hr avatar
Illuminating Talent

Test Results and Interview Guide

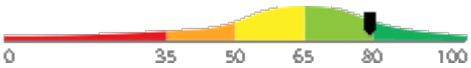
Candidate: **Elizabeth Wantsajob**
Assessment: Unity Development (Short)
Completed: July 1, 2026
Prepared for: Sara Maple
Example Company

What's Included

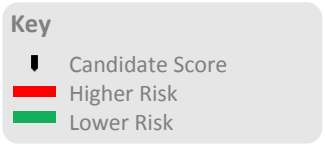
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

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

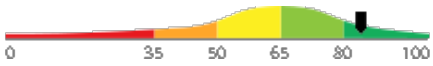
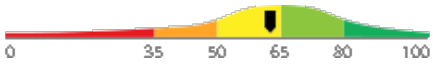
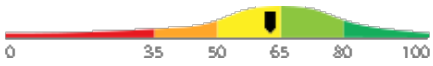
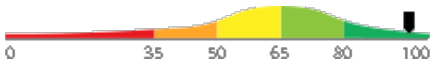
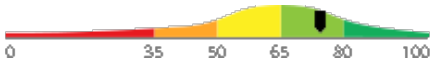
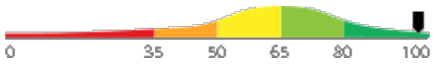
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com Unity Development (Short) July 1, 2026	79	

The candidate demonstrates a solid working knowledge of Unity game development, including proficiency with the Unity interface, scene management, C# scripting, physics components, and asset workflows. They are likely capable of contributing effectively to Unity projects, though some advanced or specialized areas may benefit from further development.



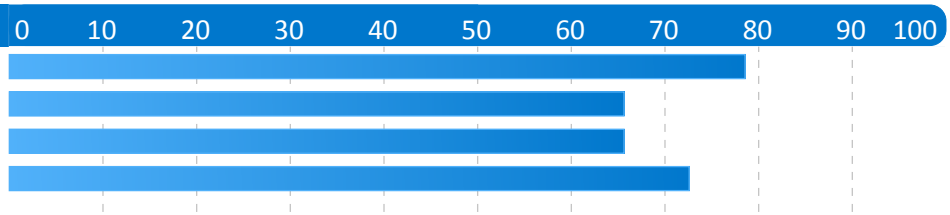
Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
C# Scripting and MonoBehaviour	84	
C# Scripting and MonoBehaviour (Coding Tasks)	62	
GameObjects, Components, and the Unity Interface (Coding Tasks)	62	
GameObjects, Components, and the Unity Interface	95	
Physics: Rigidbody and Colliders	74	
Scene Management and Build Deployment	98	

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	79th
North America	66th
United States	66th
Example Company	73rd



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:	Unity Development (Short)
Authorized:	July 1, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	July 1, 2026, 7:56:42PM EDT
Completed:	July 1, 2026, 7:56:42PM EDT
Overall Score:	79

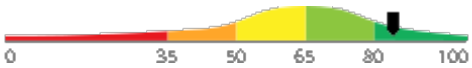
Knowledge and Skills Detail

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

C# Scripting and MonoBehaviour

Score: 84



Description:

Covers writing C# scripts in Unity to control the behavior of GameObjects. This includes understanding the MonoBehaviour lifecycle methods such as Start, Update, and FixedUpdate, and how scripts are attached to and interact with GameObjects and their components.

Interpretation:

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

The candidate exhibits a comprehensive and advanced understanding of Unity C# scripting and the MonoBehaviour lifecycle. They are highly proficient in attaching and managing scripts, controlling GameObject behavior, and leveraging component interactions effectively and efficiently.

How would you use a C# script in Unity to move a GameObject smoothly over time, and why would you use FixedUpdate instead of Update for physics-based movement?

1	2	3	4	5
Cannot describe how to move a GameObject via script or does not know the difference between Update and FixedUpdate.		Describes basic movement using Transform.Translate or similar, and knows FixedUpdate is for physics but lacks detail.		Explains Time.deltaTime for smooth movement, correctly explains FixedUpdate's fixed timestep and its importance for Rigidbody physics.

