VBScript: Math Functions

In this chapter, you will learn how to use the following VBScript functions to World Class standards:

- 1. Writing Math Equations in VBScripts
- 2. Beginning a New VBScript Program
- 3. Adding Copyright Statement to a Program
- 4. Using a Message Box to Communicate the Copyright and More
- 5. Declaring Variables in a Program with the Dimension Statement
- 6. Setting Variables in a Program
- 7. Adding Numbers in VBScript
- 8. Subtracting Numbers in VBScript
- 9. Multiplying Numbers in VBScript
- 10. Dividing Numbers in VBScript
- 11. Finding Remainders in VBScript
- 12. Computing Absolute Values in VBScript
- 13. Fixing Numbers in VBScript
- 14. Rounding Numbers in VBScript
- 15. Computing Exponents in VBScript
- 16. Computing Square Roots in VBScript
- 17. Computing Sine's in VBScript
- 18. Computing Cosines in VBScript
- 19. Ending the Program
- 20. Saving the Program
- 21. Running the Program

3-1

Writing Math Equations in VBScript

There are many arithmetic and geometric functions that we can utilize in programming a VBScript. Some arithmetic functions include adding, subtracting, multiplying and dividing, which are obvious for everyday work. The absolute value, exponent and square roots are for more serious technicians. Even greater code writers that are computing angles are using sines and cosines. These functions are easy to learn and there are examples in each section to practice every function in this chapter.

We want to start the learning process using the skills we learned in chapter 2, so in this chapter, we will continue to use message and input boxes. So lets get started.

Beginning a New VBScript Program

With the EditPad Lite program open, select **File** on the Menu Bar and select **Save**. We will save the file in the folder on either the desktop or the My Documents folder called "VBScripts". After locating the folder, name the file "math functions.vbs". The file extension *vbs* as can be easily seen means Visual Basic Script. We can run a finished VBScript file with the *vbs* extension at the command line by going to the Start button, select Run, and then Browse to locate the script program. When the plain text is compiled at runtime, the program will execute our commands in precise order.

If we want to change the font type or size, select Options on the Menu Bar and pick **Font**. The Font window will appear and we can select the font, font style and size. We will select Arial Narrow, Regular font style and 12 point size for this textbook. We suggest that programmers select text shapes that are easy to read. We are using the narrow style font to eliminate the need for word wrapping.

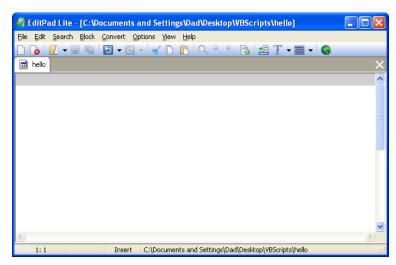


Figure 3.1 – The EditPad Lite Script Editor

Adding a Copyright Statement to a Program

At the beginning of a new program, we will expect to see an explanation or any special instructions in the form of comments such as copyright, permissions or other legal notices to inform programmers what are the rules dealing with running the code. Comments at the opening of the code could help an individual determine whether the program is right for their

application or is legal to use. The message box is a great tool when properly utilized to inform someone if they are breaking a copyright law when running the code.

As in Visual Basic, the single quote character (') will precede a comment. When the code is compiled, comments are ignored.

Begin the hello.vbs script with:

'math functions.vbs copyright (c) 2006 by charles w. robbins

Then add a short description for the program:

' this program will give the programmer exposure to math functions used in vbscripts

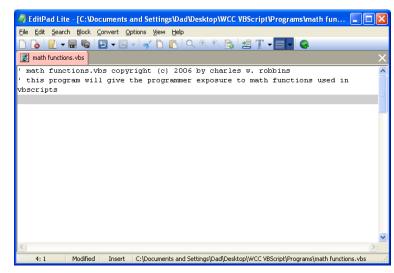


Figure 3.2 – Adding a Copyright Statement

Using a Message Box to Communicate the Copyright and More

The comments we placed in the first two lines of the program will inform the individual opening and reading the code, but those user that may run the application without checking, the message box is a great tool to alert the client to the rules of the program and what will the application do.

