GIT Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework #5 Report

Erdi Bayır
171044063

PROBLEM SOLUTION APPROACH

For Part 1:

In this part, we have to create a custom iterator class MapIterator to iterate through the keys in HashMap data structure in Java.
I chose to write an inner iterator class and I chose the Coalesced class, one of the custom HashMap classes I wrote in the 2nd part, because it is an open addressing type hashmap, similar to the hash map class in Java.

For Part 2:

In the second part, we need to write 3 different hash maps. One of them is a linked list hash map and since the implementation of this class is in the book, most of the class I wrote is similar to the book.
For the TreeSet Hash map, again using the similar class, I kept the data with the Tree Set instead of the Linkedlist object in which we keep the data and made changes in the necessary methods.
For the coalesced hash map, I used open addressing. Although most methods are similar to the other two classes, now I placed the data directly in the table we have, not in a list or tree. At the same time, I used quadratic probing when adding data into this map.

TEST CASES And Running Command Results

For Part 1:

1) Create zero parameter map iterator
2) Create map iterator with key
3) Testing next() method
4) Testing hasNext() method
5) Testing prev() method

For Part 2:

Hash Map with Linked List chain;

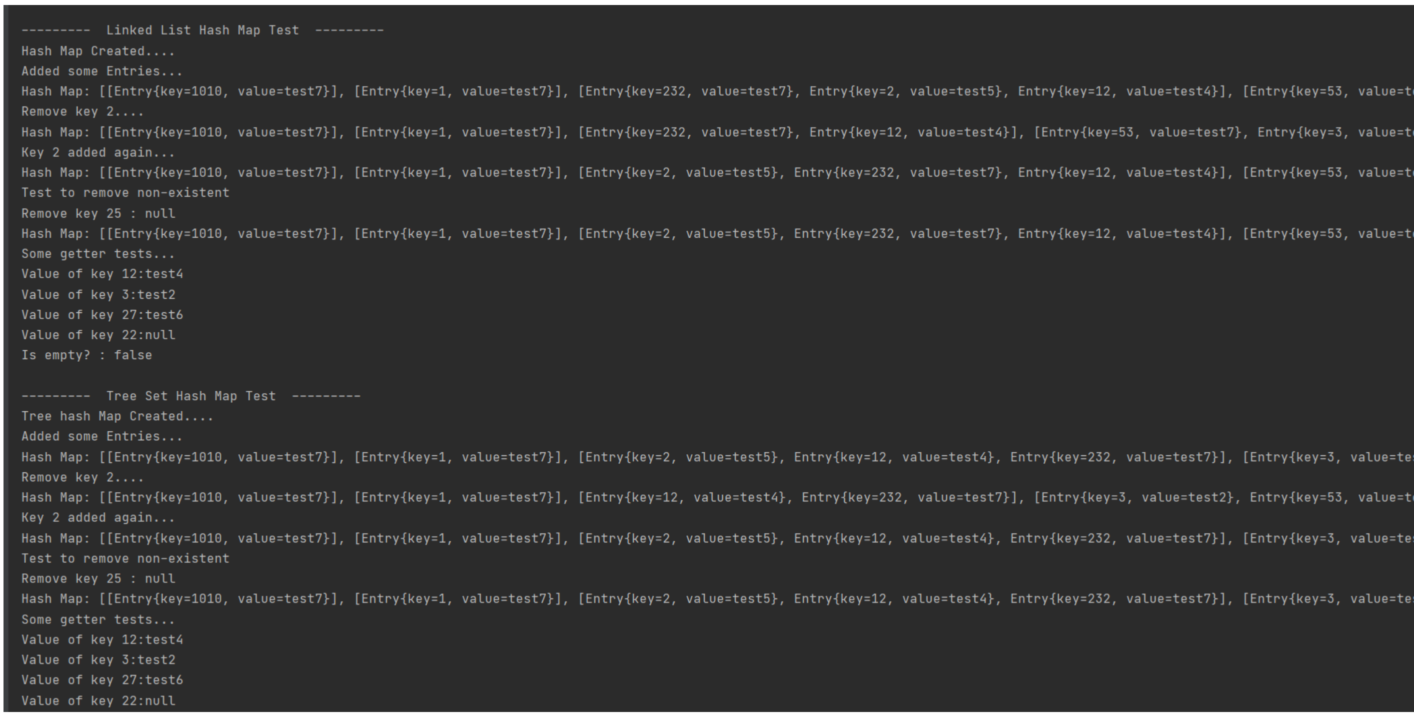
1) Create Hash Map
2) Put element to hash map
3) Remove element from hash map
4) toString method
5) Get method
6) Rehash method

