1.

**Collections** -base interface for most collection classes

**Collection** - Utility class. Group of Objects

**List** - interface class with ordered collection.Duplicates are generally permitted

**Set** - interface class with no duplicate element permitted

**Map** -an interface class which uses keys and values when mapping.

**ArrayList** - Resizable array implementation of the List interface.

**LinkedList** - Doubly-linked list implementation of the List interface. If adding and removing elements is frequent , this provides better performance than ArrayList.

**HashMap** - Hash table implementation of the Map interface.The best all-around implementation of the Map interface.

**TreeMap** - also an implementation of a Map interface .  The map is sorted according to the [natural ordering](file:///D:\Installer\jdk-8u91-docs-all\docs\api\java\lang\Comparable.html) of its keys, or by a [Comparator](file:///D:\Installer\jdk-8u91-docs-all\docs\api\java\util\Comparator.html) provided at map

creation time, depending on which constructor is used.

2. 1

**for** (Object o : list) {

System.***out***.println(o);

}

2.2

Hello

Java

Learn

World

2.3

List list = **new** LinkedList();

                ArrayList maintains indexes and element data while LinkedList maintains element data and two pointers for neighbor nodes hence the memory consumption is high in LinkedList comparatively.

2.4

         List list = **new** Vector();

          ArrayList is non-synchronized which means multiple threads can work on ArrayList at the same time. For Vector, it is synchronize. This means if one thread is working on Vector, no other thread can get a hold of it. Unlike ArrayList, only one thread can perform an operation on vector at a time.

1. **Hello  
   Learn**
2. **3**