1) What is framework in Java?

A framework is a popular and readymade architecture that contains a set of classes and interfaces.

2) What is the Collection framework in Java?

Collection Framework is a grouping of classes and interfaces that is used to store and manage the objects. It provides various classes like Vector, ArrayList, HashSet, Stack, etc. Java Collection framework can also be used for interfaces like Queue, Set, List, etc.

Java Collections Interview Questions

3) Explain Collections Class

java.util.Collections is a class consists of static methods that operate on collections. It contains polymorphic algorithms to operate on collections, “wrappers”. This class contains methods for algorithms, like binary sorting, search, shuffling, etc.

4) What is the hashCode()?

The hashCode() is a method that returns an integer hash code.

5) Distinguish between ArrayList and Vector in the Java collection framework.

ArrayList Vector

ArrayList is cannot be synchronized. Vector can be is synchronized.

It is not a legacy class. It is a legacy class.

It can increase its size by 50% of the size of the array. It can increase its size by doubling the size of the array.

ArrayList is not thread-safe. Vector is a thread-safe.

6) What is ArrayList in Java?

ArrayList is a data structure that can be stretched to accommodate additional elements within itself and shrink back to a smaller size when elements are removed. It is a very important data structure useful in handling the dynamic behavior of elements.

7) Differentiate between Iterator and ListIterator

The difference between Iterator and ListIterator is:

Iterator ListIterator

The Iterator can traverse the array elements in the forward direction. ListIterator can traverse the array elements in backward as well as forward directions.

It can be used in Queue, List, and Set. It can be used in List.

It can perform only remove operation. It can perform add, remove, and set operation while traversing the collection.

8) What is the difference between Iterator and Enumeration?

The difference between Iterator and Enumeration

Iterator Enumeration

The Iterator can traverse both legacies as well as non-legacy elements. Enumeration can traverse only legacy elements.

The Iterator is fail-fast. Enumeration is not fail-fast.

The Iterator is very slow compare to Enumeration. Enumeration is fast compare to Iterator.

The Iterator can perform remove operation while traversing the collection. The Enumeration can perform only traverse operation on the collection.

9) Define BlockingQueue

BlockingQueue is an interface used in Java that can extend the Queue. It provides concurrency in various queue operations like retrieval, insertion, deletion, etc.

The Queue waits to become non-empty at the time of retrieving any elements. BlockingQueue should not contain null elements. The implementation of this Queue is thread-safe.

The syntax of BlockingQueue is:

public interface BlockingQueue<E> extends Queue <E>

10) Explain override equals() method

The equals method is used to check the similarity between two objects. In case if the programmer wants to check an object based on the property, then it needs to be overridden.

11) What is the difference between Comparable and Comparator?

The difference between Comparable and Comparator is:

Comparable Comparator

Comparable provides compareTo() method to sort elements in Java. Comparator provides compare() method to sort elements in Java.

Comparable interface is present in java.lang package. Comparator interface is present in java. util package.

The logic of sorting must be in the same class whose object you are going to sort. The logic of sorting should be in a separate class to write different sorting based on different attributes of objects.

The class whose objects you want to sort must implement the comparable interface. Class, whose objects you want to sort, do not need to implement a comparator interface.

It provides single sorting sequences. It provides multiple sorting sequences.

This method can sort the data according to the natural sorting order. This method sorts the data according to the customized sorting order.

It affects the original class. i.e., the actual class is altered. It doesn't affect the original class, i.e., the actual class is not altered.

Implemented frequently in the API by Calendar, Wrapper classes, Date, and String. It is implemented to sort instances of third-party classes.

All wrapper classes and String class implement the comparable interface. The only implemented classes of Comparator are Collator and RuleBasedColator.

12) Explain equals() with example

Equals() verifies whether the number object is equal to the object, which is passed as an argument or not.

The syntax of the equals() method is:

public boolean equals(Object o)

This method takes two parameters 1) any object, 2) return value. It returns true if the passed argument is not null and is an object of a similar type having the same numeric value.

