hr avatar
Illuminating Talent

Test Results and Interview Guide

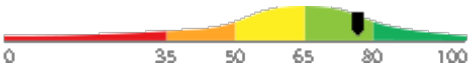
Candidate: **Elizabeth Wantsajob**
Assessment: PHP Programming
Completed: June 27, 2026
Prepared for: Sara Maple
Example Company

What's Included

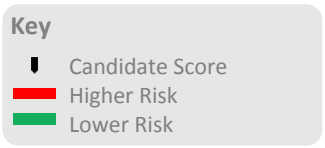
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

Important Note: The PHP Programming 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.

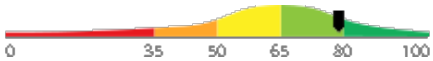
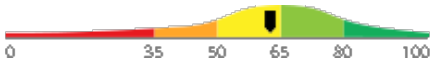
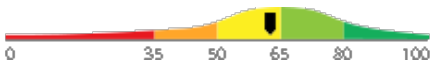
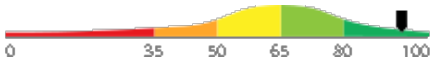
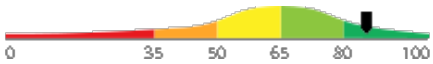
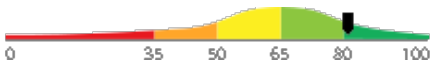
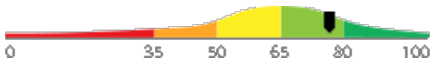
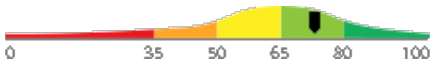
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com PHP Programming June 27, 2026	77	

The candidate exhibits a solid working knowledge of PHP programming, including syntax, design patterns, and core web application development tasks such as session management, database operations, and exception handling. Competency in object-oriented programming, data validation, and output rendering is well established, with only minor gaps expected in more advanced or specialized areas. This individual is well suited for entry-level to mid-level PHP development responsibilities with minimal onboarding required.



Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
Array Manipulation and String Processing	78	
Object-Oriented Programming (OOP) in PHP (Coding Tasks)	62	
PHP Syntax and Core Language Constructs (Coding Tasks)	62	
Database Operations with PDO	94	
Error Handling, Exceptions, and Debugging	85	
Form Handling, Sessions, and Cookies	81	
Object-Oriented Programming (OOP) in PHP	76	
PHP Syntax and Core Language Constructs	73	

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	77th
North America	63rd
United States	63rd
Example Company	70th

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:	PHP Programming
Authorized:	June 27, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	June 27, 2026, 1:20:58PM EDT
Completed:	June 27, 2026, 1:20:58PM EDT
Overall Score:	77

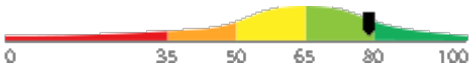
Knowledge and Skills Detail

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

Array Manipulation and String Processing

Score: 78



Description:

Covers the use of PHP's built-in array and string functions to store, search, sort, transform, and format data. These operations are among the most frequently performed tasks in PHP scripts, particularly when processing user input, preparing data for output, or working with database results.

Interpretation:

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

The candidate shows a solid and broad command of PHP's built-in array and string functions, and can reliably apply them to store, search, sort, transform, and format data in typical scripting scenarios. Minor gaps in knowledge of advanced or less commonly used functions may exist, but overall proficiency is strong.

Describe how you would extract, validate, and reformat a date string submitted by a user in an unexpected format before saving it to a database.

1	2	3	4	5
Offers no structured approach or cannot name relevant string or date functions.		Mentions string or date functions but skips validation or error handling steps.		Describes a complete pipeline using functions like strtotime(), date(), and validation checks with clear reasoning.

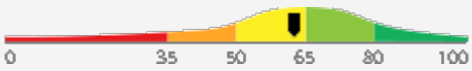
If you had an array of customer names and needed to remove duplicates and then sort them alphabetically, which PHP functions would you use and in what order?

1	2	3	4	5
Cannot name the relevant functions or describes an incorrect sequence.		Identifies one correct function but is unsure about the full sequence or syntax.		Correctly identifies array_unique() and sort() or asort(), explains the order and any caveats.

Detail

Object-Oriented Programming (OOP) in PHP (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.

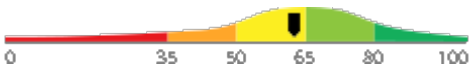
Can you explain what a pointer is in C and describe a situation where you would use one?

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

PHP Syntax and Core Language Constructs (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.

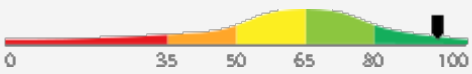
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Database Operations with PDO

Score: 94



Description:

Covers connecting to a database, executing queries, and retrieving results using PHP Data Objects (PDO), including the use of prepared statements to prevent SQL injection. Working with databases is a fundamental part of most PHP web applications, making this one of the most critical practical skills.

