**Collection Framework | ArrayList, LinkedList & Collection Interface**

**What is framework in Java?**

A framework is a popular and ready-made architecture that contains a set of classes and interfaces.

**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.

----------------------------------------------------------------------------------------- **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.

**Collection vs Collections**

* Collection is an interface that represents group of Objects
* Collections is an utility class that provides method to operate collection type(like searching ,sorting, modifying)

-----------------------------------------------------------------------------------------------------

**Explain the basic interfaces of the Java collections framework**

* Collection interface is root interface of java collections framework which represents a group of objects know as elements
* Collection interface is a part of java.util package

Methods in collection interface- add(),remove, contains(check elements),size(),clear(remove all elements), isEmpty(check empty or not),add All(we merge two list),remove all(intersection),clear(list empty) ,toarray(object to array)

* **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.

A diagram of a framework

Description automatically generated

**Explain the best practices in Java Collection Framework**

The best practices in Java Collection Framework are:

1. **Choose the Correct Type of Collection**: Pick collections based on the specific need—List for ordered elements, Set for unique items, Map for key-value pairs, and Queue for FIFO processing.
2. **Write Programs in Terms of Interfaces**: Code to interfaces (e.g., List, Set) instead of concrete classes to make future implementation changes easier without affecting the rest of the code.
3. **Use Generics for Type Safety**: Generics provide compile-time type checking, preventing ClassCastException and making your code type-safe and more reliable.
4. **Use Immutable Classes**: Prefer immutable classes like String and Integer to ensure thread safety and avoid bugs; avoid custom equals() and hashCode() unless necessary.
5. **Use the Collections Utility Class**: Use Collections for common operations like sorting, searching, and creating synchronized/empty collections, reducing boilerplate and improving reusability.

* **List->** **A List in Java is an ordered collection that allows duplicate elements and provides methods to manipulate elements, such as adding, removing, and accessing items by index**
* ordered collection (like dynamic array )
* Allow duplicates
* List is Interface and it extends to collection
* Array List ,Linked list, vector, Stack

ArrayList, Linked list, Vector, Stack

**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](https://www.guru99.com/foreach-loop-java.html).

**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.

**---------------------------------------------------------------------------------**

**What is ArrayList in Java?**

Resizable array (automatically adjust their capacity), allow duplicates, random access, Array List can store **null** values. Due to adding removing TC is O(n)

**A computer screen with white text

Description automatically generatedArray List <String, Integer> AL=new Array List<>()**;

**Internal working in ArrayList ( new size=oldsize\*3/2+1) e**.g 10->increase by 6 ->10\*3->30/2=15+1=16

**////////////////////////////////////////////////////////////////////////////////////////////////////////**

**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:

**A screen shot of a computer program

Description automatically generated**A computer screen shot of a program code

Description automatically generatedList\_object.toArray(new String[List\_object.size()])

**Linked List:- Linked list is class implementation of List and Deque interface its doubly linked list**

**List<> ls=new Linkedlist<>();**

* Linked list store elements in separate containers called nodes where each contain element and like to next node
* No need to resize like Array list
* Faster when adding and removing elements

A screenshot of a black screen

Description automatically generatedA screenshot of a computer

Description automatically generatedSame method and programmatic working as Array List

**////////////////////////////////////////////////////////////////////////////////////////////**

**Explain linked list supported by Java**

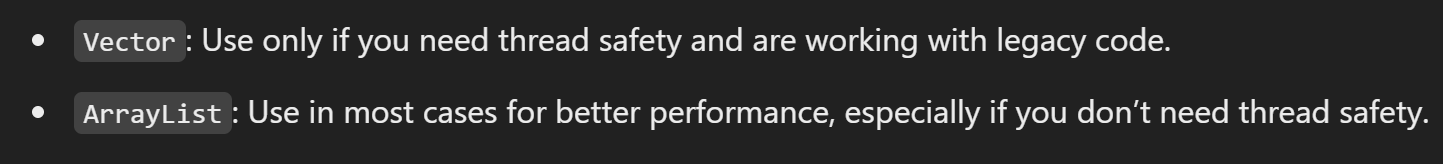
Two types of linked list supported by Java are:

* **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.

