

Lesson 3: Variables and Arithmetic

OBJECTIVES: In this lesson you will learn

- To use the arithmetic operators +, -, *, / to solve problems
- To use the assignment operator(=) to give a numeric value to a variable
- How operator precedence controls the order in which an expression is calculated
- To use the alert method to display information
- How to use the Math object in calculations

Preparing to Program

All programming languages have the ability to carry out arithmetic operations. The name computer has its root in the machine's original purpose, which was to compute values difficult or impossible to calculate by hand.

In JavaScript, arithmetic operations are carried out with the arithmetic symbols you use in math class. These symbols are called **arithmetic operators**. There are other types of operators in JavaScript that you will learn about in subsequent lessons.

JavaScript provides the following commonly used binary arithmetic operators. Binary means there are two values used, with the operator in between them.

The addition operator +

The binary + is the addition operator: $A + B$ yields the sum of A plus B.

The subtraction operator –

The binary - is the subtraction operator: $A - B$ yields the difference A minus B.

The multiplication operator *

The binary * is the multiplication operator: $A * B$ yields the product A multiplied by B.

The division operator /

The binary / is the division operator: A / B yields the dividend of A divided by B.

Simple Expressions

As you learned in the Introduction, an expression is the computer science term for a formula that returns a value. In JavaScript and other languages, an expression can be a combination of variables, **literals**, and arithmetic operators. A literal is a number itself, like 4, 92.7, etc.

Given the following variable definitions:

```
var num1 = 6;  
var num2 = 3;
```

Here are some expressions and their values:

```
num1 * 2 yields 12  
num1 + num2 yields 9  
num1/num2 yields 2
```

Expressions can be used in combination with the assignment operator to give a value to a variable.

Consider the following:

```
var length = 5;  
var width = 6;  
var area;
```

The formula for calculating the area of a rectangle is:

area is equal to length times width

If we were to write this as JavaScript, it would look like this:

```
area = length * width;
```

So if length is 5 and width is 6, then area has a value of 30. Remember that assignment always goes from right to left. So first length is multiplied times width, and then the answer is assigned to the variable area.

Expressions With More Than One Operator

Expressions can also have more than one operator. For example, the formula for calculating the perimeter of a rectangle is length plus the width times 2. So in JavaScript, the code to calculate the perimeter would look like this:

```
var length = 5;  
var width = 6;  
var perimeter;  
perimeter = length + width * 2;
```

Although this looks right, it will not produce the right answer. The correct answer is length + width (11) times 2, which yields 22. The expression above yields length (5) plus width times 2(12), or 17. Why does it end up with the wrong answer?

It has to do with a rule of mathematics called **operator precedence**. When an expression has more than one operator, the computer has to decide which operation goes first. The computer

can only carry out one operation at a time. Even though the time it takes to carry out that operation is blindingly fast, it still does one thing at a time. The computer must select the correct order in which the operations are carried out. This order is called operator precedence.

It determines which operations have precedence, or go first. It is the same order defined in your math classes when you took algebra. Multiplication and division take precedence over addition and subtraction. If more than one operator is on the same level, for example if you have addition and subtraction in the same expression, then those operations are carried out from left to right through the expression.

How can we fix our problem with calculating the perimeter? We want to override the standard order and force the addition to take place first. To do this we use parentheses. Just like in algebra, the operations inside parentheses have higher precedence and therefore are carried out first.

Here is a corrected formula for perimeter:

```
perimeter = (length + width) * 2;
```

This will produce the correct answer, because length will be added to width before it is multiplied by two. Notice you have to explicitly include the ‘*’ symbol for multiplication, otherwise you will have an error.

