Sunday, September 10, 2023

Top 10 Tricky Java interview questions and Answers

What is a tricky question? Well, tricky Java interview questions are those questions that have some surprise element on them. If you try to answer a tricky question with common sense, you will most likely fail because they require some specific knowledge. Most of the tricky Java questions come from confusing concepts like method overloading and overriding, Multi-threading which is really tricky to master character encoding, checked vs unchecked exceptions, and subtle Java programming details like Integer overflow. The most important thing to answer a tricky Java question is attitude and analytical thinking, which helps even if you don't know the answer.

Anyway in this Java article, we will see 10 Java questions that are really tricky and requires more than average knowledge of Java programming language to answer them correctly. As per my experience, there are always one or two tricky or tough Java interview questions on any core Java or Java EE interviews, so it's good to prepare tricky questions from Java in advance.

If I take an interview, I purposefully put this kind of question to gauge the depth of the candidate's understanding of Java. Another advantage of asking such a question is the surprising element, which is a key factor to put the candidate under some pressure during interviews.

Since these questions are less common, there is a good chance that many Java developer doesn't know about them. You won't find these questions even on popular Java interview courses like the **Java Interview Guide: 200+ Interview Questions and Answers**, which is nevertheless an excellent guide for Java interviews.

10 Tricky Java interview question - Answered

Here is my list of 10 tricky Java interview questions, Though I have prepared and shared a lot of difficult core Java interview questions and answers, I have chosen them as the Top 10 tricky questions because you can not guess answers to these tricky Java questions easily, you need some subtle details of Java programming language to answer these questions.

1. Question: What does the following Java program print?

```
public class Test {
    public static void main(String[] args) {
        System.out.println(Math.min(Double.MIN_VALUE, 0.0d));
    }
}
```

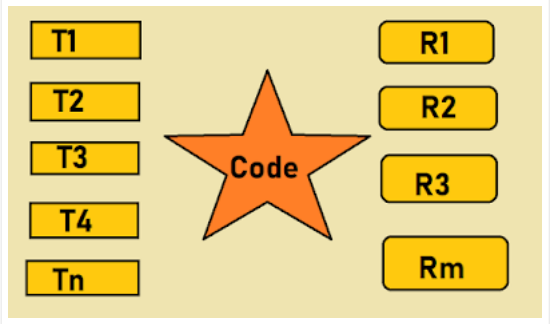
Answer: This question is tricky because unlike the Integer, where `MIN_VALUE` is negative, both the `MAX_VALUE` and `MIN_VALUE` of the `Double` class are positive numbers. The

- interview questions - java tough
- interview questions - java thread
- interview questions - coding
- interview questions - linux
- interview questions - web service
- interview questions - java advanced
- interview questions - software design
- interview questions - java inheritance
- interview questions - OOP
- interview questions - android
- interview questions - SQL
- interview questions - java tricky
- interview questions - REST
- interview questions - array
- interview questions - servlet jsp
- interview questions - EJB
- interview questions - java collection
- interview questions - design pattern
- interview questions - spring
- interview questions - hibernate
- interview questions -

`Double.MIN_VALUE is 2^(-1074)`, a double constant whose magnitude is the least among all double values.

So unlike the obvious answer, ***this program will print 0.0*** because of `Double.MIN_VALUE` is greater than 0. I have asked this question to a Java developer having experience of up to 3 to 5 years and surprisingly almost 70% of candidates got it wrong.

If you are not familiar with essential Java data types and wrapper classes like Double and Float then I highly recommend you to join a comprehensive Java course like **The Complete Java Masterclass** on Udemy to learn them, they are very very important not just for interviews but also for day to day Java work.



2. What will happen if you put the return statement or System.exit () on the try or catch block? Will finally block execute?
This is a very popular tricky Java question, and it's tricky because many programmers think that no matter what, but the finally block will always execute. This question challenge that concept by putting a return statement in the try or catch block or calling `System.exit()` from try or catch block.

The answer to this tricky question in Java is that finally, the block will execute even if you put a return statement in the try block or catch block but finally block won't run if you call `System.exit()` from the try or catch block.

3. Question: Can you override a private or static method in Java?
Another popular Java tricky question, As I said method overriding is a good topic to ask trick questions in Java. Anyway, you can not override a private or static method in Java, if you create a similar method with the same return type and same method arguments in child class then it will hide the superclass method, this is known as method hiding.

Similarly, you cannot override a private method in subclass because it's not accessible there, what you do is create another private method with the same name in the child class.

4. Question: What do the expression 1.0 / 0.0 will return? will it throw Exception? any compile-time error?
Answer: This is another tricky question from the Double class. Though Java developer knows about the double primitive type and Double class, while doing floating-point

core java
- interview questions - arraylist
- interview questions - java enum
- interview questions - java swing
- interview questions - java common
- interview questions - support
- interview questions - technical
- interview questions - java main
- interview questions - hashmap
- interview questions - java date

Blog Archive

▶ 2024 (74)

▼ 2023 (592)

▶ December (10)

▶ November (2)

▶ October (9)

▼ September (178)

How to find length/size of ArrayList in Java? Example

How to remove all elements of ArrayList in Java - ...

