**Problem Statement and steps to approach the solution**

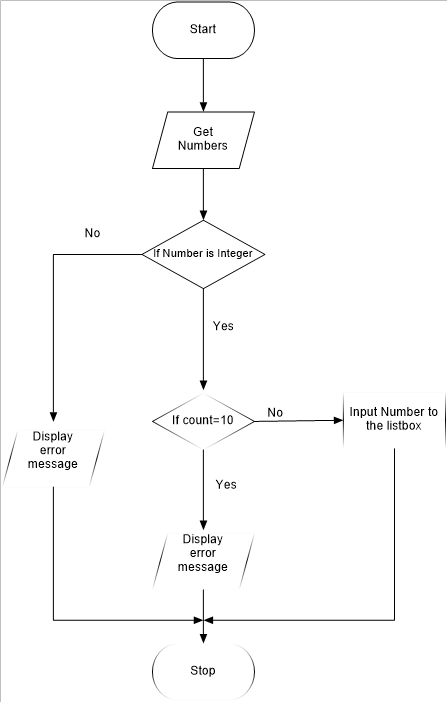
**Activity 1**

**Issue Statement**

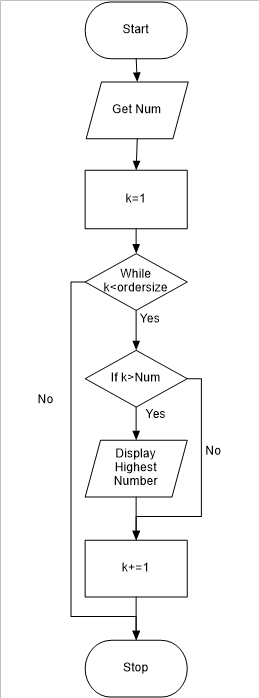
* To create a program to input numbers and display them in a list box and display the highest value, low value and the total value

**Step to solve the problem**

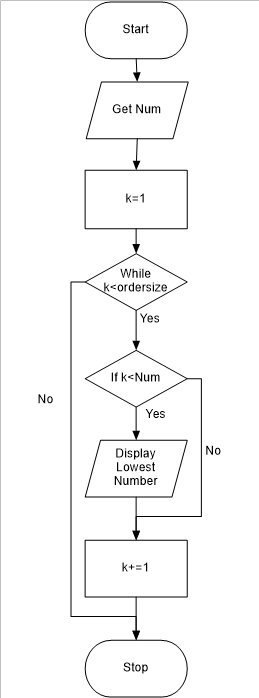
* First we should enter the number to the text box and get them to a list box the following flowchart is used to solve this problem



* Then to find the total value for loop can be used
* To find the maximum value while loop can be used the flowchart is given below



* To find the minimum value while loop can be used the flowchart is given below



**VB.Net code and the commenting for the above mentioned problem**

Public Class Form1

Private \_numList As Object

Private Property NumList(x As String) As Object

Get

Return \_numList

End Get

Set(value As Object)

\_numList = value

End Set

End Property

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

Dim num As Integer

num = GetNum(num)

'Calling the procedure'

End Sub

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

Dim Sum As Integer

For Each row As String In OrderList.Items

Sum = Sum + CInt(row)

Next

'Using for loop to add each number in the listbox'

txtTotal.Text = Sum

End Sub

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

Dim k As Integer = 1

Dim ordersize As Integer = OrderList.Items.Count

Dim h\_Num As Integer = CInt(OrderList.Items(0))

Do While (k < ordersize)

If (CInt(OrderList.Items(k)) > h\_Num) Then

h\_Num = CInt(OrderList.Items(k))

End If

k += 1

Loop

'Using while loop to find maximum value from the listbox and display the maximum value'

txtHighest.Text = h\_Num.ToString

End Sub

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

Dim k As Integer = 1

Dim ordersize As Integer = OrderList.Items.Count

Dim l\_Num As Integer = CInt(OrderList.Items(0))

Do While (k < ordersize)

If (CInt(OrderList.Items(k)) < l\_Num) Then

l\_Num = CInt(OrderList.Items(k))

End If

k += 1

Loop

'Using while loop to find minimum value from the listbox and display the minimum value'

txtLow.Text = l\_Num.ToString

End Sub

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

'Clear the screen for the next entry'

txtHighest.Text = ""

txtInsert.Text = ""

txtLow.Text = ""

txtTotal.Text = ""

OrderList.Items.Clear()

