1. In java a List is considered an ordered collection of objects, capable of storing duplicate values. It is a type safe interface which extends collection, allowing all of collections’ operations while also allowing a few of its own such as positional access, search, iteration, and range view. Lists can allow positional access and insertion of elements while preserving insertion order. List provides several methods for add, remove, get, and set based on the numerical position of elements. List is capable of search providing methods to search an element and return its numerical position. ListIterator a bidirectional iterator which extends Iterator, is used to iterate over a List element. The List interface provides a method to get the List view of the portion of given List between two indices.
2. Array List is the part of the framework in java that provides us dynamic arrays. Often slower than standard arrays, it can be helpful where lots of array manipulation is needed. The ArrayList inherits the AbstractList class and implements List interface. ArrayList allows us to randomly access the list and can’t be used for primitive types, to do so requires a wrapper class. ArrayList also has several methods to manipulate elements.
3. Although List and ArrayList behave the same and can achieve primarily the same result. The fundamental difference between List and ArrayList is List is an interface and not an actual class whereas ArrayList is a concrete class.
4. HashMap provides the basic implementation of Map interface for java. Data is stored in key, value pairs. A key must be known to access its value. The technique is named HashMap in relation to its ability to use hashing. Hashing converts a large string to a smaller one. That represents the same string. A shorter value helps speed up indexing in searches. The performance of HashMap depends on: 1. The initial capacity 2. The load factor. HashMap is by default unsynchronized supporting multithreading access, but if at least one thread manipulates it structurally then it is necessary to make it synchronized externally.
5. HashTable is similar to HashMap yet it synchronized by default. HashTable implements a hash table which maps keys to values. To retrieve objects from a hash table, the object used as keys must implement the hashCode method and the equals method. In HashTable we specify an object that is used as a key and the associated value. The key is then hashed and he resulting hash code is used as the index at which the value is stored within the table.
6. HashMap and HashTable both store key/value pairs in a hash table. When using either we specify that an object is used as a key and the associated value. One main difference is HashMap is not synchronized nor thread safe and can’t be shared between many threads without proper synchronization. HashTable is the opposite where it is thread safe and can be shared between many threads. HashMap allows one null key and multiple null values whereas HashTable doesn’t allow any null keys or values. To successfully store and retrieve objects from a HashTable, the objects used as keys must implement the hashCode method and the equals method. These methods can’t be implemented since null isn’t an object. HashMap is considered an improvement to HashTable as it was created after such.
7. Set is an unordered collection of objects in which duplicate values can’t be stored. It is an interface which extends Collection. Set consists of various method to manipulate the usage of this interface.
8. HashSet class implements the Set interface, backed by a HashMap instance. The iteration of the order of the set is not fixed and it is possible to change over time. Null elements are also permitted. Constant time performance for basic operations like add, remove, contains, and size assuming the hash function dispersing the elements properly among the buckets.
9. A ConcurrentHashMap is a thread safe class in which multiple threads are available for read without locking the ConcurrentHashMap object which is unavailable in HashMap. In this class it divides the segments according to the concurrency level. The default concurrency level is sixteen but at any time unlimited threads can perform a retrieval operation. For an update, a thread must lock the particular segment in which it wants to operate. This is known as segment locking, so by default sixteen update operations can be performed concurrently. Null values or keys are not permitted in this class.
10. The hashcode method returns the hashcode value as an integer. Hashcode value is primarily used in hashing based collections and must be overridden in every class that overrides the equals method. The equals method uses two ways of comparing objects which will return a Boolean value. The default comparison checks if two object references refer to the same object. A more in depth comparison checks if data members of objects are compared with one another. The difference between the two methods is in hashcode during execution of the application, if hashcode is invoked more than once on the same object it must return the same integer value. This is provided no info in the used in the equals(Object) comparison on the object is modified. If two objects on the equals method are equal according to the equals method, then the hashcode method must produce the same integer on each of the two objects. If two objects are unequal, then it is not necessary for the integer value produced by the hashcode method to be distinct. Producing the distinct integer on each of the two objects is better for performance improvement of hashing based Collections.