How to loop over a TreeSet in Java with Example

How to Convert Vector to Array in

arithmetic they don't pay enough attention to `Double.INFINITY`, NaN, and `-0.0` and other rules that govern the arithmetic calculations involving them.

The simple answer to this question is that it will not throw `ArithmeticExcpetion` and return `Double.INFINITY`.

Also, note that the comparison `x == Double.NaN` always evaluates to false, even if x itself is a NaN. To test if x is a NaN, one should use the method called `Double.isNaN(x)` to check if the given number is `NaN` or not. If you know SQL, this is very close to NULL there.

Btw, If you are running out of time for your interview preparation, you can also check out my book **Grokking the Java Interview** for more of such popular questions,

5. Does Java support multiple inheritances?

This is the trickiest question in Java if C++ can support direct multiple inheritances then why not Java is the argument Interviewer often give. The answer to this question is much more subtle than it looks like, because Java does support multiple inheritances of Type by allowing an interface to extend other interfaces, what Java doesn't support is multiple inheritances of implementation.

This distinction also gets blurred because of the default method of Java 8, which now provides Java, multiple inheritances of behavior as well. See why multiple inheritances are not supported in Java to answer this tricky Java question.

6. What will happen if we put a key object in a HashMap which is already there?

This tricky Java question is part of another frequently asked question, How HashMap works in Java. HashMap is also a popular topic to create a confusing and tricky question in Java.

The answer to this question is if you put the same key again then it will replace the old mapping because HashMap doesn't allow duplicate keys. The Same key will result in the same hashcode and will end up at the same position in the bucket.

Each bucket contains a linked list of `Map.Entry` object, which contains both Key and Value. Now Java will take the Key object from each entry and compare it with this new

Java? 2 Examples

How to sort a LinkedList in Java? Example Tutorial

10 Example of List in Java

How to Convert a List to a Set in Java with Example

Difference between ArrayList and HashMap in Java

How to sort HashSet in Java? Example

How to declare ArrayList with values in Java? Exam...

How to add element at first and last position of l...

Difference between Class and Record in Java?

Difference between HashMap and LinkedHashMap in Java

Difference between TreeMap and TreeSet in Java? An...

Difference between FileReader vs FileInput

key using the `equals()` method, if that returns true then the value object in that entry will be replaced by the new value. See How HashMap works in Java for more tricky Java questions from HashMap.

7. Question: What does the following Java program print?

```
public class Test {
    public static void main(String[] args) throws Exception {
        char[] chars = new char[] {'\u0097'};
        String str = new String(chars);
        byte[] bytes = str.getBytes();
        System.out.println(Arrays.toString(bytes));
    }
}
```

Answer: The trickiness of this question lies in character encoding and how String to byte array conversion works. In this program, we are first creating a String from a character array, which just has one character `'\u0097'`, after that, we are getting the byte array from that String and print that byte.

Since `\u0097` is within the 8-bit range of byte primitive type, it is reasonable to guess that the `str.getBytes()` call will return a byte array that contains one element with a value of -105 `((byte) 0x97)`.

However, that's not what the program prints and that's why this question is tricky. As a matter of fact, the output of the program is operating system and locale dependent. On a Windows XP with the US locale, the above program prints [63], if you run this program on Linux or Solaris, you will get different values.

To answer this question correctly, you need to know about how Unicode characters are represented in Java char values and in Java strings, and what role character encoding plays in `String.getBytes()`.

In simple word, to convert a string to a byte array, Java iterates through all the characters that the string represents and turn each one into a number of bytes and finally put the bytes together. The rule that maps each Unicode character into a byte array is called a character encoding.

So It's possible that if the same character encoding is not used during both encoding and decoding then the retrieved value may not be correct. When we call `str.getBytes()` without specifying a character encoding scheme, the JVM uses the default character encoding of the platform to do the job.

The default encoding scheme is an operating system and locale-dependent. On Linux, it is `UTF-8` and on Windows with a US locale, the default encoding is `Cp1252`. This explains the output we get from running this program on Windows machines with a US locale.

No matter which character encoding scheme is used, Java will always translate Unicode characters not recognized by the encoding to 63, which represents the character U+003F (the question mark, ?) in all encodings.

Stream i...

How to Create Read Only and Unmodifiable ArrayList...

Difference between HashSet and TreeSet in Java

How to handle click event in jQuery - Example

Top 5 Java Main method Interview Questions with An...

Difference between NoClassDefFoundError vs ClassNo...

Difference between Public, Package, Private and Pr...

Difference between HashSet vs TreeSet in Java? [An...

What is fail safe and fail fast Iterator in Java?

Difference between ROW_NUMBER(), RANK() and DENSE_...

Difference between Method and

8. If a method throws NullPointerException in the superclass, can we override it with a method that throws RuntimeException?

One more tricky Java question from the overloading and overriding concept. The answer is you can very well throw a superclass of RuntimeException in overridden method, but you can not do the same if it's checked Exception. See Rules of method overriding in Java for more details.