Hash Map with Tree Set chain;

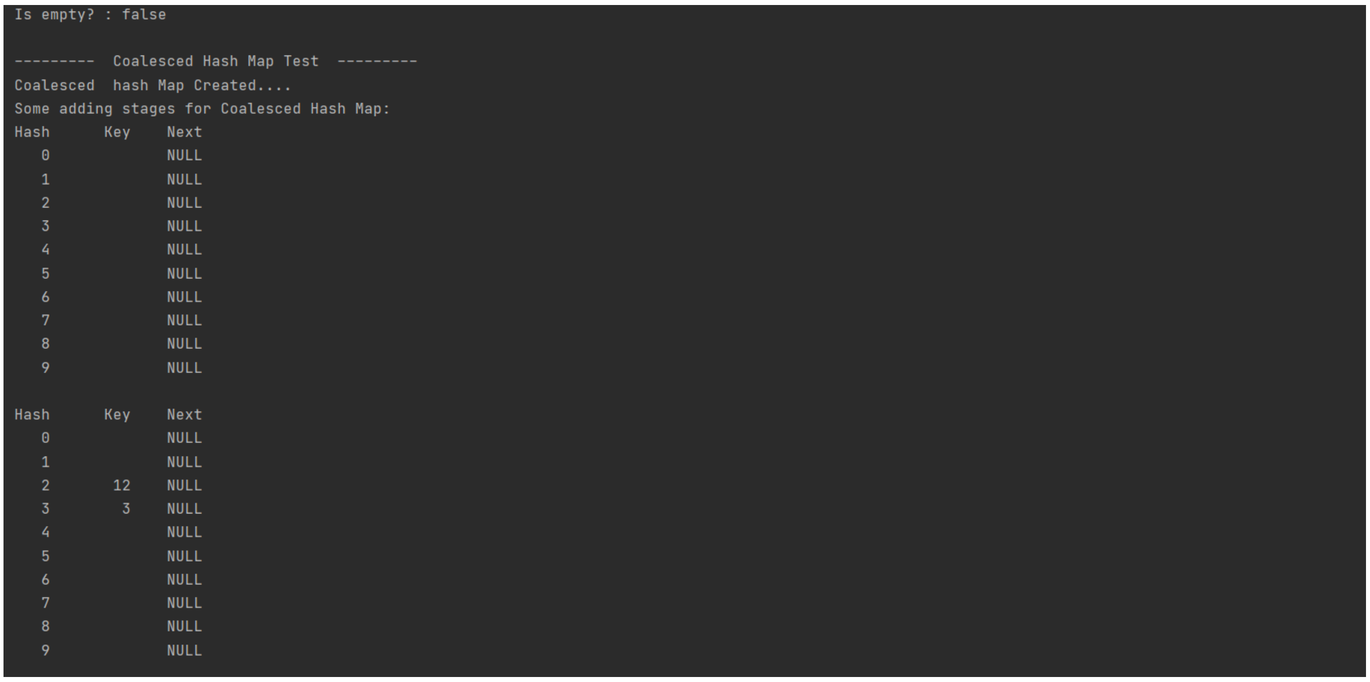
1) Create Hash Map
2) Put element to hash map
3) Remove element from hash map
4) toString method
5) Get method
6) Rehash method

