Overview (1):
- Before We Begin
  - Some administrative details
  - Some questions to consider
- ListBox Control
  - Introduction
- Date Type
  - Introduction
  - Working with the Date type

Administrative Details (1):
- Lab Exercises
  - You should be working on Ex 5-3 this week
    - Due February 27
  - Still have a few exercises and tests that were previously distributed but have not been picked up yet
    - If you have not picked up any exercise or test yet, you can after the lecture
  - I will be in the Glade Lab today after the lecture for about 30 minutes

Some Questions to Consider (1):
- What is a counted loop?
- When should we use a counted loop?
- What is the loop index?
- If we can use a counted loop, is it wrong if we use a conditional loop instead?
- Can a counted loop count “backwards”??
Introduction (1):

- As an Aside
  - Recall that an object contains properties that can be accessed, modified etc.
  - An object can also have methods associated with it
    - A method is a sub-program (think of the event handlers we know) that can take zero or more arguments and returns one value
    - Since a method is associated with (belongs to) an object, it is accessed in the same manner as an object's properties → using the "dot" notation
      $$\text{objectName.methodName}$$

Introduction (2):

- What is a ListBox Control?

Introduction (3):

- What is a ListBox Control? (cont.)
  - An object containing a list of output
    - If the data displayed in the ListBox exceeds its height, a scroll bar appears
    - Displays on each row a string value, generically called an item
    - The item must be displayed on the ListBox using the AddItem method of the ListBox
      $$\text{listBoxName.AddItem(stringExpression)}$$

Introduction (4):

- What is a ListBox Control? (cont.)
  - Example → displaying a row in a listBox called List1
    $$\text{Private Sub Form_Load()}
    \text{Dim testString As String}
    \text{testString = "This is a test of the ListBox control"}
    \text{List1.AddItem (testString)}
    \text{End Sub}$$
  - After executing the above code segment, the following is observed in the ListBox control placed on the form

Introduction (5):

- What is a ListBox Control? (cont.)
  - When we add information to the ListBox (via the "addItem" method), the new information is appended to the next line
  - But what if we don't want to append and wish to start "clean" → there is a method to clear the ListBox of any information it may currently hold thus allowing you to "start fresh"
    - The method to clear the ListBox is "Clear" and takes no arguments → ListBox.Clear

The Date Data Type
Introduction (1):
- Dates Are Common Hence the Date Type
  - Represent dates and times
    - Stored as 64-bit (8-byte) integers
    - Represent dates ranging from January 1 of the year 1 through December 31 of the year 9999
    - Represent times from 0:00:00 (midnight) through 11:59:59 PM
    - Must be enclosed within number signs (#) and be in the format M/d/yyyy → for example #5/31/1993#

Working With The Date Type (1):
- Declaring A Date Variable
  - As with any other variable declaration
    - Dim birthDay As Date
    - Dim lastDayOfSchool As Date
    - birthDay = #10/10/1999#
    - lastDayOfSchool = #1/1/9999#
  - Can also declare Date constants
    - Const birthDate As Date = #10/10/1999#
    - Const examDate As Date = #1/20/2006#

Working With The Date Type (2):
- Date to String Conversion
  - As an aside → if you convert a Date value to the String type
    - Date is rendered according to the short date format recognized by your computer
    - Time is rendered according to the time format (either 12-hour or 24-hour) in effect on your computer

Working With The Date Type (3):
- Built-in Date Related Functions
  - How can we obtain today's date ?
    - Use the "Date" command
      - Dim myDate As Date
      - myDate = date
      - Text1.text = CStr(myDate) → “1/24/2006”
  - Visual Basic contains many built-in functions that deal with the Date type
    - Allow for various processing of Dates

Working With The Date Type (4):
- Functions Relevant to Exercise 5-3
  - Converting a String to a Date Type
    - Use the CDate conversion function → takes a String argument and returns a Date type representation of it
      - Dim myDate As Date
      - Dim myString As String
      - myString = “1/1/2006”
      - myDate = CDate(myString)

Working With The Date Type (5):
- Functions Relevant to Exercise 5-3 (cont.)
  - Updating a Date object → use the DateAdd function
  - General form → DateAdd(interval, number, date1)
    - Interval → a string specifying to add years ("yyyy"), months ("m"), days ("d") etc.
    - Number → how many of the specified intervals to add
    - date1 → the Date object to which the specified interval are to be added
Working With The Date Type (5):
- Functions Relevant to Exercise 5-3 (cont.)
  - Updating a Date object Example
    - Suppose we have a Date object representing the date "1/1/2006" and we want to add 6 months to it

```
Dim myDate As Date
Dim myNewDate As Date
Dim myInteger As Integer

myInteger = 6
myDate = CDate("1/1/2006")
myNewDate = DateAdd("m", myInteger, myDate)
```

Live Demos (1):
- "Live" Examples of Counted Loops and ListBoxes
  - Lets look at some simple examples of working with counted loops and ListBox controls in Visual Basic