9. What is the issue with the following implementation of the compareTo() method in Java where an id is an integer number?

```
public int compareTo(Object o){
   Employee emp = (Employee) o;
   return this.id - e.id;
}
```

Well, there is nothing wrong with this Java question until you guarantee that id is always positive. This Java question becomes tricky when you can't guarantee that the id is positive or negative.

The tricky part is If id becomes negative then **subtraction may overflow** and produce an incorrect result. See How to override the compareTo method in Java for the complete answer to this Java tricky question for an experienced programmer.

10. How do you ensure that the N thread can access N resources without deadlock? (answer)

If you are not well versed in writing multi-threading code then this is a really tricky question for you. This Java question can be tricky even for the experienced and senior programmers, who are not really exposed to deadlock and race conditions.

The key point here is ordering, if you acquire resources in a particular order and release resources in the reverse order you can prevent deadlock. If you want to prepare multithreading and concurrency in-depth, I highly recommend **Java Multithreading for the Senior Engineering Interviews** course on Educative.

11. Question: Consider the following Java code snippet, which is initializing two variables and both are not volatile, and two threads T1 and T2 are modifying these values as follows, both are not synchronized

Construct or in Java ...

What is static in Java? Example Tutorial

Can you make an Abstract Class or Method Final in ...

What is Variable and Method Hiding in Java - Examp...

Difference between Abstract class and Interface in...

How to Fix java.lang.OutOfMemoryError: Metaspace i...

Java Interface Example Tutorial

How Constructor Chaining works in Java - Example

5 Rules of Method Overloading and Overriding in Ja...

Difference between Method Overloading and Overridi...

Can You Override Private Method in Java ? Example

How to fix java

```
int x = 0;
boolean bExit = false;

Thread 1 (not synchronized)
x = 1;
bExit = true;

Thread 2 (not synchronized)
if (bExit == true)
System.out.println("x=" + x);
```

Now tell us, is it possible for Thread 2 to print "x=0"?

Answer: It's impossible for a list of tricky Java questions to not contain anything from multi-threading. This is the simplest one I can get. The answer to this question is Yes, It's possible that thread T2 may print x=0. Why? because without any instruction to compiler e.g. synchronized or volatile, `bExit=true` might come before x=1 in compiler reordering. Also, x=1 might not become visible in Thread 2, so Thread 2 will load x=0. Now, how do you fix it?

When I asked this question to a couple of programmers they answer differently, one suggests making both threads synchronized on a common mutex, another one said to make both variables volatile. Both are correct, as they will prevent reordering and guarantee visibility.

But the best answer is you just need to make `bExit` as volatile, then Thread 2 can only print "x=1". x does not need to be volatile because x cannot be reordered to come after `bExit=true` when `bExit` is volatile.

12. What is the difference between CyclicBarrier and CountDownLatch in Java?
Relatively newer Java tricky question, only been introduced from Java 5. The main difference between both of them is that you can reuse CyclicBarrier even if the Barrier is broken, but you can not reuse CountDownLatch in Java. See CyclicBarrier vs CountDownLatch in Java for more differences.

13. What is the difference between StringBuffer and StringBuilder in Java?
Classic Java questions which some people think are tricky and some consider very easy. StringBuilder in Java was introduced in JDK 1.5, and the only difference between both of them is that StringBuffer methods like length(), capacity(), or append() are synchronized while corresponding methods in StringBuilder are not synchronized.

Because of this fundamental difference, concatenation of String using StringBuilder is faster than StringBuffer. Actually, it's considered a bad practice to use StringBuffer anymore, because, in almost 99% of scenarios, you perform string concatenation on the same thread. See StringBuilder vs StringBuffer for more differences.

14. Can you access a non-static variable in the static context?
Another tricky Java question from Java fundamentals. No, you can not access a non-static variable from the static context in Java. If you try, it will give a compile-time error.

module error "Caused by: java.lang. ..

Difference between Class and Interface in Java and...

10 Essential Object Oriented Concepts for Java Dev...

Difference between Static binding vs Dynamic bindi...

What is Inheritance in Java with example - Object ...

19 Java and OOP Method Overloading and Overriding ...

Difference between Abstraction and Polymorphism in...

Difference between instance and Object in Java

How to create a class with methods and attributes ...

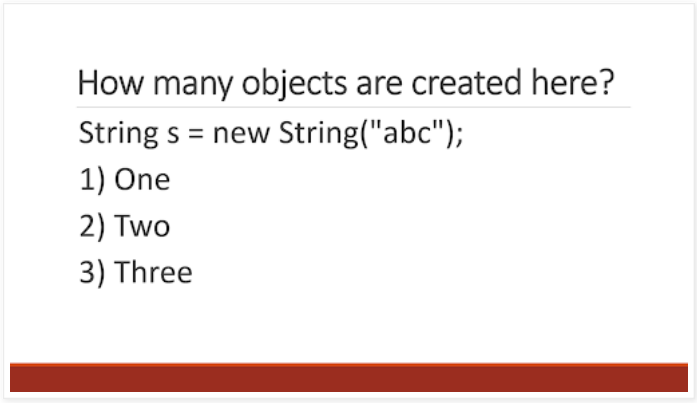
How to convert a List to Array in Java? Example T...