**------------------------------------------------------------------------------------------------------------------**

A screenshot of a computer program

Description automatically generated**Vector: -**



A black screen with white text

Description automatically generated

**-----------------------------------------------------------------------------------------------------------------------------**

**Synchronized vs Non-Synchronized**

**A screenshot of a computer

Description automatically generatedonly one thread can access a particular section of code or data at a time. It is used to prevent issues when multiple threads try to modify data simultaneously, which could lead to data corruption or unexpected behavior.**

**Non-synchronization allows multiple threads to access and modify shared resources simultaneously, which can lead to errors.**

**---------------------------------------------------------------------------------------------------------------------------**

1. **Stack: - stack is subclass of vector that implements a LIFO data Structure .it provides method push(),pop(),peek(),empty()**

**A screen shot of a computer code

Description automatically generated**

**//////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**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
* Local variables in functions are stored in the stack and removed when the function ends.
* A stack is used when we want 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
* A screen shot of a computer program

  Description automatically generated[Variables](https://www.guru99.com/java-variables.html)cannot be resized.

**A grid of text on a white surface

Description automatically generated**

**Java Queue Interface**

The Queue is interface of the java collection framework provides the functionality of the queue Data structure .it extends collection interface(FIFO)

A screenshot of a computer screen

Description automatically generatedA blue squares with black text

Description automatically generated

|  |  |
| --- | --- |
| * **offer(E e)🡪**insert elements return true false ( returns false if queue is full) | Boolean Add(E e)🡪insert element but return exception if queue is full |
| * **E poll()🡪** remove elements from queue and return null if q is empty | E remove 🡪remove but return exception if queue is empty |
| * **E peek()🡪** retrieve element which will be remove only showing and return null if queue empty | E element🡪 retrieve element if q empty throw an exception |

1. A screen shot of a computer code

   Description automatically generated**Linked List Queue**🡪

* The linked list class implements to queue Interface and provide method like peek(),poll(),offer()
* It follow FIFO

1. **Array Deque**🡪 Java.util.Deque

* array Deque is resizable array based implementation of the Deque(Double ended queue)
* A black background with white text

  Description automatically generatedA screen shot of a computer code

  Description automatically generatedit allows elements to be added or removal from both side of queue (pollFirst(),peekFirst(),offerFirst(),getFirst)

is Java, DQ 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.

//////////////////////////////////////////////////////////////////////////////////////

1. **Priority queue**🡪

* According to priority elements are remove and insert
* Elements which is higher priority are removed first if two elements have same priority they are removed in the ordered they were added
* React like minheap means always remove smallest element but we can also custom it like maxheap
* Automatically ordered from smallest element ascending

A screen shot of a computer code

Description automatically generated

**////////////////////////////////////////////////////**

**Define BlockingQueue**

**BlockingQueue is an interface that extends Queue and allows thread-safe operations with blocking behavior. It waits for the queue to become non-empty when retrieving elements or non-full when adding elements.**

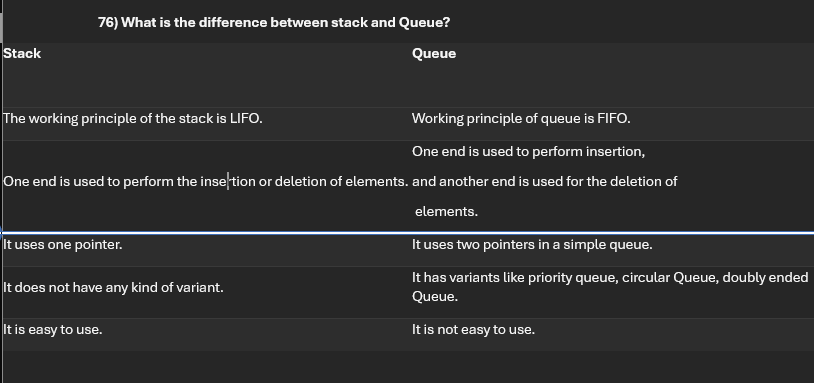
multiple threads can run concurrently, sharing resources without interfering with each other.

The syntax of BlockingQueue is:

public interface BlockingQueue<E> extends Queue <E>

**Explain Circular queues in Java**

**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.

****

**Sets🡪**

A diagram of a set

Description automatically generated

**Java Set interface**

* **The set interface of java collection framework provides the features of the mathematical set in java**
* **It Extend the collection interface it cannot contains duplicate elements**
* **Time complexity is good o(n)**
* A diagram of a hash set

  Description automatically generated

**Interfaces**

* Add()🡪add element to set
* addAll()🡪adds all element of the specified collection to the set
* remove()🡪remove specified elements from set
* remove all ()🡪remove all the elements
* retain all()🡪retains all the elements //intersection
* clear()🡪remove all the elements from set
* size()🡪return length of set
* contains🡪check elements pre