In our code add another comment:

'alert the user with a message box

The function MsgBox will launch a message box in Windows. The text or as programmers state "string" will be enclosed in quotes.

Type the following line of code:

MsgBox "Math functions.vbs copyright (c) 2006 by charles w. robbins. This program will give the programmer exposure to math functions used in vbscripts"

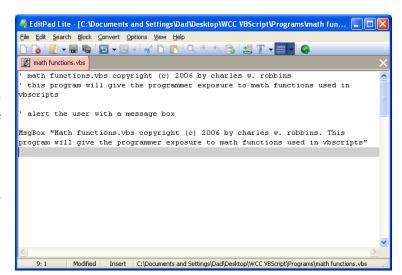


Figure 3.3 – Adding a Message Box

Declaring Variables in a Program with the Dimension Statement

When we are going to use a number, text string or object that may change throughout the life of the code, we create a variable to hold the value of that changing entity. In Visual Basic, the dimension statement is one of the ways to declare a variable at the script of procedure level. The other two ways are the Private and Public statements, which we will use in later chapters.

In our program, we will set two variables to experiment with different math functions and one variable to hold the answer to the problem.

Type the following code:

'Declare variable dim number1 dim number2 dim answer

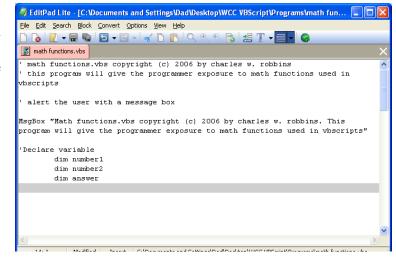


Figure 3.4 – Declaring Variables with Dim Statements

Notice that the variable name should be a word or a phrase without spaces that represents the value that the variable contains. If we want to hold a value of one's date of birth, we can call the variable, DateofBirth. The keywords Date and Birth are in sentence case with the first letter capitalized. There are no spaces in the name. Some programmers use the underscore character (_) to separate words in phrases. This is acceptable, but a double underscore (__) can cause errors if we do not detect the repeated character.

Setting Variables in a Program with an InputBox

Next, we will set the variables using the equal function (=) and an InputBox. When this section of the code is run, we want to a window appears with the prompt "Type the first number" so the user can input the value in the blank text box. After the user places the number in the text box and presses the OK command button, another window comes into view with the prompt "Type the second number" so the user can input the second value in the blank text box. We will copy this section of code in front of every math statement.

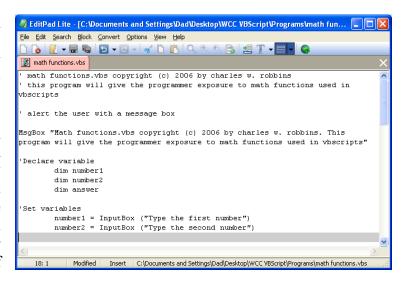


Figure 3.5 – Setting the Variables in the Script

So type the following code in the program to set the variables.

'Set variables

```
number1 = InputBox ("Type the first number")
number2 = InputBox ("Type the second number")
```

We will not set the variable **answer**, since this will be found when solving the math problem.

Adding Two Numbers in a VBScript

The first arithmetic function we will address is one of the most common, the adding function which is displayed by the icon +. The addition function allows us to add two or more numbers. The values of the numbers can be whole like 1,2,3... or decimals, positive or negative. Remember we can have more than two numbers like (2 + 3 + 7 + 4).

In this program, we will add the two variables that are holding the numbers.

Type the following code:

'Add numbers answer = number1 + number2

The variable **answer** will equal the sum of the value in variable **number1** and with the value in variable **number2**.

