Topic Name :🡪Set details Map Comparable and Comparator interface Internal working of map and set

Q1) How can we pass object of arraylist into hashmap ?

[Array List](https://www.geeksforgeeks.org/arraylist-in-java/) can be converted into [HashMap,](https://www.geeksforgeeks.org/java-util-hashmap-in-java-with-examples/) but the HashMap does not maintain the order of ArrayList. To maintain the order, we can use LinkedHashMap which is the implementation of HashMap.

Basically, there are two different ways to convert ArrayList to Hashmap**-**

1. Using ArrayList Iteration
2. Using ArrayList Iteration with LinkedHashMap

**1)Using ArrayList Iteration:**

Here, we just need to iterate on each of the elements of the ArrayList and the element can be converted into the key-value pair and store in the HashMap.

**2)Using ArrayList Iteration with LinkedHashMap :**

* Here, the  Array List is converted into a HashMap but HashMap does not maintain the order of the ArrayList.
* To maintain the order, we use LinkedHashMap which is an implementation of HashMap and helps us to maintain the order of the elements, and we can easily convert Arraylist to Hashmap.

Q2) Difference Hashmap vs hashtable?

|  |  |
| --- | --- |
| **HashMap** | **Hashtable** |
| 1) HashMap is **non synchronized**. It is not-thread safe and can't be shared between many threads without proper synchronization code. | Hashtable is **synchronized**. It is thread-safe and can be shared with many threads. |
| 2) HashMap **allows one null key and multiple null values**. | Hashtable **doesn't allow any null key or value**. |
| 3) HashMap is **fast**. | Hashtable is **slow**. |
| 4) HashMap is **traversed by Iterator**. | Hashtable is **traversed by Enumerator and Iterator**. |
| 5) Iterator in HashMap is **fail-fast**. | Enumerator in Hashtable is **not fail-fast**. |
| 6) HashMap inherits **AbstractMap** class. | Hashtable inherits **Dictionary** class. |

Q3) What is hashing. ?

Hashing is the process of converting a given key into another value. A **hash function** is used to generate the new value according to a mathematical algorithm. The result of a hash function is known as a **hash value** or simply, a **hash**.

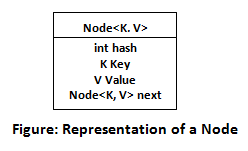
A good hash function uses a **one-way** hashing algorithm, or in other words, the hash cannot be converted back into the original key.

Q4) Internal working of hashmap ?

HashMap is a part of the Java collection framework. It uses a technique called Hashing.

It implements the map interface. It stores the data in the pair of Key and Value. HashMap contains an array of the nodes, and the node is represented as a class.

It uses an array and LinkedList data structure internally for storing Key and Value. There are four fields in HashMap.



Before understanding the internal working of HashMap, you must be aware of hashCode() and equals() method.

* **equals():** It checks the equality of two objects. It compares the Key, whether they are equal or not. It is a method of the Object class. It can be overridden.
* If you override the equals() method, then it is mandatory to override the hashCode() method.
* **hashCode():** This is the method of the object class.
* It returns the memory reference of the object in integer form. The value received from the method is used as the bucket number.
* The bucket number is the address of the element inside the map. Hash code of null Key is 0.
* **Buckets:** Array of the node is called buckets. Each node has a data structure like a LinkedList.
* More than one node can share the same bucket. It may be different in capacity.



Q4) Difference between sorted map and trees set ?

|  |  |
| --- | --- |
| **TreeSet** | **TreeMap** |
| TreeSet implements SortedSet in Java | TreeMap implements Map Interface in Java |
| TreeSet stored a single object in java. | TreeMap stores two Object one Key and one value. |
| TreeSet does not allow duplication Object in java. | TreeMap in java allows duplication of values. |
| TreeSet implements NavigableSet in Java. | TreeMap implements NavigableMap in Java. |
| TreeSet is sorted based on objects. | TreeMap is sorted based on keys. |

Q5) what is ConcurrentModificationException ?

The ConcurrentModificationException occurs when an object is tried to be modified concurrently when it is not permissible. This exception usually comes when one is working with Java Collection classes.

**For Example** - It is not permissible for a thread to modify a Collection when some other thread is iterating over it. This is because the result of the iteration becomes undefined with it.

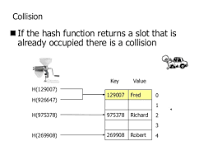
Q6) Arraylist vs hashmap ?

|  |  |
| --- | --- |
| **ArrayList** | **HashMap** |
| ArrayList implements the List interface. | HashMap implements the Map interface. |
| ArrayList implements the List interface. | HashMap implements the Map interface. |
| ArrayList stores element's value and maintains the indexes for each element. | HashMap stores elements **key** & value pair. For each value, there must be a key associated with HashMap. |
| ArrayList stores only a single object. | HashMap stores elements in Key and value pairs. |
| ArrayList allows duplicate elements. | HashMap allows duplicate values but does not allow duplicate keys. |
| ArrayList has any number of null elements. | HashMap allows only one null Key and lots of null values. |
| ArrayList is the index-based data structure supported by the array. | While HashMap is a mapped data structure that works on hashing to obtain stored values. |

Q7) Collision in hashmap ?

a hash code collision in a HashMap, is a situation where two or more key objects produce the same final hash value and hence point to the same bucket location or array index.

How is collision handled in HashMap?

