GEBZE TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

CSE222-2021

HOMEWORK 05 REPORT

AHMET FURKAN KURBAN

1801042674

1 INTRODUCTION

1.1Problem Definition

Write a custom iterator class MapIterator to iterate through the keys in a HashMap data structure in Java. This class should have the following methods:

Implement KWHashMap interface in the book using the following hashing methods to organize hash table:
- Use the chaining technique for hashing by using linked lists (available in the text book) to chain items on the same table slot.

- Use the chaining technique for hashing by using TreeSet (instead of linked list) to chain items on the same table slot.

- Use the Coalesced hashing technique

2. Problem Solution Approach

part1:

Firstly, I wrote custom ıterator class MapIterator to iterate through the keys in a HashMap data structure in Java.
public class HashMap<K,V> extends java.util.HashMap<K, V> implements MapIterable<K,V> it is main class i extends hasmap from java and implement mapıterable interface , it has iterator function it is no paremeter iterator.
public class MAPIterator implements MapIterator<K,V> it is ıterator class it implements mapıterator interface , it has paremeter iterator and some functions such as prev next hasnext.
Set<K> set=keySet();
java.util.Iterator<K> it= set.iterator();
i reach the hashmap keys and iterator easily using keyset function and then set functions.

part2:

I use the chaining technique for hashing by using linked lists to chain items on the same table slot.

i use the chaining technique for hashing by using TreeSet to chain items on the same table slot.

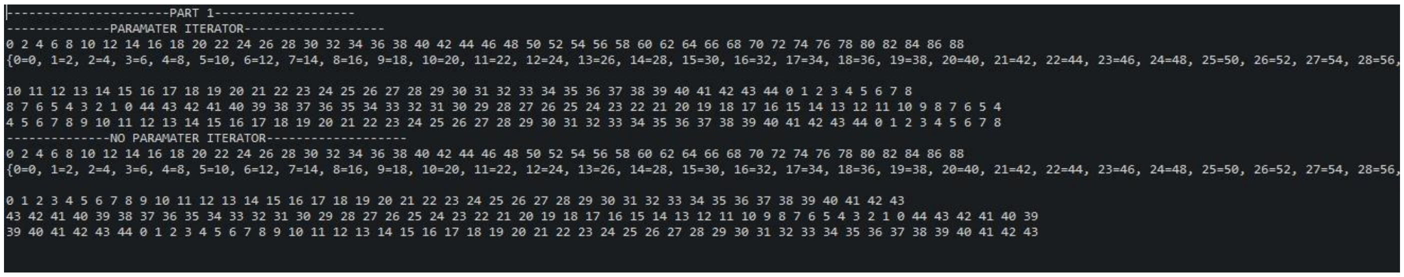
i implement comparable interface to load keys with treeset rules.

in Coalesced hashing technique i add element using key hashcode then if two elements has same index after hashing the second element will be next of first element then if one element added that has same index with these two element i add this element to the second element next , apperantly i use quadratic probing and the concept of Separate Chaining to link the colliding elements to each other through pointers. if one element removed it has no next element i change it with deleted entry then it is deleted after rehash function obviously. if removed one element that has next element the next element will be added this first element and next element index removed after all next element will be in deleted element index.

3 RESULT

3.1 Test Cases

HASHMAP ITERATOR:



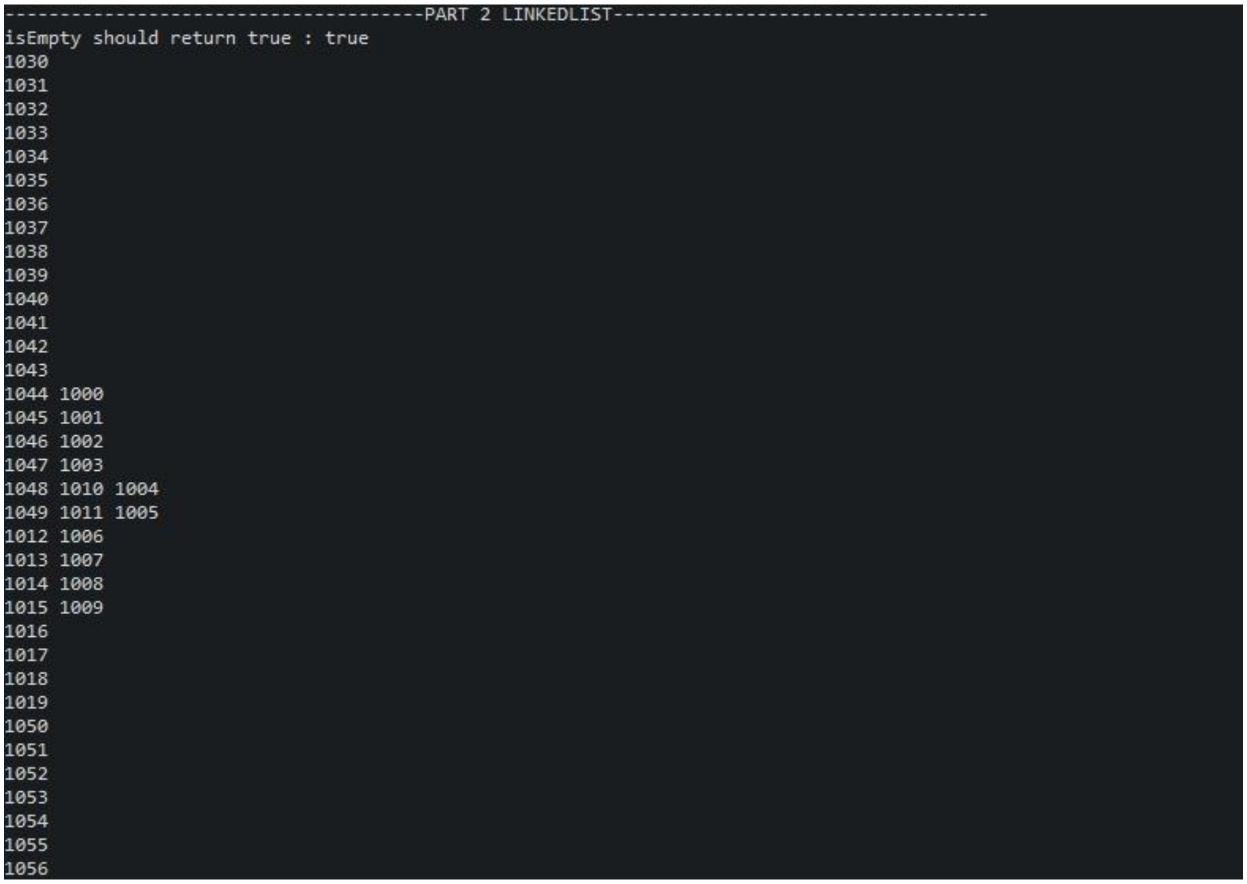
```
---------------------PART 1-------------------
--------------PARAMATER ITERATOR-------------------
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88
{0=0, 1=2, 2=4, 3=6, 4=8, 5=10, 6=12, 7=14, 8=16, 9=18, 10=20, 11=22, 12=24, 13=26, 14=28, 15=30, 16=32, 17=34, 18=36, 19=38, 20=40, 21=42, 22=44, 23=46, 24=48, 25=50, 26=52, 27=54, 28=56,

10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 0 1 2 3 4 5 6 7 8
8 7 6 5 4 3 2 1 0 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4
4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 0 1 2 3 4 5 6 7 8
--------------NO PARAMATER ITERATOR-------------------
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88
{0=0, 1=2, 2=4, 3=6, 4=8, 5=10, 6=12, 7=14, 8=16, 9=18, 10=20, 11=22, 12=24, 13=26, 14=28, 15=30, 16=32, 17=34, 18=36, 19=38, 20=40, 21=42, 22=44, 23=46, 24=48, 25=50, 26=52, 27=54, 28=56,

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 44 43 42 41 40 39
39 40 41 42 43 44 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
```