Coalesced Hash Map

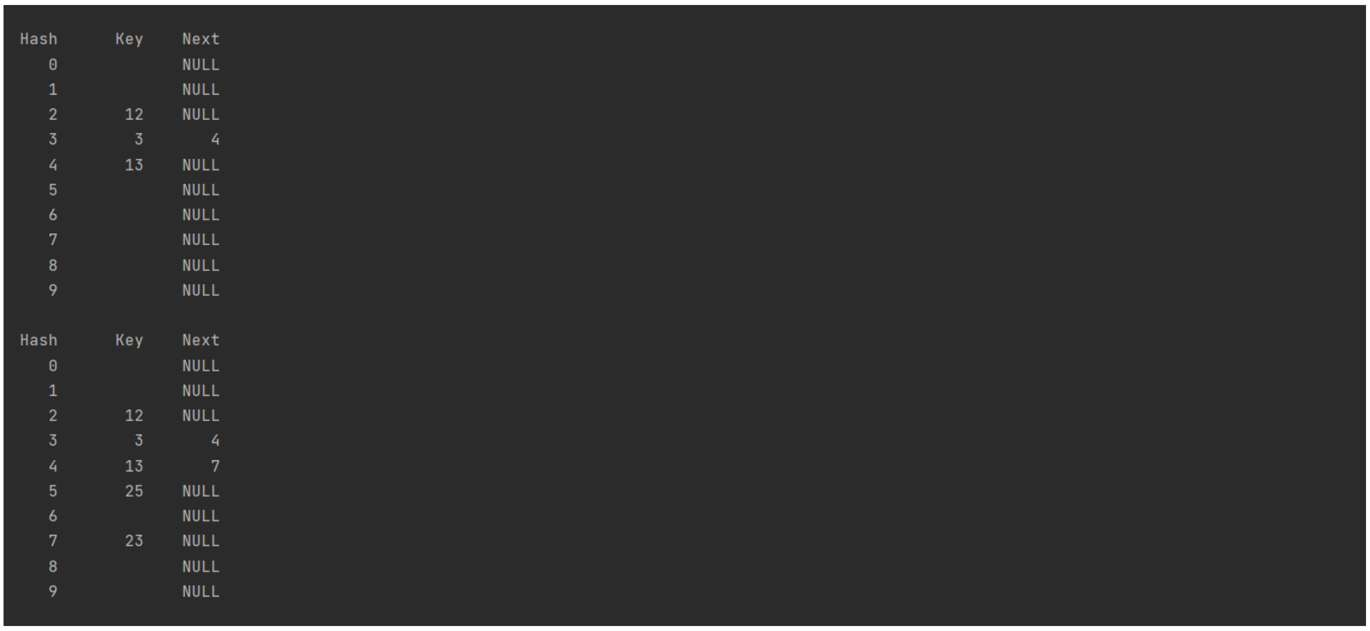
1) Create Hash Map
2) Put element to hash map
3) Remove element from hash map
4) toString method
5) Get method
6) Rehash method



