Collections Framework Interview Questions

1. What Is A Collection ?

Ans -> Collection : Java Collection Is A Framework. Whenever We Want To Represent Group Of Individual Object As A Single Intity Then We Will Use Collection.

2. In What Situation We Will Use Collection Object?

Ans -> If We Want To Perform Several Operation Like Storing Data, Searching, Sorting, Insertion, Deletion, And Updating Of Data. Then We Use Collection.

4. What Are The Three Problems Of Array Object For Creating Collection Api?

Ans -> 1. Arrays Are Fixed In Size.

2. Arrays Can Only Store Homogeneous Elements.

3. Arrays Do Not Have Built-In Methods.

5. What Is The Inbuilt Object Used In All Collections To Store Heterogeneous Objects?

Ans -> The Inbuilt Object Used In All Collections To Store Heterogeneous Objects In Java Is An Arraylist

6. When We Should Use Array Object And Collection Object, Which Will Give Us High Performance In Storing Objects?

Ans -> Collections Can Hold Heterogeneous Object, Whereas Arrays Can Only Store Homogenous Type Objects. Arrays Are Unable To Provide Ready-Made Solutions For User Needs Such As Sorting, Searching, And So On, But Collections Do. Arrays Are Not Preferred Over Collections When It Comes To Memory.

7. In Which Package Collection Api Is Given?

Ans -> Java.Util Package Contains All The Classes And Interfaces Required By The Collection Framework.

8. Why Java.Util Package Classes Are Called Collection Api Classes?

Ans -> The Classes In The Java.Util Package Are Called "Collection Api Classes" Because They Provide A Standard Way To Work With Collections Of Objects In Java Through A Set Of Interfaces And Classes.

9. What Is A Container Object Why Collection Objects Are Called Container Objects?

Ans. -> The Container Is An Object That Contains Other Objects. The Container Can Contain All Other Sheet Objects. The Objects Are Grouped Together And Have Common Settings For Font, Layout And Caption.

1

11. From Which Version Onwards Collection Api Has Been Called As Collection Framework?

Ans -> In Java, A Separate Framework Named The “Collection Framework” Has Been Defined In Jdk 1.2 Which Holds All The Java Collection Classes And Interface In It. In Java, The Collection Interface (Java. Util. Collection) And Map Interface

12. What Is The Meaning Of Legacy?

Ans -> Legacy Classes Are Those Classes Or Interfaces Which Already Existed Before The Collection In Java And Have Been Modified And Added To The Java Collection Framework

13. What Are The Legacy Collection Classes & Interfaces Available?

Ans -> The Legacy Classes And Interfaces Available In Java's Collections Framework Are:

Dictionary

Hashtable

Properties

Stack

Vector

Enumeration Interface

15. What Are The Primary Advantages Of Collection Api?

Ans. -> • Collections Are Growable In Nature. That Is Based On Our Requirement We Can Increase Or Decrease The Size.

• Collections Can Hold Both Homogeneous And Heterogeneous Elements.

• Every Collection Class Is Implemented Based On Some Standard Data Structure. Hence For Every Requirement Readymade Method Support Is Available.

19. What Are The 4 Problems Of Vector, & Its 4 Alternative Classes Given Java 1.2?

Ans -> 1. Synchronization Overhead : Vector Is Synchronized, Meaning That Every Method Call Is Thread-Safe. However, This Synchronization Adds Overhead, Which Can Impact Performance, Especially In Single-Threaded Applications Where Thread Safety Is Not A Concern.

2. Performance: Due To Its Synchronization And Dynamic Resizing Mechanism, Vector May Not Offer The Best Performance In All Scenarios, Especially When Compared To Non-Synchronized Alternatives.

3. Capacity Increment: Vector Automatically Increases Its Capacity When Needed By A Certain Amount, Which May Not Always Be Optimal. This Can Lead To Wasted Memory If The Capacity Is Increased By Too Much, Or Frequent Resizing If It's Increased By Too Little.

4. Enumeration: The Enumeration Provided By Vector Is Not Fail-Fast. If The Underlying Vector Is Modified During Iteration, It May Lead To Unpredictable Behavior Or Concurrent Modification Exceptions.

To Address These Issues, Java 1.2 Introduced Four Alternative Classes:

1. Arraylist: Arraylist Is Similar To Vector But Is Not Synchronized. It Provides Dynamic Arrays That Can Resize Themselves Automatically When Needed. This Lack Of Synchronization Makes It More Efficient In Single-Threaded Scenarios.

2. Linkedlist: Linkedlist Is A Doubly-Linked List Implementation That Provides Constant-Time Insertion And Removal Of Elements From Both Ends Of The List. While It Doesn't Offer Random Access Like Arraylist, It Can Be More Efficient For Certain Operations, Such As Frequent Insertions Or Removals In The Middle Of The List.

3. Hashset: Hashset Is An Implementation Of The Set Interface That Uses A Hash Table For Storage. It Does Not Allow Duplicate Elements And Provides Constant-Time Performance For Basic Operations Like Add, Remove, And Contains.

4. Hashmap: Hashmap Is An Implementation Of The Map Interface That Uses A Hash Table For Storage. It Provides Key-Value Mappings And Allows For Constant-Time Performance For Basic Operations Like Put, Get, And Remove.

21. What Are The Two Root Interfaces Of In Collection Framework?

Ans -> The Two Main Root Interfaces Of Java Collection Classes Are The Collection Interface And The Map Interface.

22. What Is The Difference Between Collection And Map?

Ans -> Collection :- Whenever We Want To Represent Group Of Individual Object As A Single Intity Then We Will Use Collection.

Map:- Whenever We Want To Represent A Group Of Object As A Key-Value Pair Then We Will Use Collection.

23. Why Map Is Not Created As Sub Interface Of Collection?

Ans -> Because They Are Of An Incompatible Type.

• List, Set And Queue Are A Collection Of Similar Type Of Objects But Just Values Where A Map Is A Collection Of Key And Value Pairs.

• List Set And Queue Have Add As A Method Which Takes A Value As Param To Add An Element Whereas Map Has Put As A Method Which Takes A Key And A Value As Params To Add A Key-Value Pair.

• List, Set And Queue Provide Iterate Functionality Over The Value Whereas Maps Has Keys To Iterate Over Which Is Ultimately A Set And Values As Collection.

25. What Are The Important Subclasses Of Collection & Map Interfaces?

Ans -> Collection Interface :

List: Ordered Collection Allowing Duplicates (Arraylist, Linkedlist, Vector).

Set: Unordered Collection With No Duplicates (Hashset, Linkedhashset, Treeset).

Queue: Collection Designed For Fifo (First-In-First-Out) Ordering (Linkedlist, Priorityqueue).