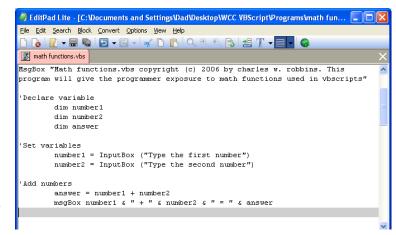


Figure 3.6 – Adding Two Numbers Together

We will use a message box to display the answer to the problem when adding the numbers together. We will use the text concatenation function (&) to connect the text string together so the message box can be easily read.

Type the following code:

msgBox number1 & " + " & number2 & " = " & answer

We will use a message box after displaying an example of each type of math function.

The following is an extract from the VBScript Quick Reference for the Addition function.

Function	Name	Description
+	Adding	The addition function will add two or more numbers
Examples		
Using integers	answer = 4 + 6	Answers 10
Using decimals	answer = 2.3 + 5.1	Answers 7.4
Using negatives	answer = 3 + -7	Answers -4

Subtracting Numbers in a VBScript

The subtraction function is similar to the adding function, but the number following the first argument is subtracted from the first number. Again there is not a limit to the number of arguments in the math statement. An easy error to make is mistaking the negative sign attached to the front of a number constructing a negative value with the minus sign in the subtraction function. Be aware of these two symbols are their meaning to prevent a troubleshooting dilemma

Now, we will subtract the second number from the first using the variables.

Type the following code:

'Subtract numbers answer = number1 - number2

The variable **answer** will equal the value of variable **number1** minus variable **number2**.

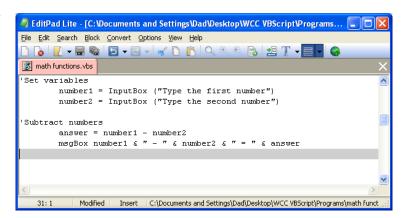


Figure 3.7 – Subtracting a Number

Type the following code for the message box.

msgBox number1 & " - " & number2 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Subtraction function.

Function	Name	Description
_	Subtracting	The subtraction function will subtract a number from the preceding number
Examples		
Using integers	answer = 4 - 6	Answers -2
Using decimals	answer = 2.3 - 5.1	Answers -2.8
Using negatives	answer = 37	Answers 10

Multiplying Numbers in a VBScript

We can use the multiplying function to multiply two or more numbers together using the asterisk icon. After practicing the simple examples in the multiplying table, we can be experimenting with compound expressions on the AutoCAD Command line like:

answer =
$$(4 + 3) * (7 - 3)$$

After computing 4 + 3 = 7 and 7 - 3 = 4, the computer calculates 7 * 4 = 28. Very neat.

Now, we will multiply the first and second numbers

Type the following code:

'Multiply numbers answer = number1 * number2

The variable **answer** will equal the value of variable **number1** times variable **number2**.

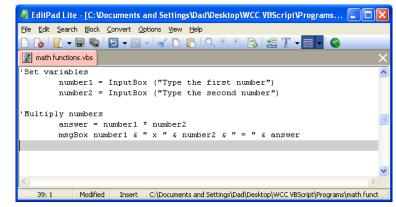


Figure 3.8 – Multiplying Numbers

Type the following code for the message box.

```
msgBox number1 & " * " & number2 & " = " & answer
```

The following is an extract from the VBScript Quick Reference for the Multiplication function.

Function	Name	Description
*	Multiplying	The multiplication function will multiply two or more numbers
Examples		
Using integers	answer = 4 * 6	Answers 24
Using decimals	answer = 2.3 * 5.1	Answers 11.73
Using negatives	answer = 3 * -7	Answers -21

Dividing Numbers in a VBScript

As the first three arithmetic functions were very similar in there handling of integers, decimals and negatives, the division function will cause some problems depending on the direction of the division symbol. Also, we can divide multiple numbers where the first argument is divided by the second and if there is a third, the process will continue until there are no more arguments, but the danger in this function exists when we change the direction of the division symbol. In any VBScript, type the expression 3/2 and the return is 1.5, and there is no problem, but type in 3\2 and the answer will be 1. When we only want the whole number in a division problem to determine the number of loops in the program, the integer division function \ will return an answer with the decimal remainder left off, so the 3\2 is 1 instead of 1.5. Some programmers make the error thinking the VBScript code will round the number up if the remainder is 5 or greater, but not so, the decimal component is just removed.