Example:

```
import java.lang.Integer;
public class Test {
  public static void main(String args[]) {
    Integer p = 5;
    Integer q = 20;
    Integer r =5;
    Short s = 5;
    System.out.println(p.equals(q));
    System.out.println(p.equals(r));
    System.out.println(p.equals(s));
  }
}
```

13) List out benefits of generic collection

The benefits of using the generic collection are:

If the programmers are using generic class, they don't require typecasting.

It is type-safe and can be checked at the time of compilation.

It provides the stability of the code by detecting bug at the compilation time.

14) Explain the method to convert ArrayList to Array and Array to ArrayList

Programmers can convert an Array to ArrayList using asList() method of Arrays class. It is a static method of Arrays class that accept the List object. The syntax of asList() method is:

```
Arrays.asList(item)
```

Java programmers can convert ArrayList to the List object using syntax:

```
List_object.toArray(new String[List_object.size()])
```

15) Give example of ArrayList

The Example of reverse ArrayList is:

```
import java.util.ArrayList;
class Test_ArrayList {
 public static void main(String[] args) {
  //Creating a generic ArrayList
  ArrayList<String> arlTest = new ArrayList<String>();
  //Size of arrayList
  System.out.println("Size of ArrayList at creation: " + arlTest.size());
```

```
//Lets add some elements to it
arlTest.add("D");
arlTest.add("U");
arlTest.add("K");
arlTest.add("E");

//Recheck the size after adding elements
System.out.println("Size of ArrayList after adding elements: " + arlTest.size());

//Display all contents of ArrayList
System.out.println("List of all elements: " + arlTest);

//Remove some elements from the list
arlTest.remove("D");
System.out.println("See contents after removing one element: " + arlTest);

//Remove element by index
arlTest.remove(2);
System.out.println("See contents after removing element by index: " + arlTest);
//Check size after removing elements
System.out.println("Size of arrayList after removing elements: " + arlTest.size());
System.out.println("List of all elements after removing elements: " + arlTest);
```

```
  //Check if the list contains "K"
  System.out.println(arlTest.contains("K"));
 }
}
```

16) Give example to sort an array in dscending order

The example of sort an array in decending order is:

```
package com.guru99;

public class SelectionSortAlgo {

	public static void main(String a[])
	{
		int[] myArray = {860,8,200,9};

		System.out.println("------Before Sort-----");

		printArray(myArray);
		selection(myArray);//sorting array using selection sort
		System.out.println("-----After Sort-----");
		printArray(myArray);
	}
```

```
        public static void selection(int[] array)
    {
        for (int i = 0; i < array.length - 1; i++)
        {  System.out.println("Sort Pass Number "+(i+1));
            int index = i;
            for (int j = i + 1; j < array.length; j++)
            {
               System.out.println("Comparing "+ array[index]  + " and " +
array[j]);
                    if (array[j] < array[index]){
                    System.out.println(array[index]  + " is greater than " +
array[j] );
                        index = j;
                    }
            }
            int smallerNumber = array[index];
            array[index] = array[i];
            array[i] = smallerNumber;
            System.out.println("Swapping Elements: New Array After
Swap");
            printArray(array);
        }
    }
```

```
    static void printArray(int[] array){


       for(int i=0; i < array.length; i++)
            {
                  System.out.print(array[i] + " ");
            }
       System.out.println();
    }
}
```

17) Explain the basic interfaces of the Java collections framework

Java collection framework is a root of the collection hierarchy. It represents a group of objects as its elements. The Java programming language does not provide a direct implementation of such interface.

Set: Set is a collection having no duplicate elements. It uses hashtable for storing elements.

List: List is an ordered collection that can contain duplicate elements. It enables developers to access any elements from its inbox. The list is like an array having a dynamic length.

MAP: It is an object which maps keys to values. It cannot contain duplicate keys. Each key can be mapped to at least one value.

18) What are the features of Java Hashmap?

Features of Java Hashmap are:

The values can be stored in a map by forming a key-value pair. The value can be retrieved using the key by passing it to the correct method.

If no element exists in the Map, it will throw a 'NoSuchElementException'.

