hr | avatar
Illuminating Talent

Test Results and Interview Guide

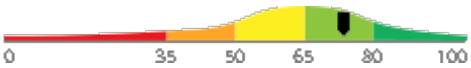
Candidate: **Elizabeth Wantsajob**
Assessment: Javascript Programming (Core) (Short)
Completed: June 27, 2026
Prepared for: Sara Maple
Example Company

What's Included

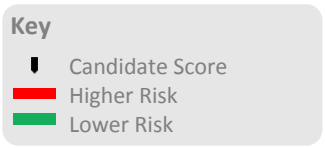
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

Important Note: The Javascript Programming (Core) (Short) 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.

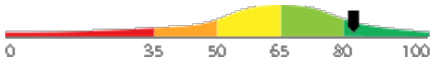
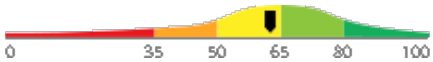
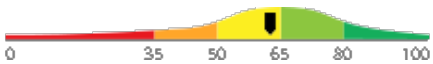
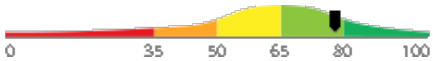
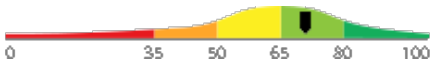
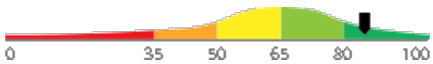
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com Javascript Programming (Core) (Short) June 27, 2026	73	

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.



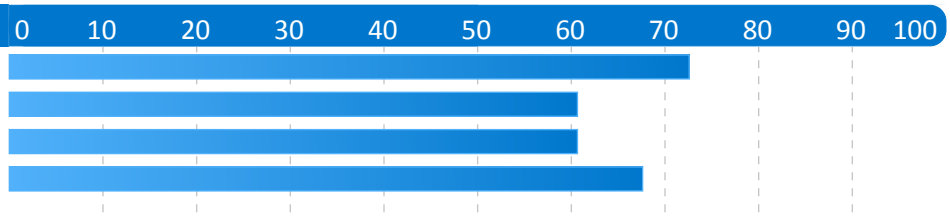
Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
Asynchronous Programming and Promises	82	
Functions and Control Flow (Coding Tasks)	62	
Variables, Data Types, and Scope (Coding Tasks)	62	
Functions and Control Flow	78	
Objects, Arrays, and Collection Structures	71	
Variables, Data Types, and Scope	85	

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
Global	73rd
North America	61st
United States	61st
Example Company	68th



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	-	-	Summary Points (AI): • (Generic Text for Sample Report) Strong performer in Drag and Drop Files tasks, indicating comfort with file management and basic computer interactions. • Demonstrates solid numerical accuracy in Recognizing and Confirming Numbers, a valuable asset in detail-oriented roles. • Moderate overall performance in Analytical Thinking and Attention to Detail, with adequate grammar skills but room for improvement. • Struggles with Reading and Analyzing Problems, which may limit effectiveness in roles requiring critical reading and complex problem-solving. • Lowest performance in Navigating Between Screens, suggesting difficulty with multi-screen software workflows that could impact productivity in computer-intensive roles. Narrative (AI): Elizabeth Wantsajob demonstrates a mixed profile of knowledge, skills, and abilities across the assessed competencies. 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. 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. 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. 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. Computed on: April 2, 2026, 11:09:49PM EDT

Detail

Candidate:	Elizabeth Wantsajob, beth.wantsajob@gmail.com
Assessment:	Javascript Programming (Core) (Short)
Authorized:	June 27, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	June 27, 2026, 1:20:37PM EDT
Completed:	June 27, 2026, 1:20:37PM EDT
Overall Score:	73

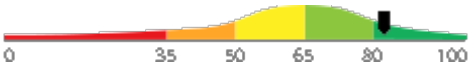
Knowledge and Skills Detail

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: 82



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	2	3	4	5
Cannot explain async/await syntax or does not know how to handle errors in an async function.		Explains async/await syntax and mentions try/catch but does not clearly contrast it with promise chaining.		Clearly contrasts async/await with chaining, demonstrates try/catch usage, and may mention handling multiple async tasks.

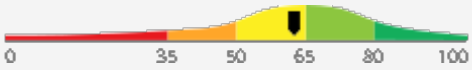
Can you explain what a Promise is in JavaScript and describe what happens when it resolves or rejects?