This is actually a common problem beginners in Java face when they try to access instance variables inside the main method.

Because main is static in Java, and instance variables are non-static, you can not access instance variable inside main. See, why you can not access a non-static variable from the static method to learn more about these tricky Java questions.

15. How many String objects are created by the following code?

you might be thinking "one" object but that's wrong. Btw, if you don't find these questions tricky enough, then you should check Joshua Bloch's other classic book, Java Puzzlers for super tricky questions. I am sure you will find them challenging enough.



More Trick Java Questions for Practice?

Now, it's practice time, here are some questions for you guys to answer, these are **contributed by readers of this blog**, big thanks to them.

1. When doesn't Singleton remain Singleton in Java?
2. is it possible to load a class by two ClassLoader?
3. is it possible for equals() to return false, even if the contents of two Objects are the same?
4. Why compareTo() should be consistent to equals() method in Java?
5. When do Double and BigDecimal give different answers for equals() and compareTo() == 0.
6. How does "has before" apply to volatile work?
7. Why is 0.1 * 3 != 0.3,
8. Why is (Integer) 1 == (Integer) 1 but (Integer) 222 != (Integer) 222 and which command arguments change this.
9. What happens when an exception is thrown by a Thread?
10. Difference between notify() and notifyAll() call?
11. Difference between System.exit() and System.halt() method?
12. Does following code legal in Java? is it an example of method overloading or overriding?

```
public String getDescription(Object obj){
   return obj.toString;
}
public String getDescription(String obj){
   return obj;
```

How to search a LinkedList in Java? Example

How to get first and last elements form ArrayList ...

Difference between synchronized ArrayList and Copy...

PriorityQueue in Java? Example Tutorial

10 Common Coding Mistakes Every Java Developers Sh...

10 Examples of HashMap in Java - Programming Tutorial

How to convert a Map to List in Java? HashMap to A...

How HashSet works in Java [Explained with Example]

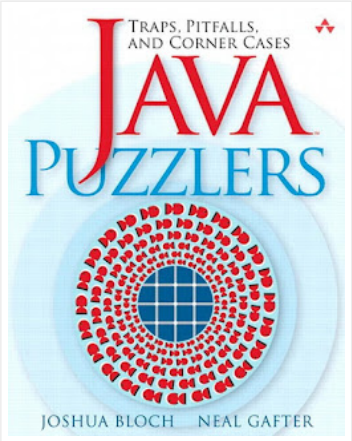
How get() and put() methods of HashMap works in Ja...

4 ways to read String from File

```
}
and
public void getDescription(String obj){
   return obj;
}
```

This was my list of Some of the most common tricky questions in Java. It's not a bad idea to prepare a tricky Java question before appearing for any core Java or J2EE interview. One or two open-ended or tricky question is quite common in Java interviews.

And, If you are looking for super challenging trick coding questions then you should check out Joshua Bloch another classic book, the **Java Puzzlers**, I am sure you'll find them really challenging to solve, I certainly did.



Hungry for more **Java Interview Question and Answer** posts, check out these articles

- 18 Java design pattern questions asked in interviews
- 10 Java coding interview questions answer for 2 to 4 years experience
- Top 21 Most Frequently Asked Java Questions and Answers
- 133 Core Java Interview Questions from last 5 years
- Top 50 Multithreading and Concurrency Interview Questions
- 21 Frequently asked SQL queries from Java Interviews
- 20 Spring and REST Interview Questions
- 75 Coding Questions to Crack Any Programming Interview
- 100+ Data Structure and Algorithm Questions
- 10 Courses to learn Java for Beginners
- My Favorite books to learn Java Programming in depth
- 25 Spring Security Interview Questions for Java Programmers
- Top 20 Spring Boot Interview Questions with Answers
- 15 Java Microservice Interview Questions with Answers
- 15 Spring Data JPA Interview Questions with Answers
- 15 Spring Cloud Interview Questions with Answers
- 15 Best Java Stream and Functional Programming Questions with Answers

Thanks for reading this article so far. If you like these *tricky Java interview questions* or have seen them on your telephonic round of interviews, then please share this post with your friends and colleagues on Facebook, Twitter, Email, etc. If you have any questions or feedback, please drop a note.

in Java - Example

How to check if a Key Object Exists in HashMap Jav...

How to sort HashMap by values in Java 8 [using Lam...

How to use ConcurrentHashSet from ConcurrentHashMa...

How to initialize HashMap with values in Java? Exa...

How to Sort HashMap in Java based on Keys and Values

How to create a gzip file in Java? Example Tutorial

Right way to check if String is empty in Java with...

Java 8 StringJoiner Example - How to join multiple...

How to Join Multiple Strings in Java 8 - String jo...

10 Points about String in Java with Examples

All the best for your Job interview.

