hr | avatar
Illuminating Talent

Test Results and Interview Guide

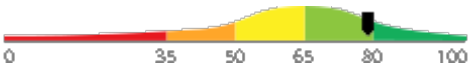
Candidate: **Elizabeth Wantsajob**
Assessment: COBOL Programming
Completed: July 3, 2026
Prepared for: Sara Maple
Example Company

What's Included

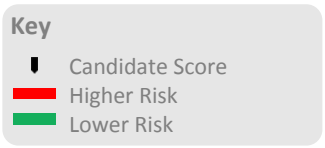
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

Important Note: The COBOL 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.

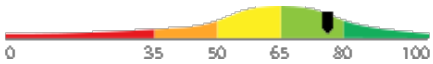
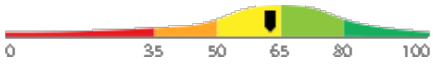
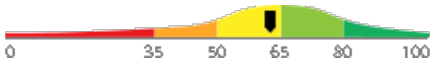
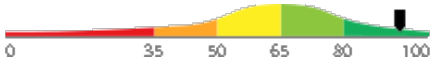
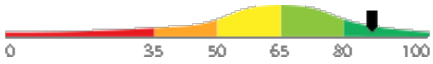
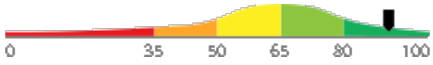
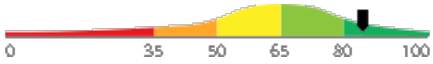
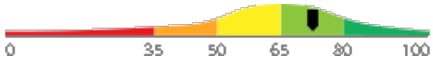
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com COBOL Programming July 3, 2026	79	

The candidate demonstrates a solid and competent understanding of COBOL programming across most core subject areas, including program structure, file handling, arithmetic and conditional logic, copybooks, and basic debugging techniques. Some proficiency gaps may exist in more specialized or advanced topics, but the candidate is well-positioned to contribute effectively to COBOL development work with moderate supervision.



Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
COBOL Program Structure and Divisions	76	
COBOL Program Structure and Divisions (Coding Tasks)	62	
Data Definition and the Data Division (Coding Tasks)	62	
Data Definition and the Data Division	93	
Debugging, Copybooks, and COPY Statements	87	
File Handling	91	
Procedure Division Logic: Arithmetic, Conditionals, and PERFORM	85	
Table Handling with OCCURS, Subscripts, and SEARCH	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	0 10 20 30 40 50 60 70 80 90 100
Global	79th	
North America	65th	
United States	65th	
Example Company	72nd	

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:	COBOL Programming
Authorized:	July 3, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	July 3, 2026, 8:26:43PM EDT
Completed:	July 3, 2026, 8:26:43PM EDT
Overall Score:	79

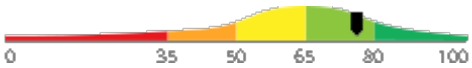
Knowledge and Skills Detail

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

COBOL Program Structure and Divisions

Score: 76



Description:

COBOL programs are organized into four divisions: Identification, Environment, Data, and Procedure. Understanding how each division is structured and what it contains is fundamental to writing and reading any COBOL program. This knowledge is applied every time a programmer creates or modifies a COBOL program.

Interpretation:

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

The candidate demonstrates a solid working knowledge of COBOL programming, including program structure, file handling, arithmetic and conditional logic, and core data processing techniques. While competent across most entry-level to mid-level concepts, there may be occasional gaps in specialized areas such as advanced table handling, string manipulation, report generation, or JCL integration that could be addressed through continued experience.

Walk me through how you would set up the basic structure of a new COBOL program that reads from an input file and writes to an output file. What would you include in each division and why?

1	2	3	4	5
Provides a disorganized or incomplete structure, missing key entries in one or more divisions.		Describes a mostly correct structure but omits some required entries or provides limited justification.		Clearly and accurately describes all required entries across all four divisions with confident, specific reasoning.