Now, we will divide the first number by the second number.

Type the following code:

'Divide numbers with the answer being a real number answer = number1 / number2

The variable **answer** will equal the value of variable **number1** divided by variable **number2**.

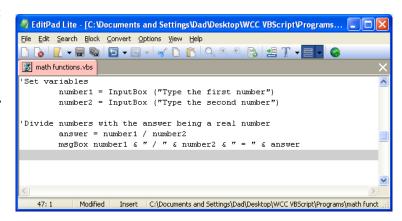


Figure 3.9 – Dividing Numbers in the Program

Type the following code for the message box.

msgBox number1 & " / " & number2 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Division function.

Function	Name	Description
/	Division	The division function will divide the first number by the second number
Examples		
Using integers	answer = 4 / 6	Answers 0.666
Using decimals	answer = 2.3 / 5.1	Answers 0.450980
Using negatives	answer = 3 / -7	Answers -0.428571428

Now we divide numbers when we only want the whole number in a division problem.

Now, we will divide the first number by the second number with the integer division symbol, \lambda.

Type the following code:

'Divide numbers with the result being an integer answer = number1 \ number2

The variable **answer** will equal the integer value of variable **number1** divided by variable **number2**.

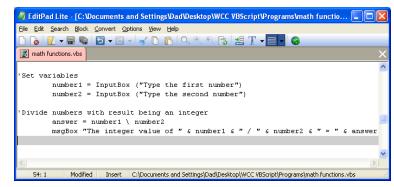


Figure 3.10 – Divide Numbers with an Integer Answer

Type the following code for the message box.

msgBox number1 & "\" & number2 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Integer Division function.

Function	Name	Description
\	Integer Division	The division function will divide the first number by the second number
Examples		
Using integers	answer = 15 \ 6	Answers 2
Using decimals	answer = 3 \ 2	Answers 1
Using negatives	answer = 13 \ -7	Answers -1

Finding Remainders with VBScripts

The Modulus Division function will return a number displaying the remainder after the second number is divided into the first number. This is a useful tool when determining whether a counter number even or odd. If you write the following expression to set the counter.

counter = 4

And the even and odd checker to determine the state of the counter.

EvenOrOdd = counter mod 2

The variable **EvenOrOdd** will be a **0** for even or a **1** for odd.

Now, we will find the remainder when we divide the first number by the second number.

Type the following code:

'Find the remainder of a division problem

answer = number1 mod number2

The variable **answer** will equal the remainder of the variable **number1** divided by the variable **number2**.

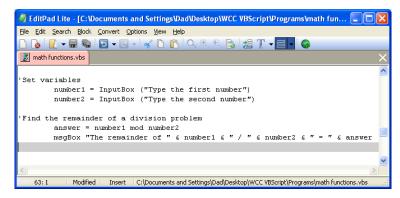


Figure 3.11 – Using the Modulus Division Function

Type the following code for the message box.

msgBox "The remainder of " & number1 & " / " & number2 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Integer Division function.

Function	Name	Description
mod	Modulus Division	The Modulus Division function will return a number displaying the remainder after the second number is divided into the first number
Examples		
Using integers	answer = 15 mod 6	Answer 3
Using decimals	answer = 3 mod 2.1	Answer 0.9
Using negatives	answer = 13 mod -7	Answer 0

Computing Absolute Values with VBScripts

The absolute value function, a function that will only allow a single case to follow the syntax **abs** will return with a positive number. This function is useful in formula writing when deciphering positive distances. In our career, we will discover that graphics program uses a starting point and ending point to explain the construction of a line. When subtracting the starting point x-value and ending point x-value depending on how the line is drawn, our answer can be a negative number. Placing the absolute value function in front of any answer will result in a positive number.

Now, we will find the absolute value of a number.

