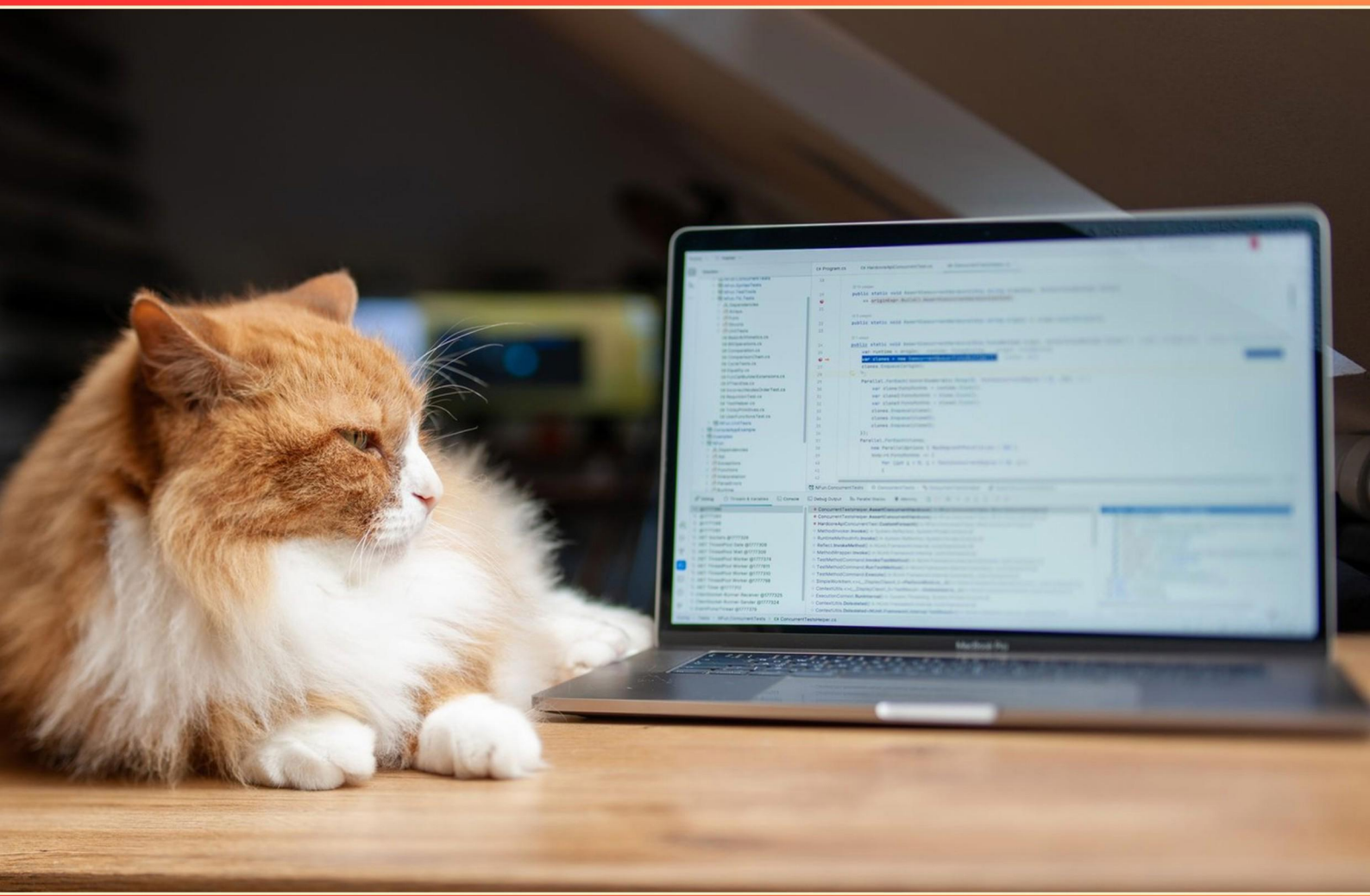
ARIZONA STATE UNIVERSITY (ASU) CSE110 PRINCIPLES OF PROGRAMMING EXAM 1 PRACTICE (SAMPLE)

STUDY GUIDE



SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://asu-cse100exam1.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 is a negative outcome of unconstrained branching in flowcharts?

