Mastering Java: The Ultimate Quiz for 'Thinking in Java (Sample)

Study Guide

Everything you need from our exam experts!

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. Does MapEntry use the key’s hashCode() for hashing?

A. Yes

B. No

C. Only if the key is a String

D. Only for null keys

2. What does the 'synchronized' keyword prevent?

A. Variable mutations

B. Race conditions

C. Syntax errors

D. All of the above

3. What does polymorphism allow in object-oriented programming?

A. Objects to change their type at runtime

B. A single interface to be used for different underlying forms of data

C. For objects to exist in multiple states simultaneously

D. For objects to automatically delete themselves once they are no longer needed

4. Which method would you use to start a previously created thread?

A. init()

B. begin()

C. start()

D. run()

5. Why can't you upcast a List<Apple> to a List<Fruit>?

A. Because the List types are invariant

B. Because a Fruit is not an Apple

C. Because of runtime type information

D. Because generics do not support covariance by default

6. What types of objects can List<?> accept via its add method?

A. Objects of any type

B. Only objects of the specified wildcard type

C. Only null

D. Only objects of the base class of the wildcard type

7. Which method must be called by the user when an InputFile object is no longer needed?

A. close()

B. dispose()

C. finalize()

D. clean()

8. What kind of types can Java generic methods operate on?

A. Only Object types

B. Specifically typed objects according to the bounds

C. Any object that extends Throwable

D. Only numeric types

9. What is true about methods in a Java interface?

A. They can have method bodies if marked static

B. They are automatically public

C. They must be abstract

D. They can be private

10. Within the SWT version of ColorBoxes, what causes the display to update after a color change?

A. redraw() method call

B. Changing the RGB values directly

C. Calling the system's native paint method

D. A call to Display.update()

Answers

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

Explanations

1. Does MapEntry use the key’s hashCode() for hashing?

A. Yes

B. No

C. Only if the key is a String

D. Only for null keys

MapEntry does use the key's hashCode() method for hashing as it is used to determine the index in the hash table where the key-value pair will be stored. Option B is incorrect because if the hash code is not used for hashing, then the key-value pair will be stored in a default index, which can result in slower retrieval time. Option C is incorrect because it implies that only String keys use the hashCode() method, which is not true as any object can override this method. Option D is incorrect because the hashCode() method is used for all keys, including null keys.

2. What does the 'synchronized' keyword prevent?

A. Variable mutations

B. Race conditions

C. Syntax errors

D. All of the above

The 'synchronized' keyword is specifically used in Java to provide mutual exclusion and monitor locking in order to prevent race conditions, which occur when multiple threads are accessing and modifying the same shared resource at the same time. Option A, variable mutations, is not a specific term related to concurrency, but it could be referring to the general concept of preventing unwanted changes to variables. Option C, syntax errors, is not related to concurrency at all. Option D, all of the above, includes incorrect options and is therefore not a viable answer.

3. What does polymorphism allow in object-oriented programming?

A. Objects to change their type at runtime

B. A single interface to be used for different underlying forms of data

C. For objects to exist in multiple states simultaneously

D. For objects to automatically delete themselves once they are no longer needed

Polymorphism allows for a single interface to be used for different underlying forms of data. This means that although an object may have many different forms or classes, it can be used interchangeably through a common interface. Option A is incorrect because objects cannot change their type at runtime. Once an object is created, its type is fixed and cannot be changed. Option C is also incorrect because objects cannot exist in multiple states simultaneously. Objects can only have a single state at any given time. Option D is incorrect because objects do not automatically delete themselves. The programmer is responsible for managing the memory of the objects they create.

4. Which method would you use to start a previously created thread?

A. init()

B. begin()

C. start()

D. run()

Once a thread is created, the start() method is used to set the thread in motion and allow it to execute concurrently with the main thread. The init(), begin(), and run() methods are not valid options for starting a thread. The init() method is used to initialize the thread, the begin() method has been deprecated and should not be used, and the run() method is used to execute code within the thread, but it does not start the thread. Therefore, the correct method to use in order to start a previously created thread is the start() method.

5. Why can't you upcast a List<Apple> to a List<Fruit>?

A. Because the List types are invariant

B. Because a Fruit is not an Apple

C. Because of runtime type information

D. Because generics do not support covariance by default

Upcasting is the process of converting an object of a subtype to its supertype. It is usually allowed when working with inheritance, but it is not possible with generic types by default. This is because generics are invariant in Java, meaning that subtypes cannot be used in place of their supertype. So, even though an Apple is a subtype of Fruit, a List<Apple> cannot be used as a List<Fruit>. This is because it would violate type safety as the List<Fruit> could end up containing other types of Fruit that are not Apples. Therefore, options A, B, and C are incorrect because they do not address the specific reason why upcasting a List<Apple> to a List<Fruit> is not allowed in Java.

6. What types of objects can List<?> accept via its add method?

A. Objects of any type

B. Only objects of the specified wildcard type

C. Only null

D. Only objects of the base class of the wildcard type

A List<?> can accept objects of any type via its add method. This means that the specific type of element being added to the list does not need to be specified. Therefore, options A and B are incorrect. Option C, which states that the list can only accept null, is also incorrect. While it is true that null is a valid object and can be added to a list, it is not the only type of object that can be added using the add method. Option D is incorrect because a List<?> can accept objects of the wildcard type as well as any of its subclasses. So it is not limited to accepting only objects of the base class of the wildcard type.

7. Which method must be called by the user when an InputFile object is no longer needed?

A. close()

B. dispose()

C. finalize()

D. clean()

The other options are incorrect because A) close() and dispose() both perform similar functions, but dispose() is used more specifically for disposing of resources used by the InputFile object. C) finalize() is used to perform any final actions on an object before it is destroyed, but it is not specific to the InputFile object. D) clean() is not a method used for handling objects and resources.

8. What kind of types can Java generic methods operate on?

A. Only Object types

B. Specifically typed objects according to the bounds

C. Any object that extends Throwable

D. Only numeric types

Generic methods in Java can operate on any specified type, making option A incorrect. Option C is incorrect because Throwable is a specific class and not all objects extend from it. Option D is incorrect because generic methods can operate on any specified type, not just numeric. The correct option, B, is the only one that accurately describes the types that generic methods can operate on, as they are specifically typed objects according to the bounds.

9. What is true about methods in a Java interface?

A. They can have method bodies if marked static

B. They are automatically public

C. They must be abstract

D. They can be private

Interface methods are automatically public, meaning they can be used and accessed by any other class. Option A is incorrect because static methods do not require an object to be called and interfaces cannot have objects. Option C is incorrect because all interface methods are abstract, meaning they do not have a method body. Option D is incorrect because all interface methods are public and therefore cannot be private.

10. Within the SWT version of ColorBoxes, what causes the display to update after a color change?

A. redraw() method call

B. Changing the RGB values directly

C. Calling the system's native paint method

D. A call to Display.update()

In the SWT version of ColorBoxes, calling the redraw() method is what causes the display to update after a color change. Option B, changing the RGB values directly, does not automatically cause the display to update and would require a redraw() call. Option C, calling the system's native paint method, is not a part of SWT and therefore would not be used to update the display. Option D, a call to Display.update(), is used to continuously update the display and is not specific to color changes.

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://thinkinginjava.examzify.com

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