Operator Precedence Table

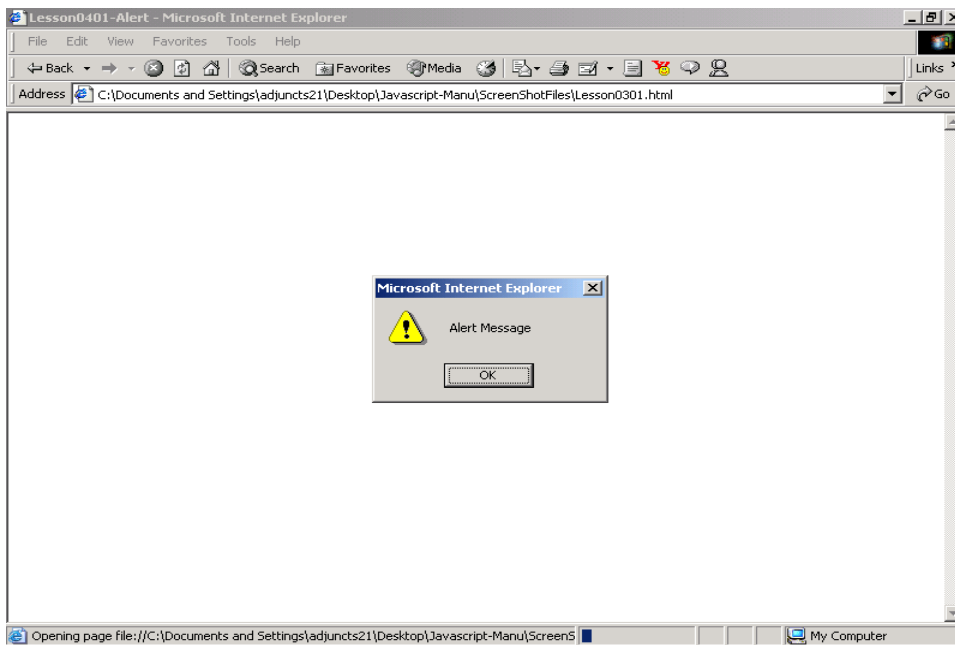
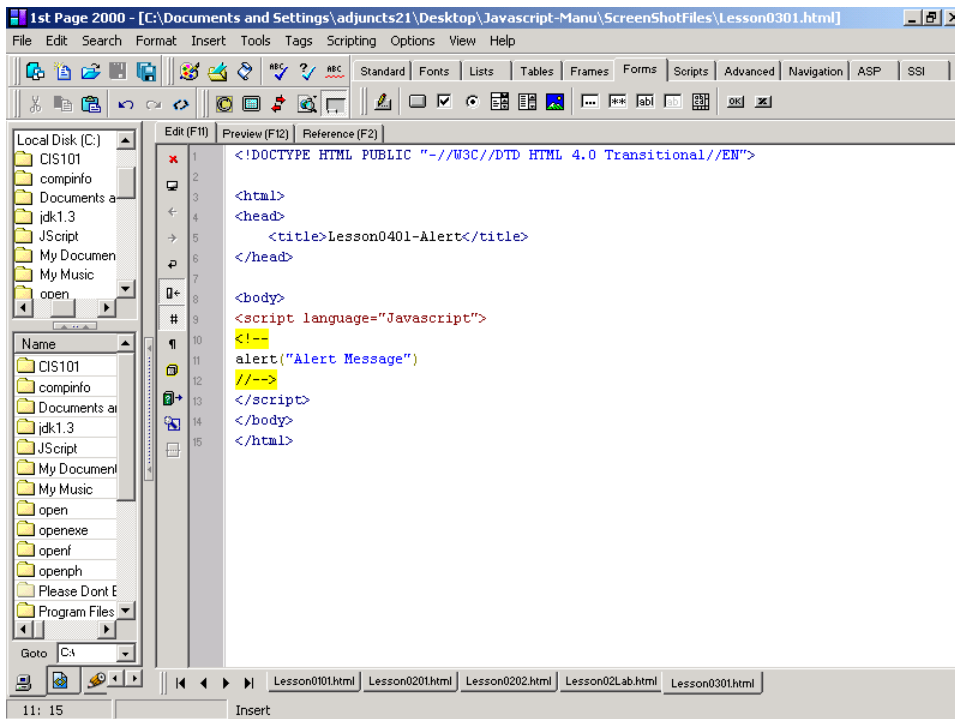
This table summarizes the operator precedence information you will need in order to perform basic arithmetic in JavaScript. The operations are carried out in order from top to bottom:

<i>Type of Operator</i>	<i>Example of Operators</i>
Parentheses (Overrides others)	()
Multiplication, Division	*, /
Addition, Subtraction	+, -
Assignment	=

The alert method

In Lesson 1 we learned how to display output using document.write. There are other ways in JavaScript to display output. The technique you will use in this lesson is the **alert method**.

