

# Test Results and Interview Guide

Candidate: **Elizabeth Wantsajob**  
Assessment: Javascript Programming (Core)  
Completed: June 27, 2026  
Prepared for: Sara Maple  
Example Company

## What's Included

- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

**Important Note:** The Javascript Programming (Core) assessment measures key factors related to high performance and tenure in this job. Attribute types measured vary by test, but can include cognitive ability, skills, knowledge, personality characteristics, emotional intelligence, and past behavioral history. This report includes a one page summary, followed by detailed results with an embedded interview guide. Note that these results should always be used as a part of a balanced candidate selection process that includes independent evaluation steps, such as interviews and reference checks.

## Overall

Candidate	Score	Interpretation
<b>Elizabeth Wantsajob</b> beth.wantsajob@gmail.com Javascript Programming (Core) June 27, 2026	<span style="font-size: 24pt; font-weight: bold; color: green;">77</span>	

The candidate demonstrates a solid and broadly competent understanding of JavaScript, including proficiency in scope management, object-oriented principles, asynchronous operations, and error isolation. This level of knowledge is consistent with a mid-level programmer who can independently contribute to JavaScript-based development efforts with moderate supervision. Minor gaps may exist in specialized or advanced areas, but overall the candidate reflects a strong working command of the language.

**Key**

- Candidate Score
- Higher Risk
- Lower Risk

## Competency Summary

Competency	Score	Interpretation
<b>Skills/Knowledge (relates to immediate readiness)</b>		
Asynchronous Programming and Promises	94	
Functions and Control Flow (Coding Tasks)	62	
Variables, Data Types, and Scope (Coding Tasks)	62	
Error Handling and Debugging	67	
Functions and Control Flow	63	
Object-Oriented Programming and Modules	95	
Objects, Arrays, and Collection Structures	94	
Variables, Data Types, and Scope	82	

## Comparison

Percentile scores indicate how the candidate compares to other test-takers within various groups. The candidate scored equal to or better than the fraction of test-takers indicated by the percentile.

Test-Taker Group	Percentile	0	10	20	30	40	50	60	70	80	90	100	
Global	77th												
North America	64th												
United States	64th												
Example Company	71st												

## Artificial Intelligence (AI) Generated Scores

This table includes one or more scores derived from a large language model AI query. AI-derived scores are non-deterministic. That is, they are not precisely repeatable. Therefore, these scores should always be treated as supplementary information and should never be used exclusively or compared to hard cutoff values.

Estimated Value	Score	Confidence	Interpretation
Knowledge, Skills, and Abilities Summary	-	-	<p>Summary Points (AI):</p> <ul style="list-style-type: none"> <li>(Generic Text for Sample Report) Strong performer in Drag and Drop Files tasks, indicating comfort with file management and basic computer interactions.</li> <li>Demonstrates solid numerical accuracy in Recognizing and Confirming Numbers, a valuable asset in detail-oriented roles.</li> <li>Moderate overall performance in Analytical Thinking and Attention to Detail, with adequate grammar skills but room for improvement.</li> <li>Struggles with Reading and Analyzing Problems, which may limit effectiveness in roles requiring critical reading and complex problem-solving.</li> <li>Lowest performance in Navigating Between Screens, suggesting difficulty with multi-screen software workflows that could impact productivity in computer-intensive roles.</li> </ul> <p>Narrative (AI): Elizabeth Wantsajob demonstrates a mixed profile of knowledge, skills, and abilities across the assessed competencies.</p> <p>Elizabeth shows a strong aptitude in Drag and Drop Files, performing well on this technical task and suggesting she is comfortable with this type of computer interaction. This is a notable strength that would translate well into roles requiring file management and basic computer navigation tasks.</p> <p>In the area of Analytical Thinking and Attention to Detail, Elizabeth performs at a moderate level. She demonstrates solid ability in Recognizing and Confirming Numbers, which suggests she is careful and accurate when working with numerical data — a valuable skill in detail-oriented work environments. Her Grammar performance is adequate but leaves room for improvement, indicating she may occasionally make written communication errors. Her weakest area within this competency is Reading and Analyzing Problems, where she struggled to consistently interpret and work through written problem scenarios. This may impact her effectiveness in roles that require critical reading, written comprehension, or complex problem-solving.</p> <p>Elizabeth's most significant area for development is Navigating Between Screens, where she scored considerably lower than the other competencies. This suggests she may have difficulty efficiently moving through software interfaces or multi-screen workflows, which could slow productivity in roles that rely heavily on navigating computer applications or data entry systems.</p> <p>Overall, Elizabeth brings some useful technical strengths, particularly in file management and numerical accuracy, but would benefit from targeted development in software navigation and analytical problem-solving to be fully effective in roles that demand these skills.</p> <p>Computed on: April 2, 2026, 11:09:49PM EDT</p>