Deque: Double-Ended Queue Allowing Insertion/Removal At Both Ends (Arraydeque, Linkedlist).

Map Interface :

Hashmap: Key-Value Pairs Stored In A Hash Table With No Ordering Guarantees.

Linkedhashmap: Maintains Insertion Order Or Access Order.

Treemap: Sorted Key-Value Pairs Based On The Natural Order Or A Custom Comparator.

Hashtable: Synchronized Version Of Hashmap (Legacy Class).

Concurrenthashmap: Thread-Safe Map Supporting High Concurrency Without Blocking.

26. What Are The New Collection Classes Added In Java 1.2, 1.4, 1.5, 1.6, 1.7, & 1.8?

Ans -> Java 1.2: Linkedlist: A Doubly Linked List Implementation Of The List Interface.

Java 1.4: Linkedhashset: A Hash Table And Linked List Implementation Of The Set Interface, With Predictable Iteration Order. Identityhashmap: A Hash Table-Based Implementation Of The Map Interface, Where Comparisons For Equality Are Based On Reference Equality Instead Of Object Equality.

Java 1.5 (Java Se 5.0): Enumset: A Specialized Implementation Of The Set Interface For Use With Enum Types.

Enummap: A Specialized Implementation Of The Map Interface For Use With Enum Keys.

Concurrentlinkedqueue: A Non-Blocking, Concurrent Implementation Of The Queue Interface Based On Linked Nodes.

Copyonwritearraylist: A Thread-Safe Variant Of Arraylist Where All Mutative Operations (Add, Set, Remove, Etc.) Are Implemented By Making A Fresh Copy Of The Underlying Array.

Java 1.7 (Java Se 7):

Linkedblockingdeque: An Optionally Bounded Deque Implementation Backed By A Linked Node Structure.

Linkedtransferqueue: A Unbounded Queue Implementation That Additionally Supports Operations That Wait For Elements To Become Available.

Arraydeque: A Resizable Array Implementation Of The Deque Interface.

Concurrentskiplistset: A Concurrent, Navigable Set Implementation Based On A Skip List Data Structure.

Concurrentskiplistmap: A Concurrent, Navigable Map Implementation Based On A Skip List Data Structure.

Java 1.8 (Java Se 8):

Stampedlock: While Not A Collection Class Itself, It Is A New Lock Implementation Introduced In Java 8 That Provides Three Modes For Controlling Read/Write Access To A Resource.

Completablefuture: Again, Not A Collection Class, But A New Feature In Java 8 Providing A Way To Perform Asynchronous Computations And Handle Their Results Using Functional Programming Techniques.

28. What Is The Data Structure Used In List Type Collections Implementation?

Ans. -> Arraylist : Underlying Data Structure: Dynamic Array

Linkedlist : Underlying Data Structure: Doubly Linked List

Vector : Underlying Data Structure: Dynamic Array

Stack : Underlying Data Structure: Inherits From Vector

29. What Is The Data Structure Used In Set And Map Collections Implementation?

Ans. -> Hashset : Underlying Data Structure: Hash Table

Linkedhashset : Underlying Data Structure: Hash Table + Linked List

Treeset : Underlying Data Structure: Red-Black Tree (A Type Of Self-Balancing Binary Search Tree)

30. What Is The Difference Between The Three Words Collection, Collection & Collections?

Ans -> Collection :- A Generic Term Referring To Any Group Of Items Or Objects

Collection :- Collection Is An Interface If We Want To Represent Group Of Individual Object As A Single Intity Then We Should Go For Collection.

Collections :- Collections Is An Utility Classes Present In Java.Util Package To Difine Sevrel Utility Methods For Collection Object Like Shorting Searching Exe.

31. Among Set, List, Map Type Collections

1) Which Collection Is Unordered Collection?

Ans -> In Java, The Primary Unordered Collections Are Hashset For Sets And Hashmap For Maps. These Collections Do Not Maintain Any Specific Order Of Their Elements Or Keys.

2) Which Collection Is Indexed Ordered Collection?

Ans -> In Java, The Primary Collection That Is Both Indexed And Ordered Is The List Interface And Its Main Implementations, Such As Arraylist And Linkedlist.

3) Which Collection Is Sorted Collection?

Ans -> Treeset Treemap Is A Sorted Collection. This Means That The Elements In A Treeset And Treemap Are Stored In Ascending Order.

4) Which Collection Is Unique Collection?

Ans -> Set Is A Unique Collection. This Means That A Set Cannot Contain Duplicate Elements.

5) Which Collection Is Duplicate Collection?

Ans -> List Is A Duplicate Collection. This Means That A List Can Contain Duplicate Elements.

6) Which Collection Is Homogeneous Collection?

Ans -> An Treeset And Treemap Is A Homogeneous Collection. This Means That All Of The Elements In An Treesetand Treemap Must Be Of The Same Type.

7) Which Collection Is Heterogeneous Collection?

Ans -> A Hashset Is A Heterogeneous Collection. This Means That The Elements In A Hashset Can Be Of Different Types.

32. What Are The 9 Core Interface Of Collection Framework?

1. What Is The Use Of Collection Interface, Does It Have Any Direct 1 Implementations?

Ans. -> The Collection Interface Is A Root Interface Of The Java Collections Framework. It Defines A Common Set Of Method That All Collections Share.

2. What Is The Use Of Set Interface, What Are The Implementations Of Set Interface?

Ans -> A Set Is A Collection That Cannot Contain Duplicate Elements. It Models The Mathematical Set Abstraction. The Set Interface Contains Only Methods Inherited From Collection And Adds The Restriction That Duplicate Elements Are Prohibited.

3) What Is The Use Of List Interface, What Are The Implementations Of List Interface?

Ans -> List Is An Interface That Allow To Store Duplicate Value Where Insertion Order Is Preserve. Implementations Of List Arraylist, Linkedlist Vector.

4) What Is The Use Of Queue Interface, What Are The Implementation Of Queue?

Ans. -> It Is The Child Interface Of Collection. If We Want To Represent A Group Of Individual Objects Prior To Processing Then We Should Go For Queue.

5) What Is The Use Of Map Interface, What Are The Implementations Of Map Interface?

Ans. -> The Map Interface In Java Is Used To Store Key-Value Pairs, Where Each Key Is Associated With A Value. It Allows Efficient Retrieval And Manipulation Of Data Based On Keys. Implementations Of The Map Interface Include Hashmap, Treemap, Linkedhashmap, Hashtable, And Concurrenthashmap.

6) What Is The Use Of Sortedset Interface, What Are The Implementations Of Sortedset?

Ans -> It Is Child Interface Of Set. If We Want To Represent A Group Of Individual Object Without Duplicate According Some Sorting Order Then We Should Go For Sortedset Implementations Treeset

7) What Is The Use Of Navigablemap Interface, What Is Its Implementation Subclass?