The alert method displays a small pop up window, sometimes called a message box. It looks like this:



Syntax:

```
alert("message")
```

The alert box will display the specified message. When the user clicks the OK button, the alert box is removed.

In the Lab

This week in lab you will use JavaScript to solve problems using variables, arithmetic operators, assignment, and the alert statement.

Start 1st Page 2000 and begin a new HTML document. Save it giving it the name lesson0301.html. Following the instructions outlined in Appendix B, place your cursor between the <body> ... </body> tags, and insert the script tags and hiding comments by using the Scripting menu.

Now type in *exactly* the following code:

```
<html>

<head>
  <title>Lesson 3: Variables and Arithmetic</title>
</head>

<body>

<script language="Javascript">

<!--

var length=10;
var width = 5;
var area = length * width;

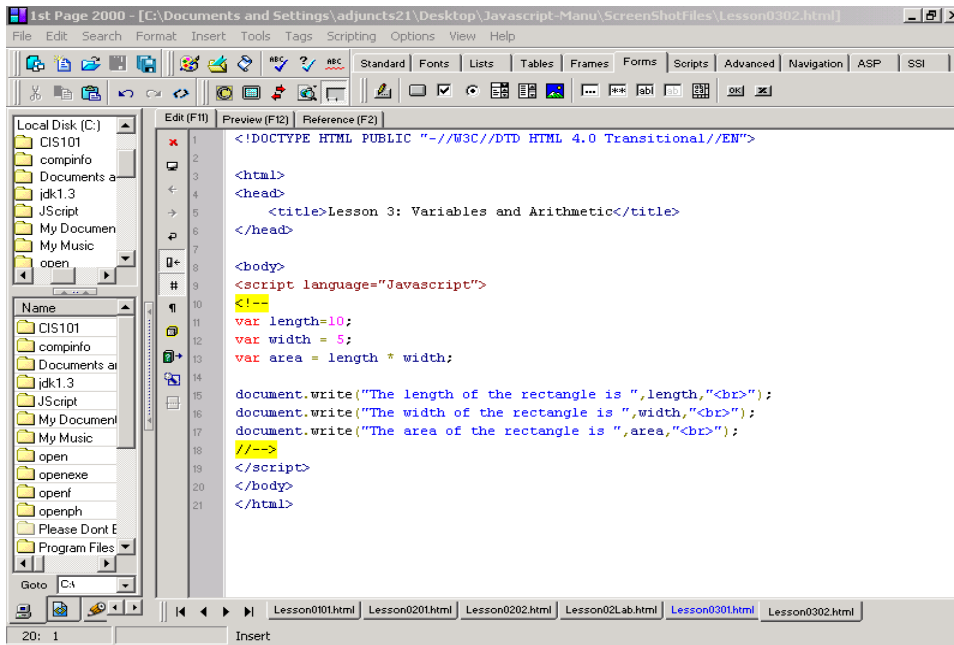
document.write("The length of the rectangle is ",length,"<br>");
document.write("The width of the rectangle is ",width,"<br>");
document.write("The area of the rectangle is ",area,"<br>");

//-->

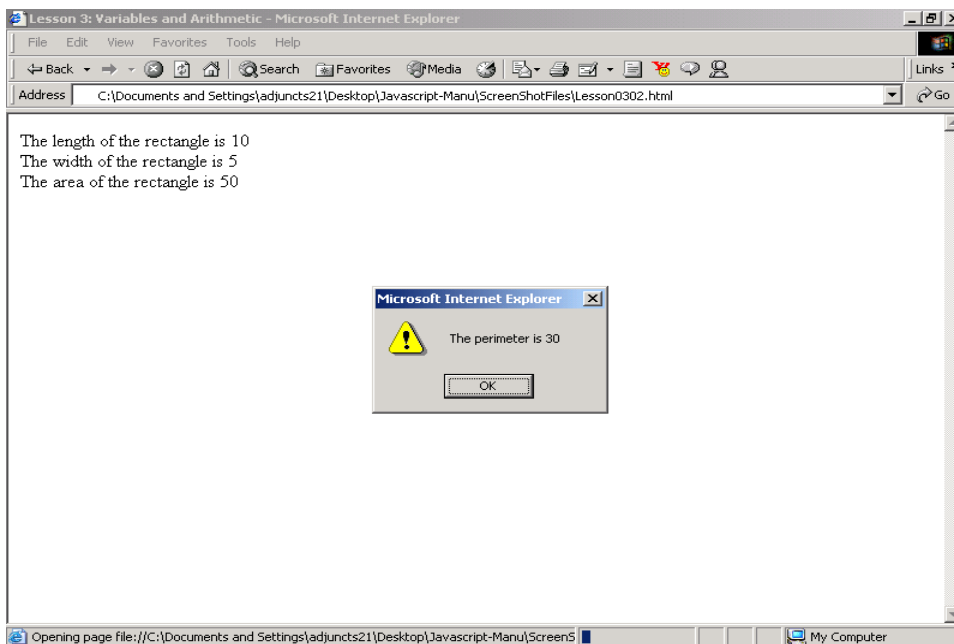
</script>

</body>

</html>
```