End Sub

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

End

'Exit from the program'

End Sub

Public Function GetNum(ByVal num As Integer) As Integer

'Procedure for displaying numbers in the listbox'

If Integer.TryParse(txtInsert.Text, num) Then

'converts string to integer'

'Here it is used to accept only the integer values'

If OrderList.Items.Count = 10 Then

'checks whether the listbox is filled with its maximum numbers'

MsgBox("10 values are already loaded", vbCritical, "Overloaded")

Else

OrderList.Items.Add(txtInsert.Text)

'Add number to the list box'

End If

Else : MsgBox("Not a number! Only numbers are allowed", vbCritical, "Error")

End If

Return num

End Function

End Class

**Screen shots for Activity 1**

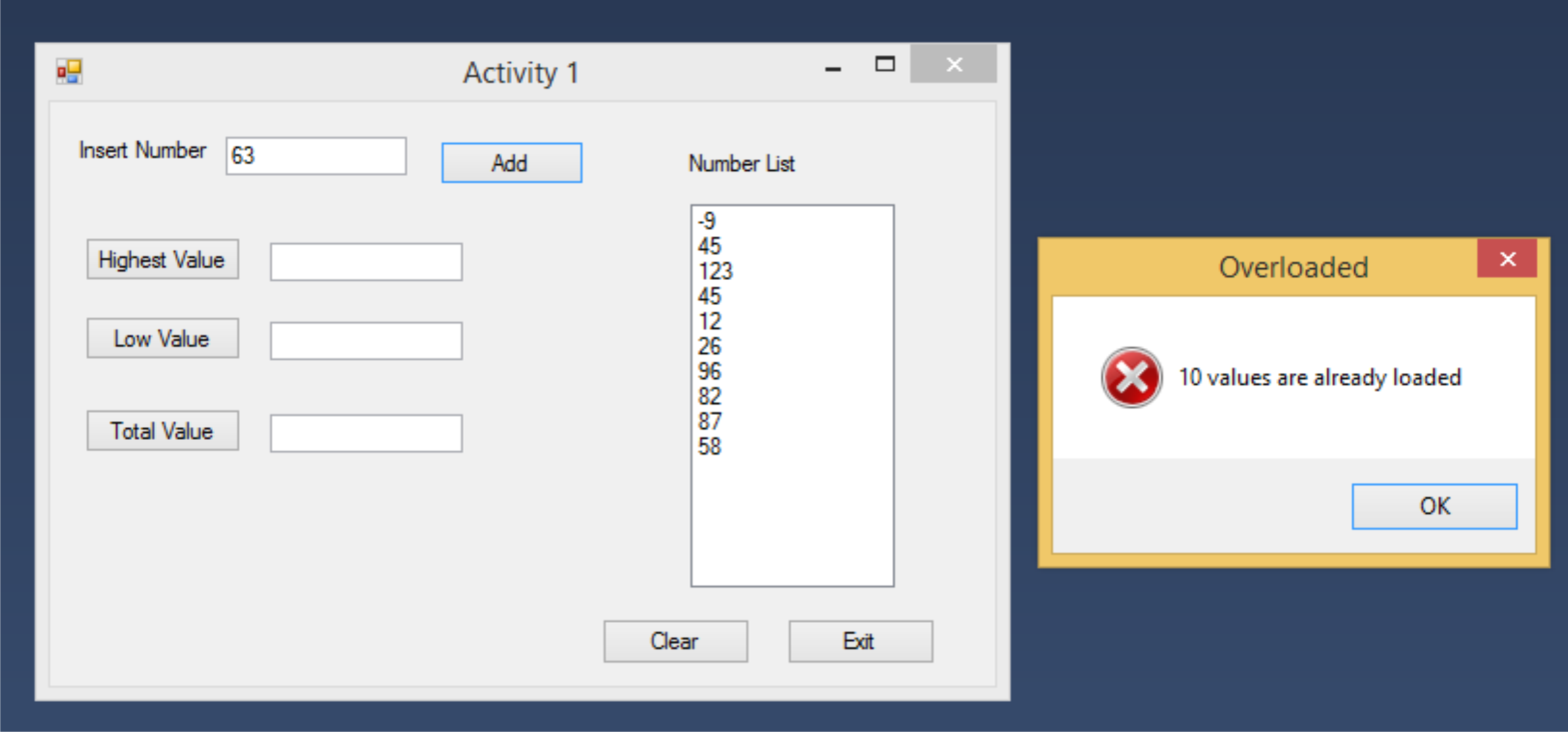


Figure 1 Error message for numbers exceeding 10 entries

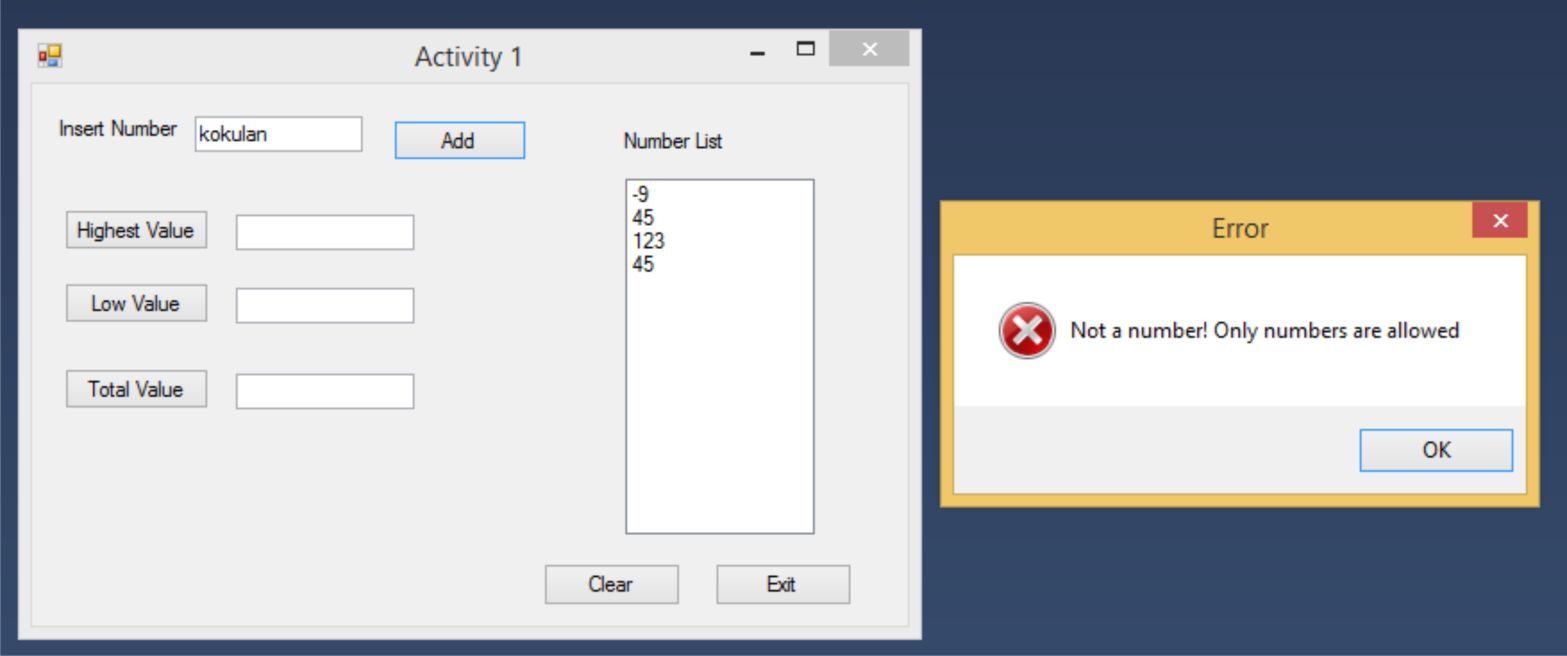


Figure 2 Error message for not inserting an Integer

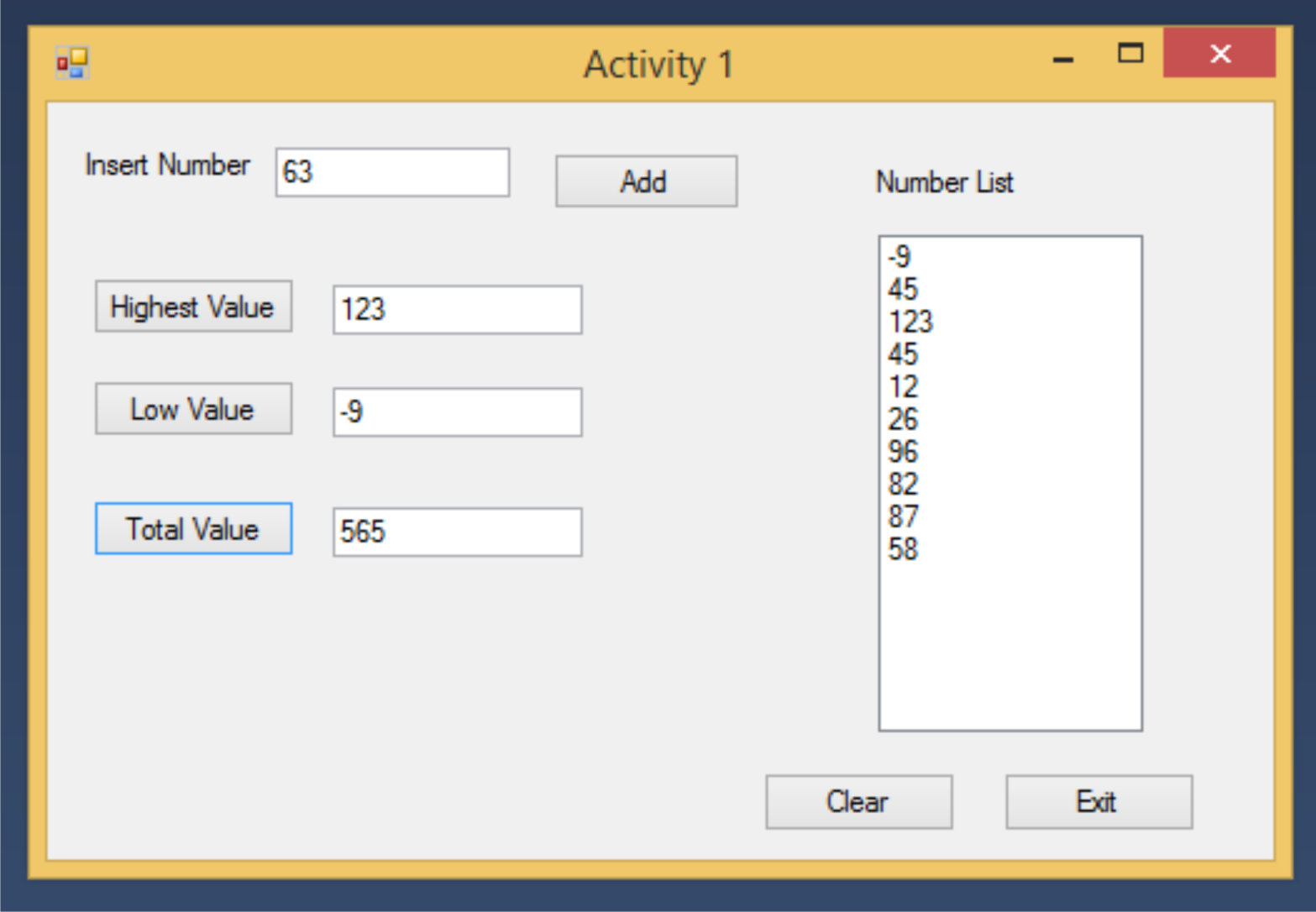


Figure Activity 1 program at a glance

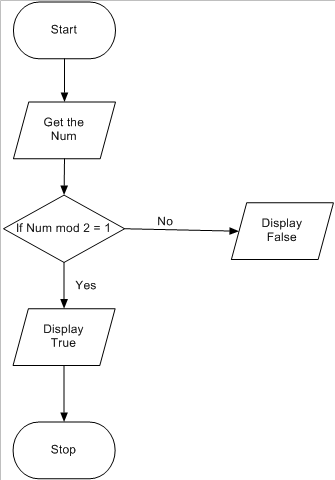
**Activity 2**

**Issue Statement**

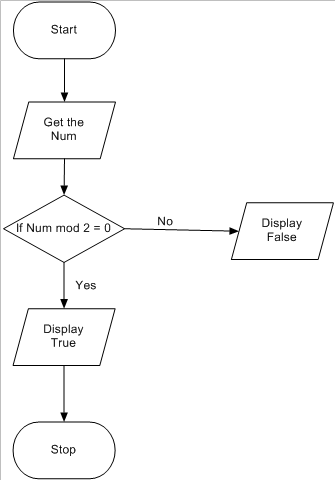
* To create a program to input numbers and display them in a list box and filter both odd and even numbers and add them to two different list boxes

**Step to solve the problem**

* First we should enter the number to the text box and get them to a list box the flowchart is same as the flowchart used in Activity 1.
* To sort the odd number if condition can be used and to get the numbers into a separate list box while loop can be used.



