hr | avatar
Illuminating Talent

Test Results and Interview Guide

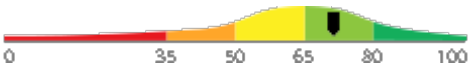
Candidate: **Elizabeth Wantsajob**
Assessment: Java Spring Boot
Completed: June 28, 2026
Prepared for: Sara Maple
Example Company

What's Included

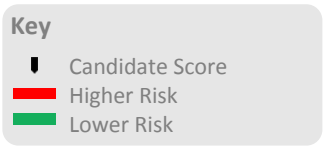
- Overall Score
- Competency Summary Table
- Comparison Matrix
- Detailed Competency Results with Interview Guide

Important Note: The Java Spring Boot 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.

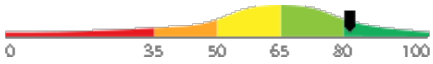
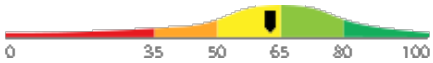
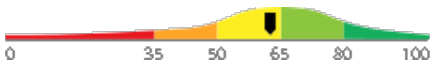
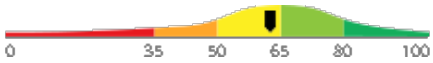
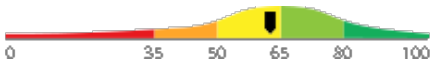
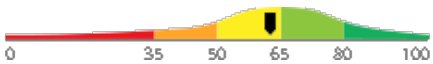
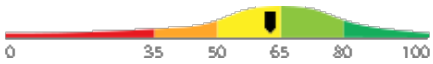
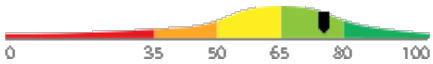
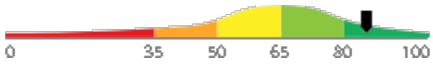
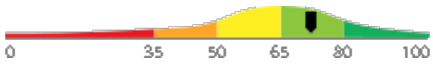
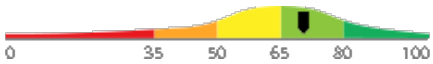
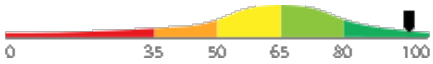
Overall

Candidate	Score	Interpretation
Elizabeth Wantsajob beth.wantsajob@gmail.com Java Spring Boot June 28, 2026	71	

The candidate demonstrates a solid and well-rounded understanding of Java Spring Boot development, including application structure, Spring MVC, Spring Data JPA, configuration management, and unit testing. They are likely capable of independently developing, debugging, and maintaining Spring Boot web applications with moderate complexity. Some refinement may be needed in specialized areas such as Spring Security, Actuator monitoring, or advanced bean lifecycle management.



Competency Summary

Competency	Score	Interpretation
Skills/Knowledge (relates to immediate readiness)		
Application Configuration	81	
Application Configuration (Coding Tasks)	62	
Exception Handling and Request Validation (Coding Tasks)	62	
REST Controllers and HTTP Request Handling (Coding Tasks)	62	
Spring Boot Core Concepts and Application Structure (Coding Tasks)	62	
Spring Data JPA and Database Operations (Coding Tasks)	62	
Testing Spring Boot Applications (Coding Tasks)	62	
Exception Handling and Request Validation	75	
REST Controllers and HTTP Request Handling	85	
Spring Boot Core Concepts and Application Structure	72	
Spring Data JPA and Database Operations	70	
Testing Spring Boot Applications	95	

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	71st
North America	59th
United States	59th
Example Company	65th

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:	Java Spring Boot
Authorized:	June 28, 2026, by Sara Maple, Example Company, qamailsaram.mike@hravatar.com
Started:	June 28, 2026, 1:00:59PM EDT
Completed:	June 28, 2026, 1:00:59PM EDT
Overall Score:	71

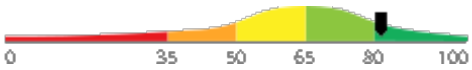
Knowledge and Skills Detail

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

Application Configuration

Score: 81



Description:

Covers configuring a Spring Boot application using application.properties or application.yml files, including setting server ports, datasource connections, logging levels, and custom application properties. Includes using @Value and @ConfigurationProperties to read configuration values in code.

Interpretation:

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

The candidate exhibits an advanced and comprehensive mastery of Spring Boot Application Configuration. They are highly proficient in leveraging both configuration file formats and annotation-based approaches to define, organize, and inject configuration values, demonstrating the expertise expected of a seasoned Spring Boot developer.

How would you define a custom configuration property in application.properties and then read its value inside a Spring Boot component or service class?