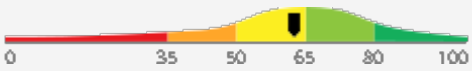
Can you name the four divisions of a COBOL program and briefly describe what each one is used for?

1	2	3	4	5
Cannot name all four divisions or confuses their purposes.		Names all four divisions but gives vague or incomplete descriptions of their purposes.		Clearly names all four divisions and accurately explains the role of each with specific examples.

Detail

COBOL Program Structure and Divisions (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.

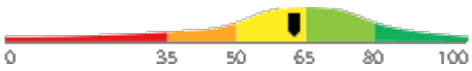
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Data Definition and the Data Division (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.

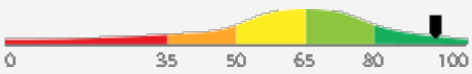
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Data Definition and the Data Division

Score: 93



Description:

The Data Division is where all data items, records, and working storage variables are defined using level numbers, PICTURE clauses, and USAGE clauses. Correctly defining data is essential for storing, processing, and displaying information in any COBOL program. Programmers use these definitions in nearly every program they write or maintain.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of COBOL Data Definition and the Data Division. They are highly proficient in accurately defining data items, records, and working storage variables, and can be expected to write and maintain correct, well-structured data definitions across a wide range of COBOL programs.

Interview Guide

Explain how you would use level numbers to define a group item and its subordinate elementary items in the Working-Storage Section. Why does this structure matter when processing data?

1	2	3	4	5
Confuses group and elementary items or provides an incorrect level number hierarchy.		Correctly describes the hierarchy but gives a limited explanation of why the structure is important.		Accurately defines the hierarchy with a clear example and explains practical benefits such as group moves and record alignment.

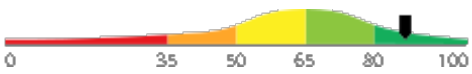
What is a PICTURE clause in COBOL, and can you give an example of how you would define a numeric field and an alphanumeric field?

1	2	3	4	5
Cannot explain PICTURE clauses or provides incorrect syntax examples.		Explains PICTURE clauses correctly but gives only one example or makes minor syntax errors.		Clearly explains PICTURE clauses and provides accurate, distinct examples for both numeric and alphanumeric fields.

Detail

Debugging, Copybooks, and COPY Statements

Score: 87



Description:

COBOL programmers use COPY statements to include reusable code and data definitions (copybooks) into programs, reducing duplication and improving consistency. Debugging involves interpreting compiler error messages and using techniques like DISPLAY statements to trace and fix program errors. Both skills are applied regularly when writing new programs and maintaining existing ones.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of COBOL debugging, copybooks, and COPY statements. They are well-equipped to proficiently leverage reusable code structures, diagnose and resolve complex program errors, and apply best practices consistently across both new development and legacy program maintenance.

Interview Guide

Walk me through how you would approach debugging a COBOL program that is producing incorrect output. What techniques or tools would you use and why?

1	2	3	4	5
Describes only one vague technique or cannot explain a structured approach to finding the problem.		Describes a reasonable approach using DISPLAY statements or compiler messages but lacks depth or structure.		Describes a clear, structured debugging process including compiler error review, DISPLAY tracing, data inspection, and logical verification.

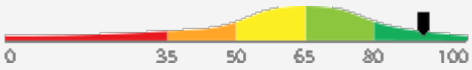
What is a copybook in COBOL and how do you include one in your program? Why would you use one instead of typing the same code directly into the program?

1	2	3	4	5
Cannot explain what a copybook is or how the COPY statement works.		Correctly explains copybooks and the COPY statement but gives a limited reason for using them.		Clearly explains copybooks, the COPY statement, and practical benefits like consistency, reuse, and easier maintenance.

Detail

File Handling

Score: 91



Description:

File handling covers how COBOL programs open, read, write, update, and close files, including sequential, indexed, and relative file organizations. It involves using SELECT statements, FD entries, and verbs like READ, WRITE, REWRITE, and DELETE. File handling is a core skill because most business COBOL programs process data stored in files.