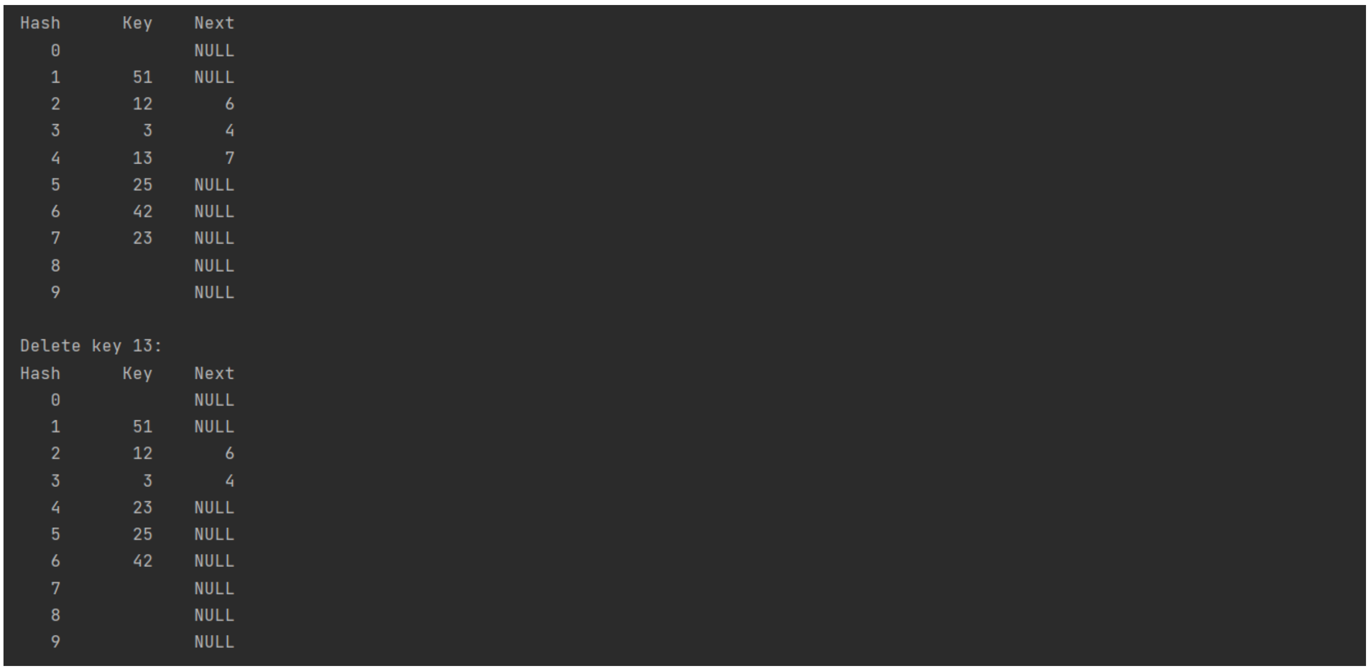
```
---------  Linked List Hash Map Test  ---------
Hash Map Created....
Added some Entries...
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=232, value=test7}, Entry{key=2, value=test5}, Entry{key=12, value=test4}], [Entry{key=53, value=te
Remove key 2....
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=232, value=test7}, Entry{key=12, value=test4}], [Entry{key=53, value=test7}, Entry{key=3, value=te
Key 2 added again...
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=2, value=test5}, Entry{key=232, value=test7}, Entry{key=12, value=test4}], [Entry{key=53, value=te
Test to remove non-existent
Remove key 25 : null
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=2, value=test5}, Entry{key=232, value=test7}, Entry{key=12, value=test4}], [Entry{key=53, value=te
Some getter tests...
Value of key 12:test4
Value of key 3:test2
Value of key 27:test6
Value of key 22:null
Is empty? : false

---------  Tree Set Hash Map Test  ---------
Tree hash Map Created....
Added some Entries...
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=2, value=test5}, Entry{key=12, value=test4}, Entry{key=232, value=test7}], [Entry{key=3, value=tes
Remove key 2....
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=12, value=test4}, Entry{key=232, value=test7}], [Entry{key=3, value=test2}, Entry{key=53, value=te
Key 2 added again...
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=2, value=test5}, Entry{key=12, value=test4}, Entry{key=232, value=test7}], [Entry{key=3, value=tes
Test to remove non-existent
Remove key 25 : null
Hash Map: [[Entry{key=1010, value=test7}], [Entry{key=1, value=test7}], [Entry{key=2, value=test5}, Entry{key=12, value=test4}, Entry{key=232, value=test7}], [Entry{key=3, value=tes
Some getter tests...
Value of key 12:test4
Value of key 3:test2
Value of key 27:test6
Value of key 22:null
```