After accurately entering the code above run it using the preview button. You should see the following output:



After you have run the program and obtained the correct output, try the program again with different numbers. Change the values for length and width to other numbers, including numbers with a fractional value, like 2.7 or 5.6, and run the program again.

Now add the following code, which will calculate the perimeter and display its value using the alert method. Insert this code after line 16 (the third document.write statement):

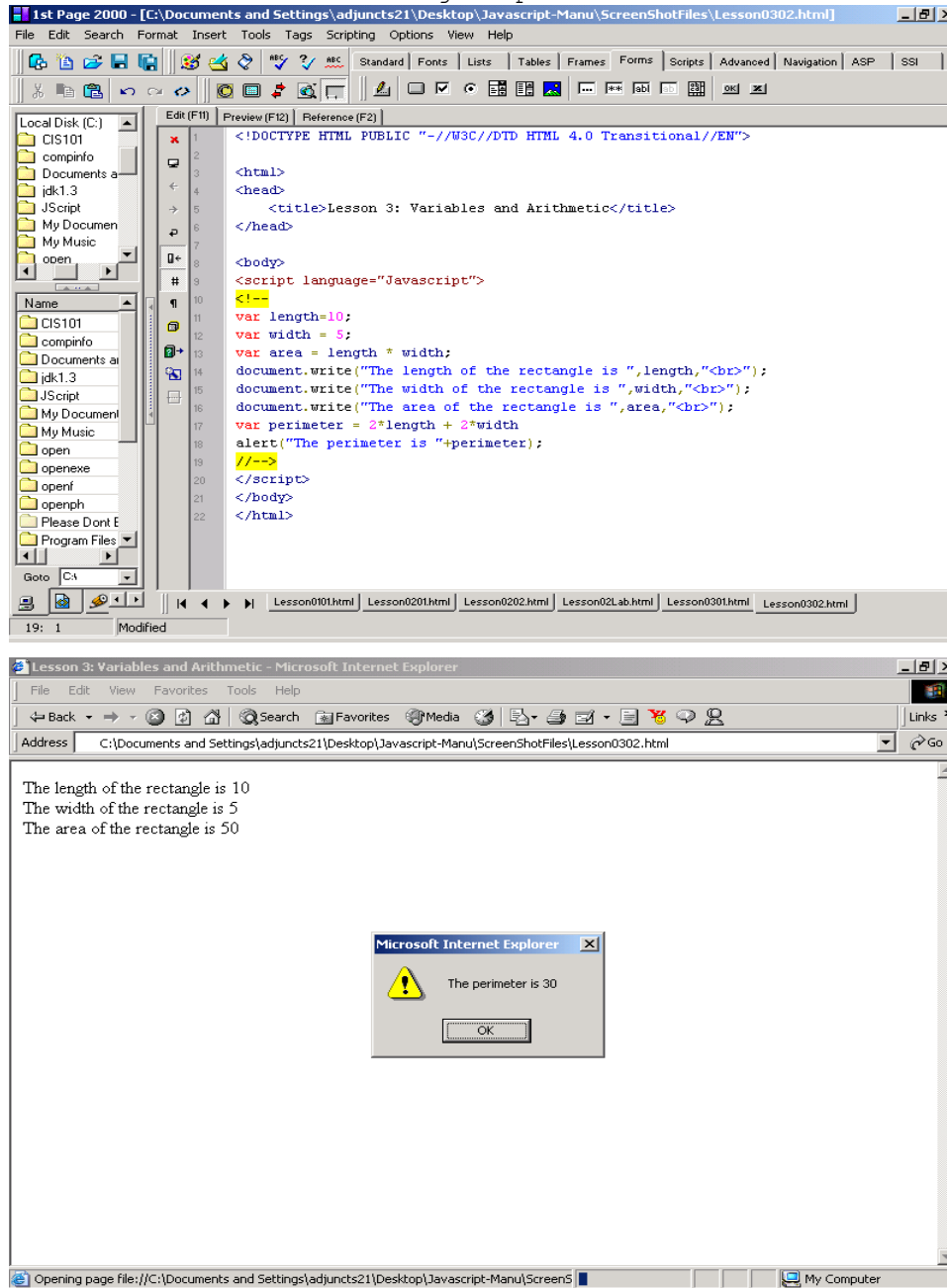
```

var perimeter = 2*length + 2*width

alert("The perimeter is "+perimeter);