Interpretation:

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

The candidate demonstrates a comprehensive and advanced mastery of COBOL file handling concepts, including sequential, indexed, and relative file organizations, as well as the full range of file control and processing constructs. They are well-equipped to independently design, develop, and maintain complex file-based COBOL programs in a professional business environment. This level of proficiency reflects a strong command of both foundational principles and advanced file handling techniques.

Interview Guide

What is the difference between sequential and indexed file organization in COBOL, and when would you choose one over the other in a business application?

1	2	3	4	5
Cannot clearly distinguish between the two or provides incorrect descriptions of how each works.		Correctly distinguishes between the two but gives a limited or generic explanation of when to use each.		Clearly explains both organizations, including key concepts like RECORD KEY for indexed files, and gives practical, well-reasoned use cases.

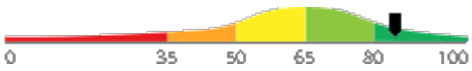
What steps are needed to read records from a sequential file in COBOL? What statements would you use and in what order?

1	2	3	4	5
Cannot describe the required steps or omits critical statements like OPEN or AT END handling.		Describes most steps correctly but misses error handling or the CLOSE statement.		Accurately describes all steps in order, including OPEN, READ with AT END, loop processing, and CLOSE, with clear reasoning.

Detail

Procedure Division Logic: Arithmetic, Conditionals, and PERFORM

Score: 85



Description:

The Procedure Division contains the executable logic of a COBOL program, including arithmetic operations (ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPUTE), conditional statements (IF, EVALUATE), and loop control using PERFORM. These constructs are used constantly to process data, make decisions, and control program flow in everyday business programs.

Interpretation:

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

The candidate demonstrates an advanced and comprehensive mastery of COBOL Procedure Division logic, including the full range of arithmetic operations, conditional statements, and loop control constructs. They are well-equipped to independently design, develop, and maintain complex COBOL business programs, making sound decisions regarding program flow and data processing logic.

Interview Guide

Describe how you would use a PERFORM loop to process all records in a table. How would you control when the loop starts and stops?

1	2	3	4	5
Cannot describe a working PERFORM loop or confuses PERFORM UNTIL with other constructs.		Describes a working loop but provides limited detail on condition control or loop termination.		Accurately describes PERFORM UNTIL or PERFORM VARYING with clear explanation of initialization, condition testing, and termination.

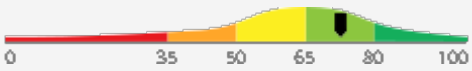
How would you use an IF statement in COBOL to check whether a numeric field is greater than zero and perform different actions based on the result?

1	2	3	4	5
Cannot write a correct IF statement or confuses syntax with other languages.		Writes a mostly correct IF statement but makes minor syntax errors or omits END-IF.		Writes a syntactically correct IF/ELSE/END-IF statement and explains the logic clearly and confidently.

Detail

Table Handling with OCCURS, Subscripts, and SEARCH

Score: 73



Description:

COBOL supports tables — arrays of data items — defined using the OCCURS clause, accessed with subscripts or indexes, and searched using SEARCH or SEARCH ALL statements. Table handling is widely used in business programs to process lists of codes, rates, records, and other repeated data structures.

Interpretation:

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

The candidate exhibits a solid and proficient understanding of COBOL table handling, including defining tables with the OCCURS clause, accessing elements via subscripts or indexes, and applying SEARCH and SEARCH ALL statements. They are well-equipped to handle most business programming tasks involving repeated data structures with minimal oversight.

Interview Guide

What is the difference between SEARCH and SEARCH ALL in COBOL, and what conditions must be met before you can use SEARCH ALL?

1	2	3	4	5
Cannot distinguish between SEARCH and SEARCH ALL or is unaware of the sorted table requirement.		Correctly distinguishes the two but gives an incomplete explanation of the ASCENDING/DESCENDING KEY requirement.		Accurately explains both statements, the binary search nature of SEARCH ALL, and the requirement for a sorted table with a defined KEY.

