Solutions to Quick Checks

Table of Contents

[Quick Check Answers 2](#_Toc52176705)

[Quick Check 1 2](#_Toc52176706)

[Quick Check 2 3](#_Toc52176707)

[Quick Check 3 3](#_Toc52176708)

[Quick Check 4 4](#_Toc52176709)

# Quick Check Answers

Quick Check 1

1. Describe the three types of program errors.

* Load-time or syntax errors in which the interpreter finds error in the syntax of the code as it initially loads it.
* Runtime errors in which the interpret has loaded the code but is unable to run it due to such factors as an unreferenced variable or a mistyped function name.
* Logic errors in which the program can be interpreted and run but has an incorrect result due to a flaw in the program’s logic.

**Feedback**: Load-time or syntax errors, runtime errors, and logic errors are the three basic types of programming errors that are caused by (a) the inability of the interpreter to read the code, (b) the inability of the interpreter to run the code, and (c) the inability of the code to arrive at a correct outcome.

1. What is the error in the following code and what type of error is it?

document.writ("Hello World");

The error is a syntax error in which document.writ should have been document.write.

**Feedback**: Syntax errors usually occur when a JavaScript keyword such as write has been misspelled or an important part of the command has been omitted, such as forgetting a closing brace in a function or command block.

1. What is the error in the following code and what type of error is it?

let firstValue = 10;  
let secondValue = 20;  
let result = firstvalue + secondValue;

This is a runtime error in which the error occurs because the firstValue variable is written as firstvalue in the final statement of the code.

**Feedback**: Runtime errors often occur when a variable or function named is misnamed or doesn’t use the correct case.

1. If the browser console reports a single syntax error, does that mean there is only one syntax error in the code?

No. The debugger will stop at the first instance of a syntax errors. There may be others in the code that have not been reached yet.

**Feedback**: The debugger can handle only one syntax error at a time because the error causes the interpreter to stop loading the code.

Quick Check 2

1. The orderCost variable in a long and elaborate program might be incorrectly calculated. Provide code to display the value of the variable within an alert box.

window.alert("orderCost = " + orderCost);

**Feedback:** Use the window.alert() method to write variable names to an alert window displayed in the browser.

1. Provide code to write the value of orderCost to the console log.

console.log("orderCost = " + orderCost);

**Feedback:** Use the console.log() method to write variable names to an alert window displayed in the browser

1. What are three reasons to use the console log approach over the alert box approach?

* Alert boxes interfere with the normal operation of the code and must be deleted after their use.
* Alert boxes do not perform well when you must trace a long sequence of operations. Imagine displaying an alert box for each iteration in a for loop that goes through one hundred iterations.
* You cannot compare the contents of one alert box with subsequent boxes, because closing the alert box removes it from the browser window.

**Feedback:** The console.log() method is generally preferable to the window.alert() method because it is less obtrusive the operations of the code. However, it does require an understanding of how to open and view the browser debugging console.

1. Why would you comment out sections of a program that is producing errors?

You comment out sections that you know are working so you can limit the sections where an error might be found within the code, making it easier to trace the code to its source.

**Feedback:** Part of the art of debugging is narrowing down the areas of the code which are not working properly. By commenting out sections of the code, you can effectively reduce the amount of code to explore.

Quick Check 3

1. What is a breakpoint?

A breakpoint is a location within the code marking the point where execution is suspended.

Feedback: Breakpoints are used in debugging to suspend the operation of the program, allowing the programmer to study the state of the program and its variables at that point in its execution.

1. Explain the difference between stepping into, stepping over, and stepping out of the program execution.

Stepping in or stepping into executes an individual line of code and then pauses until you instruct the debugger to continue. This feature gives you an opportunity to evaluate program flow and structure as code is being executed.

As you use the Step Into button to move through code, the debuggers stop at each line within every function of the JavaScript program. However, when stepping through a program to trace a logical error, it is convenient to be able to skip functions that you know are functioning correctly. The second option, known as stepping over, allows you to skip function calls. The program still executes each function that you step over, but it appears in each debugger as if a single statement executes.

The final option, stepping out, executes all remaining code in the current function. If the current function was called from another function, all remaining code in the current function executes and the debugger stops at the next statement in the calling function.

Feedback: Stepping into, stepping over, and stepping out of the program are all different ways of moving through a suspended program one statement at a time, allowing the programmer to skip past those sections that are not relevant to the debugging task.

1. What is the call stack? How do you use it to aid in debugging a program?

The call stack is the ordered list maintained by a JavaScript processor containing all the procedures, such as functions, methods, or event handlers, that have been called but have not yet finished processing. Each time a program calls a procedure, the procedure is added to the top of the call stack and then removed after it finishes executing. The ability to view a list showing the contents of the call stack is very useful when tracing logic errors in large programs with multiple functions. Viewing the call stack, along with using tracing commands, makes it easier to locate the specific function causing the problem.

Feedback: The call stack is often used when you have nested functions and need to trace of your location as the program executes that hierarchy of functions.

Quick Check 4

1. Under what circumstances will the catch command block be run by the browser?

The catch command will be run when an exception or error is found within the code.

Feedback: A catch statement is used when the program needs to override the default action of the browser in response to an error within a section of code.

1. Under what circumstances will the finally command block be run by the browser?

The finally command will always be run, whether the browser encounters an error or not.

Feedback: The finally command block is often used to perform those tasks that are necessary even in the presence of an error.

1. Provide code to generate an exception with the message, “Supply a positive value”, if the value of the age variable is less than zero.

if (age < 0) throw "Supply a positive value";

Feedback: Use the throw operator to throw an exception that can be caught and handled in a catch statement using customized methods.

1. What must be included within an error handling function to replace the browser’s default error reporting methods?

return true;

Feedback: Within an error-handling function, the return statement’s value is either true or false. If return is true, the error handling function replaces the browser’s default error handling; if return is false or omitted, the error handling function supplements the actions of the browser but does not replace them.