```

You should see the following output:



Student Modifications

- Change the first document.write statement which displays the length to an alert statement, then run the code. Notice that this alters the appearance of the text. The subsequent document.write statements do not appear until the user clicks OK.
- Add the variables base and height, and assign them values. Add a variable triangleArea and code to calculate and display the area of a triangle. The formula for the area of a triangle is:

```
triangleArea = 1/2*base*height;
```

- Add a variable radius and assign it a value. Add variables circleArea, circleCircumference, and code to calculate and display the area and the circumference of a circle. The formula for the area of a circle is:

```
circleArea = radius*radius*Math.PI;
```

The formula for the circumference of a circle is:

```
circleCircumference = 2*radius*Math.PI;
```

Math.PI is an example of a **defined constant** that is part of the **Math object**. Math.PI is defined with a value of 3.141592653589793, or the approximate value of π . Math is a built-in object which has a large number of properties and methods to handle mathematical computations.

Key Terms and Definitions

- **arithmetic operators** – The symbols + (plus), - (minus), * (multiplication), and / (division) that are used in JavaScript to carry out arithmetic.
- **literals** – numbers like 6, 9, 2.33, that can be used in an expression or to assign a value to a numeric variable.
- **operator precedence** – the order that the browser follows when evaluating expressions that contain more than one operator. The basic order is parentheses, multiplication or division, then addition or subtraction.
- **alert method** - JavaScript method that displays a pop up window with a message and an OK button. The alert window remains visible until the user clicks OK.
- **Math object** – a built in JavaScript object with a large number of properties and methods useful for carrying out mathematical calculations.
- **defined constant** – a number, like the value of π , that has been defined and assigned a value. Programmers can use defined constants in expressions to solve problems.
- **Math.PI** - a defined constant, part of the Math object, that holds an approximate value for π .

