A Little Review:
What is the Value of wicked, dude?

Take out a piece of scratch paper. See if you can answer the questions below.

Dim wicked As Integer
Dim dude As Integer
dude = 5
wicked = 2
dude = wicked * 5  // ' means multiply
wicked = dude + 1

Questions:
1. What values do dude and wicked contain at the end of this code?

Expressions

CONCEPT: Expressions are a means of performing the actual computation in a program. They are formulae made from variables and operators, e.g., calculator operations:

- weeks = days / 7  // divide value of days by 7
- totalAfterTax = totalPrice * 1.087  // multiply the two values

The Fundamental Rules of Assignment:
- The general form of an assignment statement is <variable name> <assignment symbol> <expression>
- The flow of information is always right-to-left
- The expression is evaluated before the assignment is made
  - score = score + 3

When A Decision Must Be Made:

Conditionals

Computers can be programmed to make decisions – that is, to choose one path to follow from many alternatives. Conditions are the programming tool that implements this concept

The Reason to Have Conditionals:

CONCEPT: Computer programs execute all statements in the program in order unless the program is instructed to only execute certain statements under certain conditions

For example:
- If (something is true) Then
  [do this part of the program]
End If
Operators:

- CONCEPT: Operators are used to combine expressions (logical operators) or to compare expressions (relational operators)
  - They are used in combination with values, or variables that contain values – both called operands – to complete the expression formulae
- Most programming languages have more logical operators than a pocket calculator
  - Operators like + taking 2 operands are called binary: \( a + b \)
  - Operators like – taking 1 operand are called unary: \( -a \)
- A very useful logical operator is concatenate, & in VB6, which connects two strings or variables together:
  - plural = "dog" & "s"

Basic Conditional

- Use conditionals to test to see if a condition holds:
  - If \( \text{temp} < 32 \) Then
    - \text{state} = "frozen"
    - \text{form} = "ice"
End If
- General form of basic conditional:
  - If \(< \text{T/F expression}> \) Then
    - \text{<code statements>}
  - End If
- What this means:
  - First, the \(< \text{T/F expression}> \) is evaluated
  - If the outcome is true, then the statements that follow Then are performed
  - If the outcome is false, then the statements that follow Then are skipped

General Conditional Statement

- CONCEPT: When one set of statements must be performed for the true conditions and a different set of statements are needed for the false conditions, use the If-Then-Else statement
- General form:
  - If \(< \text{T/F expression}> \) Then
    - \text{<code statements>}
  - Else
    - \text{<code statements>}
End If

  - If \( \text{sky} = \text{"clear"} \) AND \( \text{temp} > 70 \) Then
    - clothing = "tank top"
  - Else
    - clothing = "sweats"
End If
“Nested” If-Then-Else

CONCEPT: An advantage of the general conditional is that it can be imbedded within another conditional.

```vbnet
If sky = "clear" AND temp > 70 Then
    clothing = "tank top"
    If laundry = "clean" Then
        clothingColor = "purple"
    End If
Else
    clothing = "sweats"
    If ground = "muddy" Then
        shoes = "boots"
    End If
End If
```

Exercise #1

What does this print?

```vbnet
Dim x As Integer
x=10
If x=1 Then
    Print "octopus"
ElseIf x=2 Then
    Print "squid"
Else
    Print "clam"
End If
Print "mollusk"
```

Let's Move From Theory to Practice!

We want to write a program that takes an integer as input and outputs whether or not the integer is a positive number.

- How should we get the user's input?
- How do we tell if the input is positive or negative?
- What should we do with an input of zero?
- How should we output the "positive" or "negative" evaluation to the user?
  - Be Creative!
- How do we get started?

For Monday

Read Chapters 13 and 14 of the FIT course pack

Reading through Lab 9

Project 2, part 1 is due Wednesday
- You will submit through the web your project, form and executable files
- The link will be up by Monday on the Project Turn In page
Writing a VB6 Program

Sign Find is preparation for Project 2. You will finish the more complex version called Sign Finder for Lab 9. We will start and debug this program.

Form Set-up (the interface)

- Components (controls, objects, items) needed for the form
  - Heading text
  - 12 radio buttons, each with its own name
  - OK button
  - Clear button
  - Sign display

Initial Program Code Steps

- Declare a variable named sign as a string to hold the sign values for each range of dates
- For each month's click event handler, assign to sign the string value of the the sign for that range
- For the OK button click event handler
  - Make the label that will display the sign text visible
  - Make the OK button invisible
  - Make the clear button visible
  - Display the value in sign in the caption of the sign display label

What's Going On in the Program Code

```vbnet
Option Explicit
Dim sign As String

Private Sub optCan_Click()
    sign = "Cancer"
End Sub

Private Sub optCap_Click()
    sign = "Capricorn"
End Sub

Private Sub cmdOK_Click()
    cmdOk.Visible = False
    cmdClear.Visible = True
    lblSign.Visible = True
    lblSign.Caption = "Your sign is " & Sign
End Sub
```

For the January/February radio button, the assignment here saves the value that is used here.