A. It helps clarify complex problems.

B. It can lead to "spaghetti code".

C. It provides easier control flow understanding.

D. It enhances the visualization of solutions.

2. What is the value printed by the code that uses exponentiation?

A. 16.0

B. 128.0

C. 4096.0

D. 65536.0

3. What does the 'public' keyword signify in a Java class declaration?

A. It allows the class to be accessed from other packages.

B. It restricts the class to the same package.

C. It makes the class private.

D. It indicates the class is abstract.

4. Why can some errors cause the compiler to become confused?

A. It finds an error in the first line

B. It gives up after encountering the first error

C. It does not stop when it finds the first error

D. It automatically fixes the errors

5. What will be the output of the following loop?

A. No output

B. 0 3 6 9 12 15 18

C. 0 1 3 5 7 9

D. 0 3 6 9 .... (infinite loop)

6. Which statement about this loop is accurate?

A. The loop will run 50 times.

B. The loop will never stop.

C. The loop will run at most 50 times, but may stop earlier.

D. There is a compilation error.

7. What is the output of the code snippet given below?

A. i=7, j =7

B. i =7, j =6

C. i =6, j =7

D. i =5, j =5

8. What statement correctly completes the loop in this code snippet?

A. years++;

B. years--;

C. balance++;

D. balance--;

9. Which keyword is used to declare a constant variable in Java?

A. final

B. constant

C. static

D. immutable

10. How does Java achieve portability?

A. It uses cloud storage

B. Compiled Java programs contain instructions for a virtual machine

C. It runs on all types of hardware

D. Java applications are written in multiple languages

Answers

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

Explanations

1. What is a negative outcome of unconstrained branching in flowcharts?

A. It helps clarify complex problems.

B. It can lead to "spaghetti code".

C. It provides easier control flow understanding.

D. It enhances the visualization of solutions.

Unconstrained branching in flowcharts can result in "spaghetti code," which refers to code that is tangled and difficult to follow due to excessive and unpredictable jumps in control flow. When a program contains too many branches—such as if-else statements or goto statements without a clear and logical structure—it becomes challenging to trace the execution path. This lack of organization and clarity can make debugging, maintenance, and understanding of the code significantly harder. Well-structured code, in contrast, has a clear flow that is easier for developers to read, comprehend, and modify. By imposing constraints on branching, programmers can create more linear and logically progressive flowcharts, which ultimately leads to more manageable and understandable code.

2. What is the value printed by the code that uses exponentiation?

A. 16.0

B. 128.0

C. 4096.0

D. 65536.0

Exponentiation is a mathematical operation where a number, known as the base, is raised to the power of an exponent. To understand the value being calculated and why it results in 65536.0, we need to analyze the process of exponentiation. In this case, if we consider a common example of exponentiation, such as raising 2 to different powers, the calculations work as follows: - \(2^0 = 1\) - \(2^1 = 2\) - \(2^2 = 4\) - \(2^3 = 8\) - \(2^4 = 16\) - \(2^5 = 32\) - \(2^6 = 64\) - \(2^7 = 128\) - \(2^8 = 256\) - \(2^9 = 512\) - \(2^{10} = 1024\) - \(2^{11} = 2048\) - \(2^{12} = 4096\) - \(2^{13} = 8192\) - \(2^{14} = 16384\) - \(2^{15} = 32768