Posted by javin paul at 9/10/2023.08:03:00.AM

Labels: core java interview question answer, interview questions, java

74 comments:

Anonymous September 10, 2012 at 9:58 PM

Great questions, How about adding tricky questions related to programming exercise ?

Reply

Replies

Anonymous August 26, 2013 at 7:58 PM

One of the most tricky questions, I have face in a Java interview was, Does two object will always be equal, when there compareTo() method returns zero? I said, Yes, but that was not true. Though most of the classes will be equal if there compareTo() return true e.g. java.lang.String, but it's not mandatory. compareTo() may be inconsistent to equals(), which means compareTo() may return zero, but object will not be equal by equals() method. One of the prime example of this is java.math.BigDecimal class, whose equals() method return true if two BigDecimal object is equal in both value and scale e.g. 6.0 and 6.00 will not be equal, but compareTo() will return zero if both objects are compared. This was really tricky, until you had faced similar question previously. In another interview, my friend was asked this question little differently, Can we store BigDecimal class in TreeSet? obviously No, because of above reason, BigDecimal class can produce unexpected behavior when stored in SortedSet or SortedMap.

Anonymous September 12, 2016 at 6:46 AM

try here http://quizful.com/quizzes/java
as for me really tricky, I was really surprised with answers)

Emerikol December 15, 2016 at 1:05 PM

While I'm an certain these questions are getting asked, I find it ironic that being able to answer them in no way indicates any ability to be a

How to Remove all white spaces from String in Java...

3 ways to convert String to byte array in Java - E...

Java - Convert String to Boolean Example

How to convert lowercase String to uppercase in Ja...

How to Compare and Sort String by their length in ...

Difference between == and equals() method in Java?...

How to check String contains a text in Java? conta...

How to Send Asynchronous HTTP Requests using HttpC...

Difference between String and StringBuffer in Java...

7 Examples of formatting String Java -

successful programmer. That is why they are trick questions I guess.

Reply

Harish September 21, 2012 at 3:18 AM

Good work done! It will definitely help me in my interviews... :)
thanks .. :)

Reply

Summary September 27, 2012 at 2:20 AM

Hi ,

I am able to override public static method declared in base class in its subclass.
It didn't throw any compilation or runtime exception

Reply

Replies

Anonymous September 27, 2012 at 8:06 PM

It won't because compiler will treat it as different method. This is called method hiding and its indeed one of the tricky Java question. Remember that this method is not overriding super class method as static method bonded using type information. That's the reason this Java question is tricky.

Reply

Suvi October 12, 2012 at 12:56 AM

One of the tricky Java question I faced is "What is difference between Collection and Generics in Java", its not tricky because I don't know either collection or Generics but How can you compare Generics with Collection ?

Reply

Replies

satish March 29, 2018 at 1:37 PM

Correct both are diff things and they are not related at all..

Reply

Unknown October 29, 2012 at 4:36 AM

Hi
Is the line in codes correct.
It's one of ur question.

Can you access non static variable in static context?
Another tricky Java question from Java fundamentals. "No you can not access static variable in non static context in Java". Read why you can not access non-static variable from static method to learn more about this tricky Java questions.

Reply

Replies

javin paul October 29, 2012 at 6:05 AM

String.for m...

How to split String by comma in Java - Example Tut...

5 Examples of substring( ) in Java

2 ways to parse String to int in Java - Example Tu...

6 ways to convert char to String in Java - Example.. .

4 ways to concaten ate Strings in Java [Example and...

How to convert String to Float in Java and vice-v...

How to Generate Random Number between 1 to 10 - Ja...

How to Make Executabl e JAR file in Eclipse IDE - J...

Difference between @GetMa pping @PostMa pping @PutMa.. .

How to Perform Binary Tree InOrder

Hi Ankur, Thanks for pointing out, you should read opposite i.e non static variable can not accessible from static context. This always confuse if you don't remember it and that's why its one of the tricky question.

Unknown January 13, 2013 at 7:59 AM

Hi
This line is not correct.
"No you can not access static variable in non static context in Java".

Anonymous April 10, 2015 at 10:50 PM

you can not access a static variable inside the non-static one,other-way its so true.

Reply

rani December 3, 2012 at 6:30 AM

all are tricky questions?

Reply

swapnil December 6, 2012 at 3:32 AM

its really nice tricky question to read to face interview.

Reply

Anonymous December 17, 2012 at 3:41 AM

these are not tricky rather easy questions ...

Reply

Anonymous December 18, 2012 at 11:11 AM

i agree nice question to prepare before appearing for interview

Reply

Tafadzwa Kombe January 5, 2013 at 10:17 AM

These are definitely a good set of tricky questions that a candidate may face during an interview. I certainly enjoyed going through them but I have a different opinion regarding the last question: Can you access non static variable in static context?

I think the answer to that question is a YES albeit one should note that you need an object reference of the associated class to access the variable (I assume this variable is an instance variable). And likewise you can directly access a static variable in non static context too. Please refer to this link for more information http://docs.oracle.com/javase/tutorial/java/javaOO/classvars.html

Reply

Anonymous February 11, 2013 at 8:36 AM

Is the compareTo code correct ? the references look to be wrong.