HashMap stores only object references. That is why it is impossible to use primitive data types like double or int. Use wrapper class (like Integer or Double) instead.

19) What is a Stack?

A stack is a special area of computer's memory that stores temporary variables created by a function. In stack, variables are declared, stored, and initialized during runtime.

20) What is linked list?

A linked list is a data structure that can store a collection of items. In other words, linked lists can be utilized to store several objects of the same type. Each unit or element of the list is referred as a node. A node in the Linked list has its data and the address of the next node. It is like a chain. Linked Lists are used to create graphs and trees.

21) Give example of ArrayList

The example of ArrayList is:

```
import java.util.ArrayList;

class Test_ArrayList {

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

```
//Creating a generic ArrayList
ArrayList<String> arlTest = new ArrayList<String>();
//Size of arrayList
System.out.println("Size of ArrayList at creation: " + arlTest.size());
//Lets add some elements to it
arlTest.add("D");
arlTest.add("U");
arlTest.add("K");
arlTest.add("E");

//Recheck the size after adding elements
System.out.println("Size of ArrayList after adding elements: " + arlTest.size());

//Display all contents of ArrayList
System.out.println("List of all elements: " + arlTest);

//Remove some elements from the list
arlTest.remove("D");
System.out.println("See contents after removing one element: " + arlTest);

//Remove element by index
arlTest.remove(2);
```

```
 System.out.println("See contents after removing element by index: " + arlTest);