1	2	3	4	5
Cannot describe how to define or read a custom property in code.		Correctly uses @Value to inject a property value but is unaware of @ConfigurationProperties.		Explains both @Value and @ConfigurationProperties, describes binding a group of properties to a class, and mentions validation support.

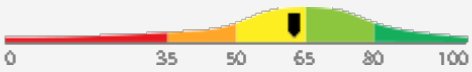
Where would you go in a Spring Boot project to change the port the application runs on, and what would you write to make that change?

1	2	3	4	5
Does not know the configuration file location or the correct property key.		Correctly identifies application.properties and the server.port property but may be unsure about .yml syntax.		Correctly identifies both application.properties and application.yml, provides the exact syntax for each, and mentions profile-specific config files.

Detail

Application Configuration (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.

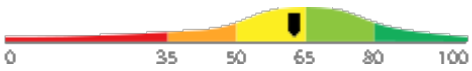
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Exception Handling and Request Validation (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.

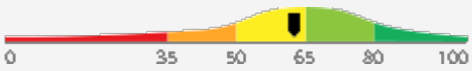
Can you explain what a pointer is in C and describe a situation where you would use one?

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

REST Controllers and HTTP Request Handling (Coding Tasks)

Score: 62



Description:

Covers the use of pointers to reference and manipulate memory addresses, along with dynamic memory allocation and deallocation using malloc, calloc, realloc, and free. Includes pointer arithmetic, dereferencing, and avoiding common issues like memory leaks and dangling pointers.

Interpretation:

The candidate exhibits average writing skills, which can hinder high performance in some jobs.

The candidate possesses a moderate working knowledge of C programming, demonstrating familiarity with core concepts including data types, control flow, functions, and basic file I/O. They may require some guidance when working with more advanced topics such as dynamic memory allocation, modular design, or debugging complex logic.

Overall AI Score:	65.0
Lines of Code:	15.0
Syntax Errors:	5.0
AI Confidence Level:	50
Match with Ideal Response (AI):	30.0
Structure:	50.0
Syntax:	30.0

Please see below to view the essay submitted.

Interview Guide

Walk me through how you would dynamically allocate memory for an array of 10 integers, use it, and then properly release it. What issues might arise if you don't follow best practices?

1	2	3	4	5
Cannot write correct allocation code; unaware of free() or memory leak risks.		Writes mostly correct malloc/free code; identifies memory leaks but misses other risks.		Correct malloc, use, and free; identifies leaks, dangling pointers, and NULL check on allocation.

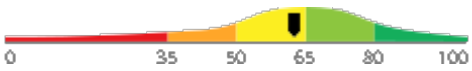
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Detail

Spring Boot Core Concepts and Application Structure (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.

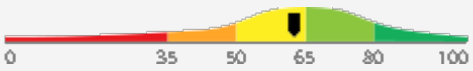
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Spring Data JPA and Database Operations (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.

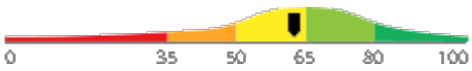
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Testing Spring Boot Applications (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.

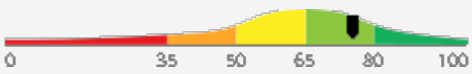
Can you explain what a pointer is in C and describe a situation where you would use one?

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

Exception Handling and Request Validation

Score: 75



Description:

Covers handling errors gracefully in a Spring Boot application using @ControllerAdvice and @ExceptionHandler to return meaningful error responses. Also includes validating incoming request data using annotations such as @Valid, @NotNull, and @Size from the Bean Validation API.

Interpretation:

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

The candidate demonstrates a solid and largely proficient understanding of exception handling and request validation in Spring Boot. They can reliably apply @ControllerAdvice, @ExceptionHandler, and Bean Validation annotations to build well-structured, robust applications, with only minor gaps in advanced usage.

Interview Guide

How would you ensure that a request body sent to a REST endpoint is validated, and what needs to be in place for validation errors to be returned to the client?

1	2	3	4	5
Cannot name the annotations used for validation or how to trigger validation in a controller.		Correctly uses @Valid on the method parameter and adds constraints like @NotNull but may not handle the MethodArgumentNotValidException.		Explains @Valid, constraint annotations, and handles MethodArgumentNotValidException in a @ControllerAdvice with a structured error response.

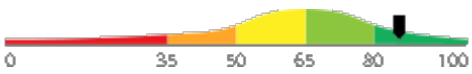
What happens in a Spring Boot application when an unhandled exception is thrown during a request, and what is one way you could customize the error response returned to the client?

1	2	3	4	5
Cannot describe the default error behavior or any method for customizing error responses.		Knows a default error response is returned and mentions @ControllerAdvice but cannot explain how to implement it.		Describes the default error page, correctly explains @ControllerAdvice with @ExceptionHandler, and gives an example of a custom error response.

