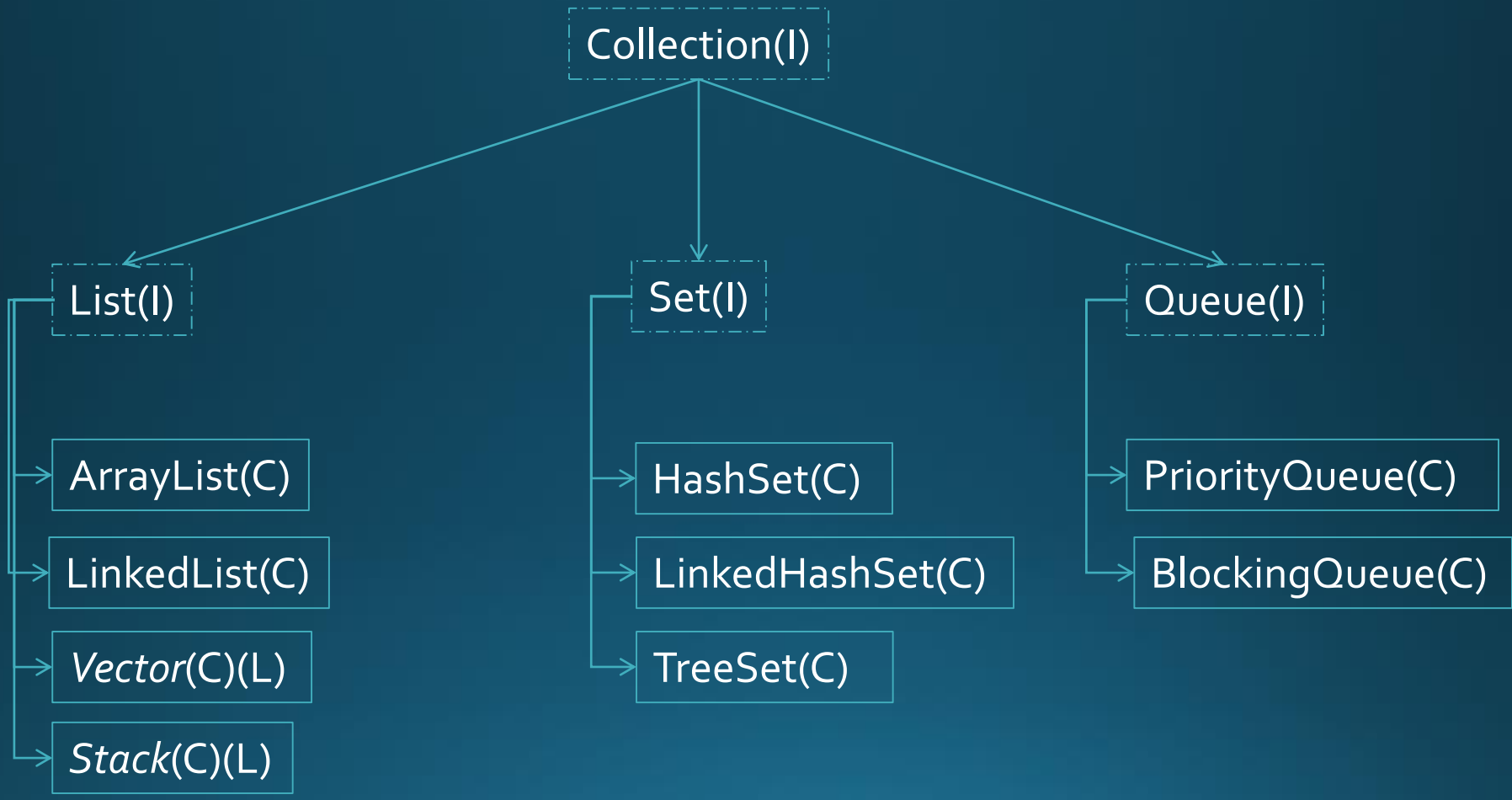
COLLECTIONS FRAMEWORK

RAM SHARMA

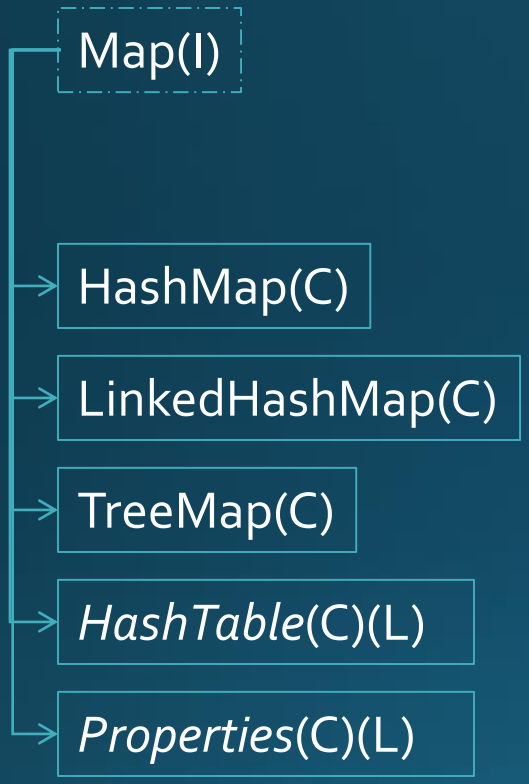
Why Collection?

- A group of individual objects as single entity is called as Collection.
- An array is a an indexed collections of fixed no of homogeneous data elements.
- Array are fixed in size.
- Array can hold only data elements of same type.
- Collection is implemented based on some data structure which provides us inbuilt methods.
- Collections framework defines several classes & interfaces that represents group of objects.

Collections Hierarchy



Collections Hierarchy(contd.)



Utility Classes

- Arrays
- Collections

Cursers

- *Enumeration*(L)
- Iterator
- ListIterator

Comparison

- Comparable
- Comparator

Difference b/w List and Set

Property	List	Set
Insertion Order	Preserved	May not be preserved
Random Insertion	supported	Not supported
Duplicate	allowed	Not allowed
null insertion	allowed	only one
Cursor	Iterator and ListIterator	Iterator
Implementation class	AL, LL, Vector, Stack	HS,LHS,TS
When to use	Insertion Order	Maintaining Unique

- ✓ ArrayList and LinkedList follow insertion order in storing data.
- ✓ HashSet uses hashcode, so doesn't follow any order.
- ✓ LinkedHashSet follow insertion order.