 //Check size after removing elements
 System.out.println("Size of arrayList after removing elements: " + arlTest.size());
 System.out.println("List of all elements after removing elements: " + arlTest);
 //Check if the list contains "K"
 System.out.println(arlTest.contains("K"));
 }
}
```

22) Explain linked list supported by Java

Two types of linked list supported by Java are:

Singly Linked list: Singly Linked list is a type of data structure. In a singly linked list, each node in the list stores the contents of the node and a reference or pointer to the next node in the list. It does not store any reference or pointer to the previous node.

Doubly linked lists: Doubly linked lists are a special type of linked list wherein traversal across the data elements can be done in both directions. This is made possible by having two links in every node, one that links to the next node and another one that connects to the previous node.

23) Explain the methods provided by the Queue interface?

Methods of Java Queue interface are:

Method	Description

boolean add(object) Inserts specified element into the Queue. It returns true in case it a success.

boolean offer(object) This method is used to insert the element into the Queue.

Object remove() It retrieves and removes the queue head.

Object poll() (): It retrieves and removes queue head or return null in case if it is empty.

Object poll() It retrieves and removes queue head or return null in case if it is empty.

Object element() Retrieves the data from the Queue, but does not remove its head.

Object peek() Retrieves the data from the Queue but does not remove its head, or in case, if the Queue is the Queue is empty, it will retrieve null.

24) Mention the methods provided by Stack class

Important methods provided by Stack class are:

push(): Push item into the stack.

empty (): This method finds that whether the stack is empty or not.

pop (): This Java collection framework method removes the object from the stack.

search (): This method searches items in the stack.

peek (): This Java method looks at the stack object without removing it.

25) Define emptySet() in the Java collections framework

Method emptySet() that returns the empty immutable set whenever programmers try to remove null elements. The set which is returned by emptySet() is serializable. The syntax of this method is:

public static final <T> Set<T> emptySet()

26) Differentiate between Collection and Collections

The difference between Collection and Collections are:

Collection	Collections
The collection is an interface.	Collections is a class.
It represents a group of objects as a single entity.	It defines various utility methods for collection objects.
The collection is the root interface of the Java Collection framework.	Collections is a general utility class.
This interface is used to derive the collection data structures.	This class contains static methods to manipulate data structure.

27) Define LinkedHashSet in the Java Collection framework?

LinkedHashSet is a subclass of the class called HashSet and implements the set interface. It is a well-ordered version of HashSet that maintains a doubly-linked List across its all elements.

28) What is the difference between failfast and failsafe?

Failfast	Failsafe

It does not allow collection modification while iterating. It allows collection modification while iterating.

It can throw ConcurrentModificationException It can't throw any exception.

It uses the original collection to traverse the elements. It uses an original collection copy to traverse the elements.

There is no requirement of extra memory. There is a requirement of extra memory.

29) List collection views of a map interface

Collection views of map interface are: 1) key set view, 2) value set view, and 3) entry set view.

30) What are the benefits of the Collection Framework in Java?

The benefits of Collection Framework in Java are:

Java collection framework offers highly efficient and effective data structures that enhance the accuracy and speed of the program.

The program developed with the Java collection framework is easy to maintain.

A developer can mix classes with other types that result in increasing the reusability of code.

The Java collection framework enables programmers to modify the primitive collection types the way they like.

31) What is a good way to sort the Collection objects in Java?

A good way to sort Java collection objects is using Comparable and Comparator interfaces. A developer can use Collections.sort(), the elements are sorted based on the order mention in compareTo().

When a developer uses Collections, sort (Comparator), it sorts the objects depend on compare() of the Comparator interface.

32) Explain Vector in Java

The vector is the same as an array. It has components that can be accessed using an index value. Vectors can contain a legacy method that is nor part of the collection framework.

33) What is the difference between Set and Map?

Set	Map
Set belongs to package-java.util.	The map belongs package- java.util.
It can extend the collection interface.	It does not extend the collection interface.
It does not allow duplicate values.	It allows duplicate values.
Set can sort only one null value.	The map can sort multiple null values.

34) Define dictionary class

The Dictionary class is a Java class that has a capability to store key-value pairs.

35) Define EnumSet

java.util.EnumSet is Set implementation that can be used with enum types. EnumSet having all elements must come from one enum type specified explicitly or implicitly. It is not synchronized, and also null keys are not allowed. EnumSet provides methods like EnumSetof(E first, E… rest), complementOf(EnumSet s), and copyOf(Collection c).

36) What are the two ways to remove duplicates from ArrayList?

Two ways to remove duplicates from ArrayList are:

HashSet: Developer can use HashSet to remove the duplicate element from the ArrayList. The drawback is it cannot preserve the insertion order.

LinkedHashSet: Developers can also maintain the order of insertion by using LinkedHashSet instead of HashSet.

37) What is IdentityHashMap?

IdentityHashMap is a class that implements Serializable, Clonable interfaces, Map, and extends AbstractMap class. It is designed for the case wherein there is a need of reference-equality semantics.

38) What is WeakHashMap?

WeakHashMap is an implementation of the Java Map. It is used to store weak references to its keys. Sorting using this Map allows a key-value pair is collected as garbage. Its key is not referenced outside WeakHashMap.

39) What are the methods to make collection thread-safe?

