Although the same operations are performed, the structure with the Linklist structure is faster than the structure with treeset structure.

1) Hashing by using linked lists:

Time:0.018

KWHashMap.java | HashTableChain.java | Main.java

```
1  public class Main {
2      public static void main(String[] args) { ///LinkedList
3          long startTime = System.currentTimeMillis();
4          HashTableChain<Integer,String> myh=new HashTableChain<Integer,String>();
5          myh.put(1,"Ferdi");
6          myh.put(2,"Ali");
7          myh.put(3,"Osman2");
8          myh.put(4,"Osman1");
9          myh.put(4,"Osman4");
10         myh.remove( key: 1);
11         System.out.println("Size:"+myh.size());
12         System.out.println("Isempty-->"+myh.isEmpty());
13         myh.Print();
14         long endTime = System.currentTimeMillis();
15         TimeCalculate(startTime,endTime);
16     }
17     public static void TimeCalculate(long startTime,long endTime){
```

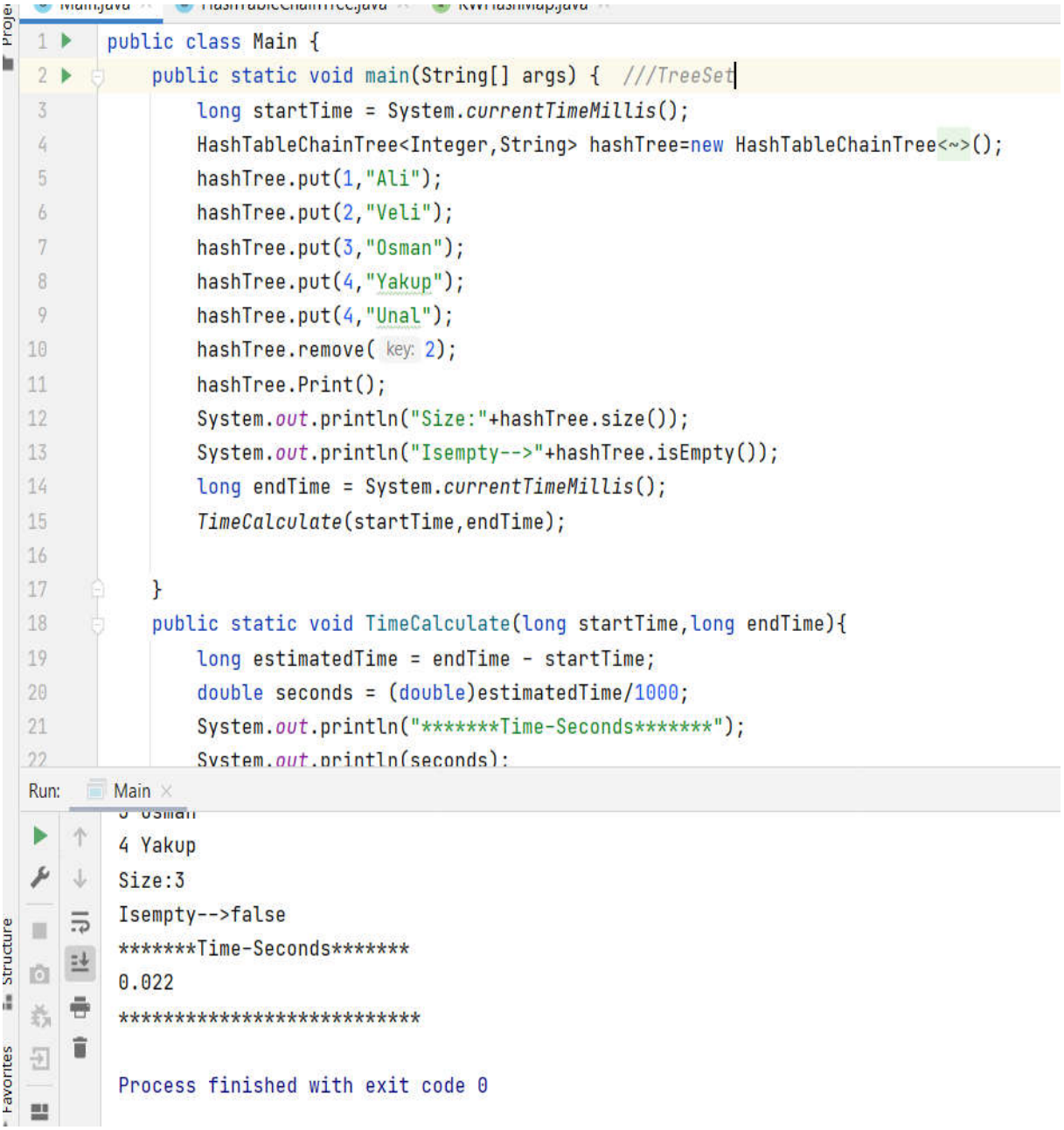
Run: Main

```
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrai
Size:4
Isempty-->false
Key:2 Value:Ali
Key:3 Value:Osman2
Key:4 Value:Osman1
Key:4 Value:Osman4
*******Time-Seconds*******
0.018
**************************

Process finished with exit code 0
```

2) Hashing by using TreeSet:

Time:0.022



```java
public class Main {
    public static void main(String[] args) {  ///TreeSet
        long startTime = System.currentTimeMillis();
        HashTableChainTree<Integer,String> hashTree=new HashTableChainTree<~>();
        hashTree.put(1,"Ali");
        hashTree.put(2,"Veli");
        hashTree.put(3,"Osman");
        hashTree.put(4,"Yakup");
        hashTree.put(4,"Unal");
        hashTree.remove( key: 2);
        hashTree.Print();
        System.out.println("Size:"+hashTree.size());
        System.out.println("Isempty-->"+hashTree.isEmpty());
        long endTime = System.currentTimeMillis();
        TimeCalculate(startTime,endTime);

    }
    public static void TimeCalculate(long startTime,long endTime){
        long estimatedTime = endTime - startTime;
        double seconds = (double)estimatedTime/1000;
        System.out.println("*******Time-Seconds*******");
        System.out.println(seconds);
```

```
Run:  Main
3 Osman
4 Yakup
Size:3
Isempty-->false
*******Time-Seconds*******
0.022
**************************

Process finished with exit code 0
```