What is the OCCURS clause in COBOL and how would you access a specific element in a table defined with OCCURS?

1	2	3	4	5
Cannot explain OCCURS or does not know how to use a subscript to access a table element.		Correctly explains OCCURS and subscript access but struggles to provide a working syntax example.		Clearly explains OCCURS, provides correct subscript syntax, and distinguishes between subscripts and indexes.

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.

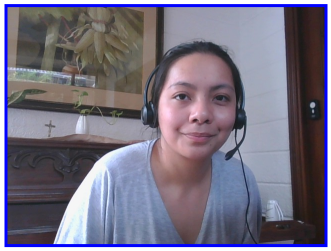
Question or Task	Response

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

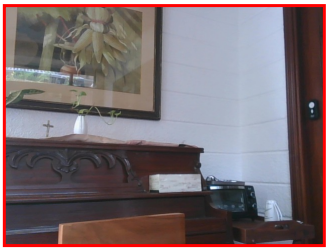
The following photos of the candidate and any identification were uploaded during the assessment session.

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%)

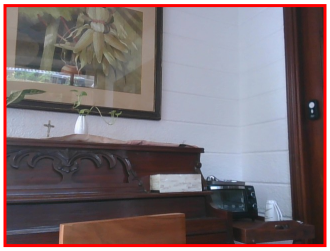


Pre/Post-Test Photo

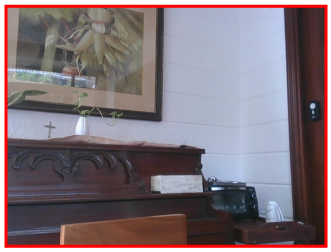
ID Photo



In-Test Error Detected (No Face Detected)

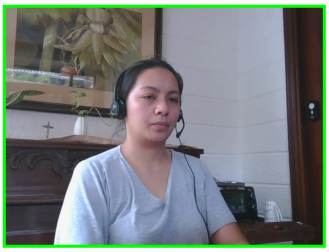


In-Test Error Detected (No Face Detected)

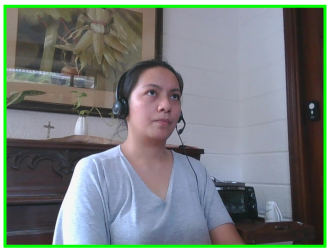


In-Test Error Detected (No Face Detected)

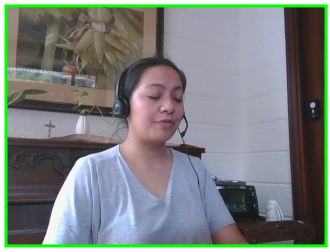
In-Test Photo



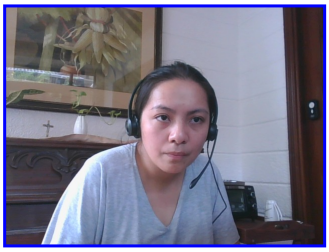
In-Test Photo



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: 20780-1, Key: 0-0, Rpt: 68, Prd: 9602, Created: 2026-07-03 20:26 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 (%)
COBOL Program Structure and Divisions	76.3578	Numeric Score	76.3578	12.5000
COBOL Program Structure and Divisions (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
Data Definition and the Data Division	93.4962	Numeric Score	93.4962	12.5000
Data Definition and the Data Division (Coding Tasks)	62.9784	Numeric Score	62.9784	12.5000
Debugging, Copybooks, and COPY Statements	87.1469	Numeric Score	87.1469	12.5000
File Handling	91.0974	Numeric Score	91.0974	12.5000
Procedure Division Logic: Arithmetic, Conditionals, and PERFORM	85.0816	Numeric Score	85.0816	12.5000
Table Handling with OCCURS, Subscripts, and SEARCH	73.1863	Numeric Score	73.1863	12.5000
Weighted Average:				79.0404
Final Overall Score:				79

Notes

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