* To sort the Even number if condition can be used and to get the numbers into a separate list box while loop can be used



**VB.Net code and the commenting for the above mentioned problem**

Public Class frmActivity2

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

Dim Num As Integer

Num = GetNum(Num)

'Calling the procedure'

End Sub

Public Function GetNum(ByVal num As Integer) As Integer

'Procedure for displaying numbers in the listbox'

If Integer.TryParse(txtInsert.Text, num) Then

'converts string to integer'

'Here it is used to accept only the integer values'

If OrderList.Items.Count = 10 Then

'checks whether the listbox is filled with its maximum numbers'

MsgBox("10 values are already loaded", vbCritical, "Overloaded")

Else

OrderList.Items.Add(txtInsert.Text)

'Add number to the list box'

End If

Else : MsgBox("Not a number! Only numbers are allowed", vbCritical, "Error")

End If

Return num

End Function

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

Dim k As Integer = 1

'setting count'

Dim ordersize As Integer = OrderList.Items.Count

Do While (k < ordersize)

'works until the count is less than the count of the number in the listbox'

If o\_Num(OrderList.Items.Item(k)) Then

'Calling the procedure and using if condition to add the odd number to the listbox'

OddList.Items.Add(OrderList.Items.Item(k))

End If

k += 1

Loop

End Sub

Private Function o\_Num(i\_Num As Integer) As Boolean

'procedure for checking the odd number'

If (i\_Num Mod 2 = 1) Then

Return True

Else

Return False

End If

'If condition to check the odd number'

End Function

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

Dim k As Integer = 1

'setting count'

Dim ordersize As Integer = OrderList.Items.Count

Do While (k < ordersize)

'works until the count is less than the count of the number in the listbox'

If e\_Num(OrderList.Items.Item(k)) Then

'Calling the procedure and using if condition to add the even number to the listbox'

EvenList.Items.Add(OrderList.Items.Item(k))

End If

k += 1

Loop

End Sub

Private Function e\_Num(o\_Num As Integer) As Boolean

'procedure for checking the even number'

If (o\_Num Mod 2 = 0) Then

Return True

Else

Return False

End If

'If condition to check the even number'