## Detail

Candidate: Elizabeth Wantsajob, beth.wantsajob@gmail.com  
 Assessment: Javascript Programming (Core)  
 Authorized: June 27, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com  
 Started: June 27, 2026, 1:06:00PM EDT  
 Completed: June 27, 2026, 1:06:00PM EDT  
 Overall Score: 77

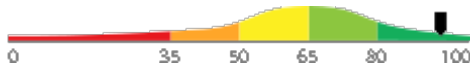
## Knowledge and Skills Detail

This section contains a list of job-related knowledge areas and skills that have been evaluated. Low scores in these areas often indicate that additional learning may be required before top performance can be achieved.

Detail
Interview Guide

### Asynchronous Programming and Promises

Score: 94



*Description:*

Covers how JavaScript handles operations that take time to complete, such as fetching data or reading files, without blocking the rest of the program. Includes understanding the event loop, using Promises to represent future values, and writing asynchronous code using `async/await` syntax and promise chaining to manage sequential or parallel tasks.

*Interpretation:*

Candidate should achieve superior job performance in this area with little or no training.

The candidate demonstrates a comprehensive and advanced mastery of asynchronous JavaScript programming, including a thorough understanding of the event loop, Promises, `async/await`, and both sequential and parallel task management. They are well-equipped to design, implement, and troubleshoot complex asynchronous workflows in professional JavaScript development contexts.

How does `async/await` improve the readability of asynchronous code compared to promise chaining, and how do you handle errors when using `async/await`?



1

Cannot explain `async/await` syntax or does not know how to handle errors in an `async` function.



2

Explains `async/await` syntax and mentions `try/catch` but does not clearly contrast it with promise chaining.



3



4

Clearly contrasts `async/await` with chaining, demonstrates `try/catch` usage, and may mention handling multiple `async` tasks.



5

Can you explain what a Promise is in JavaScript and describe what happens when it resolves or rejects?



1

Cannot define a Promise or does not understand the resolved and rejected states.



2

Defines a Promise correctly but gives a vague or incomplete explanation of how `resolve` and `reject` are handled.



3



4

Clearly defines a Promise, explains all three states, and describes how `.then()` and `.catch()` respond to each outcome.



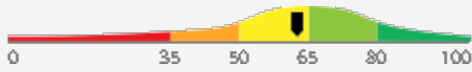
5

Detail

Interview Guide

**Functions and Control Flow (Coding Tasks)**

Score: 62



*Description:*

Covers the use of pointers to reference and manipulate memory addresses, along with dynamic memory allocation and deallocation using malloc, calloc, realloc, and free. Includes pointer arithmetic, dereferencing, and avoiding common issues like memory leaks and dangling pointers.

*Interpretation:*

The candidate exhibits average writing skills, which can hinder high performance in some jobs.

The candidate possesses a moderate working knowledge of C programming, demonstrating familiarity with core concepts including data types, control flow, functions, and basic file I/O. They may require some guidance when working with more advanced topics such as dynamic memory allocation, modular design, or debugging complex logic.

Overall AI Score:	65.0
Lines of Code:	15.0
Syntax Errors:	5.0
AI Confidence Level:	50
Match with Ideal Response (AI):	30.0
Structure:	50.0
Syntax:	30.0

Please see below to view the essay submitted.

Walk me through how you would dynamically allocate memory for an array of 10 integers, use it, and then properly release it. What issues might arise if you don't follow best practices?



1

Cannot write correct allocation code; unaware of free() or memory leak risks.



2

Writes mostly correct malloc/free code; identifies memory leaks but misses other risks.



3



4

Correct malloc, use, and free; identifies leaks, dangling pointers, and NULL check on allocation.



5

Can you explain what a pointer is in C and describe a situation where you would use one?



1

Vague or incorrect definition; cannot describe a practical use case.



2

Correct basic definition; gives a simple but valid use case with some gaps.



3



4

Clear definition with accurate use case; mentions address storage, dereferencing, or dynamic memory.