**Define EnumSet**

* EnumSet is a special type of Set in Java that only works with enum types. It stores enum values efficiently using bit vectors, and it does not allow null values. It's not synchronized and provides fast operations for enum collections. provides operations like adding, removing, and manipulating enums in a set-like structure.
* Eg. Main three
* enum Color { RED, GREEN, BLUE, YELLOW }
* EnumSet<Color> colors = EnumSet.of(Color.RED, Color.GREEN); // Red and Green
* EnumSet<Color> allColors = EnumSet.allOf(Color.class); // All colors EnumSet<Color> noColors = EnumSet.noneOf(Color.class); // Empty set

**Java Hash set**🡪

* in java hash set is commonly used if we have to access elements randomly
* hash set cannot contain duplicate hench each hash set has a unique hash code
* A screenshot of a computer code

  Description automatically generatedHash code is function like hash code(input)🡪output //hash code has unique identity

**A screenshot of a computer code

Description automatically generatedHash set Using custom objects**

**What is the hashCode ()?**

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

* hash set cannot contain duplicate hench each hash set has a unique hash code
* Hash code is function like hash code(input)🡪output //hash code has unique identity

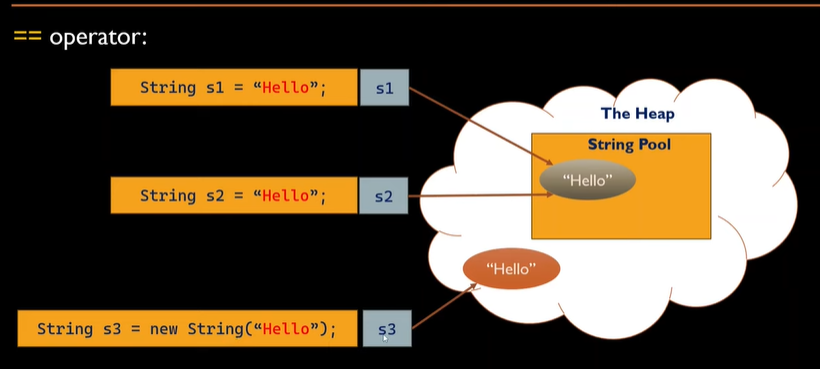
**Hash code and Equals Methods**

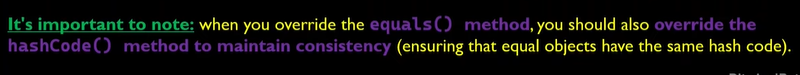
In java both method is used for comparing objects like equal or not

**Equals () and == operator**

**“== Operator”**

It compares the reference equality of two objects its check two objects references points to the same memory location or same object like below.

****

* here for non primitive(object) we can see reference (s1,s2) are refer to same String Object(hello) and in same memory location so output is True
* for primitive types (int float, char) it compares actual value of variables

///////////////////////////////////////////////////////////////////////////////////////////////////////

**Explain Linked HashSet**

LinkedHashSet is a class in Java that implements the Set interface and extends HashSet. It is a **combination of a hash table** (for fast lookups) and a **doubly linked list** (to maintain insertion order). This means it behaves like a HashSet (stores unique elements) but also maintains the order in which elements were inserted.

**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.
* A computer screen shot of a computer code

  Description automatically generated**LinkedHashSet:** Developers can also maintain the order of insertion by using LinkedHashSet instead of HashSet.

//////////////////////////////////////////////////////////////////////////

**List various classes available in sets**

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

**A screenshot of a computer

Description automatically generated**

-----------------------------------------------------------------------------------------------------------------------

**SortedSet:**

* **Definition: SortedSet is an interface in Java that defines a collection of elements sorted in a specific order (either natural or using a comparator).**
* Key Points:
  + Guarantees elements are stored in sorted order.
  + A screenshot of a computer program

    Description automatically generatedProvides methods like first(), last(), and subSet() to access sorted elements.
  + It doesn't define how elements are stored or managed, just that they must be sorted.

**Tree Set:**

* Definition: TreeSet is a class that implements the SortedSet interface and stores elements in a sorted order using a Red-Black Tree (a type of balanced binary search tree).
* Key Points:
  + Automatically sorts elements based on natural order or a custom comparator.
  + Does not allow null values.
  + Provides methods for navigating and manipulating the sorted collection, like first(), last(), and higher().
  + Operations like add(), remove(), and contains() are efficient, running in O(log n) time.