33. What Are The Important Implemented Classes Of Collection & Map Interfaces?

1) When Should We Use Arraylist, Linkedlist, Vector, And Stack?

Ans. -> Arraylist :- Arraylist Is Best Suitable If Our Frequent Operation Is Retrieval Operation.

Linkedlist :- Use Linkedlist When You Need Frequent Insertions Or Deletions In The Middle Of The List.

Vector :- Vector Is Best Suitable If Our Frequent Operation Is Retrieval Operation.

Stack :- Use Stack When You Need A Last-In-First-Out (Lifo) Data Structure.

2) When Should We Use Priorityqueue?

Ans. ->

• This Is A Data Structure Which Can Be Used To Represent A Group Of Individual Objects Prior To Processing According To Some Priority.

• The Priority Order Can Be Either Default Natural Sorting Order Or Customized Sorting Order Specified By Comparator Object.

• If We Are Depending On Natural Sorting Order Then The Objects Should Be Homogeneous And Comparable Otherwise We Will Get Classcastexception

• If We Are Defining Our Own Sorting By Comparator Then The Objects Need Not Behomogeneous And Comparable.

• Duplicate Objects Are Not Allowed.

• Insertion Order Is Not Preserved And It Is Based On Some Priority.

• Null Insertion Is Not Possible Even As 1st Element Also.

3) When Should We Use Hashset

Ans. -> Hashset :-

• You Need To Keep A Collection Of Unique Items.

• You Don't Care About The Order Of The Items.

• You Want Fast Operations For Adding, Removing, And Checking If An Item Is In The Set.

35. What Is The Difference Between Set, List, Queue And Map?

Ans -> Set:- It Is The Child Interface Of The Collection.If We Want To Represent A Group Of Individual Objects As A Single Entity Where Duplicates Are Not Allowed And Insertion Order Won't Be Preserved. Then We Should Go For Set Interface.

List:- It Is The Child Interface Of Collection. If We Want To Represent A Group Of Individual Objects As A Single Entity Where Duplicates Are Allowed And Insertion Order Preserved. Then We Should Go For List.

Queue: It Is The Child Interface Of Collection. If We Want To Represent A Group Of Individual Objects Prior To Processing Then We Should Go For Queue.

36. What Is The Difference Between Set And Sortedset?

Ans. -> Set: General Collection Without Duplicates And With No Specific Order (Except In Linkedhashset Which Maintains Insertion Order).

Sortedset: A Specialized Set That Maintains Its Elements In A Sorted Order, Providing Additional Methods To Deal With Range Views And Ordering Specifics.

37. What Is The Difference Between Map & Sortedmap?

Ans. -> Map: General Collection Of Key-Value Pairs With Unique Keys And No Specific Order For The Keys (Except In Linkedhashmap Which Maintains Insertion Order).

Sortedmap: A Specialized Map That Maintains Its Keys In A Sorted Order, Providing Additional Methods To Deal With Range Views And Ordering Specifics.

What Is The Difference Between Hashset And Hashmap?

Ans. -> Hashset

Purpose: Stores Unique Elements.

Element Type: Only Values (No Key-Value Pairs).

Order: No Guaranteed Order.

Duplicates: Does Not Allow Duplicate Elements.

Implementation: Internally Uses A Hashmap With Dummy Values.

Usage: When You Need A Collection Of Unique Items.

Hashmap

Purpose: Stores Key-Value Pairs.

Element Type: Both Keys And Values.

Order: No Guaranteed Order.

Duplicates: Keys Must Be Unique; Values Can Be Duplicated.

Implementation: Uses A Hash Table To Store Entries.

Usage: When You Need To Map Keys To Values And Retrieve Values Based On Keys.

38. What Is The Difference Between Sortedset & Navigableset ?

Ans ->Sortedset

Ordered Collection: Maintains Elements In A Sorted Order Based On Their Natural Ordering Or A Specified Comparator.

Interface: Extends The Set Interface.

Methods: Provides Methods For Basic Operations And Additional Methods For Range View And Endpoints.

Methods: First(), Last(), Subset(), Headset(), Tailset()

Common Implementation: Treeset.

Navigableset

Ordered Collection: Maintains Elements In A Sorted Order, Same As Sortedset.

Interface: Extends The Sortedset Interface.

Additional Navigation Methods: Provides Navigation Methods To Find The Closest Matches For Given Search Targets.

Methods: Inherits All Sortedset Methods And Adds Methods Like Lower(), Floor(), Ceiling(), Higher(), Pollfirst(), Polllast(), And Descendingset().

Common Implementation: Treeset.

39. What Is The Difference Between Sortedmap & Navigablemap?

Ans. -> Sortedmap

Ordered Collection: Maintains Keys In A Sorted Order Based On Their Natural Ordering Or A Specified Comparator.

Interface: Extends The Map Interface.

Methods: Provides Methods For Basic Operations And Additional Methods For Range Views And Endpoints.

Methods: Firstkey(), Lastkey(), Submap(), Headmap(), Tailmap()

Common Implementation: Treemap.

Navigablemap

Ordered Collection: Maintains Keys In A Sorted Order, Same As Sortedmap.

Interface: Extends The Sortedmap Interface.

Additional Navigation Methods: Provides Navigation Methods To Find The Closest Matches For Given Search Targets.

Methods: Inherits All Sortedmap Methods And Adds Methods Like Lowerentry(), Floorentry(), Ceilingentry(), Higherentry(), Pollfirstentry(), Polllastentry(), Descendingmap().

Common Implementation: Treemap.

40. What Is The Difference Between Array & Arraylist?

Ans. -> 1. Size: Array In Java Is Fixed In Size. We Can Not Change The Size Of Array After Creating It. Arraylist Is Dynamic In Size. When We Add Elements To An Arraylist, Its Capacity Increases Automatically.

2. Performance: In Java Array And Arraylist Give Different Performance For Different Operations. Add() Or Get(): Adding An Element To Or Retrieving An Element From An Array Or Arraylist Object Has Similar Performance. These Are Constant Time Operations. Resize(): Automatic Resize Of Arraylist Slows Down The Performance. Arraylist Is Internally Backed By An Array. In Resize() A Temporary Array Is Used To Copy Elements From Old Array To New Array.

3. Primitives: Array Can Contain Both Primitive Data Types As Well As Objects. But Arraylist Can Not Contain Primitive Data Types. It Contains Only Objects.

4. Iterator: In An Arraylist We Use An Iterator Object To Traverse The Elements. We Use For Loop For Iterating Elements In An Array.

5. Type Safety: Java Helps In Ensuring Type Safety Of Elements In An Arraylist By Using Generics. An Array Can Contain Objects Of Same Type Of Classe. If We Try To Store A Different Data Type Object In An Array Then It Throws Arraystoreexception.

