Java Collections Framework :-

⇒ A collection represent a group of object

Java Collection provide class and interfaces that helps to write code easily & efficiently

- Few Commonly used Collections –

ArrayList –

⇒ `ArrayList<Integer> arr = new ArrayList<>();`

Methods in ArrayList

1. add (int index, object element)
2. add (object o)
3. addAll (collection c)
4. addAll (int index, collection c)
5. clear()

⑥ clone()

⑦ Contains(object o)

⑧ get(int index)

⑨ indexOf(Object o)

⑩ remove(int index)

⑪ set(index, Element)

⑫ size()

Sorting in ArrayList -

```
ArrayList<Integer> list = new ArrayList();
Collections.sort(list);
```

To see the List as an Array -

⇒ System.out.println(list);

To see ArrayList is Empty

⇒ Sout(list.isEmpty())

to get the Fifth Element

⇒ Sout(list.get(4));

Linked List in Java

⇒ LinkedList is a part of Collection Framework present in java.util.package.

⇒ Every element are not stored as contiguous locations and every element is having data part and address part.

⇒ This works internally using doubly linkedlist.

#LinkedList<String> l2 = new LinkedList<String>();

- Common methods -

① l2.add("Ajay");
② l1.addLast("Paswan");
③ l2.addFirst("Mr");
④ l2.remove("Mr");
⑤ l2.remove(index)
⑥ l1.get(index)
⑦ l2.getFirst(index)
⑧ l2.peek() ⑨ PeekLast()

⑩ L.poll() → remove the head of the list

⑪ L.pollLast() → remove the last element of the list.

⑫ L.pop() → returns the last Element and remove it.

⑬ L.remove(o) → returns & remove the first occurrence.

Hashtable in Java –

⇒ Hashtable stores key/value pairs

⇒ The initial default capacity of Hashtable is 11 whereas loadfactor is 0.75.

Declaration – Hashtable<Integer, String> h2 = new Hashtable<>();

(size, Fill ratio)

Methods –

① ht2.put(1, "one");

② h2 l = new Hashtable<Integer, String>(h1);

③ remove (key) ⇒ remove the element from the hashtable.

④ Traversal -

```
For (Map.Entry<String, Integer> e : ht.entrySet()){
    Sout (e.getKey()+" "+ e.getValue());
}
```

⑤ Integer a = ht.get("Ajay");

HashMap -

⇒ It also stores key value pair, but is unsynchronized.

⇒ default initial capacity of 16 and load factor of 0.75.

- Declaration - HashMap<Integer, String> mp = new HashMap<Integer, String>();

Methods - Same as Hashtable.

HashSet

create unique collection of object.

⇒ HashSet<String> h = new HashSet<String>();

Methods -

① h.add("Ajay");

② h.remove("Ajay");

③ Iteration in hashset -

i) Iterator itr = h.iterator();

```
while(itr.hasNext()){
    sout(itr.next());
}
```

ii)

```
for (String s : h){
    sout(s);
}
```