Register | Login | Logout | Contact Us

Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Core Java Interview Q&A › Collection and Data structures › 4 Java Collection Quizzes

4 Java Collection Quizzes

Posted on September 17, 2014 by Arulkumaran Kumaraswamipillai — No Comments ↓

Q1. When is an object needs to implement the Comparable interface?

a) When adding it to a HashSet.
b) When adding it to a TreeSet.
c) When adding it to a LinkedHashSet.
d) When adding it to any Java Collection class

A1 The answer is b.

1. **HashSet** is implemented using a HashMap, hence the elements are not ordered.

2. **TreeSet** is implemented using a tree structure(red-black tree algorithm). The elements in a set are sorted. It offers several methods to deal with the ordered set like first( ), last( ), headSet( ), tailSet( ), etc. Any object you add to it must

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

Ice Breaker Interview
Core Java Interview (
Java Overview (4)
Data types (6)
constructors-methc
Reserved Key Wor
Classes (3)
Objects (8)
OOP (10)
GC (2)
Generics (5)
FP (8)
IO (7)
Multithreading (12)
Algorithms (5)
Annotations (2)
Collection and Data
♦ Find the first n
♦ Java Collectior
♥ Java Iterable \
♥♦ HashMap & H

implement the Comparable interface, otherwise it throws a runtime exception

```
Exception in thread "main" java.lang.ClassCastExc
 at java.util.TreeMap.compare(TreeMap.java:1188)
 at java.util.TreeMap.put(TreeMap.java:531)
 at java.util.TreeSet.add(TreeSet.java:255)

```

3. TreeSet maintains the elements in the order in which the public int compareTo(String o) method is implemented. Any object that implements the Comparable interface must provide the public int compareTo(String o) method implementation.

4. LinkedHashSet is between a HashSet and TreeSet. It is implemented as a HashMap with a linked list running through it. So, it maintains the elements in the order in which they are added.

Q2. Given the following code, which of the statements are true?

```

List plainList = new ArrayList();
List<String> specList = new ArrayList<String> ();

```

a) adding a non String object to specList will throw a compile-time error
b) adding a non String object to specList will throw a runtime error
c) adding a non String object to specList will be slower than adding it to a plainList
d) adding a non String object to specList will throw a checked exception
e) you can add any objects to a plainList

A2. Answer is a & e.

Java generics check happens at compile-time. Learn more at Java Interview Questions & Answers: Compile-time versus

♦ Sorting objects
02: ♦ Java 8 Stre
04: Understandi
4 Java Collectio
If Java did not ha
Java 8: Different
Part-3: Java Tre
Sorting a Map by
When to use whi
⊞ Differences Betwe
⊞ Event Driven Progr
⊞ Exceptions (2)
⊞ Java 7 (2)
⊞ Java 8 (24)
⊞ JVM (6)
⊞ Reactive Programr
⊞ Swing & AWT (2)
⊞ JEE Interview Q&A (3
⊞ Pressed for time? Jav
⊞ SQL, XML, UML, JSC
⊞ Hadoop & BigData In
⊞ Java Architecture Inte
⊞ Scala Interview Q&As
⊞ Spring, Hibernate, & I
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A 1
⊞ ▶ Free Java Intervie

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience I

runtime. Generics is a bit tricky subject to master, and a must know topic to do well in coding and written tests.

Q3. Which of the following interfaces will you be concerned about when adding an object to a HashSet?

a) Collection
b) Set
c) SortedSet
d) List
e) All the above

A3. Answer is a and b.

The Set interface extends the Collection interface, and HashSet implements the Set interface. Hence a and b are correct. You can write either

```java
Collection<String> col = new HashSet<String>();
col.add("a");

```

or

```java
Set<String> col = new HashSet<String>();
col.add("a");

```

Coding to an interface is a best practice.

c) is wrong because the SortedSet is implemented by ConcurrentSkipListSet and TreeSet. if you write

```java
SortedSet<String> col = new HashSet<String>();
col.add("a");

```

you will get a compile time error “Type mismatch”. If you write as

- ⊞ Low Latency (7)
- ⊞ Memory Managemen
- ⊞ Performance (13)
- ⊞ QoS (8)
- ⊞ Scalability (4)
- ⊞ SDLC (6)
- ⊞ Security (13)
- ⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all

- ⊞ Setting up Tutorial (6)
- ⊞ Tutorial - Diagnosis (2
- ⊞ Akka Tutorial (9)
- ⊞ Core Java Tutorials (2
- ⊞ Hadoop & Spark Tuto
- ⊞ JEE Tutorials (19)
- ⊞ Scala Tutorials (1)
- ⊞ Spring & HIbernate Tu
- ⊞ Tools Tutorials (19)
- ⊞ Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all

- ⊞ Can you write code? (
- ⊞ ♦ Complete the given
- ⊞ Converting from A to I
- ⊞ Designing your classe
- ⊞ Java Data Structures
- ⊞ Passing the unit tests
- ⊞ What is wrong with th
- ⊞ Writing Code Home A
- ⊞ Written Test Core Jav

```
SortedSet<String> col = (SortedSet)new HashSet<St
col.add("a");

```

you will get a Runtime exception

Exception in thread “main” java.lang.ClassCastException: java.util.HashSet cannot be cast to java.util.SortedSet
at com.writtentest5.Def.main(Def.java:19)

Q4. What does the following code do? Can the **LinkedHashSet** be replaced with a **HashSet**?

```
import java.util.ArrayList;
import java.util.LinkedHashSet;
import java.util.List;

public class CollectionFunction {
    public <E> List<E> function (List <E> list) {
          return new ArrayList<E>(new LinkedHashS
    }
}
```

A4. The above code **removes duplicates** from a supplied list by passing it through an implementation of a Set interface. In this case, a LinkedHashSet is used to honor the ordering by implementing a **SortedSet** interface. If ordering is not required, the **LinkedHashSet** can be replaced with a **HashSet**.

Popular Posts

♦ 11 Spring boot interview questions & answers

823 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

765 views

18 Java scenarios based interview Questions and Answers

399 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

388 views

⊞ Written Test JEE (1)

How good are your .....?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

001B: ♦ Java architecture & design concepts interview questions & answers

201 views

Bio | **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 02: ♥♦ Java Compile-time Vs Run-time Interview Q&As

When to use which Java data structure? and why? ›

Posted in Collection and Data structures**,** member-paid**,** Written Test Core Java

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.