Interpretation:

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

The candidate demonstrates an advanced and comprehensive mastery of Database Operations with PDO in PHP, including secure and efficient use of prepared statements, query execution, and result retrieval. They possess the depth of knowledge expected of an experienced PHP developer and are well-equipped to design and implement robust database-driven web applications.

Interview Guide

Explain what a prepared statement is in PDO, why it matters for security, and walk me through how you would use one to insert a record submitted from a user form.

1	2	3	4	5
Cannot explain prepared statements or does not connect them to SQL injection prevention.		Correctly explains the concept but gives an incomplete or syntactically flawed implementation example.		Clearly explains parameterized queries, binds user input safely, and walks through prepare(), execute(), and error handling.

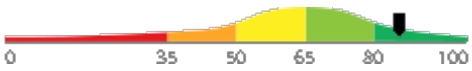
How would you connect to a MySQL database in PHP and run a simple query to retrieve all rows from a table called 'orders'?

1	2	3	4	5
Cannot describe a connection method or confuses deprecated functions like mysql_connect() with PDO.		Describes a basic PDO connection and query but omits error handling or fetching results correctly.		Demonstrates correct DSN, PDO instantiation, query execution, and fetch loop with error handling.

Detail

Error Handling, Exceptions, and Debugging

Score: 85



Description:

Covers the use of try/catch blocks, custom exception classes, error reporting settings, and logging tools to detect, handle, and record errors in PHP applications. Effective error handling and debugging are essential for maintaining reliable applications and resolving issues quickly in development and production environments.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of PHP error handling, exceptions, and debugging. They are highly proficient in designing robust error-handling architectures using custom exception classes, fine-tuned error reporting configurations, and sophisticated logging strategies, enabling them to reliably detect, handle, and resolve issues in complex PHP applications.

Interview Guide

How would you design error handling for a PHP function that queries a database and returns results, so that both database errors and unexpected data issues are caught and logged appropriately?

1	2	3	4	5
Provides no structured error handling or conflates different types of errors without distinction.		Uses a try/catch block correctly but handles only one error type or logs errors inconsistently.		Uses try/catch with specific exception types, logs errors with context, and returns a safe fallback value or response.

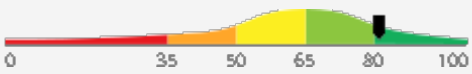
If your PHP script crashes with a fatal error on a live website, what steps would you take to find out what went wrong without exposing sensitive details to users?

1	2	3	4	5
Suggests displaying raw errors to users or has no structured approach to diagnosing the issue.		Mentions error logs or display_errors but cannot describe a complete safe debugging workflow.		Describes disabling display_errors in production, enabling error_log, checking logs, and using try/catch or a logging library.

Detail

Form Handling, Sessions, and Cookies

Score: 81



Description:

Covers the processing of HTML form submissions using superglobals, the validation and sanitization of user input, and the management of user state across requests using sessions and HTTP cookies. These tasks are core to nearly every business web application that involves user interaction.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of PHP form handling, sessions, and cookies. They are highly proficient in processing form submissions using superglobals, rigorously validating and sanitizing user input, and implementing robust session and cookie-based state management, making them well-equipped to independently lead development of complex, user-interactive business web applications.

Interview Guide

Walk me through how you would implement a basic user login system in PHP that keeps the user logged in across multiple pages without requiring them to re-enter their credentials.

1	2	3	4	5
Cannot describe session mechanics or conflates sessions with cookies incorrectly.		Describes session_start() and $_SESSION usage but omits security considerations like session regeneration.		Covers session_start(), storing user identity, session_regenerate_id(), and optionally secure cookie flags.

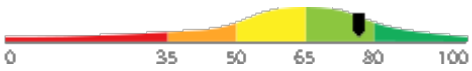
When a user submits a form on a website, how does PHP receive that data, and what is the first thing you should do with it before using it in your application?

1	2	3	4	5
Cannot identify superglobals or omits any mention of input validation or sanitization.		Correctly identifies $_POST or $_GET but gives only a vague answer about handling the data safely.		Names the correct superglobal, explains sanitization and validation, and may mention filter_input() or htmlspecialchars().

Detail	Interview Guide

Object-Oriented Programming (OOP) in PHP

Score: 76



Description:

Covers the use of classes, objects, inheritance, interfaces, traits, access modifiers, and namespaces to organize and structure PHP application code. These concepts are regularly applied when building maintainable, scalable web applications and working with modern PHP frameworks and libraries.

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 object-oriented programming in PHP, including the practical use of classes, inheritance, interfaces, access modifiers, and namespaces. They are likely capable of writing well-structured, maintainable PHP code and contributing effectively to projects built on modern PHP frameworks and libraries.

How would you use an interface and a trait differently when designing a PHP class, and when would you choose one over the other?

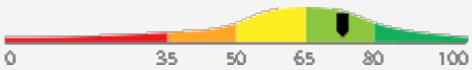
1	2	3	4	5
Cannot distinguish interfaces from traits or confuses their purposes.		Describes one correctly but struggles to articulate when to prefer one over the other.		Accurately contrasts interfaces as contracts and traits as code reuse, with a clear use-case justification.