The methods to make collection thread safe are:

Collections.synchronizedList(list);

Collections.synchronizedMap(map);

Collections.synchronizedSet(set);

40) Explain UnsupportedOperationException

UnsupportedOperationException is an exception whch is thrown on methods that are not supported by actual collection type.

For example, Developer is making a read-only list using "Collections.unmodifiableList(list)" and calling call(), add() or remove() method. It should clearly throw UnsupportedOperationException.

41) Name the collection classes that gives random element access to its elements

Collection classes that give random element access to its elements are: 1) ArrayList, 2) HashMap, 3) TreeMap, and 4) Hashtable.

42) Explain the difference between Queue and Deque.

Queue Deque

It is called a single-ended Queue It is called a double-ended Queue

Elements in the Queue are added or removed from one end Elements in the Queue are added from either end can be added and removed from the both end

It is less versatile. It is more versatile.

43) Mention the implementing List and Set interface

Class implementing List interface: 1) ArrayList, 2) Vector, and 3) LinkedList.

Class implementing Set interface: 1) HashSet, and 2) TreeSet.

44) Explain the design pattern followed by Iterator

The iterator follows the detail of the iterator design pattern. It provides developer to navigate through the objects collections using a common interface without knowing its implementation.

45) What is the peek() of the Queue interface?

Peek () is a method of queue interface. It retrieves all the elements but does not remove the queue head. In case if the Queue is empty, then this method will return null.

46) What is CopyOnWriteArrayList?

CopyOnWriteArrayList is a variant of ArrayList in which operations like add and set are implemented by creating a copy of the array. It is a thread-safe, and thereby it does not throw ConcurrentModificationException. This ArrayLists permits all the elements, including null.

47) Differentiate between ArrayList and LinkedList

The difference between ArrayList and LinkedList is:

ArrayList LinkedList

It uses a dynamic array. It uses a doubly-linked list.

ArrayList is not preferable for manipulation. LinkedList is preferable for manipulation.

ArrayList provides random access. LinkedList does not provide random access.

ArrayList s stores only objects hence it takes less overhead of memory

LinkedList stores object as well as address object; hence, it takes more overhead of memory.

48) Explain the methods of iterator interface

Methods of iterator interface are:

Method	Description
public boolean hasNext()	It returns true in the iterator has elements; otherwise, it returns false.
public Object next()	This method returns the element and moves the pointer to the next value.
public void remove()	This Java method can remove the last elements returned by the iterator. Public void remove() is less used.

49) What are the methods of the HashSet class?

Methods of HashSet class are:

Methods	Description
boolean add(Object o)	This method adds the mention element to this set if it is not already present.
boolean contains(Object o):	It returns true if the set contains the specified element.
void clear():	This method removes set elements.
boolean isEmpty():	It returns true in the case, the set has no elements.
boolean remove(Object o):	It remove the specified element from the set.

object clone(): This method returns a copy of the HashSet instance: the elements themselves are not cloned.

iterator iterator() It returns an iterator over the elements in this set.

int size(): It returns the number of elements available in the set.

50) What are the methods of Java TreeSet class?

The methods of Java TreeSet class are:

Methods Descriptions

boolean addAll(Collection c) Add all the elements in the specified collection to this set.

boolean contains(Object o) Returns true if the set contains the mention element.

boolean isEmpty() This Java method returns true if this set contains no elements.

boolean remove(Object o) Remove the specified element from the set.

void add(Object o) It adds the specified element to the set.

void clear() This Java method removes all the elements from the set.

51) Explain Linked HashSet

Java LinkedHashSet class is a Linked list and Hash table implementation of the Set interface. It contains unique elements same as a HashSet. Linked HashSet in Java also provides optional set operations that can maintain the order of insertion.

52) What are the important methods used in a linked list?

The important methods used in the linked list are:

Method Description

boolean add(Object o) It is used to append the specified element to the end of the vector.

boolean contains(Object o) It a method that returns true if this list contains the specified element.

void add (int index, Object element) Inserts the element at the specified element in the vector.

void addFirst(Object o) It is used to insert the given element at the beginning.

void addLast(Object o) It is used to append the given element to the end.

Int size() This method can be used to return the total number of elements in a list.

boolean remove(Object o) It can remove the first occurrence of the specified element from this list.

int indexOf(Object element) This Java method returns the index with the first occurrence of the mention element in this list, or -1.

int lastIndexOf(Object element)It is a Java method that returns the index with the last occurrence of the specified element in this list, or -1.

53) List various classes available in sets

Various classes available in sets are: HashSet, TreeSetand, and LinkedHashSet.

54) List methods available in Java Queue interface

boolean add(object)

boolean offer(object)

object remove()

object poll()

object element()

object peek()

55) Differentiate between List and Set.

List Set

An ordered collection of elements An unordered collection of elements

Preserves the insertion order Doesn’t preserves the insertion order

Duplicate values are allowed Duplicate values are not allowed

Any number of null values can be stored Only one null values can be stored

ListIterator can be used to traverse the List in any direction ListIterator cannot be used to traverse a Set

Contains a legacy class called vector Doesn’t contains any legacy class

56) Explain for each loop with example

For-Each Loop is another form of for loop used to traverse the array. It reduces the code significantly, and there is no use of the index or rather the counter in the loop.

Exmple of for each loop:

```
class UsingForEach {
  public static void main(String[] args) {
    String[] arrData = {"Alpha", "Beta", "Gamma", "Delta", "Sigma"};
```

```
    //The conventional approach of using the for loop
    System.out.println("Using conventional For Loop:");
    for(int i=0; i< arrData.length; i++){
      System.out.println(arrData[i]);
    }
    System.out.println("\nUsing Foreach loop:");
    //The optimized method of using the for loop - also called the foreach loop
    for (String strTemp : arrData){
      System.out.println(strTemp);
    }
  }
}
```

57) Explain diamond operator

Diamond operator enables the compiler to collect the type arguments of generic class. In Java SE, developer can substitute the parameterized constructor with an empty parameter sets (<>) known as diamond operator.

58) Explain randomaccess interface

RandomAccess interface is used by List implementations for the indication that they are supporting fast.

59) Name the collection classes that implement random access interface

Java.util package has classes that can implement random access interface are: CopyOnWriteArrayList, Stack, ArrayList, and Vector.

60) How to join multiple ArrayLists?

The list provides a addall() method multiple ArrayList in Java.

For example, consider two lists 1) areaList and 2) secondAreaList. A developer can join them using addall() like:

areaList.addAll(secondAreaList);

61) Explain deque Interface

Java.util.Deque is Java, an interface that extends Queue interface. It gives support for the insertion and deletion of elements at both the end. This Queue is also called a double-ended queue.

62) Explain Linkedhashmap

LinkedHashMap is the implementation of the Map interface. It can also extends the HashMap class. Therefore, like HashMap, LinkedHashMap enables Java developers to allow one null key and more than one null value.

63) Explain methods to remove elements from ArrayList

The methods to remove elements from ArrayList are:

Method Description

clear() This method removes the elements from ArrayList.

remove(int index) This method of ArrayList can remove the element at a particular position.

remove(Object o) It can remove the first occurrence of the mention element from the ArrayList.

removeAll() It can remove the list of elements that are in a particular collection.

removeIf(Predicate<? super E> filter) This method removes elements that satisfy the mention of a predicate.

64) Explain map. entry In Map

Map.entry is a Java interface of java.util. It has a nested interface in Map. This interface must be qualified by the name of class or interface, which it is a member. Therefore it is qualified as a Map. Entry. It represents a key and value pair that can forms element of a Map.

This method returns a view of the collection. For example, consider cityMap as a map. The developer can use entrySet() to get the set view of map having an element Map.Entry. Programmer can also use getKey() and getValue() of the Map.Entry to get the pair of key and value of the map.

65) Which method is used to sort an array in ascending order?

Java collection framework method, Collections.sort() is used to sort an array in ascending order.

66) How to measure the performance of an ArrayList?

The performance of ArrayList can be measure by:

Adding an element: Developer can add an element at the end of ArrayList using add(E e) method. It is O(1). In the worst scenario, it might go to O(n). This can happen if the developer add more elements than the array capacity.

Retrieving an element: Developer can access the array index using get(int index). The performance, in this case, can be measure using ArrayList get() is O(1).

Removing an element: In case, if the developers are removing element using the remove(int index), then the performance of ArrayList can be calculated using said remove(int index) operation is O(n – index) method.

67) Explain LinkedList class

LinkedList class in Java implements Deque and List using a doubly linked list. There is a private class node in a doubly-linked list which provides its structure. It also has an item variable for holding the value and reference to Node class. This can be used for connecting the next and previous nodes.

68) Give an example of Hashmap

The example of Hashmap is:

```
import java.util.HashMap;

import java.util.Map;