6. Length: Size Of Arraylist Can Be Obtained By Using Size() Method. Every Array Object Has Length Variable That Is Same As The Length/Size Of The Array.

7. Adding Elements: In An Arraylist We Can Use Add() Method To Add Objects. In An Array Assignment Operator Is Used For Adding Elements.

8. Multi-Dimension: An Array Can Be Multi-Dimensional. An Arraylist Is Always Of Single Dimension

41. What Is The Difference Between Arraylist & Vector?

Ans. -> Every Method Present Inside Arraylistis Non – Synchronized.

Every Method Present In Vector Is Synchronized.

At A Time Multiple Threads Are Allow To Operate On Arraylist Simultaneously And Hence Arraylist Object Is Not Thread Safe.

At A Time Only One Thread Is Allow To Operate On Vector Object And Hence Vector Object Is Always Thread Safe.

Relatively Performance Is High Because Threads Are Not Required To Wait.

Relatively Performance Is Low Because Threads Are Required To Wait.

Introduced In 1.2 Version And It Is Non – Legacy.

Introduced In 1.0 Version And It Is Legacy.

42. What Is The Difference Between Arraylist & Linkedlist?

Ans. -> 1) Arraylist Internally Uses A Dynamic Array To Store The Elements.

Linkedlist Internally Uses A Doubly Linked List To Store The Elements.

2) Manipulation With Arraylist Is Slow Because It Internally Uses An Array. If Any Element Is Removed From The Array, All The Other Elements Are Shifted In Memory.

Manipulation With Linkedlist Is Faster Than Arraylist Because It Uses A Doubly Linked List, So No Bit Shifting Is Required In Memory.

3) An Arraylist Class Can Act As A List Only Because It Implements List Only.

Linkedlist Class Can Act As A List And Queue Both Because It Implements List And Deque Interfaces.

4) Arraylist Is Better For Storing And Accessing Data.

Linkedlist Is Better For Manipulating Data.

5) The Memory Location For The Elements Of An Arraylist Is Contiguous.

The Location For The Elements Of A Linked List Is Not Contagious.

6) Generally, When An Arraylist Is Initialized, A Default Capacity Of 10 Is Assigned To The Arraylist.

There Is No Case Of Default Capacity In A Linkedlist. In Linkedlist, An Empty List Is Created When A Linkedlist Is Initialized.

7) To Be Precise, An Arraylist Is A Resizable Array.

Linkedlist Implements The Doubly Linked List Of The List Interface.

8) Memory Is Allocated At The Compile Time At The Stack Memory Location.

Memory Is Allocated During The Run Time At The Heap Memory Section.

43. What Is The Difference Between Hashset & Linkedhashset?

Ans. -> It Uses A Hashtable To Store The Elements.

It Uses A Hashtable And Doubly Linked List To Store And Maintain The Insertion Order Of The Elements.

Insertion Order Is Not Preserve

Insertion Order Is Preserve

It Requires Less Memory.

It Requires More Memory Than Hashset.

Introduced In 1.2 V

Introduced In 1.4 V

Initial Capacity 16 Both

44. What Is The Difference Between Hashmap & Linkedhashmap?

Ans. -> Hashmap Linkedhashmap

The Underlying Datastructure Is Hashtable.

The Underlying Data Structure Is Combination Of Hashtable And Linkedlist

Insertion Is Not Preserved

Insertion Order Is Preserved

Introduced In 1.2 Version.

Introduced In 1.4 Version.

45. What Is The Difference Between Hashset & Treeset?

Ans. -> The Hash Set Is Executed With The Help Of A Hashtable.

The Tree Set Is Executed With The Help Of A Tree Structure.

It Authorises A Heterogeneous Object.

It Does Not Authorise A Heterogeneous Object.

It Permits A Null Object.

It Does Not Permit The Null Object. From(1.7)

To Compare Two Objects, We Use The Equals Method.

To Compare Two Objects, We Use The Compare Method.

It Does Not Support Any Order

Treeset Supports An Object In Sorted Order.

What Differences Exist Between Hashmap And Hashtable ?

Ans. -> No Method Present In Hashmap Is Synchronized.

Every Method Present In Hashtable Is Synchronized.

At A Time Multiple Threads Are Allowed To Operate On Hashmap Object Simultaneously And Hence It Is Not Thread Safe.

At A Time Only One Thread Is Allowed To Operate On The Hashtable Object And Hence It Is Thread Safe.

Relatively Performance Is High.

Relatively Performance Is Low.

Null Is Allowed For Both Keys And Values.

Null Is Not Allowed For Both Keys And Values. Otherwise We Will Get Npe.

Introduced In 1.2 Version And It Is Non – Legacy.

Introduced In 1.0 Version And It Is Legacy.

Hashmap Is Traversed By Iterator.

Hashtable Is Traversed By Enumerator And Iterator.

47. What Is The Difference Between Identityhashmap & Hashmap ?

Ans. -> Identityhashmap Uses The Equality Operator == For Comparing Keys And Values

Hashmap Uses To Equal Method For Comparing Keys And Values

Instead It Uses The System.Identityhashcode() Method To Find The Bucket Location.

Hashmap Uses The Hashcode() Method Of Hashmap Class To Find The Bucket Location.

Identityhashmap Performs Better Than Hashmap.

Hashmap Performs Slightly Less Than The Identityhashmap.

46. What Is The Difference Between Treeset & Treemap?

18. What Is The Difference Between Vector & Hashtable?

48. What Is The Difference Between Hashmap & Weakhashmap?

Ans. -> In Case Of Hashmap, Hashmap Dominates Garbage Collector. That Is If Object Doesn’t Have Any Reference Still It Is Not Eligible For Garbage Collector If It Is Associated With

Hashmap.

But In Case Of Weakhashmap If An Object Doesn't Contain Any References Then It Is Always Eligible For Gc Even Though It Is Associated With Weakhashmap. That Is Garbage Collector

Dominates Weakhashmap.

49. What Is The Collection Object That Stores Only Unique Objects In No Order?

Ans. -> A Set Is A Collection Object That Stores Unique Objects In An Unordered Collection. A Set Is An Interface That Extends The Collection Interface In Java.

50. What Is The Collection Object That Stores Only Unique Objects In Insertion Order?

Ans. -> The Collection Object That Stores Only Unique Objects In Insertion Order Is The Linkedhashset In Java.

51. What Is The Collection Object That Stores Only Unique Objects In Sorting Order?

Ans. -> The Collection Object That Stores Only Unique Objects In Sorting Order Is The Treeset Class In Java.

