Constants and Variables

Named Constants are used to hold values that never change, at least for the life of the program. Some constants truly are constant, for example, the value of pi is always 3.14...., and the number of inches in a foot are always 12. At other times, the same value will hold for the run of a program, e.g., number of students, tax rate, etc. Declaring constants looks like the following:

```
Const Pi as Single = 3.14
Const Inchperfoot as Integer  = 12
Const Numstudent as Integer = 40
Const Welcome as String  = "Welcome to the Superette"
Const Taxrate as Single = 0.07
        ** Note that if you have a percent, it must have a 0 in front of the decimal
```

Variables are used to hold values that can change while a program is running. There are three types of variables (that we're concerned with now).

Integer -- for whole numbers, including negative whole numbers
Single -- for numbers that can have a decimal
String -- for strings of characters

Declaring variables looks like the following:

```
Dim Age as Integer
Dim GPA as  Single
Dim Student as  String
```

Good programming practice requires that all variables be declared. Our version of Visual Basic is set up to require you to declare all variables and will give you an error if you don't.

Assignments

In order to give a variable a value, you assign a value to it. The value could be a simple value, or it could be the result of an expression. The format is as follows:

```
Variable = value
Variable = expression
```

The value may be one of a number of things:

An actual value	IQ = 115
The value of another variable	YourIQ = MyIQ
The result of a computation	YourIQ = MyIQ - 40
The value of a constant	Length = footlength (where footlength was declared as 12)

Although arithmetic is fully covered in another section, here are some more examples of computing values:

The results of computing a multiplication:

```
salestax = baseprice * taxrate
```

The result of a concatenation of strings:

```
welcome = “Howdy” & “, “ & “Pardner”
```

The initial assignment can be given to a variable when it is declared:

```
Dim counter as Integer = 0
Dim totalprice as Single = 0.0
```

Input and Output to Textboxes

In order to get input values from the user interface, you can get numbers or strings that the user types in to a textbox. You can also output numbers and strings by assigning them to a textbox.

Using the textbox for values means that you will use the text property. Suppose that you put a textbox on your user interface and that you put its name property to be txtNumber. If the user types in a value, it goes into the Text property and to get that value (and convert it to an integer), you say

```
CInt(txtNumber.Text)
```

Normally, you may assign this value to a variable:

```
Dim firstnumber as integer
firstnumber = CInt(txtNumber.Text)
```

Now the variable firstnumber will have the integer value that the user typed in.

There is a text conversion function for each data type:

CInt	converts text to an integer
CSng	converts text to a single
CStr	converts text to a string (if necessary)

There are many cases when Visual Basic allows you to omit the conversion function, but it's good to know them if you get an error.

You can also use the Text property to output a value to a Textbox. For example,

```
txtNumber.Text = Cstr(intsum)
```

Another useful function for output is the Format function. It not only converts integer and single numbers to string to put as output, it formats the appearance of the number. For example, in the following program fragment, a number of type single is formatted to look like money using the currency format:

```
Dim number as Single
number = 10.9700
TextBox1.Text = Format (Number, "currency")
```

In the textbox will appear: $10.97

Other formats include:

"standard"	put commas in-between thousands and put two numbers to the right of the decimal
"percent"	display number multipled by 100 and with a percent sign %
"scientific"	use standard scientific notation