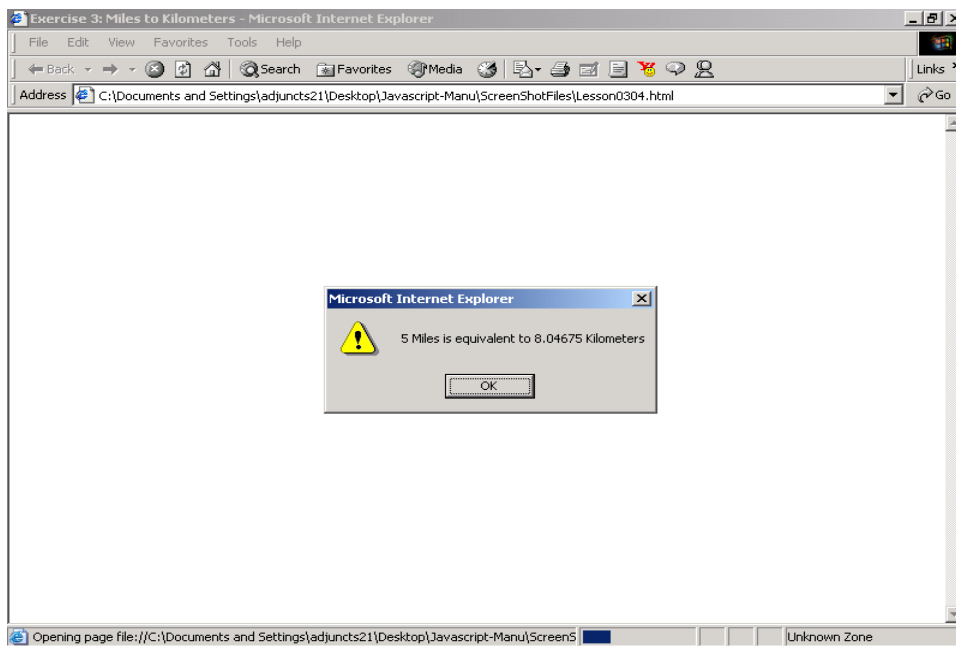
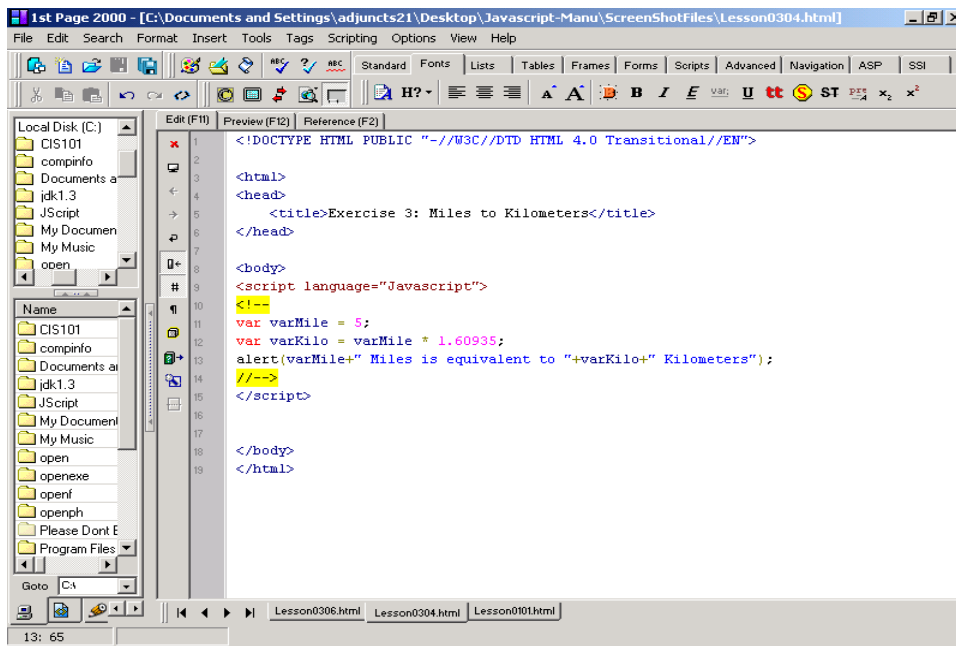
Lesson 3 Summary

In Lesson 3 you learned about arithmetic operators and how expressions are evaluated by the browser using operator precedence. You learned how to use JavaScript to solve problems, like calculating and displaying the area of common shapes like a rectangle and triangle. You used the alert method as an alternate way to display output. Finally you used the Math object and defined constant Math.PI to calculate and display the area and circumference of a circle.