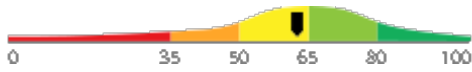
5

Detail

Interview Guide

**Variables, Data Types, and Scope (Coding Tasks)**

Score: 62



*Description:*

Covers the use of pointers to reference and manipulate memory addresses, along with dynamic memory allocation and deallocation using malloc, calloc, realloc, and free. Includes pointer arithmetic, dereferencing, and avoiding common issues like memory leaks and dangling pointers.

*Interpretation:*

The candidate exhibits average writing skills, which can hinder high performance in some jobs.

The candidate possesses a moderate working knowledge of C programming, demonstrating familiarity with core concepts including data types, control flow, functions, and basic file I/O. They may require some guidance when working with more advanced topics such as dynamic memory allocation, modular design, or debugging complex logic.

Overall AI Score:	65.0
Lines of Code:	15.0
Syntax Errors:	5.0
AI Confidence Level:	50
Match with Ideal Response (AI):	30.0
Structure:	50.0
Syntax:	30.0

Please see below to view the essay submitted.

Walk me through how you would dynamically allocate memory for an array of 10 integers, use it, and then properly release it. What issues might arise if you don't follow best practices?



1

Cannot write correct allocation code; unaware of free() or memory leak risks.



2

Writes mostly correct malloc/free code; identifies memory leaks but misses other risks.



3



4

Correct malloc, use, and free; identifies leaks, dangling pointers, and NULL check on allocation.



5

Can you explain what a pointer is in C and describe a situation where you would use one?



1

Vague or incorrect definition; cannot describe a practical use case.



2

Correct basic definition; gives a simple but valid use case with some gaps.



3



4

Clear definition with accurate use case; mentions address storage, dereferencing, or dynamic memory.



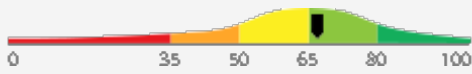
5

Detail

Interview Guide

**Error Handling and Debugging**

Score: 67



*Description:*

Covers how to detect, catch, and respond to runtime errors using try/catch/finally blocks and how to throw custom errors to communicate problems in a program. Includes using console methods (such as console.log, console.error, and console.table) to track program behavior and identify issues, as well as recognizing common sources of errors like type mismatches and null reference exceptions.

*Interpretation:*

Candidate should achieve above average job performance in this area with little or no training.

The candidate demonstrates a solid working knowledge of JavaScript error handling and debugging practices. They are generally proficient with try/catch/finally structures, custom error creation, and console-based debugging tools, and can identify and address common runtime error sources with reasonable competence.

What console methods do you use most often when debugging JavaScript code, and how do you approach narrowing down the source of a bug in a program?



1

Mentions only console.log with no structured approach to isolating or diagnosing bugs.



2

Describes a few console methods and a basic debugging process but lacks a systematic or methodical approach.



3



4

References multiple console methods, describes a clear step-by-step debugging strategy, and may mention browser or Node.js debugging tools.



5

Can you explain how a try/catch block works in JavaScript and describe a situation where you would use one in your code?



1

Cannot explain try/catch syntax or does not understand when or why it would be used.



2

Correctly describes try/catch structure but gives a vague or overly simple example of when to apply it.



3



4

Accurately explains try/catch/finally, provides a realistic use case, and may mention throwing custom errors.



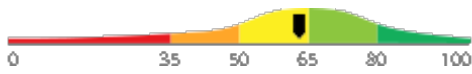
5

Detail

Interview Guide

**Functions and Control Flow**

Score: 63



*Description:*

Covers how to define and call functions using function declarations, function expressions, and arrow functions. Includes the use of control flow structures such as if/else statements, switch statements, and loops (for, while, for...of, for...in) to direct the execution path of a program based on conditions or repeated logic.

*Interpretation:*

Candidate appears capable of average job performance in this area with little or no training.

The candidate has a moderate grasp of JavaScript functions and control flow, demonstrating partial competency with function declarations, expressions, and common control structures. They can work with these concepts in straightforward scenarios but may encounter difficulty with more nuanced or complex applications.

Describe how you would choose between a for loop, a for...of loop, and a for...in loop when iterating over data in a JavaScript program.



1

Cannot differentiate the loop types or provides incorrect use cases for one or more.



2

Correctly identifies general use cases but lacks precision or misses edge cases like iterating object keys vs. array values.



3



4

Clearly distinguishes all three with accurate, practical use cases and mentions potential pitfalls of for...in on arrays.



5

Can you walk me through what an arrow function is and how it differs from a regular function declaration in JavaScript?



1

Cannot describe arrow function syntax or confuses it with a regular function declaration.