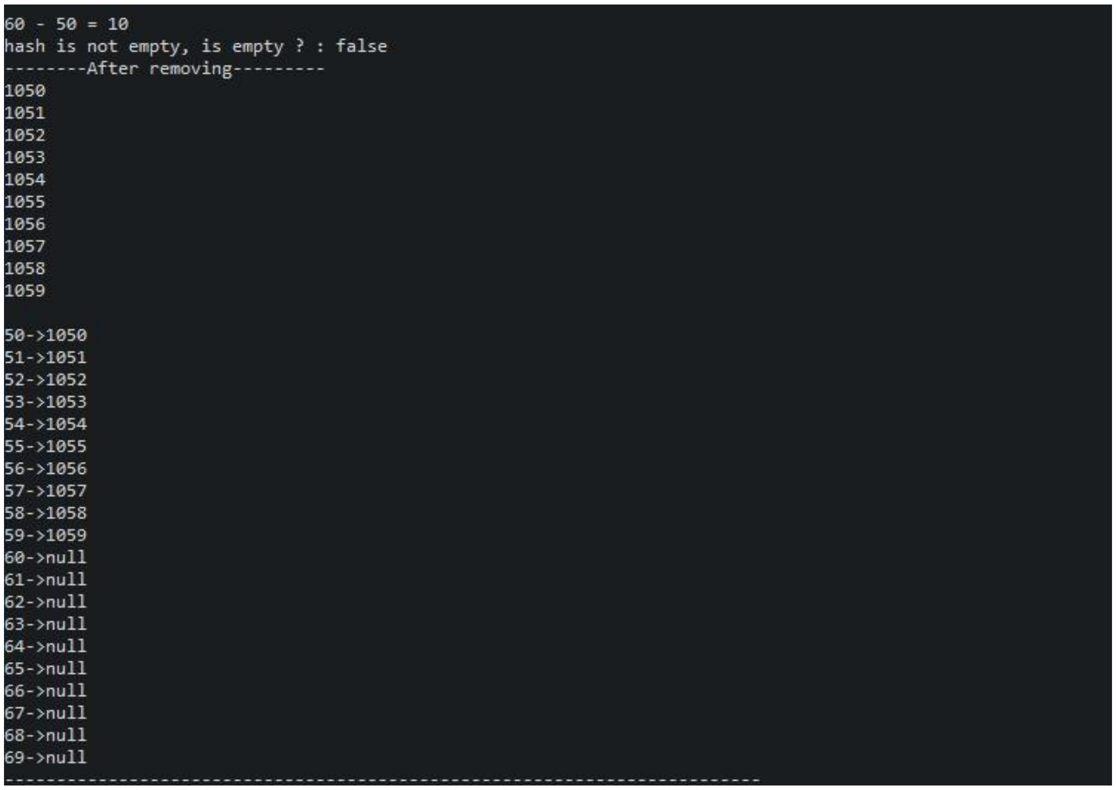
HASHMAP WITH LINKEDLIST:

Add method:



```
--------------------------------------PART 2 LINKEDLIST----------------------------------
isEmpty should return true : true
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044 1000
1045 1001
1046 1002
1047 1003
1048 1010 1004
1049 1011 1005
1012 1006
1013 1007
1014 1008
1015 1009
1016
1017
1018
1019
1050
1051
1052
1053
1054
1055
1056
```

Remove method: Get method:

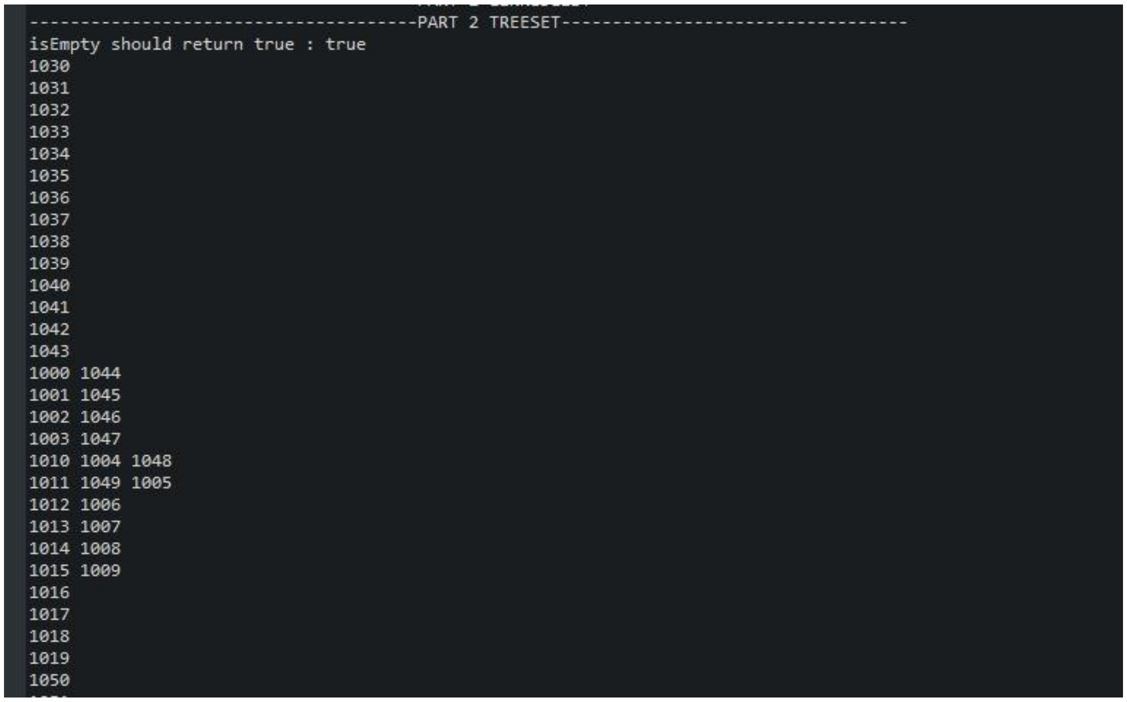


```
60 - 50 = 10
hash is not empty, is empty ? : false
--------After removing---------
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059

50->1050
51->1051
52->1052
53->1053
54->1054
55->1055
56->1056
57->1057
58->1058
59->1059
60->null
61->null
62->null
63->null
64->null
65->null
66->null
67->null
68->null
69->null
------------------------------------------------------------------------
```