Type the following code:

'Find the absolute value of a number

answer = abs(number1)

The variable **answer** will equal the positive value of the variable **number1.**

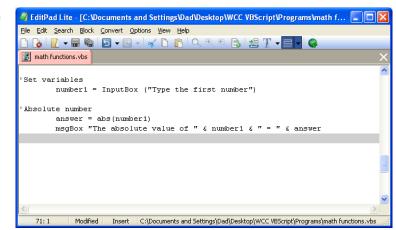


Figure 3.12 – Using the Absolute Value Function

Type the following code for the message box.

msgBox "The absolute value of " & number1 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Absolute Value function.

Function	Name	Description
abs	Absolute Value	The absolute value function will return the positive value of a number
Examples		
Using integers	answer = abs (15)	Answers 15
Using decimals	answer = abs (3.1)	Answers 3.1
Using negatives	answer = abs (-7)	Answers 7

Fixing Numbers in VBScripts

The fix function is in our list of number modifiers for arithmetic functions, but we will visit the tool again in the conversion function list. This contrivance will take a real number and remove

the decimal places leaving a whole number or integer. There is not any rounding, but the numbers to the right of the decimal place are just removed. This coding method will be useful when computing an array, where we want a whole number response.

Now, we will find the fixed (integer) value of a number.

Type the following code:

'Find the fixed value of a number answer = abs(number1)

The variable **answer** will equal the integer value of the variable **number1.**

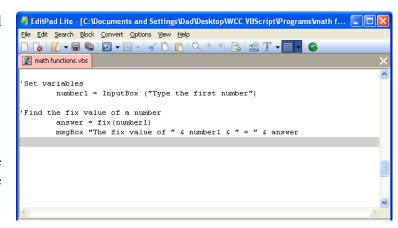


Figure 3.13 – Using the Fix Function

Type the following code for the message box.

msgBox "The fixed value of " & number1 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Fix function.

Function	Name	Description
fix	Fix	The fix function will return a whole number of a value by removing the number's decimal places
Examples		
Using integers	answer = fix (15)	Answer 15
Using decimals	answer = fix (3.1)	Answer 3
Using negatives	answer = fix (-7.5)	Answer -7

Rounding Numbers in a VBScript

The round function will take a single number and round the number to the integer value or if we set the placeholder value will round the decimal places to the placeholder value. Numbers to the right of the rounding placeholder will cause that value to increase by one if the number is 5 or above. If the number to the right of the rounding placeholder is 4 or less, the value at the precise decimal place being requested will remain the same.

Type the following code in the program to set the variables for the rounding function.

'Set variables

```
number1 = InputBox ("Type the number")
number2 = InputBox ("Type the number of decimal places")
```

Now, we round the number to a precise decimal place.

Type the following code:

'Set the decimal places for the number

answer = round(number1,number2)

The variable **answer** will be rounded to the precise number of decimal places requested.

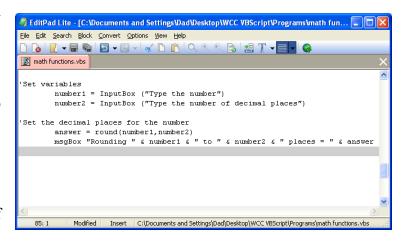


Figure 3.14 – Using the Round Function

Type the following code for the message box.

msgBox "Rounding " & number1 & " to " & number2 & " places = " & answer

The following is an extract from the VBScript Quick Reference for the Round function.

Function	Name	Description
Round	Round	The round function will return a number set to the requested decimal place.
Examples		
Using integers	answer = round (15,0)	Answer 15
Using decimals	answer = fix (3.125,2)	Answer 3.13
Using negatives	answer = fix (-7.523,1)	Answer -7.5

Computing an Exponent in VBScripts

The exponent function is used when applying formulas like the area of a circle, which is

$$A = \pi r^2$$

Setting the radius of the circle to 3 type:

$$r = 3$$

The value of π in AutoLISP is already assigned with syntax, **pi**. Remember we can type !**pi** to check any variable, so try check the value of **pi**. Now to find the area of the radius 3 circle, enter the code shown.

$$a = pi * r ^ 2$$

Type the following code in the program to set the variables for the exponent function.