[[](https://www.google.com/search?rlz=1C1JJTC_enIN974IN974&biw=1229&bih=596&sxsrf=APq-WBu9M7XdWKJIwSi5UnkS8WodtVprEg:1650257508004&q=How+is+collision+handled+in+HashMap?&tbm=isch&source=iu&ictx=1&vet=1&fir=edlNIrBNOxe-oM%252CMEFu48RMZD__0M%252C_&usg=AI4_-kQSpUE1gvc2mc7zJtoTvvrtnvIxvA&sa=X&ved=2ahUKEwjJubWk6Jz3AhXjmuYKHU_HDS8Q9QF6BAgUEAE#imgrc=edlNIrBNOxe-oM)](https://www.google.com/search?rlz=1C1JJTC_enIN974IN974&biw=1229&bih=596&sxsrf=APq-WBu9M7XdWKJIwSi5UnkS8WodtVprEg:1650257508004&q=How+is+collision+handled+in+HashMap?&tbm=isch&source=iu&ictx=1&vet=1&fir=edlNIrBNOxe-oM%252CMEFu48RMZD__0M%252C_&usg=AI4_-kQSpUE1gvc2mc7zJtoTvvrtnvIxvA&sa=X&ved=2ahUKEwjJubWk6Jz3AhXjmuYKHU_HDS8Q9QF6BAgUEAE" \l "imgrc=edlNIrBNOxe-oM)

1) HashMap handles collision by using a linked list to store map entries ended up in same array location or bucket location.

2) From Java 8 onwards, HashMap, ConcurrentHashMap, and LinkedHashMap will use the balanced tree in place of linked list to handle frequently hash collisions

Q8) what is generic?

The Java Generics programming is introduced in J2SE 5 to deal with type-safe objects. It makes the code stable by detecting the bugs at compile time.

Before generics, we can store any type of objects in the collection, i.e., non-generic. Now generics force the java programmer to store a specific type of objects.

There are mainly 3 advantages of generics. They are as follows:

**1) Type-safety:** We can hold only a single type of objects in generics. It doesn?t allow to store other objects.

**2) Type casting is not required:** There is no need to typecast the object.

**3) Compile-Time Checking:** It is checked at compile time so problem will not occur at runtime. The good programming strategy says it is far better to handle the problem at compile time than runtime.

Q9) Can we hashmap in synchronisation with multithreading ?

* HashMap is not synchronized, which means you cannot use it on multi-threaded Java programs without external synchronization.
* In another word, if you share one instance of HashMap between multiple threads, each is either adding, removing or updating entries then it's possible that HashMap loses its structure and not behave as expected.

Q10) Sorted map by value ?   
In Java, sorting HashMap by values is complicated because there is no direct method available. If we need to sort the HashMap by values, we should create a **Comparator.**

It compares two elements based on the values.

Java collections class provides a method to sort all list implementations such as LinkedList and ArrayList.

There are two overloaded sort methods():

* **sort(List list):** It sorts the elements of the List in ascending order of their natural order.
* **sort(List list, Comparator <T>):** It sorts the elements of the list according to the order included by the comparator.

**Q11)** we can pass primitive and object in hashmap as key ?

HashMap doesn't handle primitives, just objects.

You cannot store primitive types in any java.

Q12) Difference between concurrent HashMap & synchronized hashmap ?

|  |  |
| --- | --- |
| **Concurrent hash map** | Synchronized hashmap |
| It is a class that implements a Concurrent hash map and serializable interface. | It is a method in Collection class. |
| Locks the portion | Locks the whole map. |
| It doesn't allow null as a key or value. | It allows null as a key. |
| It doesn't throw concurrent modification exception. | Iterator return by synchronized map throws concurrent modification exception |

Q13) Why String used in Hashmap?

|  |
| --- |
| * Since String is immutable, its hashcode is cached at the time of creation and it doesnt need to be calculated again. * This makes it a great candidate for key in a Map and its processing is fast than other HashMap key objects. * This is why String is mostly used Object as HashMap keys. |
|  |

Q14) what is Linked hashmap.?

* Java LinkedHashMap contains values based on the key.
* Java LinkedHashMap contains unique elements.
* Java LinkedHashMap may have one null key and multiple null values.
* Java LinkedHashMap is non synchronized.
* Java LinkedHashMap maintains insertion order.
* The initial default capacity of Java HashMap class is 16 with a load factor of 0.75.

Q15) concurrent hashmap is thread safe or not ? & what he do for thread safety?

* ConcurrentHashMap class is thread-safe i.e. multiple threads can operate on a single object without any complications.
* At a time any number of threads are applicable for a read operation without locking the ConcurrentHashMap object which is not there in HashMap.

Q16) Load factor ?

The load factor is the measure that decides when to increase the capacity of the Map.

The default load factor is 75% of the capacity.

The threshold of a HashMap is approximately the product of current capacity and load factor.

Q17) time complexity of hashmap for getting element and different scenarios ?

Q18) different between map and flat map.

|  |  |
| --- | --- |
| map() | flatMap() |
| The function passed to map() operation returns a single value for a single input. | The function you pass to flatmap() operation returns an arbitrary number of values as the output. |
| One-to-one mapping occurs in map(). | One too many mapping occurs in flatMap(). |
| Only perform the mapping. | Perform mapping as well as flattening. |
| Produce a stream of value. | Produce a stream of stream value. |
| map() is used only for transformation. | flatMap() is used both for transformation and mapping. |