1	2	3	4	5
Cannot define a Promise or does not understand the resolved and rejected states.		Defines a Promise correctly but gives a vague or incomplete explanation of how resolve and reject are handled.		Clearly defines a Promise, explains all three states, and describes how .then() and .catch() respond to each outcome.

Detail

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.

Interview Guide

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	2	3	4	5
Cannot write correct allocation code; unaware of free() or memory leak risks.		Writes mostly correct malloc/free code; identifies memory leaks but misses other risks.		Correct malloc, use, and free; identifies leaks, dangling pointers, and NULL check on allocation.

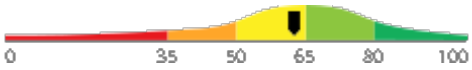
Can you explain what a pointer is in C and describe a situation where you would use one?

1	2	3	4	5
Vague or incorrect definition; cannot describe a practical use case.		Correct basic definition; gives a simple but valid use case with some gaps.		Clear definition with accurate use case; mentions address storage, dereferencing, or dynamic memory.

Detail

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.

Interview Guide

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	2	3	4	5
Cannot write correct allocation code; unaware of free() or memory leak risks.		Writes mostly correct malloc/free code; identifies memory leaks but misses other risks.		Correct malloc, use, and free; identifies leaks, dangling pointers, and NULL check on allocation.

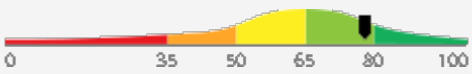
Can you explain what a pointer is in C and describe a situation where you would use one?

1	2	3	4	5
Vague or incorrect definition; cannot describe a practical use case.		Correct basic definition; gives a simple but valid use case with some gaps.		Clear definition with accurate use case; mentions address storage, dereferencing, or dynamic memory.

Detail

Functions and Control Flow

Score: 78



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

The candidate shows a solid and proficient understanding of JavaScript functions and control flow, including the use of function declarations, expressions, and arrow functions alongside a range of conditional and looping structures. They can reliably apply these concepts to direct program execution in a variety of practical situations.

Interview Guide

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	2	3	4	5
Cannot differentiate the loop types or provides incorrect use cases for one or more.		Correctly identifies general use cases but lacks precision or misses edge cases like iterating object keys vs. array values.		Clearly distinguishes all three with accurate, practical use cases and mentions potential pitfalls of for...in on arrays.

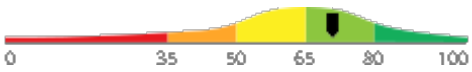
Can you walk me through what an arrow function is and how it differs from a regular function declaration in JavaScript?

1	2	3	4	5
Cannot describe arrow function syntax or confuses it with a regular function declaration.		Describes basic syntax differences but does not mention behavioral differences such as 'this' binding.		Accurately explains syntax, 'this' binding differences, and when to prefer one form over the other.

Detail	Interview Guide

Objects, Arrays, and Collection Structures

Score: 71



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

The candidate demonstrates a solid and proficient understanding of JavaScript objects, arrays, and collection structures, including the use of common array methods and object manipulation techniques. Some gaps may exist in the more advanced application of modern syntax features such as destructuring and the spread operator, but overall competence in this area is well established.

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?

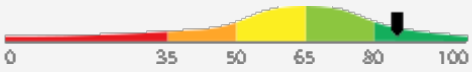
1	2	3	4	5
Cannot explain destructuring or the spread operator, or confuses their syntax and purpose.		Explains one feature correctly but struggles to provide a clear, practical example for both.		Accurately demonstrates both features with clear, real-world examples and explains the readability or efficiency benefits.

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	2	3	4	5
Cannot describe map or filter or confuses what each method returns.		Correctly describes one method but struggles to explain the return value or contrast it with the other.		Accurately explains both methods, their return values, and gives a clear, practical example for each.

Variables, Data Types, and Scope

Score: 85



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	2	3	4	5
Confuses block and function scope or cannot explain how scope affects variable declaration choices.		Correctly describes the difference but gives a generic or incomplete explanation of decision-making.		Clearly contrasts both scopes with examples and articulates a practical, reasoned approach to choosing keywords.

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	2	3	4	5
Cannot distinguish between var, let, and const or does not connect declarations to scope behavior.		Explains basic differences but struggles to describe a concrete bug scenario caused by misuse.		Clearly explains scoping differences, hoisting, and gives a specific, realistic bug example.

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.

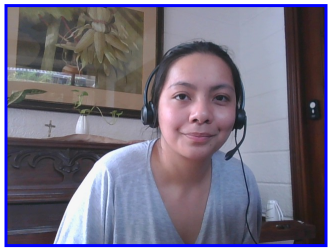
Question or Task	Response

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.
 	
The program declares a function called duplicate_array that:
1. Takes a const int pointer to a source array and its length as parameters.
2. Uses calloc to allocate a new int array of the same length.
3. Returns NULL if calloc fails.
4. Copies each element from the source array into the new array using pointer arithmetic (not array subscript notation).
5. Returns the pointer to the newly allocated copy.
 	
In main, the program:
1. Declares and initializes a stack array of 4 integers with values 5, 15, 25, 35.
2. Calls duplicate_array to create a heap-allocated copy.
3. Checks for NULL and prints an error and returns 1 if the call failed.
4. Prints each element of the duplicate using a loop.
5. Frees the duplicate array.
 	
Use appropriate indentation, common C coding conventions, and add brief inline comments where needed. Type your completed source code as your response.
	#include <stdio.h>
	#include <stdlib.h>
	
	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 < 4; i++) {
	printf("duplicate[%d] = %d\n", i, *(duplicate + i));
	}
	
	/* Free the duplicate array */
	free(duplicate);
	return 0;
	}
	
	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

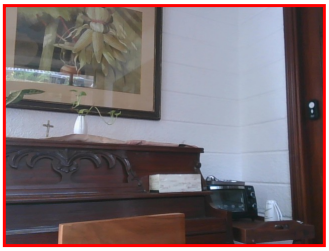
The following photos of the candidate and any identification were uploaded during the assessment session.

Photo Analysis Results	
- Risk:	Medium risk of cheating based on image inconsistencies
- 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%)

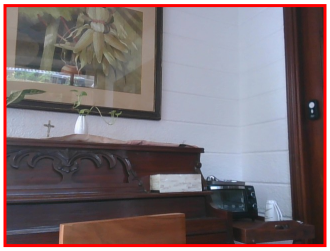


Pre/Post-Test Photo

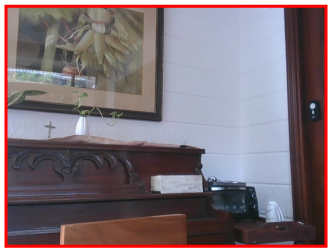
ID Photo



In-Test Error Detected (No Face Detected)

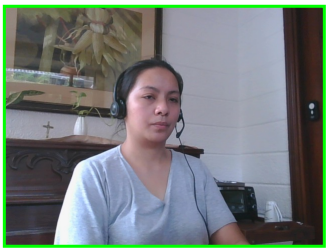


In-Test Error Detected (No Face Detected)

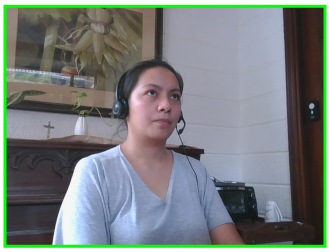


In-Test Error Detected (No Face Detected)

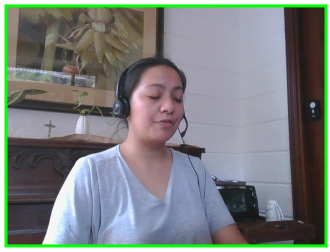
In-Test Photo



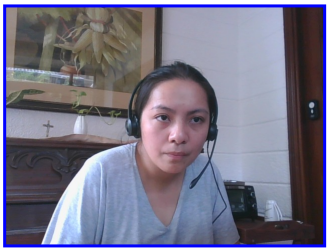
In-Test Photo



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.
- 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: 20789-1, Key: 0-0, Rpt: 68, Prd: 9611, Created: 2026-06-27 13:20 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	82.7367	Numeric Score	82.7367	16.6667
Functions and Control Flow	78.4165	Numeric Score	78.4165	16.6667
Functions and Control Flow (Coding Tasks)	62.9784	Numeric Score	62.9784	16.6667
Objects, Arrays, and Collection Structures	71.0791	Numeric Score	71.0791	16.6667
Variables, Data Types, and Scope	85.2915	Numeric Score	85.2915	16.6667
Variables, Data Types, and Scope (Coding Tasks)	62.9784	Numeric Score	62.9784	16.6667
Weighted Average:				73.9135
Final Overall Score:				73

Notes

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