Zend Certified PHP Engineer (ZCPE) Practice Test (Sample)

Study Guide

Everything you need from our exam experts!

Table of Contents

Copyright **1**

Table of Contents **2**

Introduction **3**

How to Use This Guide **4**

Questions **5**

Answers **8**

Explanations **10**

Next Steps **16**

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- **Practice answering questions under realistic conditions,**
- **Improve accuracy and speed,**
- **Review explanations to strengthen weak areas, and**
- **Approach the exam with greater confidence.**

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here’s how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don’t just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There’s no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

Questions

1. Which PHP function can be used to convert a string to an array?

A. implode()

B. split()

C. explode()

D. array_split()

2. What does the `isset()` function do in PHP?

A. Checks if a variable is set and is not NULL

B. Fetches the value of a variable

C. Creates a variable if it is not set

D. Deletes a variable from memory

3. In PHP, which constant returns the name of the current function?

A. __METHOD__

B. __FUNCTION__

C. get_function_name()

D. current_function()

4. What function is commonly used to open a file in PHP?

A. open_file()

B. file_open()

C. fopen()

D. file_get_contents()

5. What function is used to create a new session variable in PHP?

A. session_start()

B. session_create()

C. session_new()

D. session_variable()

6. What is the output of the given PHP code that creates a DOM document and loads an empty root element?

A. <root/>

B. <root></root>

C. <?xml version="1.0"?><root></root>

D. <?xml version="1.0"?><root/>

7. What is the output of `strlen("Hello")` in PHP?

A. 5

B. 4

C. 6

D. 10

8. What is the difference between "==" and "===" in PHP?

A. "==" checks for value only; "===" checks for value and type equality

B. "==" compares variables; "===" compares arrays

C. "==" is for strings; "===" is for numbers

D. "==" checks for identical objects; "===" checks for identical arrays

9. What purpose does the header() function serve in PHP?

A. It sends a cookie to the client

B. It redirects the user to a different page

C. It sends a raw HTTP header to the client

D. It prints out the HTTP headers of the page

10. What is the output of an empty constant in PHP?

A. Null

B. True

C. False

D. Empty String

Answers

1. C
2. A
3. B
4. C
5. A
6. D
7. A
8. A
9. C
10. D

Explanations

1. Which PHP function can be used to convert a string to an array?

A. implode()

B. split()

C. explode()

D. array_split()

The function that is used to convert a string to an array is `explode()`. This function takes a delimiter and a string as input, and it splits the string at each occurrence of the delimiter. The result is an array of strings, with each element corresponding to a segment of the original string that was separated by the delimiter. For example, if you have a string such as "one,two,three" and you want to convert it into an array of words, you would call `explode(',', 'one,two,three')`. The output would be an array containing ["one", "two", "three"]. The other options do not serve the purpose of converting a string to an array in the same manner. `implode()` does the reverse of what `explode()` does; it takes an array and joins its elements into a single string using a specified delimiter. The function `split()` is actually deprecated as of PHP 5.3.0 and was used to split strings based on a regular expression, but is not typically used for simple delimiter-based string conversions. `array_split()` is not a built-in PHP function and does not exist in the PHP language, making it unrelated to the task of converting strings to arrays. Understanding

2. What does the `isset()` function do in PHP?

A. Checks if a variable is set and is not NULL

B. Fetches the value of a variable

C. Creates a variable if it is not set

D. Deletes a variable from memory

The `isset()` function is designed to determine if a variable has been declared and is not NULL. When you call `isset()` with a variable as its argument, the function returns TRUE if the variable exists and is not NULL; otherwise, it returns FALSE. This function is particularly useful for checking if variables are available for use, especially when working with form data, session variables, or any situation where you might not be sure if a variable has been initialized or assigned a value. In contrast, the other options describe different behaviors that `isset()` does not perform. Fetching the value of a variable would involve simply echoing or returning the variable itself, which is not what `isset()` is designed to do. Creating a variable if it is not set is also not a function of `isset()`, as it only checks for existence rather than modifying the variable's state. Lastly, deleting a variable from memory is handled by the `unset()` function in PHP, which specifically removes a variable, while `isset()` does not modify the variable's state at all.

3. In PHP, which constant returns the name of the current function?

A. __METHOD__

B. __FUNCTION__

C. get_function_name()

D. current_function()

The constant that returns the name of the current function in PHP is __FUNCTION__. This constant is specifically designed to provide the function's name in which it is called, making it very useful for debugging and logging purposes. When used within a function, it returns a string with the exact name of that function, regardless of how many times it has been called or from where in the code. Using __FUNCTION__ helps in scenarios where one might want to implement functionality conditionally based on the function name or to log activity related to specific functions. Since PHP automatically handles the scoping and context, it ensures that the right function name is always returned as expected. The other options do not accurately achieve this specific task. __METHOD__ would return the name of the method in the context of an object, including the class name, making it unsuitable when only the function name is needed. The function get_function_name() does not exist in PHP's built-in functions. Similarly, current_function() is also not a recognized PHP function. Thus, __FUNCTION__ is indeed the correct choice for retrieving the name of a function in PHP.

4. What function is commonly used to open a file in PHP?

A. open_file()

B. file_open()

C. fopen()

D. file_get_contents()

The function commonly used to open a file in PHP is `fopen()`. This function is vital for file handling, as it allows developers to specify the file they want to work with and determine the mode in which the file should be opened (such as read, write, append, etc.). When `fopen()` is called, it takes two primary parameters: the filename and the mode. For example, `fopen("example.txt", "r")` would open the file "example.txt" in read mode. If the file is successfully opened, `fopen()` returns a file pointer that can be used by other file functions to read from or write to the file. The other options listed do not represent standard PHP functions for opening files. Functions like `file_get_contents()` are used to read the entire file into a string, rather than opening a file for writing or reading operations that could be performed later. The options `open_file()` and `file_open()` do not exist in the PHP standard library, thus making them incorrect choices.

5. What function is used to create a new session variable in PHP?

A. session_start()

B. session_create()

C. session_new()

D. session_variable()

The function used to create a new session variable in PHP is session_start(). When you call session_start(), PHP checks if a session already exists for the user. If not, it initiates a new session and generates a unique session ID. This enables the use of session variables, which are stored on the server and can persist across different pages and requests. Once the session has started, you can create and manipulate session variables using the superglobal array $_SESSION. For instance, you can assign a value to a session variable like this: $_SESSION['key'] = 'value'; The session variable now exists and can be accessed throughout the user's session, provided that session_start() is called at the beginning of any script that needs access to session data. Other options do not exist in PHP's built-in functions for session management. For example, functions like session_create(), session_new(), and session_variable() are not defined in PHP, making them invalid choices for the purpose of creating a new session variable.

6. What is the output of the given PHP code that creates a DOM document and loads an empty root element?

A. <root/>

B. <root></root>

C. <?xml version="1.0"?><root></root>

D. <?xml version="1.0"?><root/>

The output of the PHP code that creates a DOM document and loads an empty root element is correct in identifying that it includes the XML declaration followed by a self-closing root element. When you create a new DOM document in PHP and add an empty root element to it, the default behavior is to also include the XML declaration at the start of the document. This declaration specifies the version of XML being used, which in this case is "1.0". Following that, since the root element is empty, it is represented as a self-closing tag, meaning that it does not have any child elements or text content inside. In the case provided, the combination of the XML declaration and the self-closing root element accurately reflects the output format that PHP's DOMDocument would produce, hence producing the result: <?xml version="1.0"?><root/>.

7. What is the output of `strlen("Hello")` in PHP?

A. 5

B. 4

C. 6

D. 10

The function `strlen()` in PHP returns the length of a given string in bytes. When you call `strlen("Hello")`, it computes the number of characters in the string "Hello". In this case, "Hello" consists of five letters: H, e, l, l, and o. Therefore, the output of `strlen("Hello")` is 5, which represents the total number of characters present in the string. It's important to note that this function counts each character individually and does not include any hidden characters or special formatting unless they are part of the string itself. Thus, option A is the correct answer.

8. What is the difference between "==" and "===" in PHP?

A. "==" checks for value only; "===" checks for value and type equality

B. "==" compares variables; "===" compares arrays

C. "==" is for strings; "===" is for numbers

D. "==" checks for identical objects; "===" checks for identical arrays

The distinction between "==" and "===" in PHP is fundamental to understanding how comparisons are made in the language. The correct choice indicates that "==" checks for value equality, meaning that it will return true if the values being compared are equivalent, regardless of their data types. For example, the integer 0 would be considered equal to the string "0" when using "==". On the other hand, "===" checks for both value and type equality. This means that for a comparison to return true using "===," the values must not only be the same but must also be of the same type. Therefore, the integer 0 and the string "0" would not be considered equal when using "===", leading to a more strict comparison. This concept is crucial in PHP as it can impact how conditions and logic within the code behave. Understanding this difference helps developers avoid unexpected behavior when performing comparisons and is an essential part of writing robust PHP code.

9. What purpose does the header() function serve in PHP?

A. It sends a cookie to the client

B. It redirects the user to a different page

C. It sends a raw HTTP header to the client

D. It prints out the HTTP headers of the page

The header() function in PHP is primarily used for sending raw HTTP headers to the client. This function enables developers to manipulate the HTTP response directly by setting headers that instruct the client (typically a web browser) on how to handle the response. For instance, you might use header() to set the content-type of a response, indicating to the client the type of content being returned (like text/html or application/json). Additionally, it is frequently used to manage cache control or to define custom headers needed for proper functionality of the application. Choosing option C reflects a fundamental understanding of the purpose of the header() function. While redirecting the user to a different page is a common use case for the header() function (specifically by using a "Location" header), it is only one of its many functionalities. Hence, the correct answer encompasses the broader capability of the function, which is to send any raw HTTP header, not just for redirection or cookie management.

10. What is the output of an empty constant in PHP?

A. Null

B. True

C. False

D. Empty String

In PHP, trying to access an empty constant that has not been defined results in an empty string. When a constant is referenced within a script and it doesn't exist, PHP will not throw an error; instead, it treats it as an undefined value, which is effectively equivalent to an empty string. This behavior is significant because it allows for flexible coding practices, where constants are often used in conditional statements or as configuration values. If a developer attempts to use an empty constant, PHP will return an empty string, which can be useful in situations where the intention is to check for the presence of a value without causing disruption to the flow of the code. Thus, the output when referencing an empty constant is an empty string, reinforcing the idea that PHP handles undefined constants gracefully by returning a value that allows the script to continue running without interruption.

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://zendcertifiedphpengineer.examzify.com

We wish you the very best on your exam journey. You've got this!