Reply

Ankita Dutta February 20, 2013 at 12:51 AM

The explanation of the last question is little bit confusing. Actually static variables are always accessible in non static context however the reverse is not true.

traversal in Ja...

How to Remove all adjacent duplicates characters f...

How to Find Duplicate Characters in String [Java C...

How to Implement Binary Tree InOrder traversal in ...

How to remove duplicate characters from String in ...

10 Programming questions and exercises for Java Pr...

How to Find the Largest and Smallest of Three Numb...

How to find median of two sorted arrays in Java? E...

Fibonacci Series in Java Using Recursion

How to Implement a Power Function in Java? Example...

How to download

Reply

Replies

Anonymous September 23, 2013 at 8:21 AM

In static context, you can access non-static variables - by creating a new instance of the object. Hence the reverse is partly true.

Reply

Anonymous February 23, 2013 at 1:05 PM

public int compareTo(Object o){
Employee emp = (Employee) emp;
return this.id - o.id;
}

I think this is what it should be...

public int compareTo(Object o){
Employee emp = (Employee) o;
return this.id - o.id;
}

Reply

Replies

Unknown March 27, 2013 at 11:57 AM

Wrong, it should be:

public int compareTo(Object o){
Employee emp = (Employee) o;
return this.id - emp.id;
}

Anonymous February 17, 2014 at 3:31 AM

I should also mention that you shouldn't cast to Employee without knowing it's an Employee via instanceof, or just using Comparable generic interface.

Reply

Unknown February 28, 2013 at 10:22 PM

Try some more new

Reply

Anonymous March 12, 2013 at 12:43 AM

Q 1. about try and catch block
class Main
{
public static void main(String args[])
{

try
{
int a=2/0;
System.exit(0);

a file using HTTP in Java? Example

How to reverse bits of an integer in Java? [LeetCo...

How to Reverse String in Java with or without Stri...

How to check is given String is a Palindrome in Ja...

How to check if strings are rotations of each othe...

Java Program to find Armstrong numbers with Example

▶ August (41)

▶ July (24)

▶ June (2)

▶ May (27)

▶ April (169)

▶ March (50)

▶ February (48)

▶ January (32)

▶ 2022 (329)

▶ 2021 (273)

▶ 2020 (1)

▶ 2017 (1)

▶ 2016 (1)

```
}


catch(Exception e)
{

System.out.println("i am in catch block");

}

finally
{

System.out.println("finally");
}


}
}
/*OUTPUT
i am in catch block
finally*/
```

finally block get executed if we place System.exit(0) in try block

Reply

▾ Replies

Anonymous April 4, 2013 at 3:40 AM

Before calling System.exit(0) you raised excetion "int a=2/0".
Onces system.exit(0) is called finally will never called.
plz Don't confused others :)

Anonymous May 29, 2013 at 1:48 AM

2/0 is arithmetic exception so before going to system.exit it will jump to catch block and then finally but once system.exit(0) executes ...finally will not execute then .. try it with

```
class Main
{
public static void main(String args[])
{

try
{
int a=2/1;
System.exit(0);
}


catch(Exception e)
{

System.out.println("i am in catch block");

}

finally
{
```

```
System.out.println("finally");
}


}
}
```

Anonymous July 18, 2013 at 4:29 AM

as per my java knowledge,try,finally blocks will execute parallely,because if single statement(inside try block first line or empty block) is executed corresponding catch may or may not execute but finally block will execute automatically,otherwise it wont execute,so without try&catch blocks we cant write finally block in java.

Unknown May 13, 2016 at 5:23 AM

System.exit(0) is unreachable statement here.

Reply

Unknown March 18, 2013 at 12:11 AM

I was asked during an interview the difference between "arraylist" and "linkedlist" and when I should use them
Also, found a site where you can give online Java mock interviews. Its www.preparestreet.com

Reply

Replies

Anonymous October 22, 2015 at 3:55 AM

linklist is dynamic is called at runtime to save memory. where as arrylist is not dynamic whose memory locations are allocated in memory so memory is already stored for that so memory is waste. if u have shortage of memory can use linklist it is envoke at runtime and if u have enough memory can use arraylist..

Unknown November 23, 2015 at 2:01 AM

Wrong. LinkedList uses more memory, because, apart from values, it stores a reference to the previous and next entry. The difference between them in the ways of access to the elements of the list, and add / remove items in the list

Reply

Fawad August 14, 2013 at 7:22 PM

What is so tricky about these question, to me they look most simplest question, which doesn't even qualify for Interviews. Tricky questions are those, who challenge your notion e.g.

When Singleton doesn't remain Singleton in Java?
is it possible to load a class by two ClassLoader?
is it possible for equals() to return false, even if contents of two Objects are same?
Why compareTo() should be consistent to equals() method in Java?
is Following code legal in Java? is it example of method overloading or overriding?

```
public String getDescription(Object obj){
return obj.toString;
```

```
}

public String getDescription(String obj){
return obj;
}

and

public void getDescription(String obj){
return obj;
}
```

Anyone disagree with my questions not being tricky?