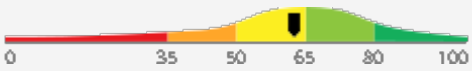
What is MonoBehaviour in Unity, and can you explain what the Start and Update methods are used for?

1	2	3	4	5
Cannot explain MonoBehaviour or confuses the purpose of Start and Update.		Correctly identifies Start as running once at initialization and Update as running every frame, but with limited detail.		Clearly explains MonoBehaviour lifecycle, distinguishes Start from Awake, and explains Update vs. FixedUpdate with use cases.

Detail

C# Scripting and MonoBehaviour (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.

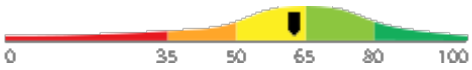
Can you explain what a pointer is in C and describe a situation where you would use one?

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

GameObjects, Components, and the Unity Interface (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.

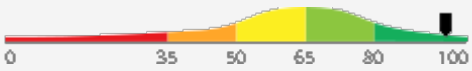
Can you explain what a pointer is in C and describe a situation where you would use one?

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

GameObjects, Components, and the Unity Interface

Score: 95



Description:

Covers the core Unity editor windows (Scene, Game, Hierarchy, Inspector, Project) and how to create, organize, and manage GameObjects and their components within a scene. This includes working with prefabs, Tags, Layers, and the Transform component to control position, rotation, and scale.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of Unity game development, spanning the full breadth of the subject including scripting, physics, UI systems, audio, camera control, asset management, and platform deployment. They are well-equipped to independently lead or contribute at a high level to complex Unity projects with minimal oversight.

Interview Guide

How would you use prefabs in Unity, and what are the advantages of using them when building a scene with many repeated objects?

1	2	3	4	5
Cannot define prefabs or gives an incorrect explanation of their purpose.		Correctly defines prefabs and mentions reusability but does not elaborate on workflow advantages.		Explains prefabs thoroughly, including instancing, editing prefab assets, and benefits for consistency and efficiency.

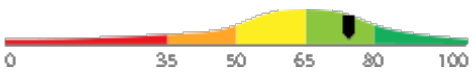
Can you walk me through what a GameObject is in Unity and give an example of how you would add and configure a component on it?

1	2	3	4	5
Vague or incorrect description of GameObjects; cannot explain components or how to add them.		Correctly describes GameObjects and components but gives a basic or incomplete example.		Clearly explains GameObjects and components, gives a concrete example, and mentions the Inspector or prefabs.

Detail

Physics: Rigidbody and Colliders

Score: 74



Description:

Covers Unity's physics system, including how to use Rigidbody components to apply physics-based movement and forces to GameObjects, and how to use Collider components to detect and respond to collisions and triggers between objects.

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 Unity's Rigidbody and Collider systems, and is capable of implementing physics-based movement, forces, and collision detection effectively in most scenarios. Minor gaps in knowledge may exist in advanced or edge-case physics configurations.

Interview Guide

What is the difference between a Collider used as a trigger versus a standard collider in Unity, and when would you use each?

1	2	3	4	5
Cannot distinguish between trigger and standard colliders or confuses the two.		Correctly identifies that triggers do not cause physical collisions and use OnTriggerEnter, but gives a limited use case.		Clearly explains both, names the relevant callback methods (OnCollisionEnter vs. OnTriggerEnter), and gives practical examples for each.

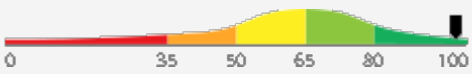
What is a Rigidbody component in Unity, and what does adding one to a GameObject do?

1	2	3	4	5
Cannot explain Rigidbody or incorrectly describes its function.		Correctly states that Rigidbody enables physics simulation but does not elaborate on forces, gravity, or collisions.		Explains Rigidbody's role in physics simulation, mentions gravity, forces, and the need for a Collider for collision detection.

Detail

Scene Management and Build Deployment

Score: 98



Description:

Covers how to manage multiple scenes in Unity, including loading and transitioning between scenes using the SceneManager. Also includes how to configure build settings and deploy a finished project to a target platform such as PC, mobile, or console.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of Unity scene management and build deployment. They are highly proficient in managing multiple scenes, implementing scene transitions using the SceneManager, and confidently configuring and deploying finished projects across a range of target platforms including PC, mobile, and console.