- ✓ TreeSet by default follow natural sorting but we can use comparator to provide customized sorting order.

Various Methods in Collection

For Collection

- boolean add(E e)
- void clear()
- boolean contains(E e)
- boolean equals(C c)
- boolean isEmpty()
- boolean remove(E e)
- int size()
- Iterator<E> iterator()

For List

- void add(int index, E e)
- E get(int index)
- int indexOf(E e)
- int lastIndexOf(E e)
- ListIterator<E> listIterator()
- E remove(int index)

Map

- A map contains values on the basis of key.
- Each key and value pair is known as an entry.
- Map contains only unique keys.
- HashMap follows no sorting method.
- LinkedHashMap follow inserting order.
- TreeMap by default provides natural ordering but in customized sorting can be provided using Comparator interface.
- HashTable is synchronized, i.e, thread safe.
- Properties is a very specialized class that's designed to hold configuration and/or resources that are usually stored in some file.

Various Methods in Map

- void clear()
- boolean containsKey(Object e)
- boolean containsValue(Object e)
- V put(K key, V value)
- V remove(Object key)
- boolean equals(Object o)
- boolean isEmpty()
- int size()
- Set<K> keySet()
- Collection<V> values()
- Set<Map.Entry<K,V>> entrySet()

Map.Entry(I)

- Object getKey()
- Object getValue()
- V setValue(V value)