public class Sample_TestMaps{

  public static void main(String[] args){

    Map<String, String> objMap = new HashMap<String, String>();
```

```
    objMap.put("Name", "Suzuki");
    objMap.put("Power", "220");
    objMap.put("Type", "2-wheeler");
    objMap.put("Price", "85000");
    System.out.println("Elements of the Map:");
    System.out.println(objMap);
  }
}
```

69) How to iterate map?

The developer cannot directly iterate map, but, this interface has two methods that gives view set of map. These methods are:

Set<Map.Entry<K, V>>entrySet(): It is a method that returns a set having the entries mention in the map. These entries are generally objected, which has type Map. Entry.

Set<K>keySet(): This Java method returns a set that having the map key.

70) Explain Treemap in Java

TreeMap is a class that implements the Map interface LinkedHashMap and HashMap. It can also implements the NavigableMap interface and can extends the AbstractMap class.

71) What is the difference between Hashmap and Hashtable?

Hashmap Hashtable

It is not synchronized. It is synchronized.

HashMap allows one key as a null value. HashTable does not allow null values.

Iterator is used to traverse HashMap. Either Iterator or Enumerator is used for traversing a HashTable.

It can be used for both HashTable, HashMap and is fail-fast. It can be used with HashTable and is fail-safe.

HashMap perform faster than the HashTable. Hashtable is not much faster as compared to HashMap.

72) Explain the internal working of HashSet in Java

HashSet in Java internally uses HashMap to store elements. It can also store unique values with no duplicate values.

In Java, HashSet developer can have add(E e) method that takes just the element to add as a parameter. It does not accept the key and value pair.

73) Explain Big-O notation with an example

The Big-O notation depicts the performance of an algorithm as the number of elements in ArrayList. A developer can use Big-O notation to choose the collection implementation. It is based on performance, time, and memory.

For example, ArrayList get(index i) is a method to perform a constant-time operation. It does not depend on the total number of elements available in the list. Therefore, the performance in Big-O notation is O(1).

74) Explain the best practices in Java Collection Framework

The best practices in Java Collection Framework are:

Choose the correct type of collection depends on the need.

Avoid rehashing or resizing by estimating the total number of elements to be stored in collection classes.

Write a Java program in terms of interfaces. This will help the developer to change it's implementation effortlessly in the future.

A developer can use Generics for type-safety.

Use immutable classes given by the Java Development Kit. Avoid implementation of equals() and hashCode() for custom classes.

A programmer should use the Collections utility class for algorithms or to get read-only, synchronized, or empty collections. This will enhance code reusability with low maintainability.

75) Explain various types of queues in Java

There are three types of queues in Java:

Priority queue: It is a special type of Queue wherein elements are sorted as per their natural ordering or custom comparator.

Circular Queue: It is a type of Queue in which user operations are performed based on the FIFO method. The last element is connected to the first position in order to make a circle.

Double-ended Queue: A double-ended queue is an abstract data type that generalizes a queue. The elements in this queue can be added or removed from either head or tail.

76) What is the difference between stack and Queue?

Stack	Queue
The working principle of the stack is LIFO.	Working principale of queue is FIFO.
One end is used to perform the insertion or deletion of elements.	One end is used to perform insertion, and another end is used for the deletion of elements.
It uses one pointer.	It uses two pointers in a simple queue.
It does not have any kind of variant.	It has variants like priority queue, circular Queue, doubly ended Queue.
It is easy to use.	It is not easy to use.

77) What is the difference between array and stack?

The difference between array and stack is:

Array	Stack
It is a collection of elements that are identified by the index.	It is a collection operation that serve as operations push and pop.
It has a elements of data types which are same.	It has a elements of data types which are different.
Elements can be removed or added into the array using random access operation.	Elements can be removed or added into a stack using LIFO operation.

78) Define Iterator()

The Iterator() is an interface that provides methods to iterate Collection. Iterator can take the place of Enumeration in Java. It allows the caller to remove elements from the collection. The method provides a generic way for traversal using elements of the collection and implementing iterator design pattern.

79) What are the various ways to iterate over a list?

Java collection Framework programmer can iterate over a list in two ways: 1) Using iterator, and 2) using it for each loop.

80) What are the advantages of the stack?

The advantages of the stack are:

It helps you to manage the data in a Last In First Out (LIFO) method, which is not possible with the Linked list and array.

When a function is called, the local variables are stored in a stack, and it is automatically destroyed once returned.

A stack is used when a variable is not used outside that function.

It allows you to control how memory is allocated and deallocated.

Stack automatically cleans up the object.

Not easily corrupted

Variables cannot be resized.

These interview questions will also help in your viva(orals)