In your own words, what is the difference between a class and an object in PHP, and can you give a simple real-world example of each?

1	2	3	4	5
Conflates class and object or cannot provide a meaningful example.		Correctly distinguishes the two but gives a vague or overly abstract example.		Clearly explains class as blueprint and object as instance, with a concrete and relevant example.

PHP Syntax and Core Language Constructs

Score: 73



Description:

Covers the foundational rules and building blocks of PHP, including variables, data types, operators, control flow statements, loops, and built-in functions. This knowledge is applied constantly when writing any PHP script and is essential for producing readable, functional code.

Interpretation:

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

The candidate exhibits a solid and well-rounded understanding of PHP programming, including object-oriented principles, database operations, session and cookie management, and error handling through exception handlers. They are likely capable of writing, debugging, and maintaining business-oriented web application scripts with moderate complexity and limited supervision.

Describe a situation where you chose a specific loop type — such as foreach, for, or while — over the others, and explain why it was the right choice for that task.

1	2	3	4	5
Cannot distinguish between loop types or gives an illogical justification.		Identifies a valid use case but offers limited reasoning about the tradeoffs.		Clearly contrasts loop types, cites a specific scenario, and explains performance or readability benefits.

Can you walk me through what happens when PHP encounters a variable that has not been defined yet, and how would you handle that situation in your code?

1	2	3	4	5
Cannot explain undefined variables or offers no handling strategy.		Mentions a warning or error but provides only a vague handling approach.		Clearly explains the notice/warning, uses isset() or null coalescing, gives a concrete 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.

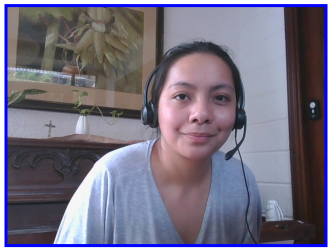
Question or Task	Response

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

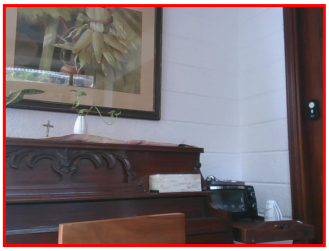
The following photos of the candidate and any identification were uploaded during the assessment session.

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%)

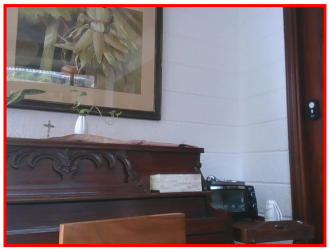


Pre/Post-Test Photo

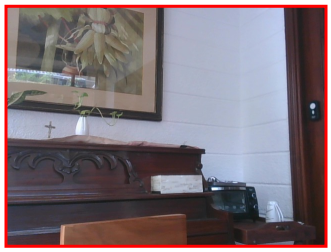
ID Photo



In-Test Error Detected (No Face Detected)

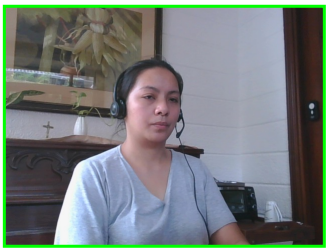


In-Test Error Detected (No Face Detected)

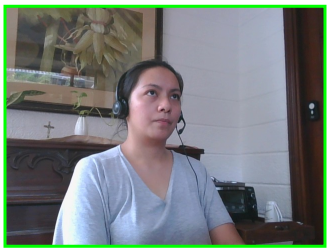


In-Test Error Detected (No Face Detected)

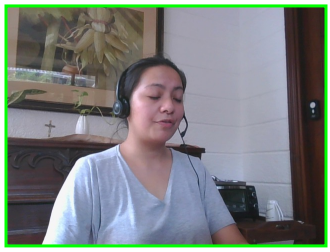
In-Test Photo



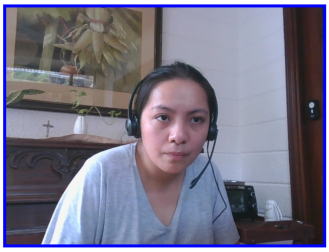
In-Test Photo



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: 20784-1, Key: 0-0, Rpt: 68, Prd: 9606, 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 (%)
Array Manipulation and String Processing	78.9543	Numeric Score	78.9543	12.5000
Database Operations with PDO	94.0525	Numeric Score	94.0525	12.5000
Error Handling, Exceptions, and Debugging	85.7133	Numeric Score	85.7133	12.5000
Form Handling, Sessions, and Cookies	81.2754	Numeric Score	81.2754	12.5000
Object-Oriented Programming (OOP) in PHP	76.8398	Numeric Score	76.8398	12.5000
Object-Oriented Programming (OOP) in PHP (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
PHP Syntax and Core Language Constructs	73.6581	Numeric Score	73.6581	12.5000
PHP Syntax and Core Language Constructs (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
Weighted Average:				77.0563
Final Overall Score:				77

Notes

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