Detail

REST Controllers and HTTP Request Handling

Score: 85



Description:

Covers building RESTful web endpoints using Spring MVC annotations such as @RestController, @GetMapping, @PostMapping, @RequestBody, @PathVariable, and @RequestParam. Includes returning appropriate HTTP responses and status codes.

Interpretation:

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

The candidate exhibits a strong and comprehensive mastery of REST Controllers and HTTP request handling within Spring Boot. They are highly proficient in applying Spring MVC annotations to design and implement well-structured RESTful endpoints, including precise handling of request data and the correct use of HTTP responses and status codes.

Interview Guide

Walk me through how you would create a REST endpoint that accepts a JSON request body, processes it, and returns a response with a specific HTTP status code.

1	2	3	4	5
Cannot name the correct annotations or describes an incomplete or incorrect implementation.		Correctly uses @PostMapping and @RequestBody and returns a ResponseEntity with a status code.		Provides a complete, accurate example including validation, ResponseEntity usage, and appropriate status codes like 201 or 400.

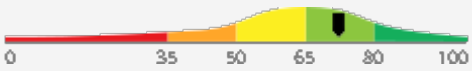
What is the difference between @Controller and @RestController in Spring Boot, and when would you use each one?

1	2	3	4	5
Cannot distinguish between the two or states they are interchangeable.		Knows @RestController combines @Controller and @ResponseBody but cannot fully explain the practical difference.		Clearly explains the difference, mentions @ResponseBody, and gives a correct use case for each annotation.

Detail

Spring Boot Core Concepts and Application Structure

Score: 72



Description:

Covers how a Spring Boot application is structured, including the main application class, auto-configuration, component scanning, dependency injection, and bean management. Includes understanding of how Spring Boot bootstraps an application and manages the application context.

Interpretation:

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

The candidate demonstrates a solid and broadly competent understanding of Java Spring Boot development, covering most key areas including application structure, dependency injection, REST API development, data persistence, and application configuration. They are likely capable of contributing meaningfully to Spring Boot projects at an entry to mid level, though some advanced or specialized topics such as Spring Security, Actuator, or complex testing scenarios may benefit from further refinement. Overall, this candidate shows a strong foundation suitable for productive work on Spring Boot-based applications.

Interview Guide

Explain the concept of dependency injection in Spring Boot and describe two different ways you can inject a dependency into a class.

1	2	3	4	5
Cannot define dependency injection or describes only one method vaguely.		Correctly defines dependency injection and names constructor and field injection with basic examples.		Explains all three injection types, discusses why constructor injection is preferred, and mentions testability benefits.

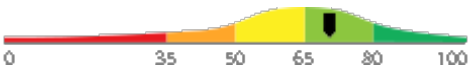
Can you describe what the @SpringBootApplication annotation does and why it is placed on the main class of a Spring Boot application?

1	2	3	4	5
Cannot explain the annotation or confuses it with unrelated concepts.		Knows it enables auto-configuration and component scanning but cannot explain all three composed annotations.		Accurately explains @EnableAutoConfiguration, @ComponentScan, and @Configuration and their combined effect.

Detail

Spring Data JPA and Database Operations

Score: 70



Description:

Covers using Spring Data JPA to interact with relational databases, including defining entity classes with @Entity, creating repository interfaces that extend JpaRepository, and using built-in and custom query methods to perform CRUD operations.

Interpretation:

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

The candidate demonstrates a solid working knowledge of Spring Data JPA and relational database operations, including defining entities, extending JpaRepository, and applying both built-in and custom query methods. Minor gaps in knowledge or edge-case scenarios may exist, but overall competence is sufficient for most practical development tasks.

Interview Guide

How would you define a repository in Spring Data JPA, and how can you create a query to find records by a specific field without writing SQL?

1	2	3	4	5
Cannot describe how to create a repository interface or write a derived query method.		Correctly extends JpaRepository and writes a basic derived query method like findByFieldName.		Explains JpaRepository, writes derived query methods, and mentions @Query as an alternative for complex queries.

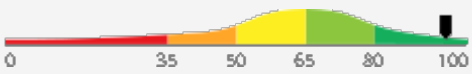
What is a JPA entity and what annotations are required at a minimum to define one in Spring Boot?

1	2	3	4	5
Cannot define a JPA entity or names incorrect or missing required annotations.		Correctly identifies @Entity and @Id but may miss @GeneratedValue or table mapping details.		Accurately describes @Entity, @Id, @GeneratedValue, and optionally @Table, with a clear explanation of their roles.

Detail

Testing Spring Boot Applications

Score: 95



Description:

Covers writing unit tests and integration tests for Spring Boot applications using JUnit, the @SpringBootTest annotation, and MockMvc for testing REST endpoints without running a full server. Includes mocking dependencies with Mockito and verifying controller and service behavior.