'Set variables

```
number1 = InputBox ("Type the number")
number2 = InputBox ("Type the number of the exponent")
```

Now, we calculate the first number to the power of the second number.

Type the following code:

'Exponents

answer = number1 ^ number2

The variable **answer** will equal the variable **number1** to the variable **number2** power.

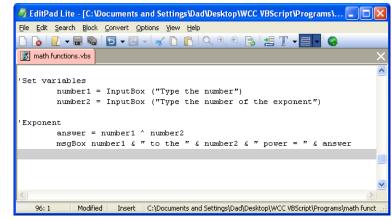


Figure 3.15 – Using the Exponent (^) Function

Type the following code for the message box.

msgBox number1 & " to the " & number2 & " power = " & answer

The following is an extract from the VBScript Quick Reference for the Exponent (^) function.

Function	Name	Description
^	Exponent	The exponent function will raise the first number to the power of the second number
Examples		
Using integers	answer = 4 ^ 3	Answers 64
Using decimals	answer = 5.5 ^ 0.2	Answers 1.40628
Using negatives	answer = -2.0 ^ 4	Answers 16.0

Computing a Square Root in VBScripts

The square root function will return a decimal number representing the side of a square area. This function only expects a single argument after the function **sqr**. Remember no negative numbers are permissible with the square root function. A complex expression that would be common place would be finding the hypotenuse of a triangle using the variables **a** and **b** that are already defined:

$$c = sqr((a ^ 2) + (b ^ 2))$$

Now, we calculate the square root of a number.

Type the following code:

'Square root answer = squ (number1)

The variable **answer** will equal the square root of variable **number1**.

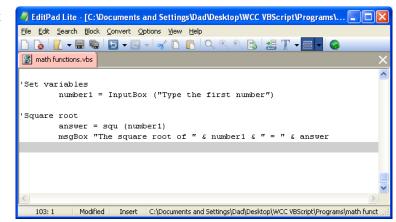


Figure 3.16 – Using the Square Root (squ) Function

Type the following code for the message box.

msgBox "The square root of " & number1 & " = " & answer

The following is an extract from the VBScript Quick Reference for the Square Root function.

Function	Name	Description
sqr	Square Root	The square root function will find the root of the square represented a number
Examples		
Using integers	answer = squ (4)	Answers 2.0
Using decimals	answer = squ (5.5)	Answers 2.34521
Using negatives	answer = squ (-155)	Negatives Not Allowed

The sine, cosine and tangent functions use a angular measurement of radians. We are moist familiar with radians when we hear the word pi. Pi radians or 3.14159 is equal to 180 degrees. To convert degrees to radians, divide the angle in degrees by 57.29578.

Computing Sine in a VBScript

The sin function is a key tool in find the length of the side opposite the angle and in the case of the standard CAD drawing system using the Cartesian coordinate system, the sine of the angle times the hypotenuse of the triangle or length of the line will represent the delta-y of the line being examined.

Type the following code in the program to set the variables for the sine function.

'Set variables number1 = InputBox ("What is the angle?")

Now, we calculate the sine of a angle in degrees.

Type the following code:

'Sine answer = sin (number1 / 57.29578)

The variable **answer** will equal the sine of the angle in degrees represented by the variable **number1**.

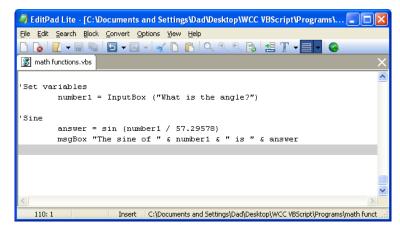


Figure 3.17 – Using the Sine Function

Type the following code for the message box.

msgBox "The sine of " & number1 & " is " & answer

The following is an extract from the VBScript Quick Reference for the Sine function.