52. Which Collection We Must Use To Store Duplicate Objects?

Ans. -> There Are A Few Different Collections In Java That Can Be Used To Store Duplicate Objects. The Most Common One Is The Arraylist. Arraylists Are Dynamic Arrays That Can Grow And Shrink In Size As Needed. They Allow You To Store Any Number Of Elements, Including Duplicates.

53. Which Collection We Must Use To Store Duplicate Objects In Multithreading Environment?

Ans. -> The Collection Object That Stores Duplicate Objects In Multithreading Environment Is The Arraylist Class In Java.

54. How Can We Remove Duplicate Objects From Arraylist?

Ans. -> Create A Program

55. How Can We Find Which Object Is Repeated & How Many Times?

56. How Can We Store Only Unique Objects In Indexed Order?

57. How Can We Get Synchronous Version Of Arraylist Object?

Ans -> Using The Collections.Synchronizedlist() Method

58. What Are The Activities Done When We Insert Or Remove An Element In Arraylist?

59. What Is The Collection Object We Must Use If More Operations Are Inserting Or Removing Objects At Middle Of The Collection?

Ans. -> According To Codegym, If You Frequently Insert Or Remove Elements In The Middle Of A Collection, You Should Use A Linkedlist.

60. Why Insertion And Deletion Operations Are Fast In Linkedlist Compared To Arraylist?

Ans. -> A Linkedlist Consumes A Bit More Memory Than An Arraylist Since Every Node Stores Two References To The Previous And Next Element. The Insertion, Addition, And Removal Operations Are Faster In A Linkedlist Because There Is No Resizing Of An Array Done In The Background.

61. Is A Cursor Object?

1) What Is The Difference Between Cursor Object & Container Object?

Ans -> A Cursor Object Is A Database Object That Allows You To Iterate Through The Results Of A Query. A Container Object Is A Data Structure That Can Store Multiple Values.

2) What Are The Three Types Of Cursor Objects Exists In Collection Api?

Ans -> We Have The Following Three Types Of Cursors In Collection Framework:

Iterator.

Listiterator.

Enumeration.

3) What Is The Difference Between Enumeration & Iterator?

Ans -> Enumeration Iterator

Enumeration Is Not A Universal Cursor As It Applies Only To Legacy Classes.

Iterator Is A Universal Cursor As It Is Applicable For All The Collection Classes.

Enumeration Does Not Have The Remove() Method.

Iterator Has The Remove() Method

Enumeration Interface Acts As A Read Only Interface, One Can Not Do Any Modifications To Collection While Traversing The Elements Of The Collection.

Iterator Can Do Modifications (E.G Using Remove() Method It Removes The Element From The Collection During Traversal

Enumeration Is A Legacy Interface Which Is Used For Traversing Vector, Hashtable.

Iterator Is Not A Legacy Interface. Iterator Can Be Used For The Traversal Of Hashmap, Linkedlist, Arraylist, Hashset, Treemap, Treeset .

4) What Is The Difference Between Iterator & Listiterator?

Ans -> Iterator Listiterator

Can Traverse Elements Present In Collection Only In The Forward Direction

Can Traverse Elements Present In Collection Both In Forward And Backward Directions.

Helps To Traverse Map, List And Set Can Only Traverse List And Not The Other Two.

Indexes Cannot Be Obtained By Using Iterator.

It Has Methods Like Nextindex() And Previousindex() To Obtain Indexes Of Elements At Any Time While Traversing List.

Cannot Modify Or Replace Elements Present In Collection We Can Modify Or Replace Elements With The Help Of Set(E E)

Cannot Add Elements And It Throws Concurrentmodificationexception. Can Easily Add Elements To A Collection At Any Time.

Certain Methods Of Iterator Are Next(), Remove() And Hasnext(). Certain Methods Of Listiterator Are Next(), Previous(), Hasnext(), Hasprevious(), Add(E E).

5) Why Listiterator Can Be A Bidirectional Cursor?

Ans -> Listiterator Is A Bi-Directional Cursor In Java Because It Supports Both Forward And Backward Direction Iteration. Listiterator Is An Interface That Allows You To Traverse A List In Both Directions And Modify The List While Iterating Over It.

6) Why Iterator Does Not Have Add() Method?

Ans. -> An Iterator's Main Purpose Is To Enumerate Through A Collection. All Collections Have An Add() Method. However, It Doesn't Make Sense To Add To An Iterator Because The Collection May Or May Not Be Ordered. For Example, A Hashset Is An Ordered Collection.

7) What Are The Various Rules We Must Follow In Using Iterator?

Ans -> Create An Instance Of The Iterator Interface From The Collection Of Objects You Wish To Travel Across

Obtain An Iterator To The Start Of The Collection By Calling The Collection's Iterator() Method

Set Up A Loop That Makes A Call To Hasnext()

Have The Loop Iterate As Long As Hasnext() Returns True

Within The Loop, Obtain Each Element By Calling Next()

Use The Remove() Method To Remove The Last Element Returned By The Next()

Use The Foreachremaining() Method To Perform The Specified Action For Each Remaining Element Of The Collection

8) Why We Cannot Call Itr.Remove() Method Before Itr.Next() Method Call?

Ans -> The Reason Why We Cannot Call Itr.Remove() Method Before Itr.Next() Method Call Is Because The Iterator Needs To Know Which Element To Remove. The Next() Method Returns The Current Element In The Collection, And The Remove() Method Removes The Current Element. If We Call Remove() Before Next(), The Iterator Will Not Know Which Element To Remove, And It Will Throw An Exception.

9) When Will You Get Nosuchelementexception Illegalstateexception & Concurrent Modification Exception?

Ans. -> Nosuchelementexception :- This Exception Typically Occurs When Attempting To Access An Element From An Empty Collection Or When Trying To Retrieve The Next Element From An Iterator Beyond The End Of The Collection. For Example, Calling Methods Like Next() On An Empty Iterator Or Attempting To Access Elements From An Empty List Using Methods Like Get() Can Result In This Exception.

Illegalstateexception :- This Exception Is Thrown To Indicate That A Method Has Been Invoked At An Illegal Or Inappropriate Time Or In An Inappropriate Context. For Example, Calling Methods Like Remove() On An Iterator Before Calling Next() Or Attempting To Modify A Collection While It's Being Iterated Can Result In This Exception.

Concurrentmodificationexception :- This Exception Occurs When A Collection Is Modified (E.G., Adding Or Removing Elements) While It Is Being Iterated Over. It Indicates That The Structure Of The Collection Has Been Modified Unexpectedly During Iteration, Violating The Fail-Fast Behavior Of Certain Collection Implementations Like Arraylist Or Hashmap. For Example, Adding Or Removing Elements From A List While Iterating Over It Using An Iterator Can Result In This Exception.

11) How Can We Get Enumeration Object On Collection Framework Classes?