Lesson 3 Exercises

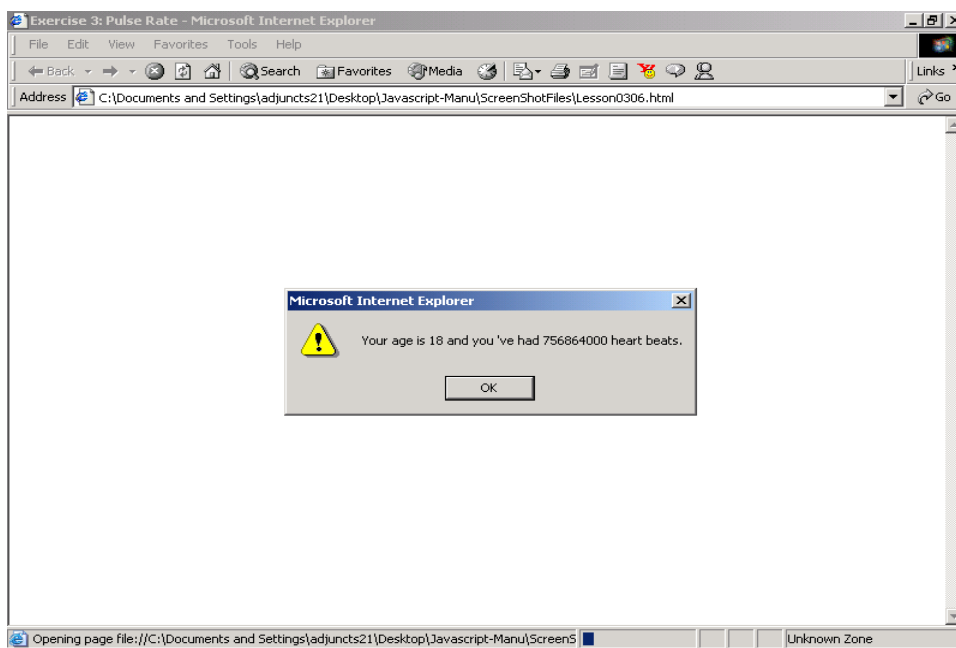
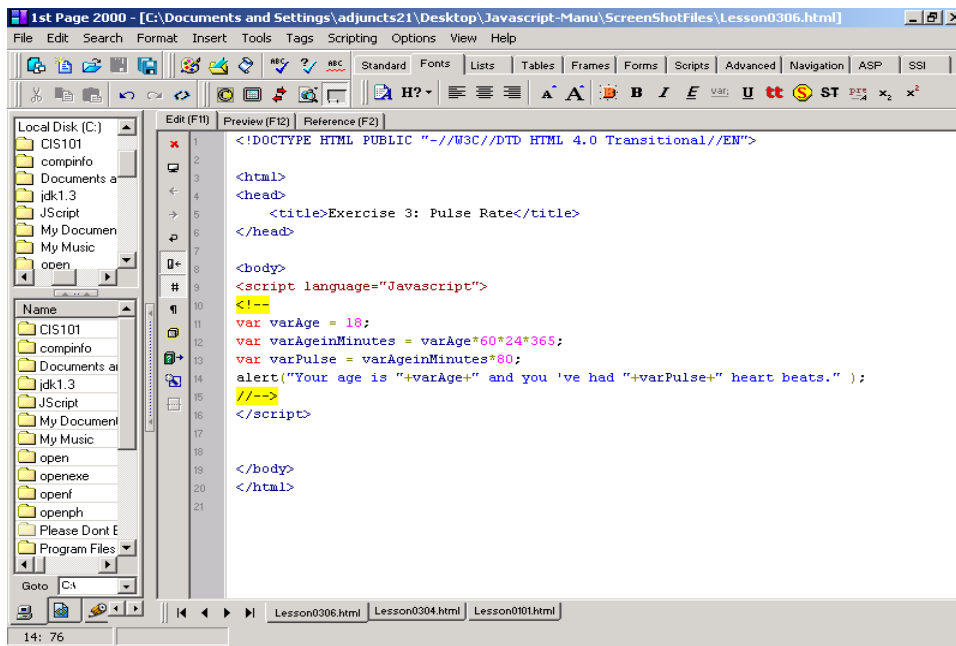
3_1. Write a JavaScript program that converts a distance in miles into a distance in kilometers. One mile is equal to 1.60935 kilometers. If you multiple the number of miles times 1.60935, you will calculate the number of kilometers. Declare miles as a variable and give it a value. Declare kilometers as a variable. Use the formula above for converting miles to kilometers to assign a value to kilometers. Display the results in an alert box. Include the original number of miles and the calculated distance in kilometers in the message displayed in the alert box.