Function	Name	Description
sin	Sine	The sine function will return the length of the side opposite the angle of a right sided triangle when the hypotenuse is 1
Examples		
Using integers	answer = sin (2)	Answer 0.909297
Using decimals	answer = sin (0.5)	Answer 0.479426
Using negatives	answer = sin (-0.2)	Answer -0.198669

Computing Cosine in a VBScript

The **cos** function is likewise a tool in find the length of the side adjacent to the angle and in the case of the standard CAD drawing system using the Cartesian coordinate system, the cosine of the angle times the hypotenuse of the triangle or length of the line will represent the delta-x of the line being examined.

$$delta_x = cos (angle / 57.29578)$$

Type the following code in the program to set the variables for the cosine function.

'Set variables

number1 = InputBox ("What is the angle?")

Now, we calculate the cosine of a angle in degrees.

Type the following code:

'Cosine answer = cos (number1 / 57.29578)

The variable **answer** will equal the sine of the angle in degrees represented by the variable **number1**.

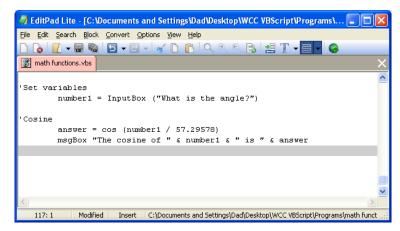


Figure 3.18 – Using the Cosine Function

Type the following code for the message box.

msgBox "The cosine of " & number1 & " is " & answer

The following is an extract from the VBScript Quick Reference for the Cosine function.

Function	Name	Description
cos	Cosine	The cosine function will return the length of the side adjacent to the angle of a right sided triangle when the hypotenuse is 1
Examples		
Using integers	answer = cos (2)	Answer -0.416147
Using decimals	answer = cos (0.5)	Answer 0.877583
Using negatives	answer = cos (-0.2)	Answer 0.980067

Ending the Program

To end this program, we will type a comment saying so. In the future, this will be more elaborate, but for now we will just get used to announcing the natural divisions of the script.

Type the following code:

'End of program

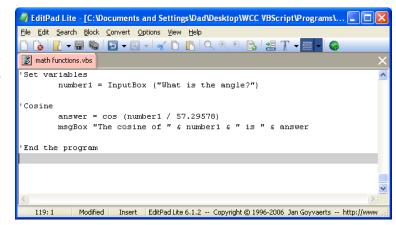


Figure 3.19 – Ending the Program

Programs creating and placing text in a message box are very easy to write once we have achieved writing the first program with these new functions. There are addition exercises for simple routines in the appendixes of this manual. Written below is the entire hello.vbs code for your benefit.

```
'math functions.vbs copyright (c) 2006 by charles w. robbins
```

MsgBox "Math functions.vbs copyright (c) 2006 by charles w. robbins. This program will give the

```
programmer exposure to math functions used in vbscripts"
'Declare variable
    dim number1
    dim number2
    dim answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
'Add numbers
    answer = number1 + number2
    msgBox number1 & " + " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
'Subtract numbers
    answer = number1 - number2
    msgBox number1 & " - " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
'Multiply numbers
    answer = number1 * number2
    msgBox number1 & " x " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
```

^{&#}x27;this program will give the programmer exposure to math functions used in vbscripts