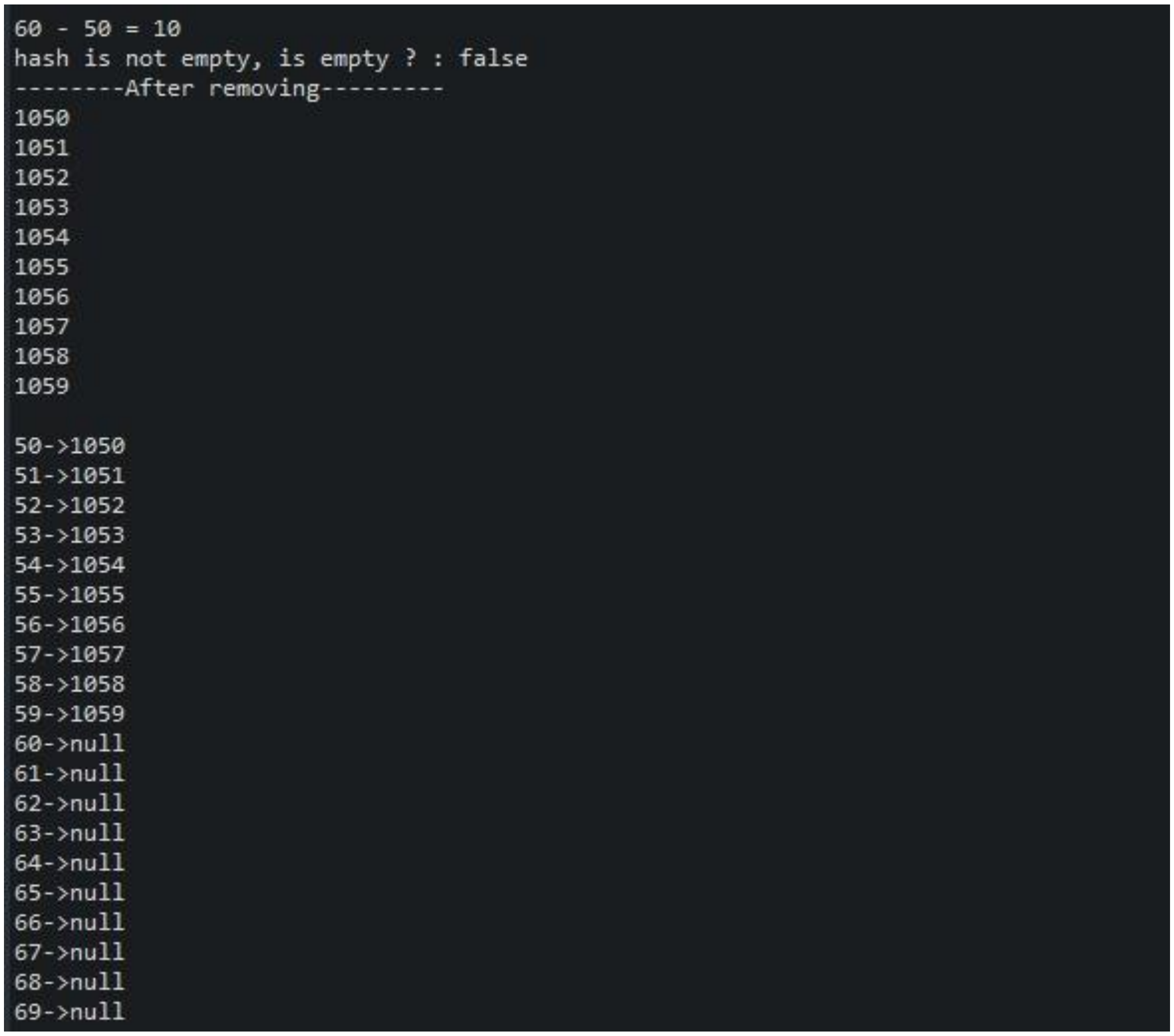
HASHMAP WITH TREESET:

Add method:



```
-------------------------------------PART 2 TREESET---------------------------------
isEmpty should return true : true
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1000 1044
1001 1045
1002 1046
1003 1047
1010 1004 1048
1011 1049 1005
1012 1006
1013 1007
1014 1008
1015 1009
1016
1017
1018
1019
1050
```

Remove method Get method:

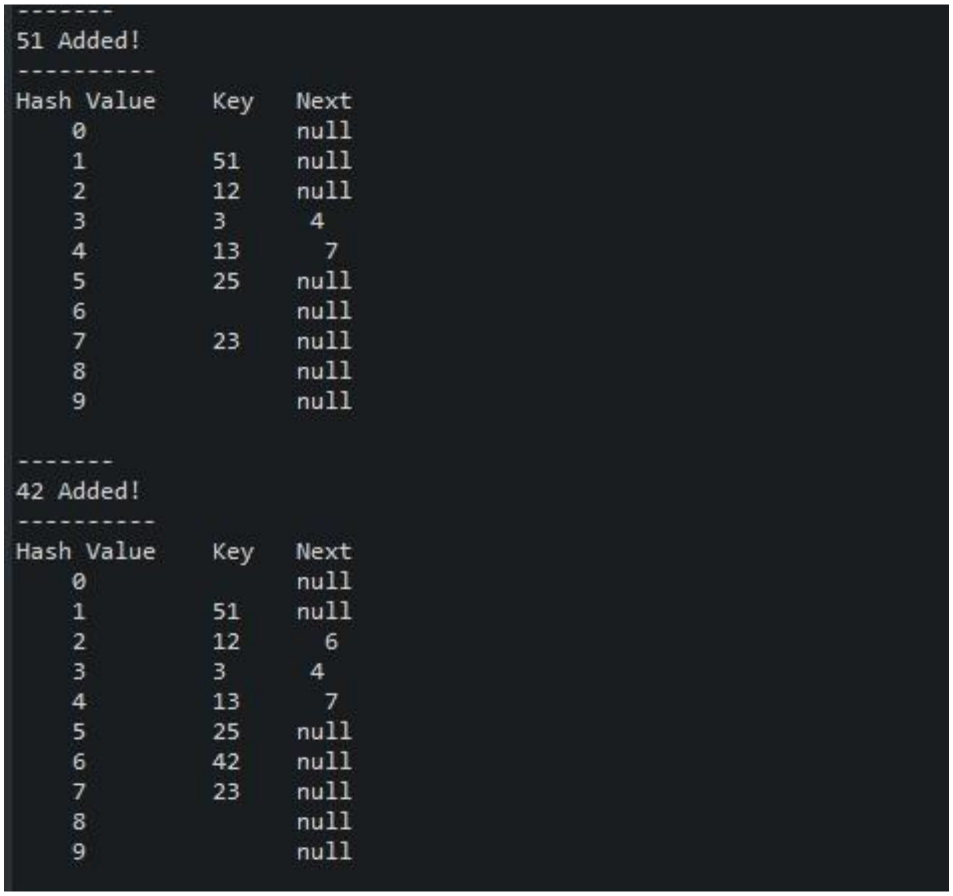


```
60 - 50 = 10
hash is not empty, is empty ? : false
--------After removing---------
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059

50->1050
51->1051
52->1052
53->1053
54->1054
55->1055
56->1056
57->1057
58->1058
59->1059
60->null
61->null
62->null
63->null
64->null
65->null
66->null
67->null
68->null
69->null
```

HASHMAP WITH OPEN ADRESSING :