Ans -> The Enumeration() Method Of The Java.Util.Collections Class Returns An Enumeration Over A Specified Collection. This Method Provides Interoperability With Legacy Apis That Require An Enumeration As Input.

12) How Can Make Sure When We Return Collection From A Method It Is Allowed Only From Reading Operations?

13) How Can Retrieve Elements From Map Using Iterator?

Ans -> Use A While Loop With The Hasnext() And Next() Methods To Iterate Through The Set And Retrieve Each Key-Value Pair.

14) What Is The Difference Between Litr.Add() & List.Add(Index, Obj) & List.Add(Obj)?

Ans. -> The List.Add() Method Adds An Element To The End Of A List. The List.Add(Index, Object) Method Adds An Element At The Specified Index In A List. The List.Add(Obj) Method Adds The Specified Object To The End Of The List.

15) What Is Map.Entry?

16) What Is The Difference Between Sequential Access & Random Access?

Ans. -> Sequential Access Is When Data Is Accessed In A Sequencee. Random Access Is When Data Can Be Accessed From Any Location.

17) Is It Possible To Retrieve Objects From Collection Randomly Using Cursor Objects?

Ans -> It's Not Possible To Use A Cursor In A Non-Sequential Way

18) What Is The Difference Between List.Get(Index) & Itr.Next()?

Ans. -> The Main Difference Between List.Get(Index) And Itr.Next() In Java Is That List.Get(Index) Returns The Element At The Specified Index In The List, While Itr.Next() Returns The Next Element In The List Based On The Current Position Of The Iterator.

62 What Are The Different Ways To Retrieve Objects From Set/List/Queue/Map Collections?

63. What Are The Benefits Added To Collection Programming With

1) Auto Boxing And Unboxing?

2) Enhanced For-Loop

3) Generics

64. What Is The Difference Between List.Remove(5) & Set.Remove(5)?

Ans. -> The Main Difference Between List.Remove(5) And Set.Remove(5) In Java Is That List.Remove(5) Removes The Element At Index 5 From The List, While Set.Remove(5) Removes The Element That Is Equal To The Value 5 From The Set.

65. What Are The Operations Performed By Addall(), Putall(), Containsall(), Removeall(), Retainall() Methods?

1) What Is The Difference Between Add() & Addall()?

Ans. -> In Java, The Add() Method Adds A Single Item To A List, While The Addall() Method Adds A Collection Of Items To A List.

2) What Is The Difference Between Put & Putall()?

Ans. -> Put(Object Key, Object Value) Allows You To Add A Single Entry Into The While Map. Putall(Map T) Adds All Entries Contained In The Map T To The Specified Map. Putall() Is Useful While Combining Two Maps.

3) What Is The Difference Between Contains() & Contains All()?

Ans. -> In Java, Contains() Checks If A Single Object Exists In A List, While Containsall() Checks If A List Contains All The Elements In A Given Collection.

4) What Is The Difference Between Remove() & Removeall()?

Ans. -> In Java, The Remove() Method Removes The First Occurrence Of A Specified Element, While The Removeall() Method Removes All Occurrences Of A Specified Collection Of Elements.

5) What Is The Difference Between Removeall() & Retainall()?

6) What Is The Difference Between Removeall() & Clear()?

66. Which Method We Must Use To Create Collection Of Collections?

67. Which Method We Must Use To Create Collection Of Maps?

68. What Is The Difference Between Capacity & Size?

Ans. -> Capacity :- Capacity Refers To The Maximum Number Of Elements That A Collection Can Hold Before It Needs To Resize Its Internal Data Structure.

It Is Specific To Certain Types Of Collections, Such As Arraylist Or Hashmap, Where The Underlying Data Structure (E.G., An Array Or A Hash Table) Has A Fixed Initial Size That May Need To Be Expanded As Elements Are Added.

When The Number Of Elements In The Collection Exceeds Its Capacity, The Collection May Need To Reallocate Memory Or Resize Its Internal Data Structure To Accommodate More Elements.

Size :- Size Refers To The Current Number Of Elements That Are Actually Present In The Collection.

It Represents The Count Of Elements That Have Been Added To The Collection And Have Not Been Removed.

Size Is Dynamic And Changes As Elements Are Added To Or Removed From The Collection.

Methods Like Size() Are Used To Retrieve The Current Size Of The Collection.

69. What Is The Default Capacity Of Arraylist, Vector, Hashset, Hashmap, Hashtable?

Ans. -> 10

70. When Collection Is Filled, What Will Happen If We Add New Object?

71. How Must Vector Capacity Is Incremented?

Ans. -> New Capacity = Current Capacity \* 2.

72. How Much Arraylist Capacity Is Incremented?

Ans. -> New Capacity = (Current Capacity \* 3/2) + 1.

73. What Is The Meaning Of Load Factor?

Ams. -> After Filling How Much Ratio New Object Will Be Created The Ratio Is Called Fill Ratio Or Load Factor.

74. What Is The Load Factor Of Set & Map Objects?

Ans -> 0.75%

75. Which Collection Object Does Not Allow Null?

Ans -> Hashtable Doesn't Allow Any Null Keys Or Values, But Hashmap Lets You Have One Null Key And Several Null Values.

76. Which Collection Object Allows Only One Null?

Ans -> Hashset Allows Only One Null Key, But Hashmap Can Allow One Null Key + Multiple Null Values.

77. Which Collection Allows Multiple Nulls?

Ans -> In Java, The Arraylist Collection Allows Multiple Null Elements. This Means You Can Add Multiple Null Values To An Arraylist Without Any Issues.

78. How Many Nulls We Can Store Key & As Valon Hashmap Treemap?

Ans -> In Java, A Hashmap Can Store One Null Key And Multiple Null Values. A Treemap Can't Store A Null Key, But It Can Contain Multiple Null Values.

79. Is It Possible To Store Null In Hashtable Either As Key Or Value?

Ans -> No, A Hashtable Does Not Allow Null Values For Either Keys Or Values.

80. Is It Possible To Store Heterogeneous Objects In Hashset & Hashmap?

Ans -> Yes, It Is Possible To Store Heterogeneous Objects In Both Hashset And Hashmap. Hashset Is A Collection That Stores Unique Elements, And Hashmap Is A Map That Stores Key-Value Pairs. Both Hashset And Hashmap Can Store Objects Of Any Type, Including Primitive Types, Wrapper Classes, And User-Defined Classes.

81. Is It Possible To Store Heterogeneous Objects In Treeset & Treemap?

Ans -> No, It Is Not Possible To Store Heterogeneous Objects In Treeset And Treemap. Both Treeset And Treemap Are Sorted Collections, And In Order To Sort Objects, They Need To Be Able To Compare Them. This Means That All Objects In A Treeset Or Treemap Must Be Of The Same Type And Must Implement The Comparable Interface.