For example, if the number of miles is 5, then your program should produce the following output:



3_2. Your pulse rate is the number of times your heart beats per minute (usually about 70 per minute for an adult and higher for children.) Write a JavaScript program with a variable to represent your age in years. Determine the number of minutes in a year: $60 \times 24 \times 365$, and multiply this by your age in years. Then determine the approximate number of times your heart has beaten by multiplying this result by 80, an average rate that takes into account the faster rate for children. Display the results using an alert box.

For example, if your age is 18, this is what your program should produce:



3_3. Write a JavaScript program to convert Celsius temperature values to Fahrenheit. The formula for the conversion is:

$$\text{Fahrenheit} = 9 * \text{Celsius} / 5 + 32$$

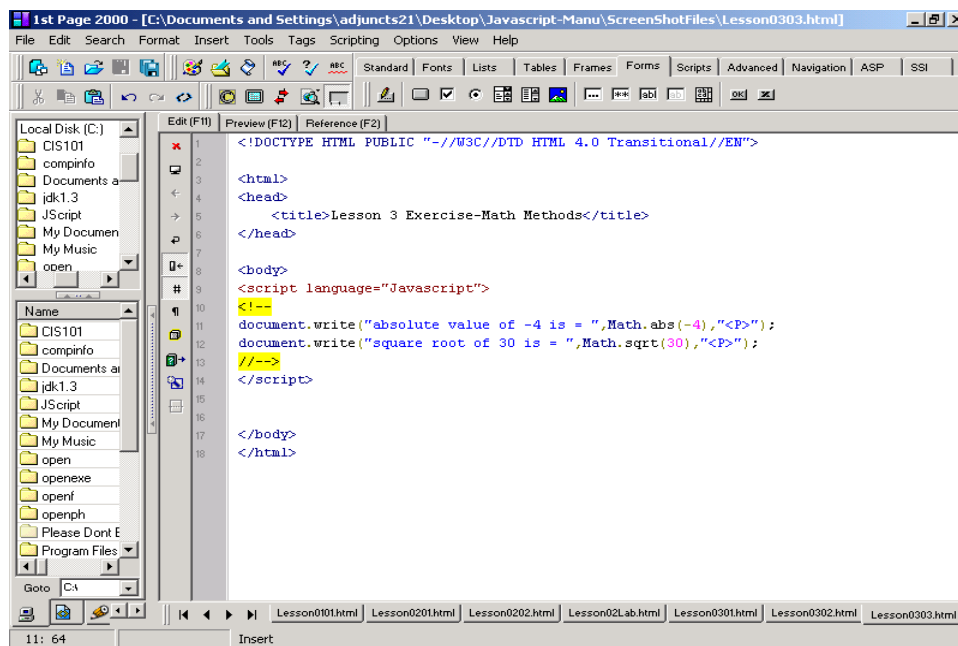
Use different variables for Celsius and Fahrenheit, and display the results in an alert box.

3_4. Here is a list of some of the methods available in the Math object:

Method syntax	Arguments	Returns
Math.abs(num)	number	absolute value of num
Math.ceil(num)	number	The least integer greater than or equal to num
Math.cos(num)	number (angle in radians)	cos(num)
Math.floor(num)	number	The greatest integer less than or equal to num
Math.log(num)	number >0	ln(num)
Math.max(num1,num2)	both are numbers	The greater of num1 and num2
Math.min(num1,num2)	both are numbers	The smaller of num1 and num2
Math.pow(num1,num2)	both are numbers	num ^{num2}
Math.round(num)	number	Rounded off integer
Math.sin(num)	number	sin(num)
Math.sqrt(num)	number ≥0	√num
Math.tan(num)	number	tan(num)

Use document.write statements to display the results of some of these Math methods. For example, the code

```
document.write("absolute value of -4 is = ",Math.abs(-4),"<P>")
document.write("square root of 30 is = ",Math.sqrt(30),"<P>")
```



displays the following results:

