hr | avatar
Illuminating Talent

Test Results and Interview Guide

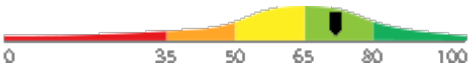
Candidate: **Elizabeth Wantsajob**
Assessment: jQuery Programming and Usage
Completed: June 28, 2026
Prepared for: Sara Maple
Example Company

What's Included

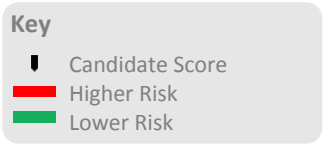
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

Important Note: The jQuery Programming and Usage 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.

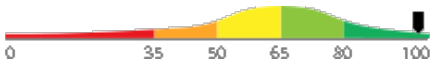
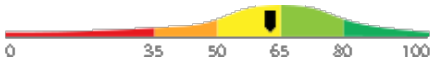
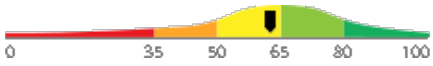
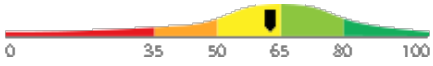
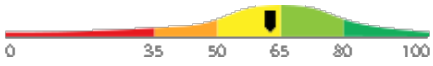
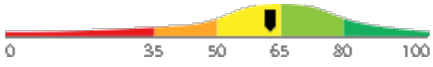
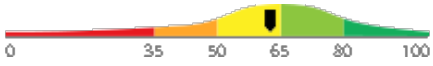
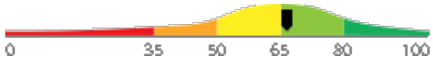
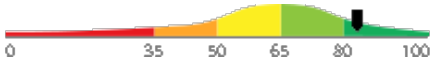
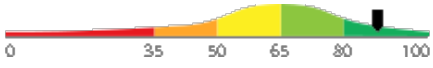
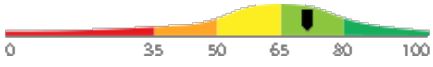
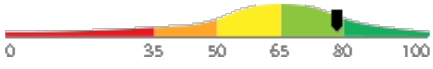
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com jQuery Programming and Usage June 28, 2026	72	

The candidate exhibits a solid and competent command of jQuery, including DOM traversal and manipulation, event delegation, effects, and AJAX interactions using callbacks and promises. Minor gaps may exist in advanced topics such as deferred objects, library conflict resolution, or complex chaining patterns, but the candidate is well-equipped to handle most entry-level to mid-level jQuery development tasks.



Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
AJAX Requests and Response Handling	97	
AJAX Requests and Response Handling (Coding Tasks)	62	
DOM Manipulation: Creating, Adding, and Removing Elements (Coding Tasks)	62	
DOM Selection and Manipulation (Coding Tasks)	62	
Document Ready and Script Initialization (Coding Tasks)	62	
Effects and Animations (Coding Tasks)	62	
Event Handling (Coding Tasks)	62	
DOM Manipulation: Creating, Adding, and Removing Elements	67	
DOM Selection and Manipulation	83	
Document Ready and Script Initialization	88	
Effects and Animations	71	
Event Handling	78	

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	72nd
North America	59th
United States	59th
Example Company	66th

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:	jQuery Programming and Usage
Authorized:	June 28, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	June 28, 2026, 12:59:35PM EDT
Completed:	June 28, 2026, 12:59:35PM EDT
Overall Score:	72

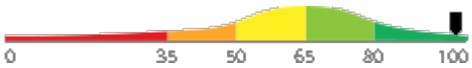
Knowledge and Skills Detail

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

AJAX Requests and Response Handling

Score: 97



Description:

Covers how to send and receive data from a server without reloading the page using jQuery methods like $.ajax(), $.get(), and $.post(). Includes handling server responses using callbacks and promises, and understanding how to update the page with returned data.

Interpretation:

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

The candidate demonstrates a comprehensive and advanced understanding of jQuery AJAX methods, response handling, and asynchronous page updates. They are well-equipped to design, implement, and troubleshoot complex AJAX-driven interactions, including sophisticated use of promises and dynamic DOM manipulation based on server-returned data.

Walk me through how you would use $.ajax() to send a POST request to a server, handle a successful response by updating part of the page, and display an error message if the request fails.