Interpretation:

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

The candidate demonstrates a comprehensive and advanced mastery of testing Spring Boot applications, encompassing unit and integration testing, REST endpoint validation using MockMvc, and sophisticated dependency mocking with Mockito. They are well-equipped to independently design and implement robust test suites, verify complex controller and service behavior, and uphold high standards of test quality.

Interview Guide

How would you write a test for a REST controller endpoint in Spring Boot without starting the full application server, and what tools or annotations would you use?

1	2	3	4	5
Cannot describe how to test a controller or names incorrect tools.		Correctly identifies MockMvc and @WebMvcTest but may not fully explain how to set up the test or assert the response.		Accurately explains @WebMvcTest, MockMvc setup, performing a mock HTTP request, and asserting status and response body.

What is the difference between a unit test and an integration test in the context of a Spring Boot application, and can you name a tool or annotation used for each?

1	2	3	4	5
Cannot distinguish between unit and integration tests or cannot name relevant tools.		Correctly distinguishes the two test types and names JUnit for unit tests and @SpringBootTest for integration tests.		Clearly explains both test types, names Mockito for mocking in unit tests, and explains @SpringBootTest and MockMvc for integration testing.

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.

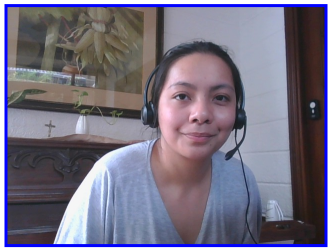
Question or Task	Response

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

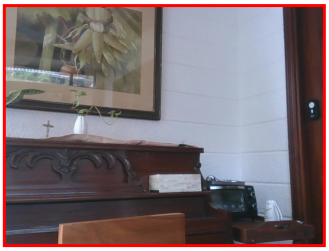
The following photos of the candidate and any identification were uploaded during the assessment session.

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%)

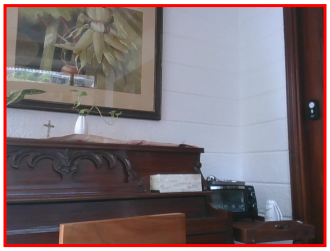


Pre/Post-Test Photo

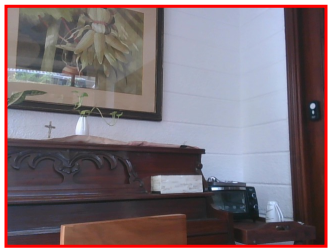
ID Photo



In-Test Error Detected (No Face Detected)

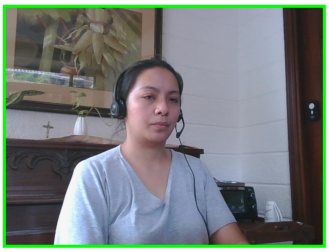


In-Test Error Detected (No Face Detected)

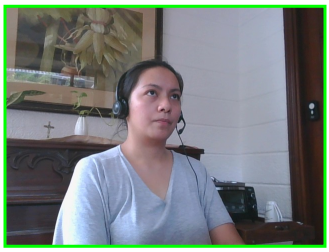


In-Test Error Detected (No Face Detected)

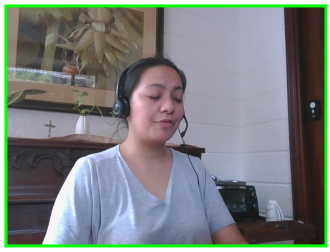
In-Test Photo



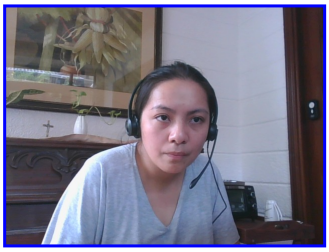
In-Test Photo



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: 20815-1, Key: 0-0, Rpt: 68, Prd: 9637, Created: 2026-06-28 13:01 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 (%)
Application Configuration	81.9628	Numeric Score	81.9628	8.3333
Application Configuration (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Exception Handling and Request Validation	75.8003	Numeric Score	75.8003	8.3333
Exception Handling and Request Validation (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
REST Controllers and HTTP Request Handling	85.8715	Numeric Score	85.8715	8.3333
REST Controllers and HTTP Request Handling (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Spring Boot Core Concepts and Application Structure	72.6373	Numeric Score	72.6373	8.3333
Spring Boot Core Concepts and Application Structure (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Spring Data JPA and Database Operations	70.7122	Numeric Score	70.7122	8.3333
Spring Data JPA and Database Operations (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Testing Spring Boot Applications	95.7362	Numeric Score	95.7362	8.3333
Testing Spring Boot Applications (Coding Tasks)	62.9784	Numeric Score	62.9784	8.3333
Weighted Average:				71.7159
Final Overall Score:				71

Notes

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