Q 1. Returns and removes the entry associated with the specified key from a TreeMap Eg:TreeMap: {One=1, Three=3, Two=2} Removed value = 2 Is the entry {Three=3} removed? True Updated TreeMap: {One=1}

->

import java.util.TreeMap;

public class TreeMapExample {

public static void main(String[] args) {

// Initialize the TreeMap

TreeMap<String, Integer> treeMap = new TreeMap<>();

treeMap.put("One", 1);

treeMap.put("Three", 3);

treeMap.put("Two", 2);

// Display the original TreeMap

System.out.println("TreeMap: " + treeMap);

// Remove entry with key "Two"

Integer removedValue = treeMap.remove("Two");

System.out.println("Removed value = " + removedValue);

// Check if the entry with key "Three" is removed

boolean isThreeRemoved = treeMap.remove("Three", 3);

System.out.println("Is the entry {Three=3} removed? " + isThreeRemoved);

// Display the updated TreeMap

System.out.println("Updated TreeMap: " + treeMap);

}

}

Output

TreeMap: {One=1, Three=3, Two=2} Removed value = 2 Is the entry {Three=3} removed? True Updated TreeMap: {One=1}

Q2 .Navigate over the elements of the treemap.(All methods)

->

import java.util.Iterator;

import java.util.Map;

import java.util.NavigableMap;

import java.util.TreeMap;

public class TreeMapNavigationExample {

public static void main(String[] args) {

// Initialize the TreeMap with some values

TreeMap<String, Integer> treeMap = new TreeMap<>();

treeMap.put("One", 1);

treeMap.put("Two", 2);

treeMap.put("Three", 3);

System.out.println("TreeMap: " + treeMap);

// 1. Using Entry Set (forEach loop)

System.out.println("\nIterating using entrySet():");

for (Map.Entry<String, Integer> entry : treeMap.entrySet()) {

System.out.println(entry.getKey() + " = " + entry.getValue());

}

// 2. Using Key Set

System.out.println("\nIterating over keys using keySet():");

for (String key : treeMap.keySet()) {

System.out.println("Key: " + key);

}

// 3. Using Values Collection

System.out.println("\nIterating over values using values():");

for (Integer value : treeMap.values()) {

System.out.println("Value: " + value);

}

// 4. Using Iterator on Entry Set

System.out.println("\nIterating using Iterator:");

Iterator<Map.Entry<String, Integer>> iterator = treeMap.entrySet().iterator();

while (iterator.hasNext()) {

Map.Entry<String, Integer> entry = iterator.next();

System.out.println(entry.getKey() + " = " + entry.getValue());

}

// 5. NavigableMap methods

System.out.println("\nUsing NavigableMap methods:");

System.out.println("First entry: " + treeMap.firstEntry());

System.out.println("Last entry: " + treeMap.lastEntry());

System.out.println("Higher entry than 'One': " + treeMap.higherEntry("One"));

System.out.println("Lower entry than 'Three': " + treeMap.lowerEntry("Three"));

// 6. DescendingMap for reverse order

System.out.println("\nIterating in reverse order using descendingMap():");

for (Map.Entry<String, Integer> entry : treeMap.descendingMap().entrySet()) {

System.out.println(entry.getKey() + " = " + entry.getValue());

}

// 7. Submaps

System.out.println("\nUsing SubMaps:");

// SubMap from "One" to "Three"

NavigableMap<String, Integer> subMap = treeMap.subMap("One", true, "Three", true);

System.out.println("SubMap from 'One' to 'Three': " + subMap);

// HeadMap up to "Three"

NavigableMap<String, Integer> headMap = treeMap.headMap("Three", true);

System.out.println("HeadMap up to 'Three': " + headMap);

// TailMap from "One"

NavigableMap<String, Integer> tailMap = treeMap.tailMap("One", true);

System.out.println("TailMap from 'One': " + tailMap);

}

}

Output

TreeMap: {One=1, Three=3, Two=2}

Iterating using entrySet():

One = 1

Three = 3

Two = 2

Iterating over keys using keySet():

Key: One

Key: Three

Key: Two

Iterating over values using values():

Value: 1

Value: 3

Value: 2

Iterating using Iterator:

One = 1

Three = 3

Two = 2

Using NavigableMap methods:

First entry: One=1

Last entry: Two=2

Higher entry than 'One': Three=3

Lower entry than 'Three': One=1

Iterating in reverse order using descendingMap():

Two = 2

Three = 3

One = 1

Using SubMaps:

SubMap from 'One' to 'Three': {One=1, Three=3}

HeadMap up to 'Three': {One=1, Three=3}

TailMap from 'One': {One=1, Three=3, Two=2}