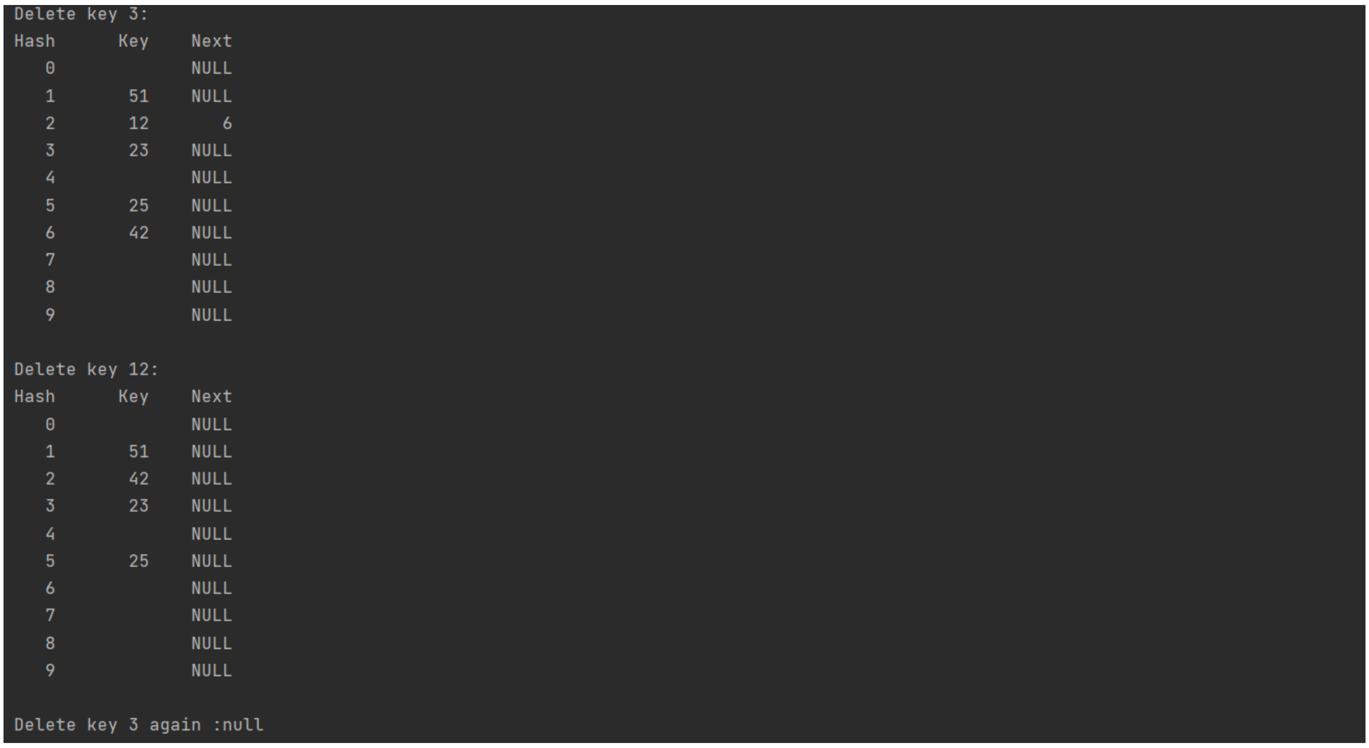
```
Is empty? : false

---------  Coalesced Hash Map Test  ---------
Coalesced  hash Map Created....
Some adding stages for Coalesced Hash Map:
Hash      Key    Next
   0             NULL
   1             NULL
   2             NULL
   3             NULL
   4             NULL
   5             NULL
   6             NULL
   7             NULL
   8             NULL
   9             NULL

Hash      Key    Next
   0             NULL
   1             NULL
   2       12    NULL
   3        3    NULL
   4             NULL
   5             NULL
   6             NULL
   7             NULL
   8             NULL
   9             NULL
```