End Function

End Class

**Screen shots for Activity 2**

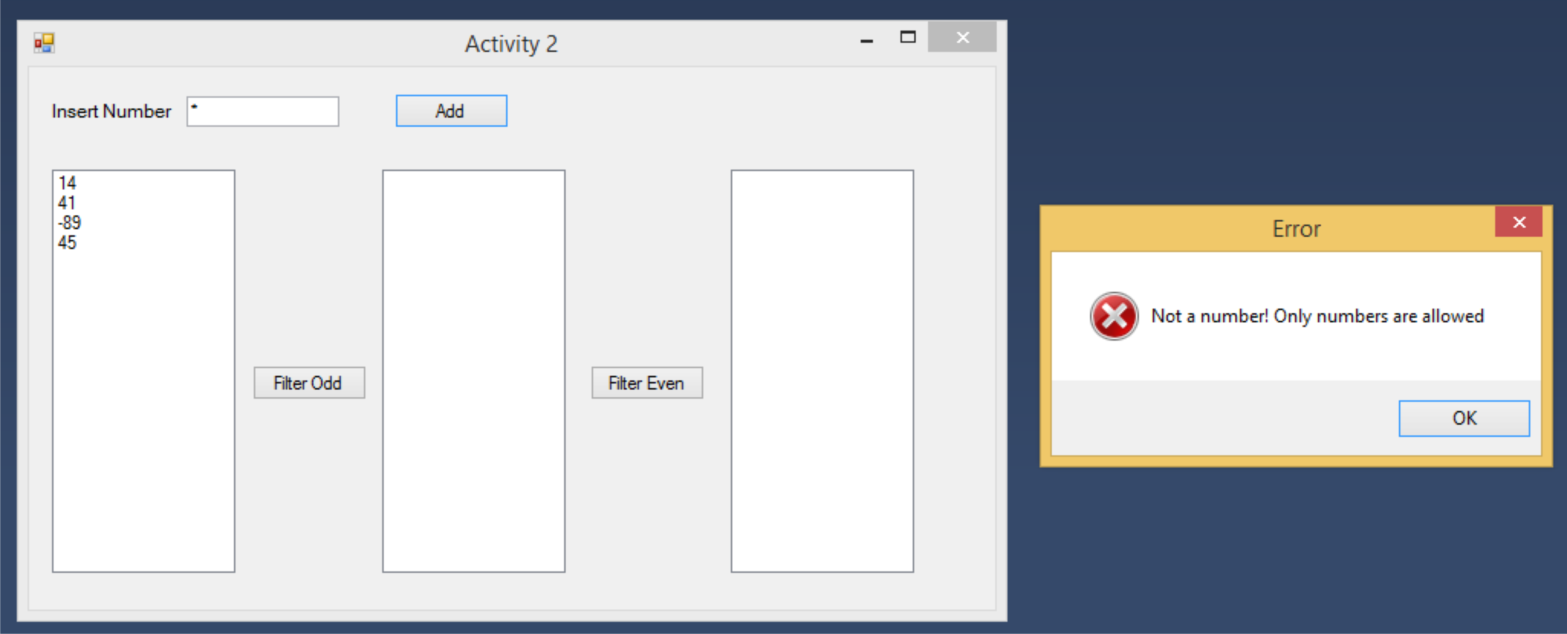


Figure error message for not inserting an integer