2

Describes basic syntax differences but does not mention behavioral differences such as 'this' binding.



3



4

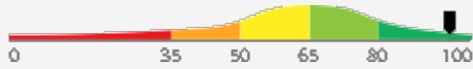
Accurately explains syntax, 'this' binding differences, and when to prefer one form over the other.



5

**Detail**
**Interview Guide**
**Object-Oriented Programming and Modules**

Score: 95


*Description:*

Covers how to use JavaScript classes to define objects with shared properties and methods, including inheritance using the extends keyword and the use of the constructor method. Also includes how to organize code into reusable modules using import and export statements to keep programs maintainable and well-structured.

*Interpretation:*

Candidate should achieve superior job performance in this area with little or no training.

The candidate demonstrates a strong and comprehensive command of JavaScript object-oriented programming and module organization. They are highly proficient in defining classes with shared properties and methods, implementing inheritance, and structuring maintainable, well-organized code using import and export statements.

How do you use JavaScript modules to organize your code across multiple files, and what is the difference between a named export and a default export?



1

Cannot explain how modules work or does not know the difference between named and default exports.



2

Correctly explains one export type but gives an incomplete or imprecise description of the other.



3



4

Accurately explains both export types with syntax examples and describes practical reasons for choosing one over the other.



5

---

Can you describe what a JavaScript class is and explain how you would use it to create multiple objects that share the same properties and methods?



1

Cannot describe a class or does not understand how a class is used to create object instances.



2

Describes a class and its constructor but struggles to explain how instances share methods through the prototype.



3



4

Clearly explains class syntax, the constructor, instance creation, and how methods are shared across instances.



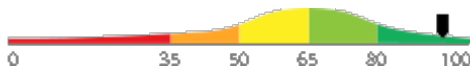
5

Detail

Interview Guide

**Objects, Arrays, and Collection Structures**

Score: 94



*Description:*

Covers how to create, access, and manipulate objects and arrays, which are the most commonly used data structures in JavaScript. Includes working with object properties and methods, array methods (such as map, filter, reduce, and forEach), and modern syntax features like destructuring and the spread operator to efficiently organize and transform collections of data.

*Interpretation:*

Candidate should achieve superior job performance in this area with little or no training.

The candidate exhibits an advanced and comprehensive mastery of JavaScript objects, arrays, and collection structures. They are highly proficient in creating, accessing, and manipulating data using object properties and methods, array transformation methods, and modern syntax features, demonstrating the ability to efficiently organize and transform complex collections of data.

How would you use destructuring and the spread operator to simplify working with objects or arrays in your code? Can you give a practical example of each?



1

Cannot explain destructuring or the spread operator, or confuses their syntax and purpose.



2

Explains one feature correctly but struggles to provide a clear, practical example for both.



3



4

Accurately demonstrates both features with clear, real-world examples and explains the readability or efficiency benefits.



5

Can you describe how you would use an array method like map or filter to work with a list of items, and walk me through what the method returns?



1

Cannot describe map or filter or confuses what each method returns.



2

Correctly describes one method but struggles to explain the return value or contrast it with the other.



3



4

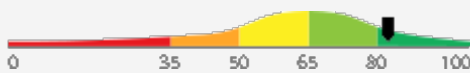
Accurately explains both methods, their return values, and gives a clear, practical example for each.



5

**Variables, Data Types, and Scope**

Score: 82



*Description:*

Covers how to declare variables using var, let, and const, and how JavaScript's core data types (strings, numbers, booleans, null, undefined, objects, and arrays) are used to store and represent information. Includes understanding how scope rules (global, function, and block scope) determine where variables are accessible within a program.

*Interpretation:*

Candidate should achieve superior job performance in this area with little or no training.

The candidate demonstrates a comprehensive and advanced mastery of JavaScript programming syntax, structure, keywords, and design patterns. They are well-equipped to confidently implement reusable components, manage asynchronous operations, handle runtime errors, and optimize application performance across a broad range of business programming scenarios.

How does block scope differ from function scope in JavaScript, and how does that affect how you decide which variable declaration keyword to use?



1

Confuses block and function scope or cannot explain how scope affects variable declaration choices.



2

Correctly describes the difference but gives a generic or incomplete explanation of decision-making.



3



4

Clearly contrasts both scopes with examples and articulates a practical, reasoned approach to choosing keywords.



5

Can you explain the difference between var, let, and const, and describe a situation where using the wrong one could cause a bug in your code?



1

Cannot distinguish between var, let, and const or does not connect declarations to scope behavior.



