WESTERN GOVERNORS UNIVERSITY (WGU) C173 SCRIPTING AND PROGRAMMING PRACTICE TEST (SAMPLE)

STUDY GUIDE

SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://wgu-c173.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 distinguishes statically typed languages from dynamically typed languages?**

   A. Statically typed languages are checked at runtime

   B. Statically typed languages require data types to be defined beforehand

   C. Dynamically typed languages do not support type inference

   D. Statically typed languages allow variable types to change during execution

2. **During which phase does a programmer create a second version of a program based on customer feedback?**

   A. Implementation

   B. Planning

   C. Testing

   D. Documentation

3. **When comparing compiled and interpreted languages, which statement is true?**

   A. Compiled languages run slower than interpreted languages

   B. Interpreted languages typically convert to machine code

   C. Compiled languages translate the source code prior to execution

   D. Interpreted languages are always more efficient

4. **Which characteristic specifically describes interpreted languages?**

   A. They can be run on any machine having the right interpreter

   B. They compile code into machine language beforehand

   C. They execute code in parallel threads

   D. They are faster than compiled languages

5. **How is the control of a loop determined in programming?**

   A. By the number of data elements

   B. By conditions that dictate the number of iterations

   C. By the value of the first iteration

   D. By user input only

6. What data structure is used to store a collection of similar items?

A. Array

B. Dictionary

C. String

D. Integer

7. Which UML diagram focuses on the order of messages passed between objects?

A. Class Diagram

B. Sequence Diagram

C. Collaboration Diagram

D. Use Case Diagram

8. When does a programmer enter the implementation phase of the waterfall approach?

A. When the requirements are defined

B. When programming has commenced

C. When the system is being tested

D. When goals are analyzed

9. What is the main difference between `==` and `===` in JavaScript?

A. `==` checks for both value and type equality

B. `===` checks only for value equality

C. `==` checks for value equality

D. `===` checks for value and logical equality

10. What is the purpose of initializing the `self` parameter in methods?

A. To create a global reference

B. To communicate with other classes

C. To refer to the current instance of a class

D. To store constant values

Answers

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

Explanations

1. What distinguishes statically typed languages from dynamically typed languages?

A. Statically typed languages are checked at runtime

B. Statically typed languages require data types to be defined beforehand

C. Dynamically typed languages do not support type inference

D. Statically typed languages allow variable types to change during execution

Statically typed languages require data types to be defined beforehand, meaning that each variable must be assigned a type at compile time. This characteristic allows for type checking to occur during the compilation process, which helps catch type-related errors before the program is executed. By having a defined type for each variable, statically typed languages can leverage this information to optimize performance and memory management. This approach often results in more predictable behavior, as the types of variables are known ahead of time. In contrast, dynamically typed languages determine variable types at runtime, allowing for more flexibility but also increasing the potential for runtime errors due to type mismatches. Thus, the defining feature of static typing is the necessity to declare types in advance, contrasting with the fluid nature of dynamic typing.

2. During which phase does a programmer create a second version of a program based on customer feedback?

A. Implementation

B. Planning

C. Testing

D. Documentation

The phase during which a programmer creates a second version of a program based on customer feedback is the implementation phase. This phase involves taking the initial version of a program, which has likely gone through some level of testing and review, and making improvements or changes based on user input and feedback. Implementation typically follows the initial development and may include iterative cycles of coding, testing, and revising to enhance functionality or fix any issues identified by users. The programmer uses the feedback received during testing or from end-users to refine the software, ensuring it aligns more closely with user expectations and requirements. In contrast, planning is focused on defining the project scope, requirements, and overall design without involving direct changes to the program. The testing phase is primarily concerned with identifying bugs and verifying that the software meets its specifications rather than creating new versions based on feedback. Documentation involves creating manuals and guides but does not include modifying the program itself.

3. When comparing compiled and interpreted languages, which statement is true?

A. Compiled languages run slower than interpreted languages

B. Interpreted languages typically convert to machine code

C. Compiled languages translate the source code prior to execution

D. Interpreted languages are always more efficient

Compiled languages translate the source code prior to execution, resulting in a machine code file that can be directly executed by the computer's hardware. This pre-execution translation allows programs written in compiled languages to run faster since the process of converting high-level code to machine code occurs only once, and the already compiled code is executed multiple times without the overhead of translation during runtime. This approach contrasts with interpreted languages, which convert code into machine-readable instructions on-the-fly during execution, typically leading to slower performance since translation occurs each time the program runs. The direct relationship between compilation and execution speed underscores why this statement accurately reflects the functioning of compiled languages.

4. Which characteristic specifically describes interpreted languages?

A. They can be run on any machine having the right interpreter

B. They compile code into machine language beforehand

C. They execute code in parallel threads

D. They are faster than compiled languages

The characteristic that specifically describes interpreted languages is that they can be run on any machine having the right interpreter. This means that unlike compiled languages, which are translated into machine code specific to a particular operating system or hardware architecture before execution, interpreted languages are processed at runtime by an interpreter. This allows the same code to run on different platforms as long as there is a compatible interpreter available. The flexibility provided by this characteristic enhances portability because the source code can remain unchanged while allowing it to be executed on various systems. As a result, developers can write their code once and leverage the interpreter's capabilities to run it in diverse environments, making interpreted languages highly adaptable for different operating systems. The other options do not accurately reflect the nature of interpreted languages. For instance, not all interpreted languages execute code in parallel threads, nor do they generally outperform compiled languages in speed, which is a common trait of compiled languages due to the direct translation of code to machine language before execution. Moreover, the claim that they compile code ahead of time does not apply, as interpretation does not involve pre-compilation into machine code.