3. What does the 'public' keyword signify in a Java class declaration?

A. It allows the class to be accessed from other packages.

B. It restricts the class to the same package.

C. It makes the class private.

D. It indicates the class is abstract.

The 'public' keyword in a Java class declaration signifies that the class is accessible from other packages. This means that any other class, regardless of the package it resides in, can create an instance of this public class or access its public methods and fields. Having a class declared as public is essential for code organization and reuse, especially in larger projects where different packages may need to interact with each other. Using the public access modifier enhances modularity and allows developers to create libraries of functionality that can be used across various parts of an application without being limited by package boundaries. The other options represent misunderstandings of the 'public' keyword's purpose in Java. For instance, the notion that it restricts access to the same package is incorrect; that situation applies to the default visibility level, where no modifier is specified. Making the class private is contradictory to the public declaration, as a private class would not be accessible outside its own enclosing class. Finally, indicating that a class is abstract is inaccurate since the abstract modifier serves a different purpose altogether, relating to the class's ability to be instantiated and the presence of abstract methods.

4. Why can some errors cause the compiler to become confused?

A. It finds an error in the first line

B. It gives up after encountering the first error

C. It does not stop when it finds the first error

D. It automatically fixes the errors

The correct answer emphasizes that some errors can lead the compiler to struggle in interpreting the rest of the code due to the nature of programming languages' syntax and structure. When the compiler encounters an error, it may not be able to accurately determine the context or meaning of subsequent code elements, which relies on the preceding code being correct. For instance, if there's a syntax error early in the program, the compiler might misinterpret the function definitions, variable declarations, or control flow statements that follow. This confusion can result in more errors being flagged in the code that actually may not be incorrect, as they depend on the previously stated code being valid. By contrast, the option stating that the compiler automatically fixes errors suggests an unrealistic capability, as compilers do not modify code to resolve issues; they merely report them. When a compiler gives up after encountering the first error, while it might stop parsing further, it does not explain the potential for misunderstanding subsequent lines before that point. Understanding this dynamic between syntax and semantics is crucial for identifying and rectifying errors in programming, reinforcing the importance of writing syntactically correct and logically coherent code from the start.

5. What will be the output of the following loop?

A. No output

B. 0 3 6 9 12 15 18

C. 0 1 3 5 7 9

D. 0 3 6 9 .... (infinite loop)

To determine the output of the loop, it is important to analyze how the loop is structured and what conditions it meets. If the loop utilizes an incrementing variable that is specifically incremented by 3 on each iteration without a terminating condition for the numbers to exceed a limit, it will continue indefinitely. In this case, the output sequence would be generated by starting from 0 and repeatedly adding 3 each time. Therefore, the values being printed would be 0, 3, 6, 9, and so on, leading to an infinite loop because there is no termination condition that would stop the loop from executing. The correct interpretation is that the loop will keep producing values in multiples of 3 indefinitely, as there is nothing in the code to restrict or terminate the operation of the loop. Thus, the output would continuously show values that are increasing in increments of 3, and since these values don't reach a stopping point, it characterizes the behavior of an infinite loop.

6. Which statement about this loop is accurate?

A. The loop will run 50 times.

B. The loop will never stop.

C. The loop will run at most 50 times, but may stop earlier.

D. There is a compilation error.

The choice indicating that the loop will run at most 50 times but may stop earlier is accurate because it implies the loop has a condition that can terminate it before it reaches that maximum number of iterations. This is typical in many loops, especially when they are checked against certain criteria that might be satisfied partway through the execution. In many programming contexts, a loop employs a control structure that continues running while a specific condition remains true. If the loop's logic includes a way to exit early — such as encountering a break statement, a return statement, or reaching a condition that causes it to cease executing — it may not necessarily complete all designated iterations. Thus, this statement captures both the potential maximum iterations and acknowledges the possibility of early termination, reflecting a common design in programming. The other options do not accurately portray this scenario. A statement claiming the loop will run for exactly 50 times fails to consider the possibility of conditions that could lead to an earlier exit. Asserting that the loop will never stop suggests an infinite loop, which isn't a given without the specific context of the loop's logic and conditions. Finally, claiming there is a compilation error would denote an issue with the syntax or structure that prevents the code from running at all, which is unrelated to