^{&#}x27; alert the user with a message box

```
'Divide numbers with the answer being a real number
    answer = number1 / number2
    msgBox number1 & " / " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
'Divide numbers with result being an integer
    answer = number1 \ number2
    msgBox "The integer value of " & number1 & " / " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
    number2 = InputBox ("Type the second number")
'Find the remainder of a division problem
    answer = number1 mod number2
    msgBox "The remainder of " & number1 & " / " & number2 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
'Find the absolute value of a number
    answer = abs(number1)
    msgBox "The absolute value of " & number1 & " = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
'Find the fix value of a number
    answer = fix(number1)
    msgBox "The fix value of " & number1 & " = " & answer
'Set variables
    number1 = InputBox ("Type the number")
    number2 = InputBox ("Type the number of decimal places")
'Set the decimal places for the number
    answer = round(number1,number2)
    msgBox "Rounding " & number1 & " to " & number2 & " places = " & answer
'Set variables
    number1 = InputBox ("Type the number")
    number2 = InputBox ("Type the number of the exponent")
```

```
'Exponent
    answer = number1 ^ number2
    msgBox number1 & " to the " & number2 & " power = " & answer
'Set variables
    number1 = InputBox ("Type the first number")
'Square root
    answer = squ (number1)
    msgBox "The square root of " & number1 & " = " & answer
'Set variables
    number1 = InputBox ("What is the angle?")
'Sine
    answer = sin (number1 / 57.29578)
    msgBox "The sine of " & number1 & " is " & answer
'Set variables
    number1 = InputBox ("What is the angle?")
'Cosine
    answer = cos (number1 / 57.29578)
    msgBox "The cosine of " & number1 & " is " & answer
'End the program
```

Saving the Program

Now that the program is finished, we need to double check our typing with the text in this manual and then save our program to our folder named "VBScripts".

Make sure the Save in list box is displaying the VBScripts folder and the File name is "math functions.vbs" as shown in Figure 3.20.

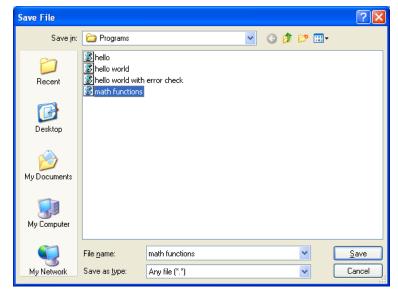


Figure 3.20 – Saving the Program

Running the Program

After noting that the program is saved, press the Start button and pick Run on the Start menu. The Run window will appear on the desktop as shown in Figure 3.21. Select the Browse command button, to open the Browse window.

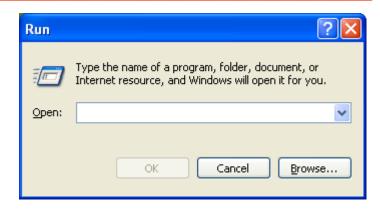


Figure 3.21 – Running the Program

In the Browse window, we need to open the VBScripts folder and select the hello script. If we cannot see the file listed, change the Files of type to "All Files" as shown in Figure 3.22.

Select the Open command button to return to the Run window as shown in Figure 3.23. Press

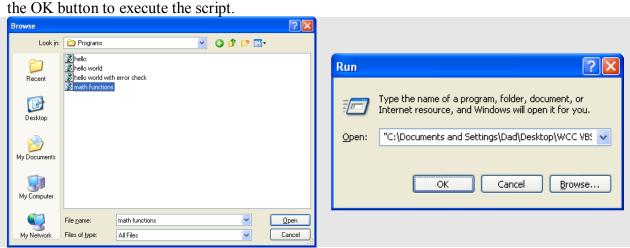


Figure 3.22 – Browse for the Program

Figure 3.23 – Executing the Program

The first message box to appear will be the regular text message with the program name and the description of the program as shown in Figure 3.24. Press the OK command button to close the message box.



Figure 3.24 – The First Message Box in the Script

The second message will appear with the current date and time read from the personal computer as shown in Figure 3.25. Press the OK command button and the script is done.

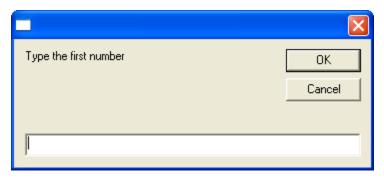


Figure 3.25 – The Second Message in the Script

There are many variations of this script we can practice and obtain information from a personal computer. While we are practicing the message boxes and strings, we learn how to use variables, strings and comments. These are skills that we want to commit to memory.

* World Class CAD Challenge 9-5 * - Write a Script that displays a message box showing script name, copyright date and author. The other messages will display information from the computer after computing the 13 different math functions shown in this chapter,

Continue this drill four times using some other messages, each time completing the VBScript in less than 30 minutes to maintain your World Class ranking.