1	2	3	4	5
Cannot write a working $.ajax() call or does not know how to handle success and error separately.		Writes a mostly correct $.ajax() call with 'type: POST', a success callback that updates the DOM, and an error callback.		Writes a complete, correct $.ajax() call using proper options, handles success and error clearly, and may mention using .done() and .fail() promise methods.

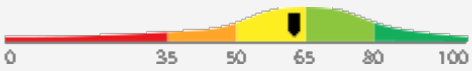
What is AJAX, and can you describe how jQuery helps you make a simple request to a server to get some data?

1	2	3	4	5
Cannot explain AJAX or cannot name any jQuery method used to make a request.		Correctly explains AJAX as a way to get data without reloading and names $.get() or $.ajax() with a basic example.		Explains AJAX clearly, demonstrates $.get() or $.ajax() with a URL and callback, and mentions handling the response data.

Detail

AJAX Requests and Response Handling (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.

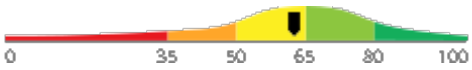
Can you explain what a pointer is in C and describe a situation where you would use one?

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

DOM Manipulation: Creating, Adding, and Removing Elements (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.

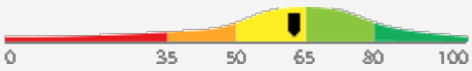
Can you explain what a pointer is in C and describe a situation where you would use one?

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

DOM Selection and Manipulation (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.

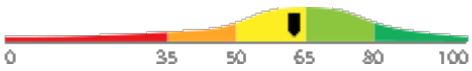
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Document Ready and Script Initialization (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.

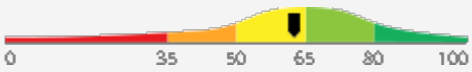
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Effects and Animations (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.

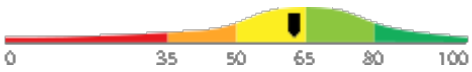
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Event Handling (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 concept 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.

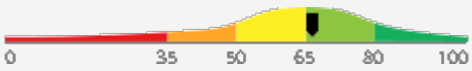
Can you explain what a pointer is in C and describe a situation where you would use one?

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

DOM Manipulation: Creating, Adding, and Removing Elements

Score: 67



Description:

Covers how to dynamically create new HTML elements, insert them into the page, remove existing elements, and clone elements using jQuery methods like .append(), .prepend(), .remove(), .clone(), and .html(). This is a core skill for building interactive and dynamic web pages.

Interpretation:

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

The candidate exhibits a solid proficiency in jQuery DOM manipulation, including the ability to dynamically create, insert, clone, and remove HTML elements using standard jQuery methods. They are well-equipped to contribute to interactive web page development with minimal supervision.

Interview Guide

Describe how you would dynamically build a list of items from an array of values using jQuery and insert the completed list into the page. What methods would you use and why?

1	2	3	4	5
Cannot describe a working approach; confuses methods or cannot connect iteration to DOM insertion.		Describes using $.each() or a loop to build list items and .append() to add them to a ul element in the DOM.		Describes an efficient approach, builds elements in a variable before a single DOM insertion for performance, and correctly uses jQuery creation and insertion methods.

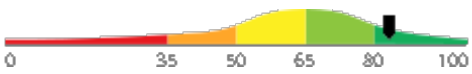
How would you use jQuery to add a new paragraph with some text to the end of a div on the page?

1	2	3	4	5
Cannot name the correct method or writes code that would not produce the intended result.		Correctly uses .append() or .html() to add a new paragraph element with text inside a target div.		Uses .append() correctly with a created element or HTML string, and may mention alternative methods like .prepend() or .after() with an explanation of the differences.

Detail

DOM Selection and Manipulation

Score: 83



Description:

Covers the use of jQuery selectors, filters, and traversal methods to find and interact with HTML elements on a page. Includes reading and changing element content, attributes, CSS properties, and classes. This is the most fundamental and frequently used aspect of jQuery in everyday web development.

Interpretation:

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

The candidate demonstrates a comprehensive and advanced understanding of jQuery, reflecting strong proficiency across all major areas including DOM manipulation, event handling, AJAX with callbacks and deferred objects, animations, method chaining, and debugging. This level of knowledge indicates the ability to independently write, maintain, and troubleshoot sophisticated jQuery-based web applications with minimal oversight.

