GTU Department of Computer Engineering

CSE 222/505 - Spring 2021

Homework 5 Report

KAHRAMAN ARDA KILIÇ

1901042648

1. PROBLEM SOLUTION APPROACH

There are 4 different implementation of the hashing technique of the HashMap. First one is open-addressing. I've used this technique in the class for the Part 1. Second is chaining technique with LinkedList. This implementation is taken from the book. Third one is same as the second one but without LinkedList. I've used TreeSet to implement it. TreeSet and LinkedList is taken from the java.util. The fourth one is a bit different than others. It is something like the combination of others.

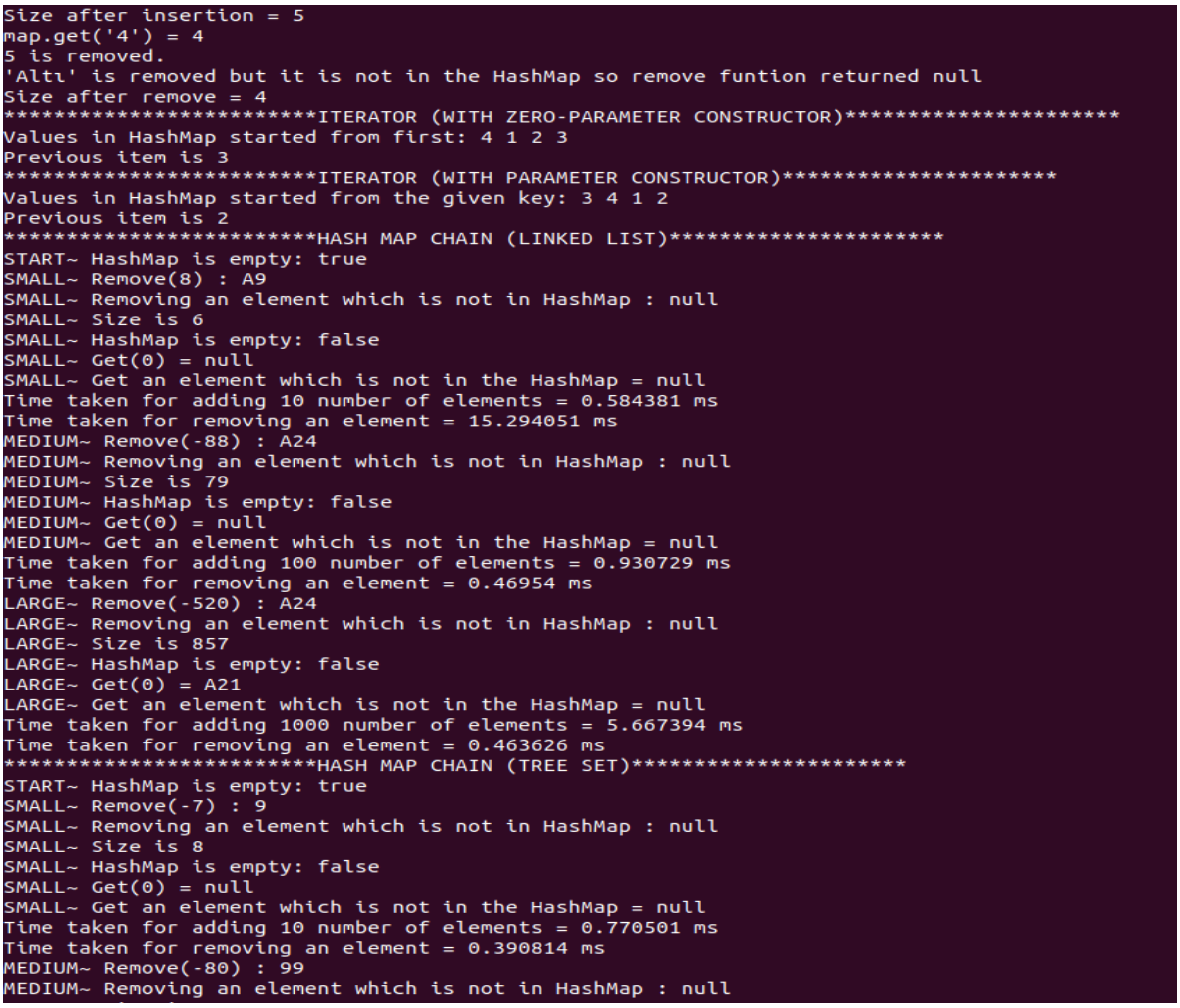
All implementations except Coalesced Hashing are using rehash() function to increase the capacity. The implementation which is using Coalesced Hashing Method does not increase its size. The user can use the constructor to give a default capacity.