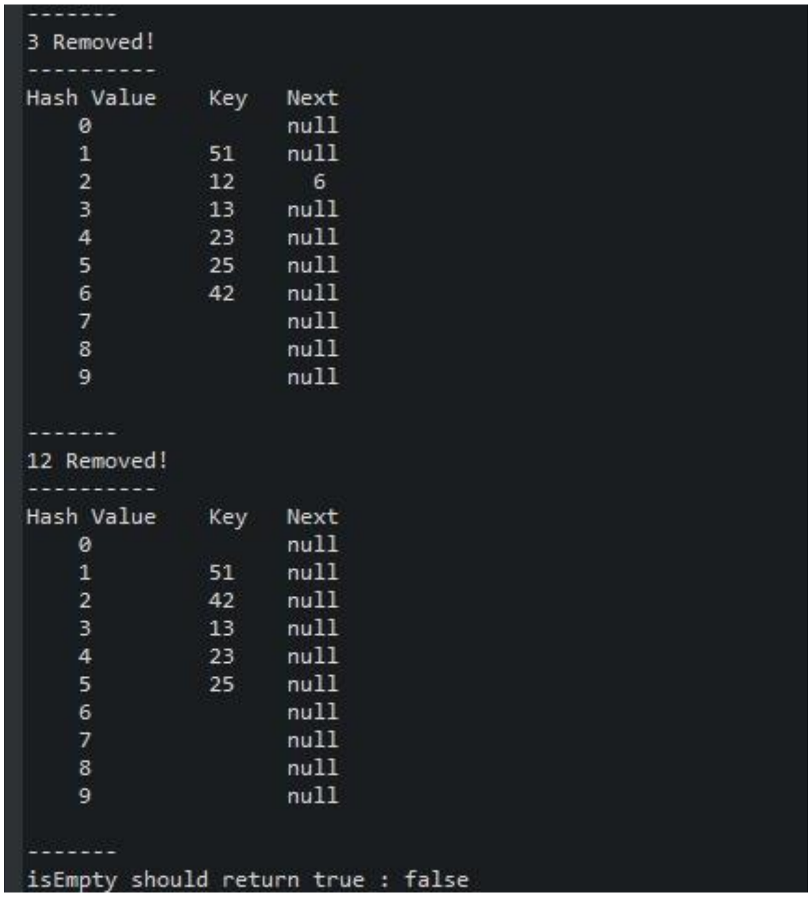
Add method:



```
-------
51 Added!
----------
Hash Value    Key   Next
    0               null
    1         51    null
    2         12    null
    3         3      4
    4         13      7
    5         25    null
    6               null
    7         23    null
    8               null
    9               null

-------
42 Added!
----------
Hash Value    Key   Next
    0               null
    1         51    null
    2         12      6
    3         3      4
    4         13      7
    5         25    null
    6         42    null
    7         23    null
    8               null
    9               null
```

Remove method:

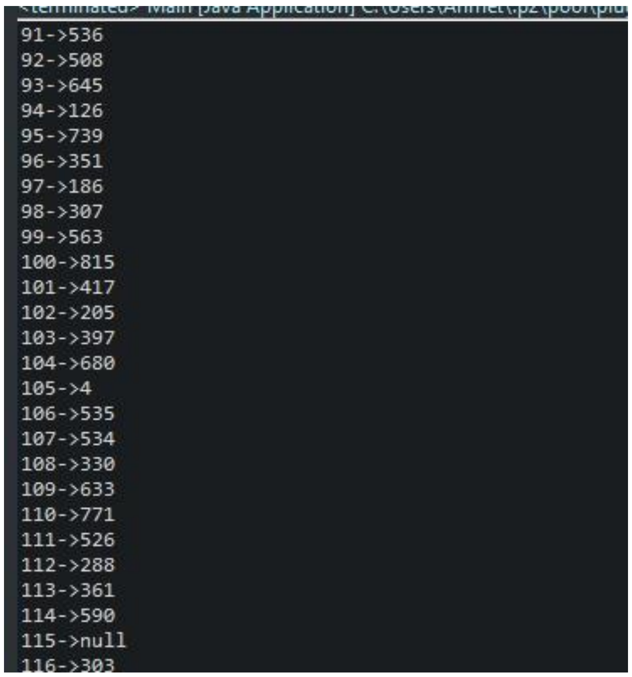


```
-------
3 Removed!
----------
Hash Value    Key   Next
    0               null
    1         51    null
    2         12      6
    3         13    null
    4         23    null
    5         25    null
    6         42    null
    7               null
    8               null
    9               null

-------
12 Removed!
----------
Hash Value    Key   Next
    0               null
    1         51    null
    2         42    null
    3         13    null
    4         23    null
    5         25    null
    6               null
    7               null
    8               null
    9               null

-------
isEmpty should return true : false
```

Get method:



```
<terminated> Main [Java Application] C:\Users\Ahmet\.p2\pool\plu
91->536
92->508
93->645
94->126
95->739
96->351
97->186
98->307
99->563
100->815
101->417
102->205
103->397
104->680
105->4
106->535
107->534
108->330
109->633
110->771
111->526
112->288
113->361
114->590
115->null
116->303
```