2

Explains basic differences but struggles to describe a concrete bug scenario caused by misuse.



3



4

Clearly explains scoping differences, hoisting, and gives a specific, realistic bug example.



5

## IT Coding Tasks

During the assessment, the candidate was asked to write one or more programs or scripts. Their responses are included below for review.

Question or Task	Response
<p>Complete the provided partial C program by filling in the missing sections marked with TODO comments. Your completion must use standard C keywords and libraries.</p> <p>The program declares a function called <code>duplicate_array</code> that:</p> <ol style="list-style-type: none"> <li>1. Takes a const int pointer to a source array and its length as parameters.</li> <li>2. Uses <code>calloc</code> to allocate a new int array of the same length.</li> <li>3. Returns NULL if <code>calloc</code> fails.</li> <li>4. Copies each element from the source array into the new array using pointer arithmetic (not array subscript notation).</li> <li>5. Returns the pointer to the newly allocated copy.</li> </ol> <p>In main, the program:</p> <ol style="list-style-type: none"> <li>1. Declares and initializes a stack array of 4 integers with values 5, 15, 25, 35.</li> <li>2. Calls <code>duplicate_array</code> to create a heap-allocated copy.</li> <li>3. Checks for NULL and prints an error and returns 1 if the call failed.</li> <li>4. Prints each element of the duplicate using a loop.</li> <li>5. Frees the duplicate array.</li> </ol> <p>Use appropriate indentation, common C coding conventions, and add brief inline comments where needed. Type your completed source code as your response.</p>	<pre>#include &lt;stdio.h&gt; #include &lt;stdlib.h&gt;  int *duplicate_array(const int *src, int length) {     /* TODO: Use calloc to allocate a new array of 'length' integers, return        NULL if calloc fails, copy elements from src using pointer arithmetic,        and return the new pointer. */     calloc(303); }  int main(void) {     /* TODO: Declare and initialize a stack array of 4 integers: 5, 15, 25, 35,        then call duplicate_array and store the result. Check for NULL and        print an error message returning 1 if it failed. */     array[4]={5,15,25,35};      int i;      /* Print each element of the duplicate */     for (i = 0; i &lt; 4; i++) {         printf("duplicate[%d] = %d\n", i, *(duplicate + i));     }      /* Free the duplicate array */     free(duplicate);     return 0; }</pre>

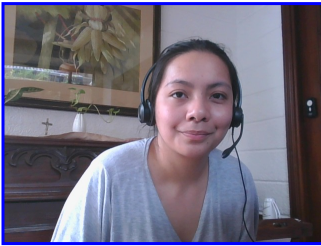
**Comments (AI):** The code segment has several syntax errors and incomplete implementation. The `duplicate_array` function does not correctly allocate memory or copy elements. The main function has syntax errors and does not properly call the `duplicate_array` function. However, the structure and intent of the code are somewhat clear, and the code attempts to follow the requirements.

## Identity Confirmation Photos

The following photos of the candidate and any identification were uploaded during the assessment session.

### Photo Analysis Results

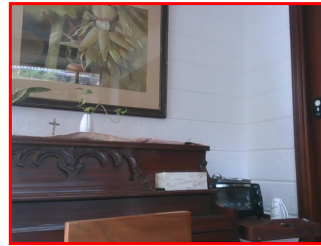
<b>- Risk:</b>	<b>Medium risk of cheating based on image inconsistencies</b>
- Percent match among processed faces	100%
- Total images processed	17
- Total images with valid faces	14 (82%)
- Total pairs of faces compared	13
- Pairs in which faces matched	13 (100%)



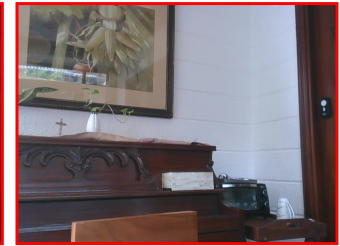
Pre/Post-Test Photo



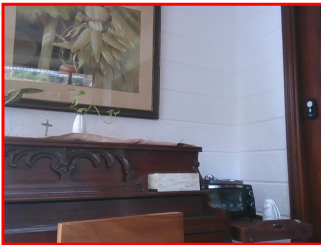
ID Photo



In-Test Error Detected (No Face Detected)



In-Test Error Detected (No Face Detected)



In-Test Error Detected (No Face Detected)