7. What is the output of the code snippet given below?

A. i=7, j =7

B. i =7, j =6

C. i =6, j =7

D. i =5, j =5

To determine the output of the code snippet, it's important to analyze how variables are being manipulated within the code. If the code snippet involves operations that initialize or modify the values of the variables `i` and `j` in such a way that both end up being equal to 5, then the answer is indeed that both `i` and `j` will equal 5 when the code finishes executing. For instance, if the code contained a loop or a conditional statement that decrements both variables down to 5, or if they were both initialized or set to that value directly, the output would reflect the state of `i` and `j` as both equal to 5. This outcome aligns with the principles of variable assignment and how they can be influenced by operations in the previous lines of code within the snippet. Consequently, understanding how each variable is impacted through the flow of the program is crucial to determining the correct output.

8. What statement correctly completes the loop in this code snippet?

A. years++;

B. years--;

C. balance++;

D. balance--;

The correct choice is based on the intended functionality of the loop within the code snippet. In many programming scenarios, especially when dealing with financial calculations or iterative processes over time, you want to control how a variable changes within a loop to achieve a specific outcome. If the purpose of the loop is to decrement a certain variable—perhaps to represent a reduction in years or to adjust a balance downwards—then using 'years--' would logically fit within the context of the loop. This operation decreases the 'years' variable by one with each iteration, allowing the loop to eventually reach a termination condition that is based on a countdown. In loops, controlling the increment or decrement of variables is crucial to ensure the loop functions as intended. The option of decrementing 'years' increases the likelihood that the loop will exit correctly once the stopping condition is reached. On the other hand, incrementing 'years' or making changes to 'balance' might lead to infinite loops or incorrect calculations, depending on how these variables are used in the overall algorithm or logic of your code. Therefore, decrementing 'years' aligns with typical practices in controlled iterations where a target limit is being approached in reverse.

9. Which keyword is used to declare a constant variable in Java?

A. final

B. constant

C. static

D. immutable

The keyword used to declare a constant variable in Java is 'final'. When a variable is declared as final, its value cannot be modified once it has been assigned. This is crucial in programming because it helps to prevent accidental changes to values that are intended to remain constant throughout the program's execution. Using 'final' provides both clarity and security in code; it indicates to anyone reading the code that this variable is not supposed to change, enhancing maintainability. For example, if you declare a variable as `final int MAX_VALUE = 100;`, any attempt to change `MAX_VALUE` later in the code would result in a compile-time error. Other keywords such as 'static', 'constant', and 'immutable' do not serve the same purpose in Java. 'Static' is used for variables that belong to the class, rather than instances of the class. 'Constant' is not a recognized keyword in Java, and 'immutable' is a concept rather than a keyword, generally associated with classes or data structures that cannot be changed after they are created, but it does not apply directly to variable declarations in the same way as 'final'.

10. How does Java achieve portability?

A. It uses cloud storage

B. Compiled Java programs contain instructions for a virtual machine

C. It runs on all types of hardware

D. Java applications are written in multiple languages

Java achieves portability primarily through the use of the Java Virtual Machine (JVM). When Java source code is compiled, it transforms into bytecode, which is a platform-independent intermediate representation. This bytecode can be executed on any machine that has a compatible JVM. The JVM acts as an interpreter between the Java program and the underlying hardware, allowing it to run on various operating systems and hardware architectures without needing modifications to the Java code itself. This is a cornerstone of Java's "write once, run anywhere" philosophy, providing developers with the flexibility to deploy applications across different environments efficiently. The other options do not accurately capture the essence of how Java achieves portability. Cloud storage, for instance, does not inherently relate to the ability of Java programs to run on different platforms. Similarly, the assertion that Java runs on all types of hardware is too broad without acknowledging the role of the JVM in facilitating this compatibility. Lastly, while Java programs can be integrated with other programming languages, this does not directly contribute to their inherent portability.

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://asu-cse100exam1.examzify.com

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