Interview Guide

Can you walk me through the steps you would take to build and deploy a Unity project to a specific platform, such as Android or PC?

1	2	3	4	5
Cannot describe the build process or is unaware of platform-specific settings.		Describes opening Build Settings and selecting a platform but misses key steps like switching platform or player settings.		Describes the full process: opening Build Settings, switching platform, configuring Player Settings, and building the project.

How would you load a different scene in Unity from a script, and what do you need to set up beforehand to make that work?

1	2	3	4	5
Cannot name SceneManager or describe how scenes are loaded; unaware of build settings requirement.		Mentions SceneManager.LoadScene but is unaware that scenes must be added to Build Settings first.		Correctly explains SceneManager.LoadScene, the need to add scenes to Build Settings, and may mention async loading.

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.

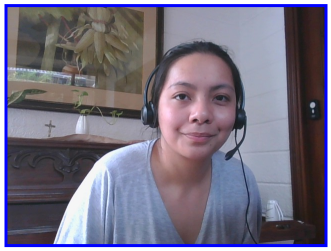
Question or Task	Response

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

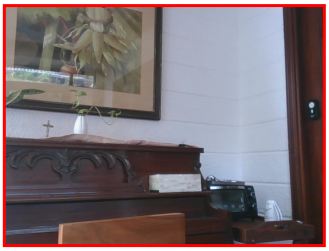
The following photos of the candidate and any identification were uploaded during the assessment session.

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%)

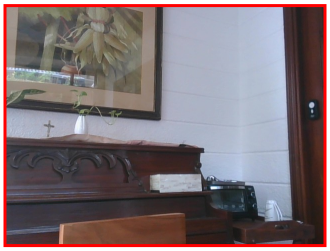


Pre/Post-Test Photo

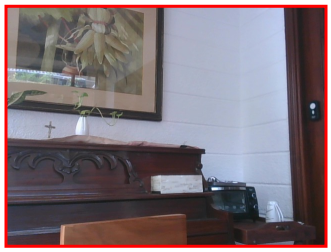
ID Photo



In-Test Error Detected (No Face Detected)

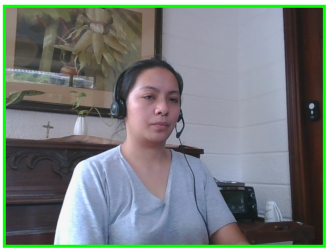


In-Test Error Detected (No Face Detected)

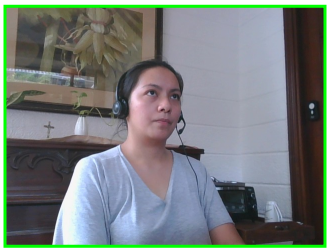


In-Test Error Detected (No Face Detected)

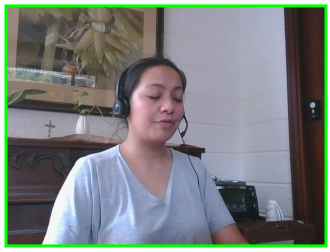
In-Test Photo



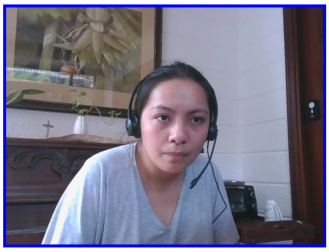
In-Test Photo



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: 20886-1, Key: 0-0, Rpt: 68, Prd: 9706, Created: 2026-07-01 19:56 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 (%)
C# Scripting and MonoBehaviour	84.3651	Numeric Score	84.3651	16.6667
C# Scripting and MonoBehaviour (Coding Tasks)	62.9784	Numeric Score	62.9784	16.6667
GameObjects, Components, and the Unity Interface	95.6178	Numeric Score	95.6178	16.6667
GameObjects, Components, and the Unity Interface (Coding Tasks)	62.9784	Numeric Score	62.9784	16.6667
Physics: Rigidbody and Colliders	74.8579	Numeric Score	74.8579	16.6667
Scene Management and Build Deployment	98.0246	Numeric Score	98.0246	16.6667
Weighted Average:				79.8037
Final Overall Score:				79

Notes

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