5. How is the control of a loop determined in programming?

A. By the number of data elements

B. By conditions that dictate the number of iterations

C. By the value of the first iteration

D. By user input only

The control of a loop is determined by conditions that dictate the number of iterations because these conditions define how many times the loop will execute before terminating. In programming, loops are commonly controlled by conditional statements, often structured as "while," "for," or "do-while" loops. For instance, in a "while" loop, the loop continues to execute as long as a specified condition evaluates to true. Once the condition becomes false, the loop will terminate. In "for" loops, the number of iterations is often defined by a counter and a limit, allowing precise control of how many times the loop runs based on the initial condition set for the loop. This method of controlling loops allows programmers to perform repetitive tasks efficiently and ensures that the loop operates for an intended number of iterations based on defined criteria rather than relying solely on the first iteration's value or on user input. The flexibility of using conditions enables the development of complex algorithms that can adapt to varying requirements dynamically.

6. What data structure is used to store a collection of similar items?

A. Array

B. Dictionary

C. String

D. Integer

The correct choice is an array because it is specifically designed to store a collection of similar items. Arrays are a fundamental data structure in programming that allow you to store multiple values in a single variable, and all elements within an array typically share the same data type. This makes arrays efficient for handling groups of data, such as lists of numbers, strings, or custom objects. In programming, when you want to manipulate a collection of similar items, arrays provide indexed access. This means that each element can be retrieved or modified using its index position in the array. For example, if you have an array of integers, you can quickly access any particular integer using its index. This is particularly useful for operations like sorting, searching, or iterating through the collection. Other options do not fit the requirement as an array does. A dictionary, for instance, is used for storing key-value pairs, making it more suitable for associative data rather than a simple collection of similar items. A string represents a sequence of characters and is not designed to manage multiple values as a collection. An integer, on the other hand, is a singular numerical value and cannot be used to store collections of items. Therefore, the array is the most appropriate choice for this context.

7. Which UML diagram focuses on the order of messages passed between objects?

A. Class Diagram

B. Sequence Diagram

C. Collaboration Diagram

D. Use Case Diagram

The focus of a Sequence Diagram is on the order of messages passed between objects. This type of UML diagram is specifically designed to detail how objects interact in a sequential manner over time. In a Sequence Diagram, the vertical axis typically represents time, with the various objects displayed horizontally across the top. As events occur, they are shown as arrows indicating messages sent from one object to another, along with the sequence of these interactions. Therefore, it effectively illustrates how the objects collaborate to fulfill a specific use case or functionality through the flow of messages, highlighting not just the interactions but also the timing of those messages. In contrast, other diagrams like Class Diagrams focus more on the structure of the system, showing classes and their relationships independently of time. Collaboration Diagrams emphasize the roles of the objects in a system rather than the timing of the exchanges. Use Case Diagrams outline the interactions between users (or actors) and the system as a whole, focusing on use cases rather than the details of the object interactions over time. Thus, for understanding message order and timing between objects, the Sequence Diagram is the appropriate choice.

8. When does a programmer enter the implementation phase of the waterfall approach?

A. When the requirements are defined

B. When programming has commenced

C. When the system is being tested

D. When goals are analyzed

In the waterfall approach to software development, the implementation phase begins after the requirements have been thoroughly defined and all preliminary phases, such as analysis and design, are completed. This phase is marked by the actual programming work where the developers start writing code according to the specifications laid out in the earlier phases. This means that when programming has commenced, it signifies that the project has moved from planning and designing stages into the active development of the software. During this phase, the focus shifts to turning the theoretical plans into functional software.

9. What is the main difference between `==` and `===` in JavaScript?

A. `==` checks for both value and type equality

B. `===` checks only for value equality

C. `==` checks for value equality

D. `===` checks for value and logical equality

The main difference between `==` and `===` in JavaScript lies in how they handle comparisons of values, specifically regarding type coercion. The operator `==` performs a comparison of values and allows for type coercion, meaning that if the values being compared are of different types, JavaScript will attempt to convert one or both values to a common type before making the comparison. For example, if you compare a string to a number, JavaScript will convert the string to a number for the comparison. On the other hand, `===` is known as the strict equality operator. It checks both the value and the type for equality without performing any type coercion. This means that if the values being compared are not of the same type, `===` will return false, even if the values might be logically equivalent when converted. Therefore, the correct understanding is that `==` checks for value equality with type coercion allowed, while `===` requires both the value and the type to be the same, leading to more predictable and less error-prone code.

10. What is the purpose of initializing the `self` parameter in methods?

A. To create a global reference

B. To communicate with other classes

C. To refer to the current instance of a class

D. To store constant values

The purpose of initializing the `self` parameter in methods is to refer to the current instance of a class. It acts as a reference to the specific object that is calling the method, allowing you to access variables and methods associated with that object. When you define a method within a class, `self` is required as the first parameter so you can work with the instance's attributes and call other methods on that instance without needing to pass the object itself explicitly each time. This design is essential for object-oriented programming in Python, promoting encapsulation and allowing each object to maintain its own state.

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://wgu-c173.examzify.com

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