82. Is It Possible To Store Duplicate Objects In Hashset & Hashmap?

Ans -> No, It Is Not Possible To Store Duplicate Objects In Hashset And Hashmap. Both Hashset And Hashmap Are Based On The Concept Of Hash Tables, Which Means That They Use A Hash Function To Map Objects To Their Corresponding Locations In Memory. If You Try To Add A Duplicate Object To A Hashset Or Hashmap, The Hash Function Will Return The Same Location For Both Objects, And The Newer Object Will Overwrite The Older Object.

83. Explain How Set & Map Collections Will Stop Storing Duplicate Objects?

1) What Will Be Used In Storing Objects In Set & Map Collections?

2) Why Set & Map Collection Internally Uses Hashcode() & Equals() Methods?

3) Will All Subclasses Of Set & Map Use Hashcode() & Equals() Methods?

Ans. -> Yes, All Subclasses Of Set And Map Will Use The Hashcode() And Equals() Methods. The Hashcode() Method Is Used To Determine The Hash Code Of An Object, Which Is A Unique Identifier For That Object. The Equals() Method Is Used To Compare Two Objects For Equality.

4) For Finding Duplicate Object "==" Operator & Equals() Methods Are Sufficient, Then For What Purpose Hashcode() Method Is Used?

5) What Is A Bucket, What Is The Use Of It, When A New Bucket Is Created?

6) How One Group Of Objects Are Differentiated From Another Group Of Objects?

7) Why We Must Override Hashcode() & Equals() Methods In Our Class For Storing Its Objects In Hash Based Collection Classes Such As Hashset, Linkedhashset, Hashmap, Linkedhashmap, Hashtable?

8) What Will Be The Problem If We Do Not Override Hashcode() & Equals() Methods In Subclass?

9) Why Immutable Object Recommended To Use As Key In Map?

84. Explain How Treeset Can Store Objects In Sorting Order?

1) What Is The Default Sorting Order Of Treeset?

Ans -> The Default Sorting Order Of Treeset In Java Is Ascending.

2) How Can We Passing Sorting Order Of Our Objects To Treeset Class?

3) What Am The Difference Between Comrorable & Comprar Interfaces?

Ans. -> Comparable Comparator

Comparable Provides A Single Sorting Sequence. In Other Words, We Can Sort The Collection On The Basis Of A Single Element Such As Id, Name, And Price.

The Comparator Provides Multiple Sorting Sequences. In Other Words, We Can Sort The Collection On The Basis Of Multiple Elements Such As Id, Name, And Price Etc.

Comparable Affects The Original Class, I.E., The Actual Class Is Modified.

Comparator Doesn't Affect The Original Class, I.E., The Actual Class Is Not Modified.

Comparable Provides Compareto() Method To Sort Elements.

Comparator Provides Compare() Method To Sort Elements.

Comparable Is Present In Java.Lang Package.

A Comparator Is Present In The Java.Util Package.

We Can Sort The List Elements Of Comparable Type By Collections.Sort(List) Method.

We Can Sort The List Elements Of Comparator Type By Collections.Sort(List, Comparator) Method.

4) What Is The Meaning Of Natural Sorting Order & Custom Sorting Order?

5) What Is The Meaning Of Custom Comparator?

6) What Do You Mean By Non-Comparable Objects?

7) When Will Treeset Use Comparable Interface For Sorting Elements?

8) When Will Tree Set Use Comparator Interface For Sorting Elements?

9) What Are The Steps We Must Follow To Develop A Class To Add Its Objects To Treeset Or To Treemap As Key?

10) What Types Of Objects Are Only Allowed To Store In Treeset & Treemap?

11) How Can We Store Non-Comparable Objects In Treeset?

12) How Can Store A Class Objects In Different Sorting Orders In Treeset Objects?

13) Why Compareto() Method Has One Parameter & Compare() Method Has Two Parameters?

14) Compareto() & Equals() Methods Are Meant From Comparing Objects Of A Class, Then Why Compareto() Method Return Type Is Int & Equals() Method Return Type Is Boolean?

85. What Are The Three Methods We Must Override In Subclass To Store Its Objects In All Types Of Collection Objects?

86. What Is The Use Of Collections Class?

1) How Can We Find Number Of Occurrences Of A Given Object?

2) How Can We Sort Elements In Arraylist?

3) How Can We Get Synchronous Version Of Arraylist?

4) How Can We Get Immutable Collection Object?

5) While Passing A Collection As Argument To A Function, How Can We Make Sure The Function Will Not Be Able To Modify It?

87.What Is The Use Of Array Class?

1) How Can Sort Objects In Array?

2) How Can We Search An Object In Array?

3) How Can You Print All Elements Of Array Without Using For Loop Text Explicitly?

88. What Is The Use Of Stack?

1) Are Both Lifo & Filo Same?

2) How Should We Add Objects To Stack?

3) What Is The Difference Between Pop() And Peek() Methods?

4) What Is The Difference Between Insertion Index And Searching Index In Stack?

5) Stack Is A Subclass Of Which Collection?

6) Can We Use List Interface Methods On Stack Object For Adding, Retrieving & Removing Its Objects?

89. What Is The Use Of Properties Class?

90. What Is The Difference Between Properties Class & Resourcebundle Class?

91. What Is The Use Of Stringtokenizer Class?

92. What Is The Use Of Randomaccess Interface, Is It A Marker Interface?

93. Explain In A Project, When Will We Create Collection Of Objects, Collection Of Collections & Collection Of Maps Created?

94. Explain The Meaning Of Collection Can Be Used As Data Transferable Object & As Mini Db?

============================================================================================================

What Is Difference Between Hashset And Linkedhashset ?

Difference Between Comparator And Comparable

11) What Is The Difference Between Encapsulation And Abstraction ?

Q46. What Is The Difference Between List And Set?

Ans -> The List And Set Both Extend The Collection Interface. However, There Are Some Differences Between The Both Which Are Listed Below.

The List Can Contain Duplicate Elements Whereas Set Includes Unique Items.

The List Is An Ordered Collection Which Maintains The Insertion Order Whereas Set Is An Unordered Collection Which Does Not Preserve The Insertion Order.

The List Interface Contains A Single Legacy Class Which Is Vector Class Whereas Set Interface Does Not Have Any Legacy Class.

The List Interface Can Allow N Number Of Null Values Whereas Set Interface Only Allows A Single Null Value.

Q47. What Is The Difference Between Set And Map?

Ans -> The Differences Between The Set And Map Are Given Below.

Set Contains Values Only Whereas Map Contains Key And Values Both.

Set Contains Unique Values Whereas Map Can Contain Unique Keys With Duplicate Values.

