

## JavaScript

Candidates for this exam should be able to recognize and write syntactically correct JavaScript code that will logically solve a given problem and use data types supported by JavaScript.

Candidates are expected to have at least 150 hours of instruction or hands-on experience with the JavaScript programming language. Candidates should be familiar with JavaScript features and capabilities, and understand how to write, debug, and maintain well-formed, well-documented JavaScript code.

To be successful on the test, the candidate is also expected to have the following prerequisite knowledge and skills:

- 8th grade reading skills
- Algebra I
- Fundamental knowledge of HTML
- Fundamental knowledge of CSS

### 1. JavaScript Operators, Methods, and Keywords

#### 1.1 Complete and debug code that uses assignment and arithmetic operators

- Assignment, increment, decrement, addition, subtraction, division, multiplication, modulus, compound assignment operators (`+=`, `-=`, `*=`, `/=`, `%=`)

#### 1.2 Apply JavaScript best practices

- Comments, indentation, naming conventions, `noscript`, constants, reserved keywords, debugger keyword, setting breakpoints, `console.log`

#### 1.3 Evaluate the use of internal and external scripts

- When to use, how to use, and what happens when scripts are used at multiple levels

#### 1.4 Implement exception handling

- `try`, `catch`, `finally`

#### 1.5 Complete and debug code that interacts with the Browser Object Model (BOM)

- Displaying dialogs, determining screen size

### 2. Variables, Data Types, and Functions

#### 2.1 Declare and use variables of primitive data types

- Number, Boolean, String, null, undefined, type of operator, type-checking functions, use `strict`, converting between data types (`parseInt`, `parseFloat`), formatting numbers, string operations, `eval()`, `toFixed()`, `toLocaleString()`, `toPrecision()`, single quote vs. double quote (nesting), initialization

#### 2.2 Declare and use arrays

- Single-dimensional arrays; multi-dimensional arrays; iteration; initialization; defining, sorting, and searching an array; `push`, `pop`, `shift`, and `unshift` methods; `length` property; accessing an array element



# IT SPECIALIST EXAM OBJECTIVES

## 2.3 Complete and debug code that uses objects

- Properties, methods, instantiation, Date object, retrieving date and time parts, localizing date format (MM/DD vs DD/MM), adding and subtracting dates

## 2.4 Complete and debug code that uses built-in Math functions

- random, round, abs, floor, ceil, min, max, pow, sqrt

## 2.5 Complete and debug functions that accept parameters and return values

- Reusable code, local vs. global scope, redefining variables, passing parameters, value vs. reference, return values

## 3. Decisions and Loops

### 3.1 Evaluate expressions that use logical and comparison operators

- !=, <, >, <=, >=, !, ==, &&, ||

### 3.2 Complete and debug decision statements

- Single alternative (if), dual alternative (if else), multiple alternative (switch), nested if

### 3.3 Complete and debug loops

- for, for in, while, do while, break, continue

## 4. Document Object Model

### 4.1 Identify and construct the Document Object Model (DOM) tree

- window, document, body, other HTML elements

### 4.2 Identify and handle document, form, keyboard, and mouse events

- onload, onfocus, onblur, onchange, onkeydown, onkeyup, onkeypress, onclick, onmouseover, onmouseout

### 4.3 Complete and debug code that outputs to an HTML document

- document.write, innerHTML, textContent

### 4.4 Complete and debug code that locates, modifies, and adds HTML elements and attributes to documents

- getElementById, getElementsByTagName, getElementsByClassName, setAttribute, createElement

### 4.5 Create events using event handlers and listeners

- DOM events, HTML attribute event, addEventListener

## 5. HTML Forms

### 5.1 Complete and debug code that retrieves form input and sets form field values

- Retrieving form values; identifying the DOM path; getting values from different types of elements; prepopulating, masking, and updating values

### 5.2 Complete and debug code that performs input validation

- Case, string comparisons, Not-A-Number (NaN), not blank

### 5.3 Describe the form submission process

- onsubmit, POST vs. GET, potential targets for submission

