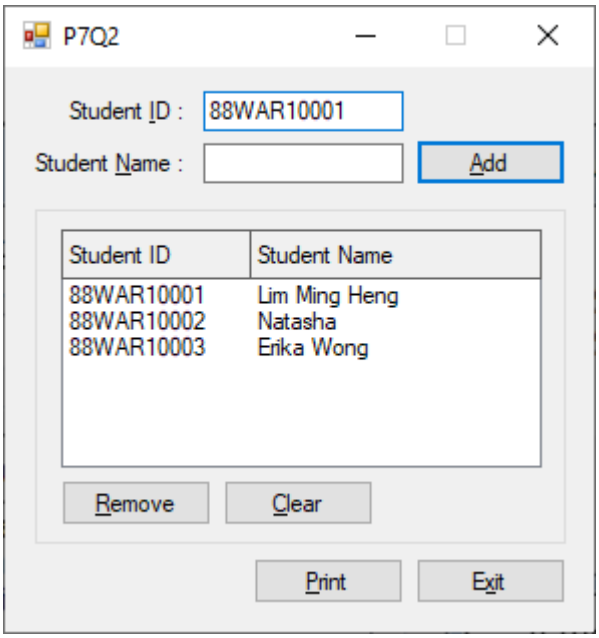
```
Imports System.Text
```

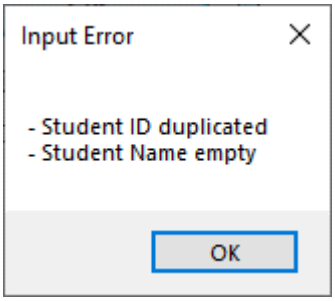
http://msdn.microsoft.com/en-us/library/bb513985.aspx

- When **btnRemove** is clicked, remove the selected item from **lstStudent**.
- When **btnClear** is clicked, clear all the items from **lstSudent**.
- When **btnExit** is clicked, close the form.

NOTE: We will program the Print button (**btnPrint**) later.

- Test your solution:

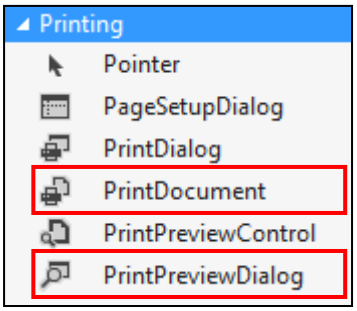


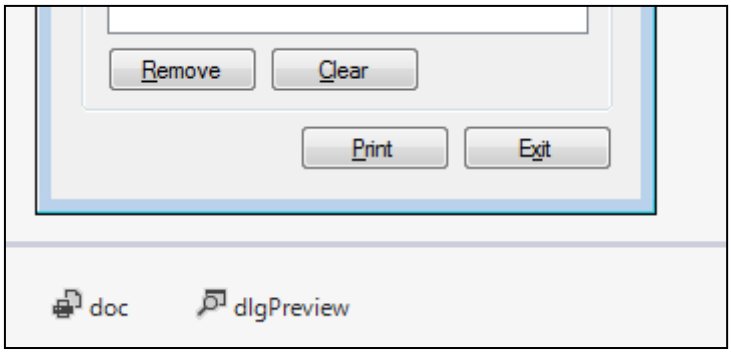


If there are input errors

Q2.(b)

- Continue to work on the form **FrmP7Q2**.
- Add the given image file **Erika.png** (or your own image file) to the project as image resource.
- From Toolbox, drag-and-drop a PrintDocument control to the form. Name the control as **doc**. Set its **OriginAtMargins** property to **True**.
- From Toolbox, drag-and-drop a PrintPreviewDialog control to the form. Name the control as **dlgPreview**. Set its **UseAntiAlias** property to **True**.





- Double-click on the PrintDocument control **doc** to generate its **PrintPage** event handler.

  Complete the **doc_PrintPage** event handler with the following codes to print the document with student records (i.e. Student ID and Name) added to **lstStudent**:

```
Private Sub doc_PrintPage(sender As Object, e As Printing.PrintPageEventArgs) _
Handles doc.PrintPage
    ' (1) Fonts
    Dim fontHeader As New Font("Calibri", 24, FontStyle.Bold)
    Dim fontSubHeader As New Font("Calibri", 12)
    Dim fontBody As New Font("Consolas", 10)

    ' (2) Prepare header and sub-header
    Dim header As String = "Student Listing"
    Dim subHeader As String = String.Format(
        "Printed on {0:dd-MMMM-yyyy hh:mm:ss tt}" & vbNewLine &
        "Prepared by SOMEBODY", DateTime.Now
    )

    ' (3) Prepare body
    Dim body As New StringBuilder()

    body.AppendLine("No   Student ID   Student Name")
    body.AppendLine("--   ----------   ----------------------------")

    Dim cnt As Integer = 0
    Dim parts() As String
    For Each item In lstStudent.Items
        cnt += 1
        parts = CStr(item).Split(CChar(vbTab))
        body.AppendFormat("{0,2}   {1,10}   {2,-30}" & vbNewLine,
                          cnt, parts(0), parts(1))
    Next

    body.AppendLine()
    body.AppendFormat("{0,2} record(s)", cnt)

    ' (4) Print
    With e.Graphics
        .DrawImage(My.Resources.Erika, 0, 0, 80, 100)
        .DrawString(header, fontHeader, Brushes.Purple, 100, 0)
        .DrawString(subHeader, fontSubHeader, Brushes.Black, 100, 40)
        .DrawString(body.ToString(), fontBody, Brushes.Black, 0, 120)
    End With
End Sub
```

Format the current date and time

Line count

Student ID and Name in **lstStudent** is separated by a **TAB**, thus it is necessary to split the item text. The **Split()** method returns the split parts as a string array

Print the image

Print the texts

http://msdn.microsoft.com/en-us/library/microsoft.visualbasic.strings.format.aspx
http://msdn.microsoft.com/en-us/library/System.String.Split.aspx

- When **btnPrint** is clicked, show the print preview dialog **dlgPreview**, with a preview of the print document **doc**.

  Complete the **btnPrint_Click** event handler with the following codes:

```
Private Sub btnPrint_Click(sender As Object, e As EventArgs) _
Handles btnPrint.Click
    dlgPreview.Document = doc
    dlgPreview.ShowDialog(Me)
End Sub
```

- Test your solution: