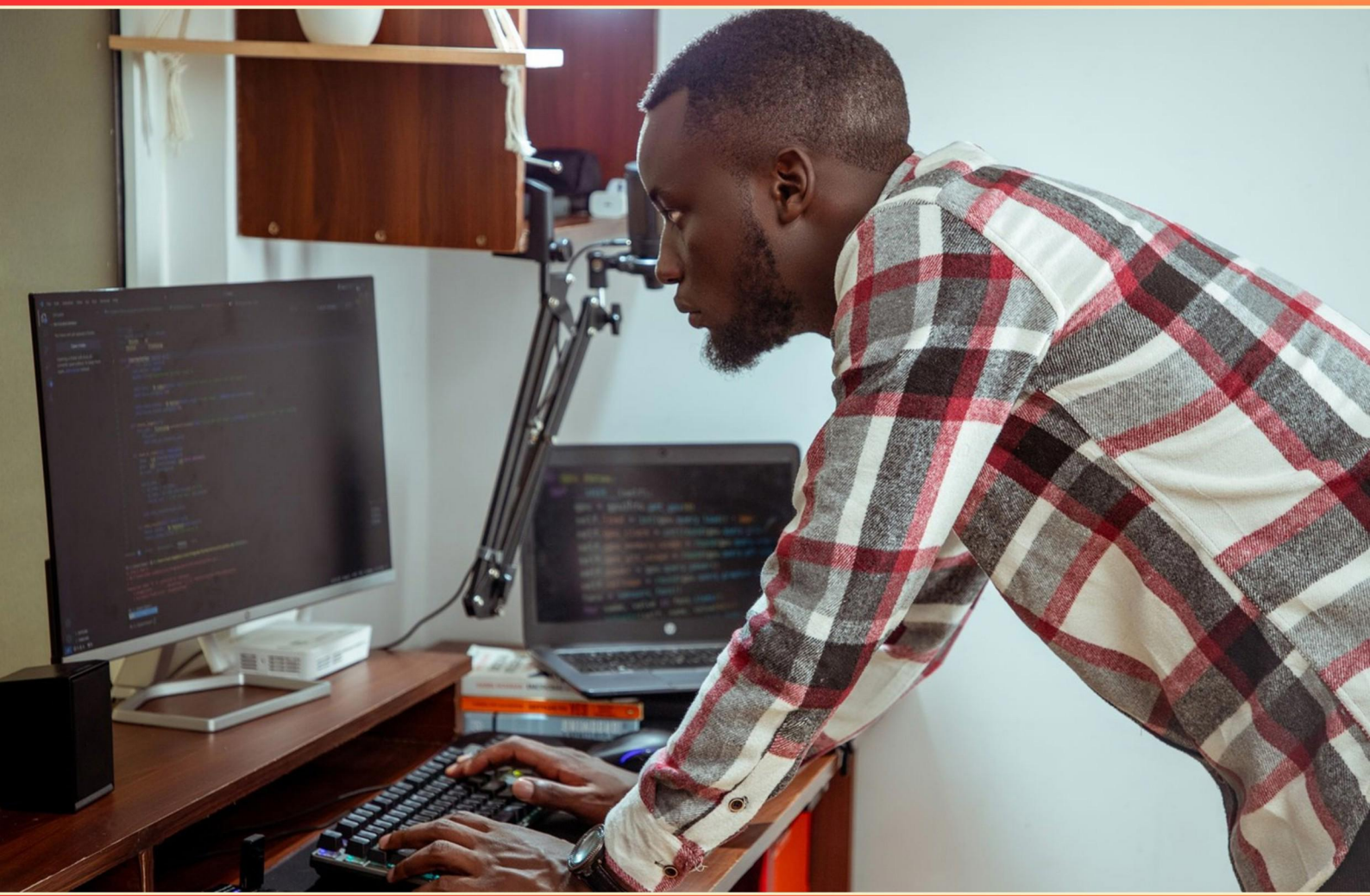
ZEND CERTIFIED PHP ENGINEER (ZCPE) PRACTICE TEST (SAMPLE)

STUDY GUIDE



SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://zendcertifiedphpengineer.examzify.com

Table of Contents

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. 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

2. How do you define a class in PHP?

A. Using the class keyword followed by the class name

B. Using the define keyword followed by the class name

C. Using the create keyword followed by the class name

D. Using the function keyword followed by the class name

3. What is an anonymous function in PHP?

A. A function with a user-defined name

B. A function that cannot be used as a callback

C. A function without a specified name

D. A built-in PHP function

4. What is the primary purpose of PHP?

A. To serve as a client-side scripting language for web development

B. To serve as a server-side scripting language for web development

C. To manage databases

D. To create desktop applications

5. What will be the output of the following PHP code: echo 5 + "10 apples";?

A. 15 apples

B. 15

C. 5 apples

D. 10 apples

6. What does PDO stand for in PHP?

A. PHP Document Object

B. PHP Data Objects

C. PHP Dynamic Output

D. PHP Database Operations

7. Which operator would you use to ensure two values or variables are not equal in PHP?

A. ===

B. !=

C. !==

D. ==

8. How do you handle exceptions in PHP?

A. Using if-else statements

B. Using try, catch, and finally blocks

C. Using error_reporting()

D. Using exception_error()

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

A. 5

B. 4

C. 6

D. 10

10. Which operator is used for checking type and value equality in PHP?

A. !=

B. ===

C. !==

D. +

Answers

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

Explanations

1. 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.

2. How do you define a class in PHP?

A. Using the class keyword followed by the class name

B. Using the define keyword followed by the class name

C. Using the create keyword followed by the class name

D. Using the function keyword followed by the class name

To define a class in PHP, you use the class keyword followed by the name of the class. This syntax is essential as it clearly indicates to the PHP interpreter that a new class definition is being created. The class declaration allows you to encapsulate properties and methods that define the behavior and characteristics of that class. For instance: ```php class MyClass { public $property; public function myMethod() { // Method code goes here } } ``` In this example, "MyClass" is defined using the class keyword, demonstrating how to encapsulate behavior and state. This approach forms the backbone of object-oriented programming in PHP, allowing developers to create reusable and modular code components. The other choices provided do not accurately reflect the syntax used in PHP for class definitions. The define keyword is used for constants, the create keyword does not exist in PHP, and the function keyword is used for defining functions, not classes. Thus, the correct option succinctly outlines the necessary framework for class creation in PHP.

3. What is an anonymous function in PHP?

A. A function with a user-defined name

B. A function that cannot be used as a callback

C. A function without a specified name

D. A built-in PHP function

An anonymous function in PHP is defined as a function that does not have a specified name. This allows it to be created and used in situations where it might not need a formal name, such as when passing it as a callback function or defining it inline. Anonymous functions, also referred to as closures, can be particularly useful in scenarios such as array manipulation, where a temporary function needs to be executed on each element of an array. They provide a convenient way to encapsulate functionality without cluttering the global namespace with unnecessary function names. The concept of having a function that lacks a name allows for more flexible programming paradigms, such as functional programming, where functions can be treated as first-class citizens and passed around like any other type of data. Other options do not accurately describe an anonymous function. A function with a user-defined name explicitly indicates that it is named, while options referring to callbacks and built-in functions do not capture the essence of what makes an anonymous function unique.

4. What is the primary purpose of PHP?

A. To serve as a client-side scripting language for web development

B. To serve as a server-side scripting language for web development

C. To manage databases

D. To create desktop applications

The primary purpose of PHP is to serve as a server-side scripting language for web development. PHP enables the creation of dynamic web pages by processing scripts on the server before the page is sent to the client's browser. This allows developers to interact with databases, generate HTML code based on user input, and produce content that can be customized in real-time, creating a more engaging user experience. Server-side scripting is essential in modern web applications, where functionality such as user authentication, form submission processing, and data retrieval from databases need to occur securely and efficiently on the server. By executing scripts on the server, PHP can protect sensitive data, manage sessions, and leverage server resources effectively. The other options do not accurately describe PHP's core functionality. While PHP can be used to manage databases and can be part of desktop applications in combination with other technologies, its primary domain remains server-side scripting for dynamic web development. Similarly, PHP is not designed as a client-side scripting language, which typically involves technologies like JavaScript that execute in the user's browser.

5. What will be the output of the following PHP code: echo 5 + "10 apples";?

A. 15 apples

B. 15

C. 5 apples

D. 10 apples

The output of the code `echo 5 + "10 apples";` is indeed 15. This result is due to how PHP handles the addition operation when one of the operands is a string that contains numeric characters and additional non-numeric characters. When the addition operator `+` is used, PHP attempts to convert the string to a number. In this case, the string "10 apples" begins with the digits '10', which PHP successfully converts to the integer value 10. The operation then performs the addition: 5 (the integer on the left) plus 10 (the converted value from the string) results in 15. The non-numeric portion of the string, "apples", is ignored in this operation as PHP primarily focuses on the numeric part at the beginning of the string. Therefore, the final output is simply the numeric result of the addition, which is 15. This behavior is consistent with PHP's type juggling feature, where the language attempts to convert between types as needed during operations like arithmetic.

6. What does PDO stand for in PHP?

A. PHP Document Object

B. PHP Data Objects

C. PHP Dynamic Output

D. PHP Database Operations

The term PDO stands for PHP Data Objects, which is a database access layer providing a uniform interface for accessing multiple types of databases in PHP. It allows developers to perform database operations in a secure and efficient manner by offering prepared statements, which help mitigate SQL injection vulnerabilities. PDO is designed to support different database management systems (DBMSs), including MySQL, PostgreSQL, and SQLite, among others, without requiring changes to the PHP codebase. This abstraction makes it easier to switch between different databases if necessary while maintaining the same code structure, promoting code reusability and flexibility. Each of the other options does not accurately reflect the purpose or functionality of PDO. 'PHP Document Object' suggests a focus on document structure, which is not related to database interactions. 'PHP Dynamic Output' implies a feature oriented towards output generation rather than database handling, and 'PHP Database Operations' is somewhat close but lacks the specificity and definition provided by "Data Objects." Thus, identifying PDO as PHP Data Objects is essential for understanding its role in PHP database interaction.

7. Which operator would you use to ensure two values or variables are not equal in PHP?

A. ===

B. !=

C. !==

D. ==

The operator used to ensure that two values or variables are not equal in PHP is the inequality operator, represented as `!=`. This operator checks if the values being compared are not the same. If they are not equal, it returns true; otherwise, it returns false. In contrast to other operators, `!=` performs type juggling, meaning it will evaluate the equality of two values after converting them to a common type if they are of different types. For example, the expression `0 != "0"` would evaluate to false because they are considered equal after type conversion. It's worth noting that the strict inequality operator, `!==`, checks for both value and type. This means that if the values are not equal or if they are of different types, it returns true. The triple equals operator, `===`, checks for strict equality, meaning both the value and type must be the same for it to return true. Therefore, when you want to assert that two values are not equal, `!=` is the appropriate choice as it allows for flexibility in type comparison.

8. How do you handle exceptions in PHP?

A. Using if-else statements

B. Using try, catch, and finally blocks

C. Using error_reporting()

D. Using exception_error()

Using try, catch, and finally blocks is the established way to handle exceptions in PHP, allowing for structured and effective management of error conditions. This approach enables developers to separate normal program flow from error handling, enhancing code clarity and maintainability. When an exception is thrown in PHP, execution can be transferred to a catch block where the exception can be processed, logged, or handled appropriately. The try block contains code that may throw an exception, and if an exception occurs, the script stops executing further code within that block and jumps to the associated catch block, which can deal with the exception. The finally block, if used, will execute regardless of whether an exception was thrown or caught, making it suitable for releasing resources or performing cleanup tasks. This structured error handling is crucial in building robust applications, especially when dealing with unpredictable runtime errors that can disrupt normal execution. In contrast, other options such as using if-else statements or error reporting tools do not provide this level of control or organization over error management, leading to less effective handling of exceptions.

9. 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.

10. Which operator is used for checking type and value equality in PHP?

A. !=

B. ===

C. !==

D. +

The operator used for checking both type and value equality in PHP is the triple equal sign, represented as ===. This operator ensures that not only are the values being compared equal, but it also confirms that they are of the same type. For example, using === to compare an integer with a string that contains the same number will return false because, despite having the same numerical value, they are different types (integer vs. string). This strict comparison is helpful in scenarios where type integrity is crucial, such as when dealing with data from various sources or in function parameter checks. The clarity it provides by enforcing both type and value checks helps prevent subtle bugs that can arise from implicit type juggling in PHP. In contrast, other options involve different comparisons. For instance, != and !== are used for value comparison without strict type checking and can lead to type coercion. The use of + does not relate to comparison at all, as it is the addition operator in PHP. Therefore, the triple equal sign is the most precise for ensuring both data type fidelity and value equality in PHP applications.

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!