Reply

Replies

Anonymous February 16, 2014 at 6:20 AM

Actually your questions are better than Javin's Questions...

Peter Lawrey May 5, 2014 at 9:17 AM

Trickier) When do Double and BigDecimal give different answers for equals() and compareTo() == 0.

Reply

Java Online August 29, 2013 at 2:05 AM

Hi,
Good collection of tricky questions, However I feel that the questions in the interview becomes tricky because we need to answer the same as what interviewer in thinking is correct, this is the most tricky part. It is always better to clarify the question correctly by rephrasing it.

Reply

Peter Lawrey May 5, 2014 at 9:13 AM

Some trickier questions.

1a) how do you prevent a return statement from calling finally.
1b) how do you make a System.exit() call the finally block.

3) Not sure multiple inheritance is the trickest question in Java. How about;
3a) how does "has before" apply to volatile work?
3b) why is 0.1 * 3 != 0.3,
3c) why is (Integer) 1 == (Integer) 1 but (Integer) 222 != (Integer) 222 and which command arguments change this.

7) how can you release synchronized locks in an order which is not the reverse order? (You can do this BTW)

Good answers to some common questions, I look forward to some trickier questions. ;)

Reply

Replies

Fawad June 27, 2014 at 10:29 PM

I would add :
What happens when exception is thrown in a Thread?
Difference between notify and notifyAll call?
Difference between System.exit() and System.halt() method?

do you agree these are much trickier question for an average Java programmers?

Unknown October 1, 2015 at 9:33 PM

Is there a System.halt() method exists in Java? I think it is Runtime.getRuntime().halt(status), right?

Reply

Tom Henricksen October 6, 2014 at 5:32 AM

Thanks for sharing these. I need some good Java interview questions and a few of these might help me out.

Reply

Replies

javin paul October 6, 2014 at 5:35 AM

No problem Tom, thanks for dropping by, If you have not read already, you may find my article about 10 Questions to make Programming Interview Less Costly useful as well.

Reply

Anonymous December 10, 2014 at 1:57 AM

Answer to the question **"is it possible to load a class by two ClassLoader?"** is Yes, it quite possible if you are using a custom class loader or working on managed environment which uses classloader e.g. web and application server. This is one more reason why you should use instanceof instead of getClass() while overriding equals() mehtod, otherwise equals() will return false even if object are same but classloader is different.

Reply

Unknown December 24, 2014 at 3:25 AM

hi..

this is hari.can you any one send me jsp&servlets 1.6 year interview questions.

this is my id:harikrishnapulipati@gmail.com
please help me...

Reply

Anonymous January 9, 2015 at 8:59 PM

How would you describe Enum<E extends Enum<E> > ?

Reply

Anonymous January 19, 2015 at 5:21 AM

About Q2: What will happen if you put return statement or System.exit () on try or catch block ? Will finally block execute?

Here is a code fragment (Java class) I found a long time ago on some C++ forum. It is called "JavaSucks" (sorry) and its content says it all:

```
public class JavaSucks {
public static void main(String[] args) {
for (;;) {
try {
System.out.println("Java sucks");
} catch (Exception e) {
System.exit(0);
} finally {
continue;
}
}

}
}
```

It will compile without errors? If yes, what do you think it will produce on console as output?
Compile it and run it. Result: no compilation errors, just one warning. It will print infinitely the string "JavaSucks"...

Reply

Anonymous September 30, 2015 at 12:45 AM

Pass by value and pass by reference is also a good tricky Java question. Btw, Java is always pass by value, even for objects its pass by value, its just that reference or handle to the object is passed.

Reply

Unknown October 12, 2015 at 11:23 AM

can we declare constructor as private??? if yes then for which purpose??

Reply

Replies

Vimal Panchal March 23, 2017 at 8:20 AM

We declare constructor as private if we want to implement singleton pattern. Also, If you want to create only 1 instance of your custom class then you can use private constructor. Please,write me if I am missing anything.

Reply

javin paul October 13, 2015 at 5:50 AM

Hi Sandeep, Yes, you can declare a private constructor in Java. You can do to prevent instantiation of class outside the class, for example, Singleton pattern is one of the prime examples of the private constructor. In Singleton, the class itself is responsible for creating the instance and managing it so constructor is made private.

Reply

Anonymous October 16, 2015 at 10:15 AM

A fiend who is a really good programmer was given some of these in an interview a few months ago.

His response was: "I'd fire who every wrote such lousy code. I never have to worry about such things, because I write defensive and don't allow such nonsense in my code base"

He didn't get the job, but he's right.Most of this "trick" questions are are just academic and should never occur to in real life. It's lousy people think these type of questions some how gauge the productivity and ability of a programmer who avoids such pitfalls to begin with and hence doesn't know the answers to the way out of them.