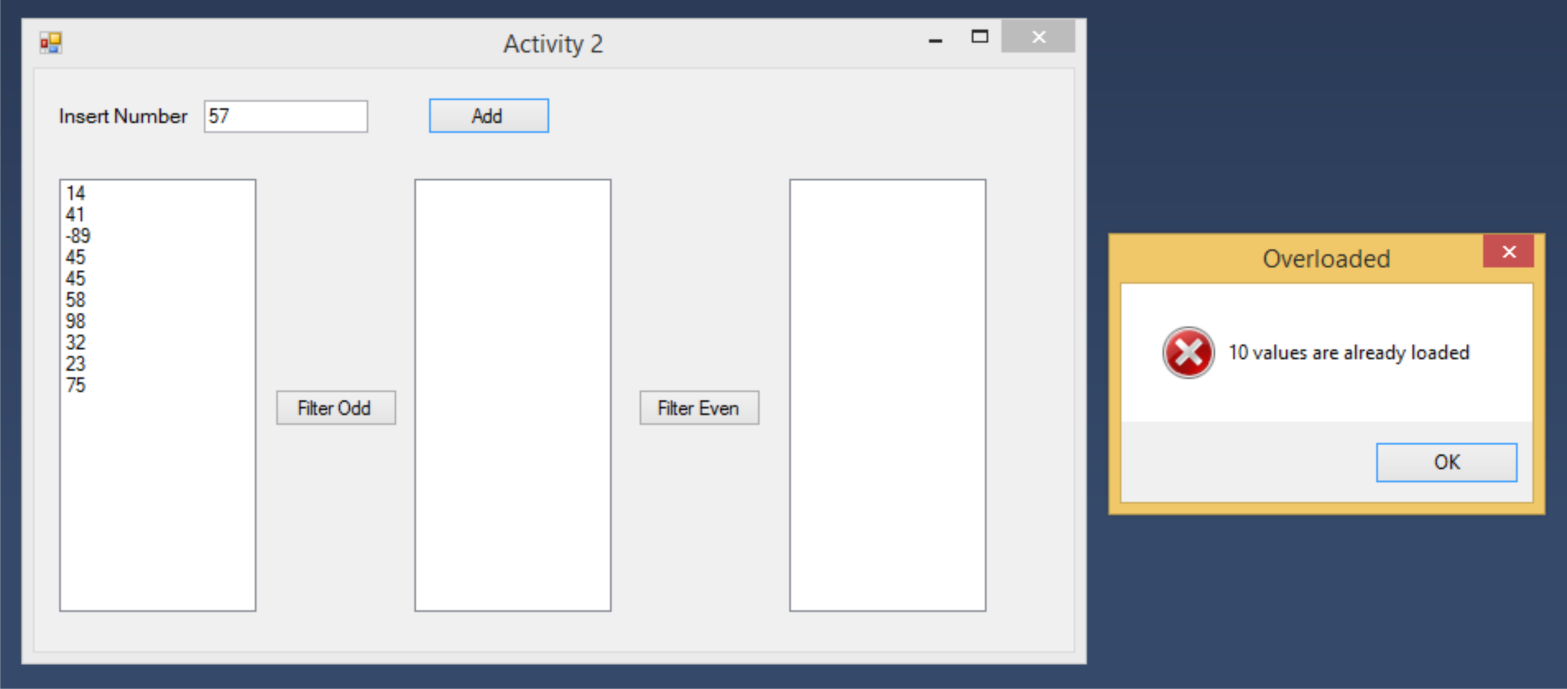


Figure 5Error message for numbers exceeding the limit

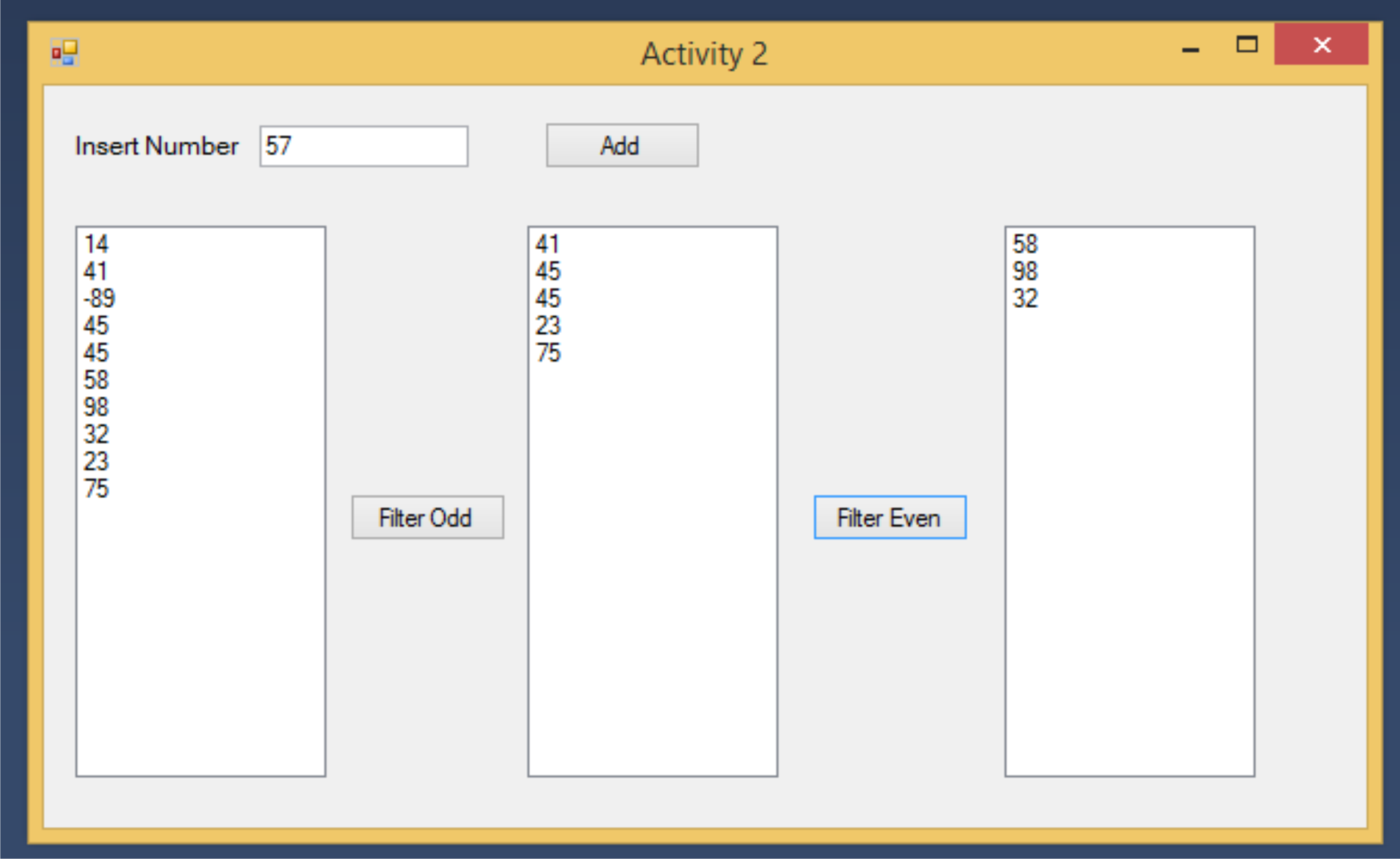


Figure 6 Overall View of Activity 2