You are going to run frequently into JavaScript code that doesn't work. In fact, MOST code fails to operate correctly the first time you test it, for one reason or another.  
  
Debugging JavaScript syntax errors is much harder, in general, than debugging HTML errors. JavaScript is VERY strict with regards to syntax.  
  
There are two types of syntax errors, viz., load-time errors and run-time errors.  
  
**Load-time errors**:  
  
An error that violates the grammatical rules of the language when the script is loaded. These errors are the major mistakes that prevent the script from functioning before it has a chance to start. It is during the loading process that JavaScript spots any serious errors that will cause your script to fail right off the bat. The script cannot be run until the page has been successfully loaded.   
  
The error may be that the code has an extra parenthesis, or is missing a quote, brace, semi-colon, etc. Or you may declare a variable that is a JavaScript reserve word, such as class. Or you may enter parseint, instead of parseInt. JavaScript is case-sensitive.  
  
  
**Run-time errors:**  
  
A run-time error occurs when the JavaScript interpreter encounters a problem during the execution of a program. This usually happens when the interpreter reads code it can't handle.   
  
The most common type of run-time error occurs when you try to access an object or variable that doesn't exist.  
  
For example:  
  
<script type="text/javascript">  
document.write(greeting);  
</script>  
  
In IE, you would get:  
  
Line: 8  
Char: 1  
Error: 'greeting' is undefined  
  
You get a similar error message in Firefox.  
  
If you then changed the script to:  
  
<script type="text/javascript">  
var greeting = "Hi there";  
document.write(greeeting);  
</script>  
  
You would get an error message that 'greeeting' is undefined.   
  
Another type of run-time error occurs when you misapply one of JavaScript's objects. For example, document="test"; results in an error ("document cannot be set by assignment.") because you cannot assign a value directly to the document object.  
  
 **Firefox:**  
  
I have found that debugging is fairly easy in Firefox. Click Tools/Error Console/Errors to find JavaScript errors in Firefox. The line number and approximate position of the error is displayed. Note that you can also check for CSS errors by clicking Tools/Error Console/Warnings.  
  
Suppose you have a script containing the following:  
  
var x = window.prompt("enter a value" "");  
  
By clicking Tools/JavaScript Console/Errors, you will see something similar to:  
  
Error: Missing ) after argument list  
Line: 11  
var x = window.prompt("enter a value" "");  
  
Most Firefox versions also show an arrow pointing to approximately where the error occurred. In this case, the arrow will point to the first " after value.  
  
You may wish to also download Firefox's add-on debugger called [Firebug](https://addons.mozilla.org/en-US/firefox/addon/1843).  
  
**Internet Explorer:**  
  
If you are testing your code in older versions of Internet Explorer, and the status bar in the lower left corner of the browser shows an error icon or says 'error on page' or 'done but with errors', the gray error box may appear or you may need to click on the error icon at the lower left of the page to see the JavaScript error and what line contains the error. For testing purposes, I suggest you have "Disable script debugging" as UNCHECKED, under Tools/Internet Options/Advanced on your browser.  
Suppose you have a script containing the following:  
  
var x = window.prompt("enter a value" "");  
  
When it is run you will see the yellow error icon indicating a syntax error. After clicking it, the error message box will say something like:  
  
Line: 11  
Char: 39  
Error: Expected ')'  
Code: 0  
  
This is not all that enlightening. Besides the trouble of finding line 11 and character 39, the actual error may be on a different line. After inspecting the code, you should eventually see that the error is that "enter a value" should be followed by a comma.  
  
Starting with IE8, there is a JavaScript debugger as part of its development tools. To use it, you must ensure that it's enabled. Again you need to have "Disable script debugging" as UNCHECKED, under Tools/Internet Options/Advanced. The script debugger is accessed by clicking Tools/Developer Tools, which opens the debugger window. Now click the Script tab at the top left and the Start Debugging button.  
  
**Chrome**:  
  
In Chrome, click the "Control this page" button to the right of the address bar and select Tools/Javascript console. Then click on the error link at the right of the page to see which line has the error along with the corresponding error message.  
 **Safari:**  
  
In Safari, you must enable the Develop menu. To do so, choose Edit/Preferences and then click the Advanced tab. There is a check box entitled "Show develop menu in menubar" that should be checked. Once the setting is enabled, a menu named "Develop" appears in the Safari menu bar. The Develop menu provides several options for debugging and otherwise working with the page that is currently loaded. You can click Show Error Console to display a list of JavaScript and other errors. The console displays the error message, the URL of the error, and the line number for the error.  
  
  
There is a freeware editor called Crimson Editor for Windows. It has line numbering that can be turned on, and line numbers can also be included when printing hard copy. It's a pretty nice editor with syntax coloring for various language files such as html and css.  
  
It's available for download at <http://www.crimsoneditor.com/>  
A free and good editor for both HTML and for locating line numbers is [HTML-Kit](http://www.chami.com/html-kit/download/). It has plugins for JavaScript coding.  
  
You can also locate line numbers in [Notepad++](http://notepad-plus-plus.org/release/5.8.7).  
  
Here is a good site for [Debugging JavaScript Error](http://javascript.internet.com/debug-guide.html)s.  
  
Here is a list of some of the most common errors that students make:  
  
Spelling: JavaScript is CASE-SENSITIVE. Check your spellings!  
  
Incorrectly-placed Carriage Returns: Students often place carriage returns in the middle of parentheses for methods or functions, which usually doesn't work.  
  
Spaces: Names may have NO SPACES in them. Some arguments may also not have spaces in them, depending on the method in question. Don't forget to follow the naming conventions that I outlined in the earlier modules!  
  
Missing Quote Marks: One quote mark missing, and your JavaScript program will fail. The same holds true for missing parentheses or curly-braces, etc. I get caught by this myself surprisingly often.  
  
Here is a [free JavaScript debugger](http://webdeveloper.earthweb.com/webjs/jsutility/item.php/481771)  
  
Here is a site for checking for javascript syntax errors online. I tried it and it looks useful.  
  
<http://jshint.com/>  
  
Note - when using this site, I suggest entering the entire HTML code that encompasses the javascript. But leave out the 1st statement, i.e., the one below.  
  
<?xml version="1.0" encoding="UTF-8"?>