Reply

Anonymous October 19, 2015 at 8:56 PM

Some questions are really tricky especially "Why 01*3 != 0.3 in Java?", I doubt even experienced Java developers can answer with clarity and confidence. I have asked this question to my friends having 5 and 6 years of experience and my technical lead having 10 years of experience Java, but they couldn't provide me good answer. All they could say is, since some floating point numbers cannot be represented precisely in Java, hence 0.1*0.3 != 0.3

Reply

Anonymous December 3, 2015 at 6:54 PM

I think in practice questions "How does "has before" apply to volatile work?", you mean how "happens before" concept work with volatile variables? right? as far as I know there are certain rules which decides which update will happen first. I remember reading about it on Java concurrency in Practice, which says that a volatile write will happen before volatile read.

Reply

Javin December 5, 2015 at 8:06 AM

I have recently shared few more Java interview questions especially for developer with 1 to 4 years of experience, you can see it here

Reply

Unknown December 17, 2015 at 8:38 AM

Is null key allowed to stored in HashMap? If yes, how it is handled?
What will happen if two different HashMap key objects have same hashcode?
Visit http://modernpathshala.com/Learn/Java/Interview for more java interview questions and answers.

Reply

Anonymous May 18, 2016 at 10:07 PM

```
public class Test {
public static void main(String[] args) {
System.out.println(Math.min(Double.MIN_VALUE, 0.0d));
}
}
```

These type of questions will get you people in the industry who have crammed the language and don't know the design practices. No way to gauge a good programmer from a bad one

Reply

Anonymous June 8, 2016 at 9:01 PM

What would the following program will print
public class Main {

public static void main(String[] args) {
Collection c = new HashSet();
print(c);
}

public static void print(Collection c){
System.out.println("Collection");
}

public static void print(Set s){
System.out.println("Set");
}

public static void print(HashSet hs){
System.out.println("HashSet");
}

}
I said "HashSet" but it was wrong.

Reply

Replies

Unknown June 28, 2016 at 4:02 AM

Overriding static methods is called as method hiding.In method hiding method resolution is always based on reference type so it will print Collection.

Reply

javin paul June 9, 2016 at 5:09 AM

It will print "Collection" because methods are static and so they are bonded during compile time and at that time only type information is available because object is created at runtime.

Reply

Jake Beasley July 5, 2016 at 10:58 AM

Honestly, these questions are poor indicators of how effective they will be on the job. They may be entertaining, but they are really bad questions when screening job candidates for real work.

On the job you want people who are creative, capable of exploring available options, and capable of executing well on the tasks that they are assigned to them, large or small. These sort of "gotcha" questions do not actually tell you if they are good developers - just whether they have "happened" to stumbled upon obscure facts or details in their careers or have spend an inordinate amount of time reading java docs.

Reply

Replies

javin paul July 26, 2016 at 5:52 PM

Hello @Jake, I don't believe the go/no-go decision is based upon these tricky questions. They are mainly used to add the surprise element or check the depth of candidates knowledge in any particular topic. It's a

good indicator of candidate's deep and thorough understanding but as you said, don't expect every Java developer knows these subtle details.

Reply

Jangbahadur Patel July 22, 2016 at 7:57 PM

Wow Really Nice Collections of Interview Questions . Thank you :)

Reply

Replies

javin paul July 26, 2016 at 5:50 PM

Hello @Jang Bahadur, thanks you like these tricky Java questions, If you have any, please share with us as well :-)

Reply

Anonymous August 16, 2016 at 10:03 PM

One of these questions, CyclicBarrier vs CountDownLatch are asked to me on screening round, I managed to answer it well, thanks to internet and you guys.

Reply

Anonymous October 12, 2016 at 9:26 AM

web services consume or produce the web services if consume explain how

Reply

Replies

javin paul September 25, 2018 at 4:08 AM

It can do both. for example, HTTP GET request return something from server but by using POST and PUT you can also create new objects into Server.

Reply

adarsh January 23, 2017 at 3:27 AM

Very helpful

Reply

B-L-O-G November 22, 2017 at 5:40 PM

public int compareTo(Object o){ Employee emp = (Employee) o; return this.id - e.id; }

the issue with this code:
Compilation error.. E is undefined.

Reply

Replies

javin paul September 25, 2018 at 4:07 AM

What is E? there is no E here, btw, yes it is not using generics which it could and that way you don't need that cast.

Reply

Unknown September 9, 2018 at 7:33 AM

TreeSet allowed null? if it is allowed what is the senario. if it is not allowed what is the senario? please tell me with example

Reply

Replies

javin paul September 25, 2018 at 4:09 AM

I don't think TreeSet allows null becuase it's a SortedSet and it has to compare elements so if you try to compare null with anything it will result in NullPointerException, btw, it might just allow in case you have just one element. not sure but you can check that by running it yourself.

Unknown May 4, 2020 at 6:26 AM

Really good collection of interview quetion. Thanks

Reply

Hamid October 27, 2023 at 7:22 AM

Well, three is nothing wrong ---> Well, there is nothing wrong

Reply

Replies

javin paul October 28, 2023 at 1:27 AM

yes, indeed, the spelling is wrong :-)

Reply

Enter Comment

Feel free to comment, ask questions if you have any doubt.

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)