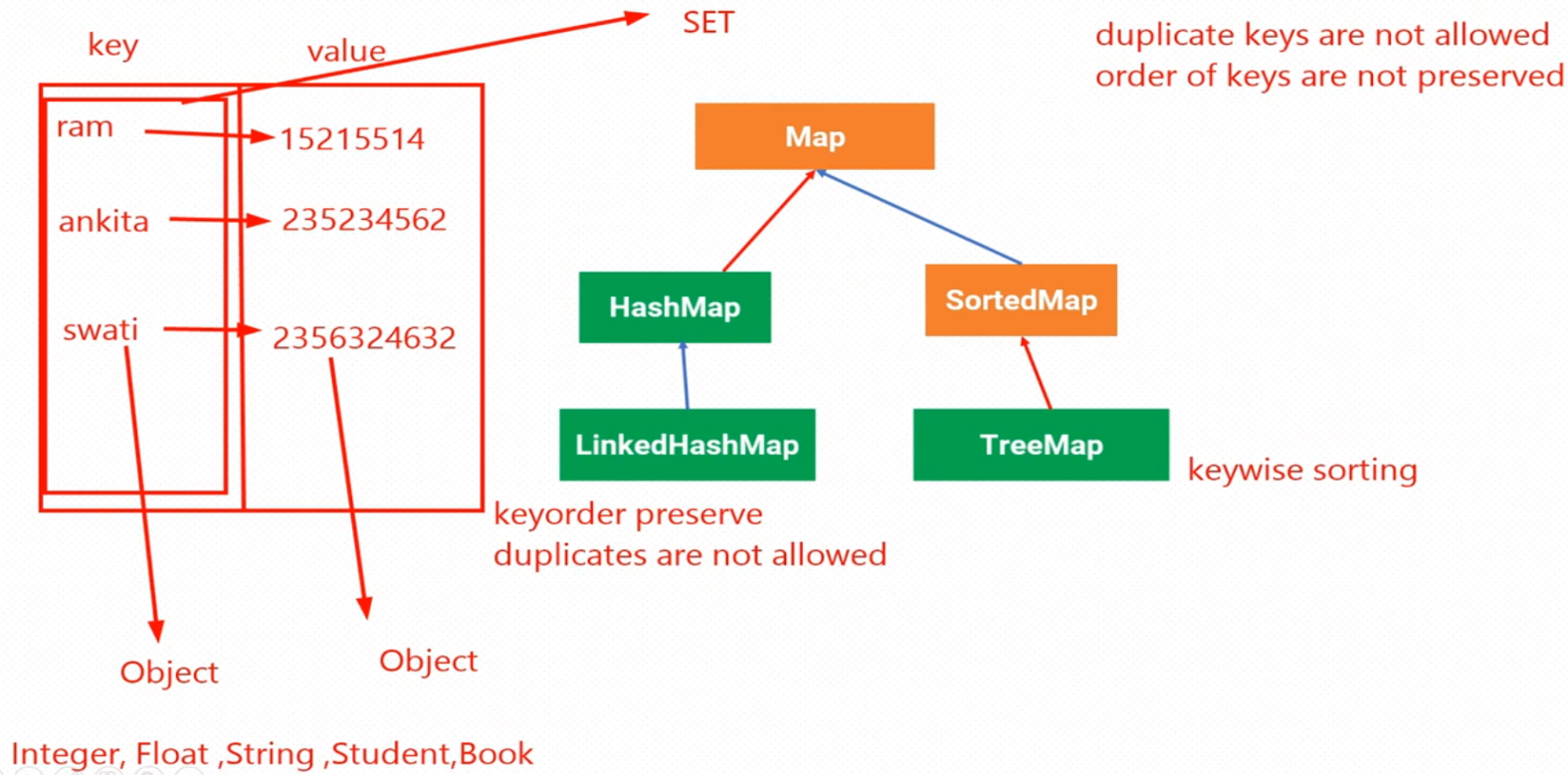


Green Boxes Represents Class.

Orange boxes represents Interface.

Blue lines represents extend relationship.

Red lines represents implements relationship.



**List**

1. Adding element in List

List <String> list = new ArrayList<String>(); or new LinkedList<String>();

list.**add**(“element1”); -> add at element at the end

list.add(index, “element”); -> add at particular index

1. Removing element in list

We can use **remove()** method

remove(int index) -> remove from specific index.

remove(Obj obj) -> remove first occurrence of specific object

1. How to get element from List at particular index

**get(int index)** -> returns the object at provided index.

1. How to get index of specific element in List.

**indexOf(Object obj**) -> return the index of first occurrence of obj.

returns -1 if obj is not present in list.

1. Traversing a list

Use stream.**forEach**()

**Set**

1. Adding element to Set

set.**add**(“element”) -> return false if element is already present in set.

1. Removing element from Set

set.**remove**(Obj obj) -> returns true if obj is present into set.

1. Traversing a set

We have 3 methods

1. Using iterator
2. Using enhanced for loop -> for(String stock : setOfStocks){System.out.println(stock);}
3. Using forEach loop -> setOfStock.forEach(i -> System.out.println(i))

**Map**

1. Adding element to Map.

put(key,value)

1. Removing element from Map.

remove(key) -> remove element having this particular key.

1. Traversing a map -> use map.forEach((k,v)->{//do anything with key & value})
2. Getting list of keys in Map ->

keySet() method returns a Set of keys contained in the map.

1. Getting List of values from Map ->

values() method returns a Collection view of the values.

1. Getting value corresponding to a key ->

get(Object key)

1. To check whether a key is present in Map

containsKey(key) -> returns true if key is present in Map otherwise returns false.

**Queue(LIFO)**

1. queue.**add**(“element”);
2. queue.**remove**() //removes from top of queue.
3. Traversing
4. Using enhanced for loop
5. Using iterator
6. Using forEach

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Add | Remove | Traverse | Retrieve |
| LIST | list.add(el) list.add(index,el) | list.remove(index)  list.remove(obj) | list.forEach | list.get(index) list.indexOf(obj) |
| Queue | queue.add(el) | queue.remove() | 1.Enhanced For loop  2.Iterators 3.forEach | N/A |
| SET | set.add(el) | set.remove(obj) | 1.Enhanced For loop  2.Iterators 3.forEach | N/A |
| Map | map.put(key,value) | map.remove(key) | map.forEach(  (key,value) -> {  }) | map.keySet() map.values() map.get(key) map.containsKey(key) |