Important Points

3 types of collections (interface):

* List (Ordered Collections)
  + Vector Class
    - It is synchronized-thread safe, legacy
  + ArrayList
    - We need frequent get and set
    - It has random access and fast
  + LinkedList
    - Un synchronized, frequent insertion and deletion, tree like structure, slower than arraylist, implements List and Queue
* Set (Unique Collections)(Object class must override equals and hashCode() methods as set are unique collection)
  + HashSet
    - Unsynchronized, frequent get and set, no order, unique
  + LinkedHashSet
    - Unsynchronized, frequent insertion and deletion, insertion order
* SortedSet (TreeSet)
  + TreeSet
    - Unsynchronized, ordered set
* Map
  + HashTable – legacy, synchronized
  + HashMap – allows null key and value
  + Linked Hash Map
  + Sorted Map
* Java.util has one more Interface which is Map
  + The above interface in implemented by the below 2 classes
  + Assert(class). This has sort method
  + Collections(class) This has sort () method
* Sort method has a contract with Comparable and Comparator
  + Comparable – it has compareTo(Object obj) method
    - Natural order of sorting
    - Implemented by the Object class
  + Comparator – it has a compare(Object obj1, Object obj2)
    - As per requirement of sorting
    - Implemented by any class and further used for sorting

Iterator are of 2 types:

* Iterator – one way
* ListIterator – two way

Execution steps for class:

* Static – Single copy per class, shared resources. Has always a class reference. It does not have any object reference.
  + It is a keyword in java which can be used variables, methods, block, static inner class (class within a class)
* Init block
* Superclass
* Final keyword – It is a keyword in java which can be used with variables and methods and class.
  + Final Variable – must be initialized and cannot be re-initialized
  + Methods – cannot be overridden
  + Final class – Cannot be subclassed, eg : all Wrapper class and String class in java are final. i.e we cannot extend these classes.
* Generics – It is used to enforce type safety in java

IO handling:

* Interfaces – InputStream, OutputStream, Reader, Writer

Exception

* Unchecked exception – when we do not have to mention the try-catch block, then this is called unchecked exception. It will not be asked during coding to handle the exception
* Checked exception – When we need to give the try-catch block then they are checked exception