**Map interface🡪**

* In java elements of map are stored in key/value pairs. Keys are unique value associated with individual value
* A diagram of values

  Description automatically generatedA screenshot of a cell phone

  Description automatically generatedA map cannot contains duplicate keys and each key is associated with single value
* Put (k, v)🡪for insert element with key value pair
* PutAll()🡪insert all entries from from the specified map to this map
* PutifAbset(k, v)🡪if absent keypair added
* Get(value or key)🡪get values
* getorDefault(“key” defaultvalue)🡪if value is not present then get defaultvalue
* containsKey(k)🡪checks the specified key is present or not
* containsValue()🡪to check value
* remove()🡪to remove key and using this we can remove key pair
* keyset()🡪we get set of key\
* values()🡪we get set of value
* entrySet()🡪all key and value in present in map
* all complexity O(n)

A screenshot of a computer program

Description automatically generated

**Here are the types of Map in Java:**

1. **HashMap**
2. **LinkedHashMap**
3. **TreeMap**
4. **Hashtable**
5. **WeakHashMap**
6. **IdentityHashMap**
7. **ConcurrentHashMap**

**What is hashmap🡪**

A HashMap is a collection in Java used to store key-value pairs, allowing fast access to values based on their corresponding keys. It is unordered and efficient for quick lookups.

**A screenshot of a computer program

Description automatically generated**

**/////////////////////////////////////////////////////////////////////////**

What is hashtable in java

A **Hashtable** is a collection class in Java that implements the **Map** interface and is used to store key-value pairs. It is similar to **HashMap**, but with a few differences:

Hashtable does not allow null keys or null values. If you try to insert a null key or value, it throws a NullPointerException.

**A screenshot of a black screen

Description automatically generated**

**Explain Linkedhashmap**

LinkedHashMap is a class that implements the Map interface and extends HashMap. It provides the same functionality as HashMap

but with one key difference: it maintains the insertion order of the entries

**A screen shot of a computer

Description automatically generated**Allows One null Key and Multiple null Values: Just **like HashMap**

**/////////////////////////////////////////**

**Explain Tree map in Java**

A screenshot of a computer program

Description automatically generatedA computer screen shot of a computer code

Description automatically generatedTreeMap is a class that implements the Map interface and stores data in sorted order based on keys. It uses a Red-Black Tree internally to maintain the sorted order of keys. It also provides additional navigation methods like firstKey(), lastKey()

**////////////////////////////////////////////////////////////////////////**

**38) What is WeakHashMap?**

* **WeakHashMap** is a special type of map where the **keys** are stored as **weak references**. This allows the garbage collector to automatically remove entries when their keys are no longer in use.
* It’s useful for **caching** and **memory-sensitive applications**, where you want the map to automatically clean up unused entries without needing explicit removal.

////////////////////////////////////////////////////////////

A screenshot of a black screen

Description automatically generated

**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.

----------------------------------------------------------------------------------------------------------------------------

**Differentiate between Iterator and ListIterator**

A screenshot of a computer

Description automatically generated

* iterator in java is an object that allows you to traverse through collection (like Arraylist, Hashset)
* it provides methods to access and remove elements during iteration
*  process of repeatedly accessing each element in a collection or a sequence, one after another. This can be done using loops (e.g., for, while) or with an Iterator.

A computer code on a black background

Description automatically generated

--------------------------------------------------------------------------------------------------------------------------**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. |

**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.

-------------------------------------------------------------------------------------------------------------------

**What is the difference between Comparable and Comparator?**

**A screenshot of a computer

Description automatically generated------------------------------------------------------------------------------------------------------------**

**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.

-----------------------------------------------------------------------------------------------------------

**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:

A screenshot of a computer

Description automatically generatedpublic static final <T> Set<T> emptySet()

**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.

--------------------------------------------------------------------------------------------

**Define dictionary class**

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

------------------------------------------------------------------------------------

**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);

----------------------------------------------------------------------------------------

**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.

---------------------------------------------------------------------------------------

**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.

---------------------------------------------------------------------------------------------

**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.

-----------------------------------------------------------------------------------------

**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.

-------------------------------------------------------------------------------

**A screenshot of a video game

Description automatically generated**

**-----------------------------------------------------------------------------------------**

**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.

-----------------------------------------------------------------------

**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.

----------------------------------------------------------------------

**Explain randomaccess interface**

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

**-----------------------------------------------------------------**

**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.

-------------------------------------------------

**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);

-------------------------------------------

**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.

-------------------------------------------------------------

**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).