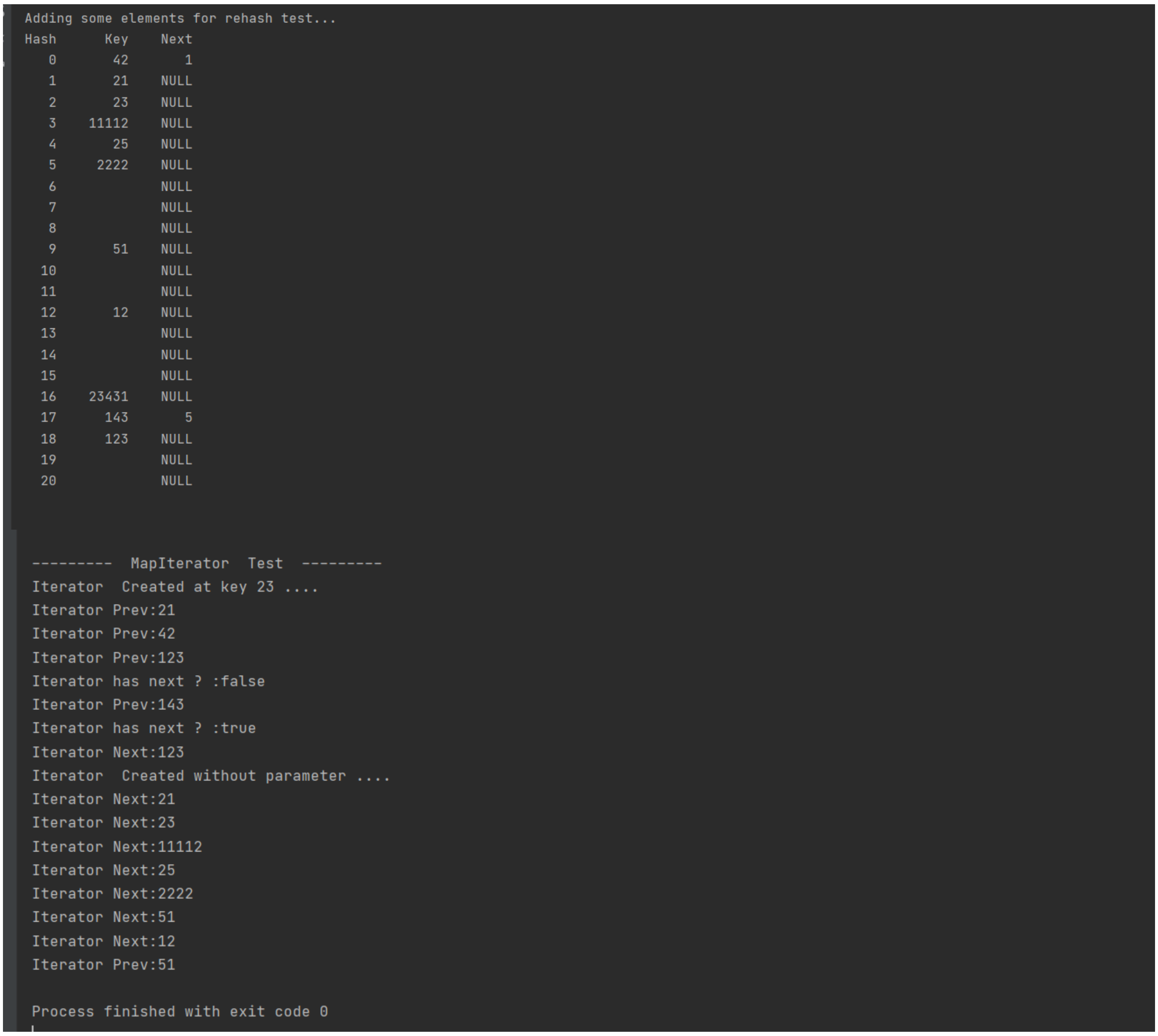
```
Hash      Key    Next
   0             NULL
   1             NULL
   2       12    NULL
   3        3       4
   4       13    NULL
   5             NULL
   6             NULL
   7             NULL
   8             NULL
   9             NULL

Hash      Key    Next
   0             NULL
   1             NULL
   2       12    NULL
   3        3       4
   4       13       7
   5       25    NULL
   6             NULL
   7       23    NULL
   8             NULL
   9             NULL
```



```
Hash      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

Delete key 13:
Hash      Key    Next
   0             NULL
   1       51    NULL
   2       12       6
   3        3       4
   4       23    NULL
   5       25    NULL
   6       42    NULL
   7             NULL
   8             NULL
   9             NULL
```



```
Delete key 3:
Hash      Key    Next
   0             NULL
   1       51    NULL
   2       12       6
   3       23    NULL
   4             NULL
   5       25    NULL
   6       42    NULL
   7             NULL
   8             NULL
   9             NULL

Delete key 12:
Hash      Key    Next
   0             NULL
   1       51    NULL
   2       42    NULL
   3       23    NULL
   4             NULL
   5       25    NULL
   6             NULL
   7             NULL
   8             NULL
   9             NULL

Delete key 3 again :null
```



```
Adding some elements for rehash test...
Hash      Key    Next
   0       42       1
   1       21    NULL
   2       23    NULL
   3    11112    NULL
   4       25    NULL
   5     2222    NULL
   6             NULL
   7             NULL
   8             NULL
   9       51    NULL
  10             NULL
  11             NULL
  12       12    NULL
  13             NULL
  14             NULL
  15             NULL
  16    23431    NULL
  17      143       5
  18      123    NULL
  19             NULL
  20             NULL


 ---------  MapIterator  Test  ---------
Iterator  Created at key 23 ....
Iterator Prev:21
Iterator Prev:42
Iterator Prev:123
Iterator has next ? :false
Iterator Prev:143
Iterator has next ? :true
Iterator Next:123
Iterator  Created without parameter ....
Iterator Next:21
Iterator Next:23
Iterator Next:11112
Iterator Next:25
Iterator Next:2222
Iterator Next:51
Iterator Next:12
Iterator Prev:51

Process finished with exit code 0
```

Performance ANALYSIS

Total Execution Time of Hash Map With Linked List chain = About 21 millis.
Total Execution Time of Hash Map With Tree Set chain = About 3 millis.
Total Execution Time of Coalesced Hash Map = About 36 millis.

Tree Set was the fastest working Hashmap. Although the Put and Remove methods are very similar to the Linked List hash map, it works much faster. The reason why the slowest employee is Coalesced is that it can only retain one data per index, so it prone to rehash.At the same time It is losing too much time to find an empty index in the table.
In addition, in the remove method, it creates an extra waste of time since it transfers the data in the next index to the current index to be deleted.