Interview Guide

Describe how you would use jQuery to find all list items inside a specific unordered list, change their text color to blue, and add a CSS class to each one. Walk through the code you would write.

1	2	3	4	5
Cannot combine selection, CSS changes, and class addition; confuses methods or syntax.		Correctly uses a selector and applies .css() and .addClass() but may not chain them or use traversal efficiently.		Writes clean, chained jQuery code using a scoped selector, .css(), and .addClass(); may mention performance benefits of chaining.

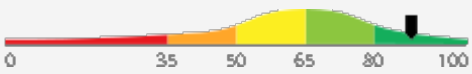
Can you explain what a jQuery selector is and give an example of how you would use one to find an element on a page?

1	2	3	4	5
Vague or incorrect description; cannot provide a working example.		Correctly explains selectors and gives a basic example like $('#id') or $('.class').		Explains selectors clearly, gives multiple examples, and mentions filters or traversal methods like .find() or .children().

Detail

Document Ready and Script Initialization

Score: 88



Description:

Covers how to ensure jQuery code runs only after the HTML document has fully loaded using $(document).ready() or the shorthand $(function(){}). Includes understanding why this matters for accessing DOM elements and how it fits into the overall structure of a jQuery-powered page.

Interpretation:

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

The candidate demonstrates an advanced and comprehensive understanding of jQuery's Document Ready concept and script initialization practices. They are highly proficient in implementing and explaining both syntactical forms, the implications for DOM element access, and how this pattern integrates into the broader architecture of a jQuery-powered page.

Interview Guide

If a teammate told you their jQuery code wasn't finding any elements on the page even though the selectors looked correct, what would be your first steps to diagnose and fix the problem?

1	2	3	4	5
Cannot identify missing document ready as a likely cause or suggests unrelated fixes.		Identifies that the code may be running before the DOM is loaded and suggests wrapping it in a document ready function.		Identifies the document ready issue, mentions checking the browser console for errors, verifying selector spelling, and confirms the script load order as part of a systematic diagnosis.

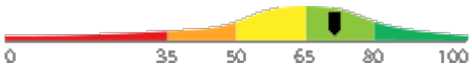
Why is it important to wrap your jQuery code inside a document ready function, and how do you write one?

1	2	3	4	5
Cannot explain why it is needed or cannot write the correct syntax.		Correctly explains that it ensures the DOM is loaded before the code runs and writes $(document).ready() or $(function(){}) correctly.		Explains the reason clearly, writes the correct syntax, and may mention the shorthand form or the difference between document ready and window load.

Detail

Effects and Animations

Score: 71



Description:

Covers jQuery's built-in methods for showing, hiding, fading, and sliding elements, as well as using the .animate() method to create custom animations. These methods are widely used to improve user experience by adding visual feedback and transitions to web pages.

Interpretation:

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

The candidate demonstrates a solid and competent understanding of jQuery effects and animations, including proficient use of built-in methods for fading, sliding, and element visibility. They are likely capable of implementing custom animations and applying visual transitions effectively to improve user experience.

Interview Guide

Explain how you would use jQuery to create a smooth fade-in effect for an element after it is added to the page, and how you would run additional code only after the animation has fully completed.

1	2	3	4	5
Cannot describe a working fade-in approach or does not know how to run code after an animation finishes.		Correctly uses .fadeIn() and knows that a callback function passed to it runs after the animation completes.		Clearly explains .fadeIn() with a callback, may mention speed options, and could reference the promise-based alternative using .promise().done().

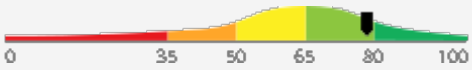
How would you use jQuery to hide an element on the page when a user clicks a button, and then show it again when they click another button?

1	2	3	4	5
Cannot name the correct methods or writes code that would not toggle visibility correctly.		Correctly uses .hide() and .show() inside click event handlers for each button.		Uses .hide() and .show() correctly, may mention .toggle() as a simpler alternative, and may note optional speed parameters.

Detail

Event Handling

Score: 78



Description:

Covers how to listen for and respond to user and browser events such as clicks, keypresses, and form submissions using jQuery's .on(), .off(), and related methods. Includes understanding event delegation for dynamically added elements and how to prevent default browser behavior.

Interpretation:

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

The candidate shows a solid and largely proficient understanding of jQuery event handling, including attaching and removing event listeners and applying event delegation to dynamic elements. Minor gaps may exist in edge cases or advanced usage patterns, but they are generally capable of implementing event-driven functionality effectively.

