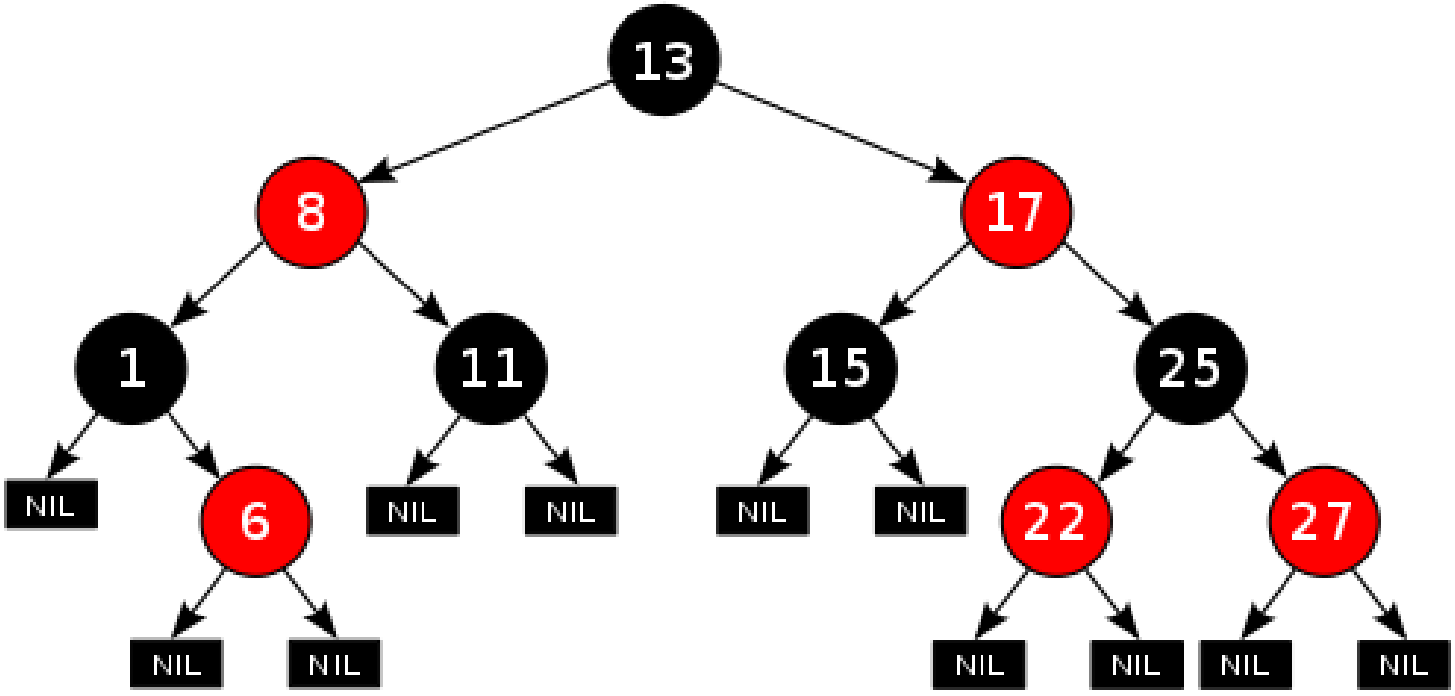
# Data Structure

# Lab #2

# Red Black Tree




Omar Mohamed Emam ID: 44

## INodeClass:

The implementation of inode interface with the same implementation as the previous lab

## IRedBlackTreeClass:

The implementation of IRedBlackTree interface:

1. getRoot: return the root of the given Red black tree.

2. isEmpty: return whether the given tree isEmpty or not.

3. clear: Clear all keys in the given tree.

4. search: return the value associated with the given key or null if no value is found.

5. contains: return true if the tree contains the given key and false otherwise.

6. insert: Insert the given key in the tree while maintaining the red black tree properties. If

the key is already present in the tree, update its value.

7. delete: Delete the node associated with the given key. Return true in case of success and

false otherwise.

In addition to some additional functions described in code comments

## ITreeMapClass:

The implementation of ITreeMap interface:

1. ceilingEntry: Returns a key-value mapping associated with the least key greater than or

equal to the given key, or null if there is no such key.

2. ceilingKey: Returns the least key greater than or equal to the given key, or null if there

is no such key.

3. clear: Removes all of the mappings from this map.

4. containsKey: Returns true if this map contains a mapping for the specified key.

5. containsValue: Returns true if this map maps one or more keys to the specified value.

6. entrySet: Returns a Set view of the mappings contained in this map in ascending key

order.

7. firstEntry: Returns a key-value mapping associated with the least key in this map, or

null if the map is empty.

8. firstKey: Returns the first (lowest) key currently in this map, or null if the map is empty.

9. floorEntry: Returns a key-value mapping associated with the greatest key less than or

equal to the given key, or null if there is no such key.

10. floorKey: Returns the greatest key less than or equal to the given key, or null if there is

no such key.

11. get: Returns the value to which the specified key is mapped, or null if this map contains

no mapping for the key.

12. headMap: Returns a view of the portion of this map whose keys are strictly less than

toKey in ascending order.

13. headMap: Returns a view of the portion of this map whose keys are less than (or equal

to, if inclusive is true) toKey in ascending order..

14. keySet: Returns a Set view of the keys contained in this map.

15. lastEntry: Returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.

16. lastKey: Returns the last (highest) key currently in this map.

17. pollFirstElement: Removes and returns a key-value mapping associated with the least

key in this map, or null if the map is empty.

18. pollLastEntry: Removes and returns a key-value mapping associated with the greatest

key in this map, or null if the map is empty.

19. put: Associates the specified value with the specified key in this map.

20. putAll: Copies all of the mappings from the specified map to this map.

21. remove: Removes the mapping for this key from this TreeMap if present.

22. size: Returns the number of key-value mappings in this map.

23. values: Returns a Collection view of the values contained in this map

## Sample Run:

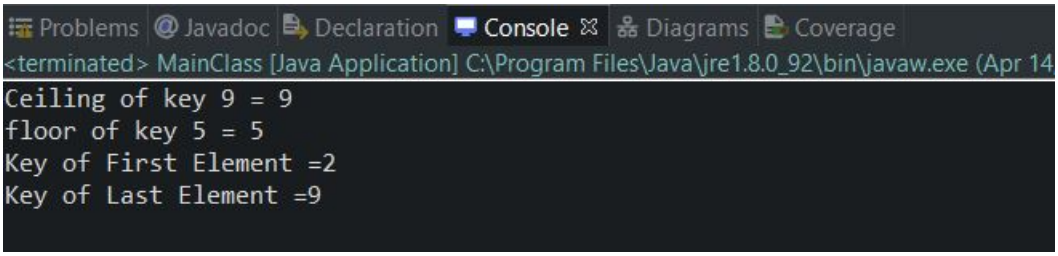
```java
package eg.edu.alexu.csd.filestructure.redblacktree;

import java.util.Map;

public class MainClass {
    public static  void main(String[] args) {
        ITreeMapClass tree = new ITreeMapClass();

        tree.put(2,2);
        tree.put(4,4);
        tree.put(3,3);
        tree.put(9,9);
        tree.put(8,8);
        tree.put(3,3);
        tree.put(5,5);


        System.out.println("Ceiling of key 9 = "+ tree.ceilingKey(9));
        System.out.println("floor of key 5 = "+ tree.floorKey(5));
        System.out.println("Key of First Element ="+ tree.firstKey());
        System.out.println("Key of Last Element ="+ tree.lastKey());
```




## Tests Run :