Set Holds A Single Number Of Null Value Whereas Map Can Include A Single Null Key With N Number Of Null Values.

Update Your Skills Form Our Experts: Core Java Online Training

Q48. What Is The Difference Between Collection And Collections?

Ans -> The Differences Between The Collection And Collections Are Given Below.

The Collection Is An Interface Whereas Collections Is A Class.

The Collection Interface Provides The Standard Functionality Of Data Structure To List, Set, And Queue. However, Collections Class Is To Sort And

Synchronize The Collection Elements.

The Collection Interface Provides The Methods That Can Be Used For Data Structure Whereas Collections Class Provides The Static Methods Which

Can Be Used For Various Operation On A Collection.

Q49. What Is The Advantage Of The Generic Collection?

Ans. -> There Are Three Main Advantages Of Using The Generic Collection.

If We Use The Generic Class, We Don’t Need Typecasting.

It Is Type-Safe And Checked At Compile Time.

Generic Con􀃕rms The Stability Of The Code By Making It Bug Detectable At Compile Time.

Q34. What’s The Difference Between An Array And Vector?

Ans -> An Array Groups Data Of Same Primitive Type And Is Static In Nature While Vectors Are Dynamic In Nature And Can Hold Data Of Different Data Types.

Q54. What’s Difference Between Stack And Queue?

Ans -> Stack And Queue Both Are Used As Placeholder For A Collection Of Data. The Primary Difference Between A Stack And A Queue Is That Stack Is Based

On Last In First Out (Lifo) Principle While A Queue Is Based On Fifo (First In First Out) Principle.

Q96. Which Api Is Provided By Java For Operations On Set Of Objects?

Ans -> Java Provides A Collection Api Which Provides Many Useful Methods Which Can Be Applied On A Set Of Objects. Some Of The Important Classes

Provided By Collection Api Include Arraylist, Hashmap, Treeset And Treemap.

4 What Differences Exist Between Iterator And Listiterator ?

Ans. -> Can Traverse Lists, Sets, And Maps

Can Only Traverse Lists

Can Only Traverse The Collection In The Forward Direction

Can Traverse The Collection In Both Forward And Backward Directions

There Is No Method To Obtain An Index Of The Element

It Is Possible To Obtain The Index Of The Element

Cannot Add Elements To The Collection

Can Add Elements To The Collection Using Add(E E)

Hasnext(), Next(), Remove() Are Some Common Methods

Hasnext(), Next(), Hasprevious(), Previous(), Remove, Add(E E), Previousindex(), Set(Ee) Some Common Methods

5 What Is Difference Between Fail-Fast And Fail-Safe ?

Ans. -> This Iterator Directly Works O The Collection Object Itself

This Iterator Works O The Clone Or Copy Of The Collection Object

It Doesn’t Allow Tp Modify The Collection While Oterating Will Throw Concurrentmodificationexception

It Will Allow To Modify The Collection While Iterating It Doesn’t Throw Any Exception

This Iteratore Doesn’t Require Any Extra Memory

This One Requires Extra Memory Consumes Heap

Example Of This Iterator Are Arraylist, Hashmap, Vector Hashset And These Classe Are Java Util Package

Example Of This Iterator Are Concurrenthashmap, Copyonwritearraylist, And Class Are In Java,Util.Concureent Package

6 How Hashmap Works In Java ?

7 What Is The Importance Of Hashcode() And Equals() Methods ?

What Is Difference Between Enumeration And Iterator Interface?

Ans. -> 01. Using Enumeration, You Can Only Traverse The Collection. You Can’t Do Any Modifications To Collection While Traversing It.

Using Iterator, You Can Remove An Element Of The Collection While Traversing It.

02. Enumeration Is Introduced In Jdk 1.0

Iterator Is Introduced From Jdk 1.2

03. Enumeration Is Used To Traverse The Legacy Classes Like Vector, Stack And Hashtable.

Iterator Is Used To Iterate Most Of The Classes In The Collection Framework Like Arraylist, Hashset, Hashmap, Linkedlist Etc.

04. Methods : Hasmoreelements() And Nextelement()

Methods : Hasnext(), Next() And Remove()

05. Enumeration Is Fail-Safe In Nature.

Iterator Is Fail-Fast In Nature.

06. Enumeration Is Not Safe And Secured Due To It’s Fail-Safe Nature.

Iterator Is Safer And Secured Than Enumeration.

What Is A Default Capacity Of Arraylist, Vector, Hashmap, Hashtable And Hashset?

Ans. -> Collections Capacity

Arraylist 10

Vector 10

Hashset 16

Hashmap 16

Hashtable 11

11 What Is Comparable And Comparator Interface ? List Their Differences ?

Ans. -> Present In Java.Lang Package

Present In Java.Util Package

It Is Meant For Default Natural Sorting Order.

It Is Meant For Customized Sorting Order.

Defines Only One Method Compareto().

Defines 2 Methods Compare() And Equals().

All Wrapper Classes And String Class Implements Comparable Interface.

The Only Implemented Classes Of Comparator Are Collator And Rulebasecollator.

12 What Is Java Priority Queue ?

13 What Do You Know About The Big-O Notation And Can You Give Some Examples With Respect To Different Data Structures ?

14 What Is The Tradeoff Between Using An Unordered Array Versus An Ordered Array ?

15 What Are Some Of The Best Practices Relating To The Java Collection Framework ?

16 What’s The Difference Between Enumeration And Iterator Interfaces ?

17 What Is The Difference Between Hashset And Treeset ?

201) Explain About Collection Interface In Java ?

202) List The Interfaces Which Extends Collection Interface ?

203) Explain List Interface ?

204) Explain Methods Specific To List Interface ?

206) Explain About Arraylist ?

208) What Is Vector?

209) Difference Between Arraylist And Vector ?

210) Define Linked List And Its Features With Signature ?

211) Define Iterator And Methods In Iterator?

212) In Which Order The Iterator Iterates Over Collection?

212) Explain Listiterator And Methods In Listiterator?

213) Explain About Sets ?

214) Implementations Of Set Interface ?

215) Explain Hashset And Its Features ?

216) Explain Tree Set And Its Features?

217) When Do We Use Hashset Over Treeset?

218) What Is Linked Hashset And Its Features?

219) Explain About Map Interface In Java?

220) What Is Linked Hashmap And Its Features?

221) What Is Sortedmap Interface?

222) What Is Hashtable And Explain Features Of Hashtable?

224) Difference Between Arraylist And Linkedlist?

226) What Is Concurrent Hashmap And Its Features ?

228) Explain Copyonwritearraylist And When Do We Use Copyonwritearraylist?

229) Explain About Fail Fast Iterators In Java?

230) Explain About Fail Safe Iterators In Java?