Interview Guide

Explain the difference between binding a click event directly to an element versus using event delegation. When would you choose one approach over the other?

1	2	3	4	5
Cannot explain the difference or incorrectly describes how delegation works.		Correctly explains that delegation uses a parent element and handles dynamically added children, with a basic example.		Clearly explains both approaches, gives practical examples, and accurately describes when delegation is necessary for dynamic content.

How would you use jQuery to make something happen when a user clicks a button on a page?

1	2	3	4	5
Cannot name the correct method or writes non-functional code; confuses event names or syntax.		Correctly uses .on('click', ...) or .click() with a basic callback function.		Uses .on() correctly, explains the callback, and may mention event object or preventing default behavior.

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.

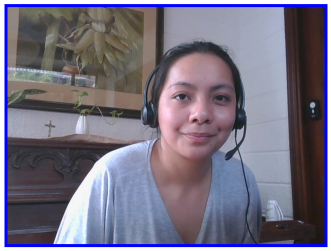
Question or Task	Response

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

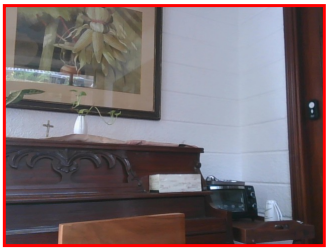
The following photos of the candidate and any identification were uploaded during the assessment session.

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%)

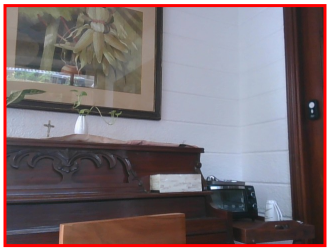


Pre/Post-Test Photo

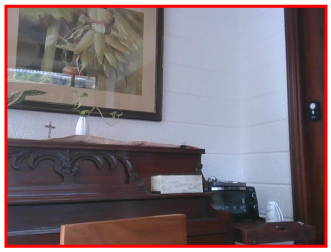
ID Photo



In-Test Error Detected (No Face Detected)

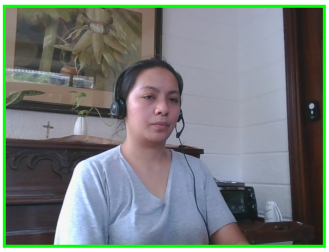


In-Test Error Detected (No Face Detected)

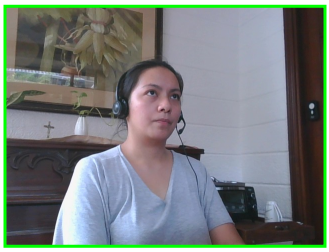


In-Test Error Detected (No Face Detected)

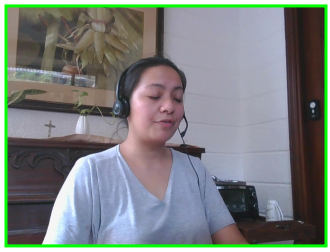
In-Test Photo



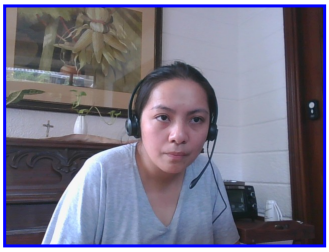
In-Test Photo



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: 20818-1, Key: 0-0, Rpt: 68, Prd: 9640, Created: 2026-06-28 12:59 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 (%)
AJAX Requests and Response Handling	97.8892	Numeric Score	97.8892	8.3333
AJAX Requests and Response Handling (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
DOM Manipulation: Creating, Adding, and Removing Elements	67.0777	Numeric Score	67.0777	8.3333
DOM Manipulation: Creating, Adding, and Removing Elements (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
DOM Selection and Manipulation	83.6226	Numeric Score	83.6226	8.3333
DOM Selection and Manipulation (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Document Ready and Script Initialization	88.2329	Numeric Score	88.2329	8.3333
Document Ready and Script Initialization (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Effects and Animations	71.7844	Numeric Score	71.7844	8.3333
Effects and Animations (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Event Handling	78.8791	Numeric Score	78.8791	8.3333
Event Handling (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Weighted Average:				72.1130
Final Overall Score:				72

Notes

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