Overview (1):

- Before We Begin
  - Some administrative details
  - Some questions to consider
- Topic Overview
  - Introduction to Topic C
- Boolean Expressions
  - Introduction to Boolean expressions
  - Boolean expressions in greater detail
  - Visual Basic example

Before We Begin
Administrative Details (1):

- Lab Exercise 4-4
  - This week, you should be working on Ex. 4-4 from your textbook
  - Due Monday, February 6 2006 before noon
    - As usual, place in the assignment drop-box located on the 1st floor of the CSE building just by the elevator and CSE undergraduate offices
    - Wednesday's office hours will be held in the Glade lab

Administrative Details (1):

- Test 1
  - Weight → 15%
  - Wednesday, February 8 2006 (entire period)
  - Recall that you do not have to write the test (if you don't, weight of the test will transferred to weight of final)
    - Probably a good idea to write it however!
  - More about the test on Friday

Some Questions to Consider (1):

- Describe how we can determine whether to declare a variable global or local?
- Why not simply declare all variables global?
- Do variables have to be explicitly declared?
- What is the "Option Explicit" statement?
- Why use the "Option Explicit" statement?
Introduction (1):

- Up Until This Point, All Our Statements Relied on Sequential Processing
  - We have used operators and variables to form expressions and assigned value resulting from the expression to a variable or object property
  - These statements have been executed sequentially → one after the other in the order they appear within the code of the event handler (e.g., in sequence)
  - This is of course very limiting → not adequate for many tasks we want to accomplish!

Introduction (2):

- Limitations of Sequential Processing
  - Does not provide the programmer the option of taking separate "program paths" depending on the outcome of certain operations
    - User input example → perhaps we want to perform one task if the user enters "x" and another task if the user enters "y"
    - Error checking example → what if we can examine a user's input to determine whether it is valid or not and perform one task if it is valid and another if it is not?
Introduction to Computer Use II

**Introduction (3):**
- **Overview of Topic C**
  - We will learn how to write projects (programs) that can take one action or another based on a condition.
  - How we can go about making the selection of which statements to execute within our code.
- **Main concepts of this topic**
  - Comparison operators
  - Selection statements
  - Boolean operators
  - Option button and checkbox controls
  - Validation of user input

---

**Boolean Expressions**

---

**Introduction (1):**
- **True / False Considerations**
  - Any decision as to whether or not to take one course of action or whether to take one course of action instead of another is essentially the result of a true / false consideration.
  - One action is taken only, not both!
  - For example → "Do I have enough money to do this?"
    - If the expression is true then I do have enough money hence I can "do this" otherwise, expression is false and I "cannot do this" since I don't have enough money.
Introduction to Computer Use II

Introduction (2):

- True / False Considerations (cont.)
  - In a computer program you can also set up an expression that results in a true / false answer and determine program execution based on this outcome.
  - This is actually a very powerful asset of any computer programming language including Visual Basic!
  - A decision made by the computer is formed as a question → Is a given condition true or false?
    - If true, do one thing
    - If false, do something else

Introduction (3):

- True / False Considerations (cont.)
  - For example

    If the sun is shining Then (condition) go to the beach (action to take if condition is true)
    Else
        go to class (action to take if condition is false)
    End If

Introduction (4):

- True / False Considerations (cont.)
  - Flowchart illustrations of previous examples
Introduction (5):

- True / False Considerations (cont.)
  - Another example
    - In this example, no action is explicitly taken if the condition is not true → action taken only if condition is true

```
If you don't succeed Then
  try, try again
End If
```

Introduction (6):

- True / False Considerations (cont.)
  - Flowchart illustrations of previous examples

![Flowchart]

Boolean Expressions (1):

- What is a Boolean Expression?
  - In a programming language, True / False conditions are expressed with Boolean expressions
  - Essentially a comparison between two values (variables)
  - Evaluate to either True or False
  - But aren't there many types of comparisons that can be made → yes!
    - Visual Basic does allow for various different comparisons to be made with various comparison operators

CSE 1530 Winter 2006
Bill Kapralos
Boolean Expressions (2):

- What is a Boolean Expression? (cont.)
  - Commonly used Visual Basic comparison operators

<table>
<thead>
<tr>
<th>OPERATOR</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to</td>
</tr>
<tr>
<td>=</td>
<td>Equal to</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Not equal to</td>
</tr>
</tbody>
</table>

Boolean Expressions (3):

- Using Comparison Operators
  - Examples
    - `myMoney > priceOfItem` → compare whether "myMoney" is greater than `priceOfItem` and if it is, the expression evaluates to True (e.g., entire expression is replaced with True) otherwise it evaluates to False
    - `myMoney = priceOfItem` → Check whether `myMoney` is equal to `priceOfItem` and if it is, the expression evaluate to True otherwise it evaluates to False

Boolean Expressions (4):

- Using Comparison Operators (cont.)
  - Be careful when using the comparison operators!
    - Should ensure that the data types of both the values (variables) are the same
    - Visual Basic will of course attempt to convert values for you but remember → result may not necessarily be what you expect
    - At times the automatic conversion by Visual Basic may give expected results → `2 > 2.3` will evaluate to False as expected (integer and double)
    - Result may not be correct when comparing Single and Double values → (Double converted to Single)

CSE 1530 Winter 2006

Bill Kapralos
Boolean Expressions (5):

- **Equality vs. Assignment**
  - But we have seen that the "=" operator is the assignment operator ??? But we just saw it is also used as a comparison operator ???
  - In most other programming languages, a separate operator is used for the equality operator → for example, in C/C++ the comparison operator is "==".
  - There are some dangers associated with this → for example, if we want to compare two variables but we use the assignment operator accidentally varA = varB instead of varA == varB.

Boolean Expressions (6):

- **Equality vs. Assignment (cont.)**
  - In Visual Basic, the "=" operator has a dual meaning → correct meaning determined based on context!
    - Can be used as the assignment operator.
    - Can also be used as the comparison operator.
    - Determining correct meaning based on context can lead to confusion at times → what does the following mean and under what circumstances might the following be valid?
      
      varOne = varTwo = varThree

Boolean Expressions (7):

- **Equality vs. Assignment (cont.)**
  - Lets experiment with Boolean expressions by working with Exercise 4-1.
    - [Experiment with Comparison Operators](image)
    - Simple program to enter two values & compare them.
    - We will experiment with various types by changing the program code of course!


**Boolean Expressions (8):**

- Equality vs. Assignment (cont.)
  - Let's take a closer look
    - `varThree = (varOne = varTwo)`
  - Meaning by context → what happens if the brackets are removed??
    - `varThree = varOne = varTwo`
  - What happens if `varThree` is not a Boolean?