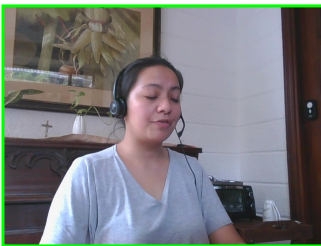
In-Test Photo



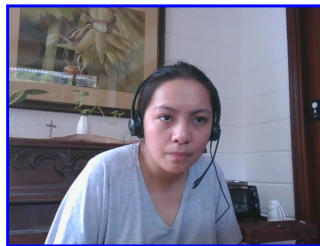
In-Test Photo



In-Test Photo



In-Test Photo



Pre/Post-Test Photo

## Resume or CV

Summary

Updated on

Motivated career professional with extensive experience in office administration and management. Proven track record of improving efficiency, reducing costs, and enhancing office operations through strategic initiatives and technology implementation.

### Objective

I am seeking a role where I can use my many skills and my exceptional judgment and empathy for customers to make a difference to a growing company.

### Education

- Associate of Applied Science in Office Administration, Portland Community College, 2020

### Experience

- General Office Clerk, Paramount Office Management, 09/2023 – Present
- Administrative Assistant, Global Enterprises Inc., 04/2021 – 08/2023
- Administrative Assistant, Innovative Business Solutions Ltd., 07/2019 – 03/2021

### Other Qualifications

- Microsoft Office Specialist (MOS) Certification
- Certified Administrative Professional (CAP)
- International Association of Administrative Professionals (IAAP) Certification

## Report Preparation Notes

- Hiring decisions should never be based on a single source of information. The most effective use of this assessment report is as a part of a multi-faceted program of candidate evaluation that includes resume review, interviews, and reference checks.
- Overall vs Percentiles Scores: The overall score reflects the success in the test, based on the mean (average) and standard deviation of the test scores. The percentile score reflects the percentage of test-takers who scored equal or below this overall score. We recommend you use the Overall Score as your primary evaluation criteria. However, percentile scores can often be useful in comparing specific candidates against one another and with a group, such as for test takers in a certain organization or within a certain account.
- Note that comparison information is calculated based on completed instances of this assessment at that time the assessment is scored. As additional instances are completed, the comparative data may change. You can always update a report to the current values by clicking on 'Recalculate Percentiles' within the online results viewing pages at [www.hravatar.com](http://www.hravatar.com).
- Most competency scores are norm-based, which means that they can be interpreted in terms of their distance from the average or mean score. For all scales, a score equal to the mean receives a score of 65 and scores above and below this value are set so that a score change of 15 equals one standard deviation.
- For linear competencies, higher is better across the entire scale. For these scales a score between 65 and 80 (light green) represents 0 to 1 standard deviation above the mean and a score above 80 (dark green) represents more than one standard deviation above the mean. Similarly, a score of 50 - 65 (yellow) represents 0 to 1 standard deviation below the mean, while a score of 35 - 50 (orange) equates to 1 to 2 standard deviations below the mean, and a score below 35 represents more than 2 standard deviations below the mean.
- Sim ID: 20788-1, Key: 0-0, Rpt: 68, Prd: 9610, Created: 2026-06-27 13:06 EDT
- UA: Mozilla/5.0 (Windows NT 6.3; Trident/7.0; Touch; rv:11.0) like Gecko

## Score Calculation Detail

The following table provides a summary of how the overall score was calculated from each of the individual competency scores. First, all competency scores are calculated on a scale of 0-100. Note that some competencies use their color category rather than their actual numeric score in the overall calculation. For these, a standard score associated with the assigned color category is used in the overall score calculation rather than the actual numeric score. This is reflected in the "Score Value Used" column. Next, a weighted average of scores is computed using individual competency weights, typically set using job analysis data provided by the US Government Occupational Information Network (O\*Net).

Competency	Score	How applied to overall	Score Value Used	Weight (%)
Asynchronous Programming and Promises	94.2074	Numeric Score	94.2074	12.5000
Error Handling and Debugging	67.4171	Numeric Score	67.4171	12.5000
Functions and Control Flow	63.1622	Numeric Score	63.1622	12.5000
Functions and Control Flow (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
Object-Oriented Programming and Modules	95.7153	Numeric Score	95.7153	12.5000
Objects, Arrays, and Collection Structures	94.5087	Numeric Score	94.5087	12.5000
Variables, Data Types, and Scope	82.6907	Numeric Score	82.6907	12.5000
Variables, Data Types, and Scope (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
Weighted Average:				77.9573
Final Overall Score:				77

## Notes

(This area is intentionally blank - it's reserved as space for your notes.)