2. TEST CASES

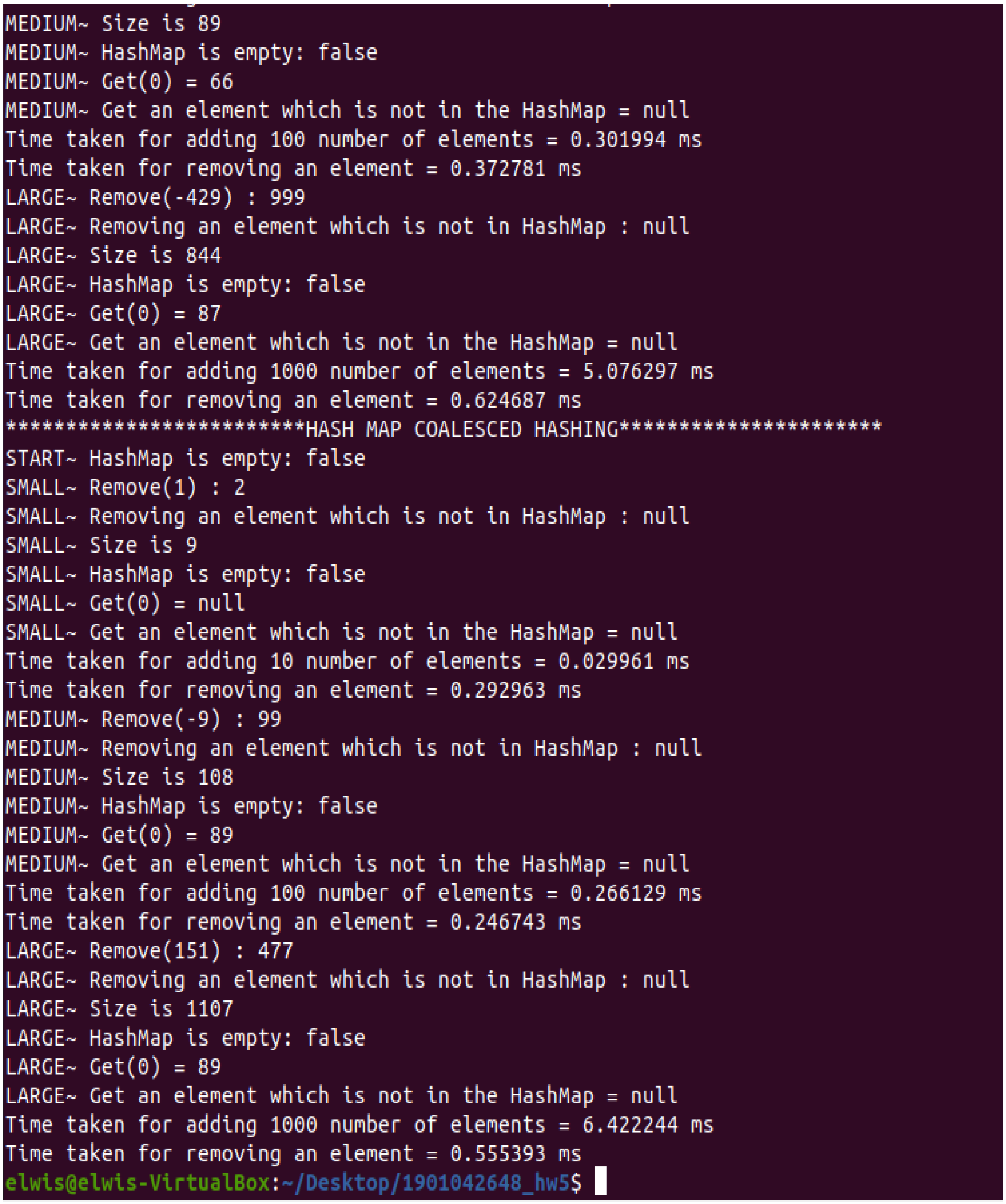
As it described in the Homework paper, I've used 3 different sizes to test the data structures. They are SMALL, MEDIUM and LARGE. The user of this class can change these values in the Driver Class. For default, they are 10, 100, 1000 respectively.

For each data structures, there are 3 different loops to put the values into them. After each for loop, I remove an entry by using the removedKey value which is obtained as a last added key. To remove an element which is not in the structure, I put the {Small,Medium,Large} * 2 to find a value out of the range. Also I used the same technique to get a value from the HashMap.

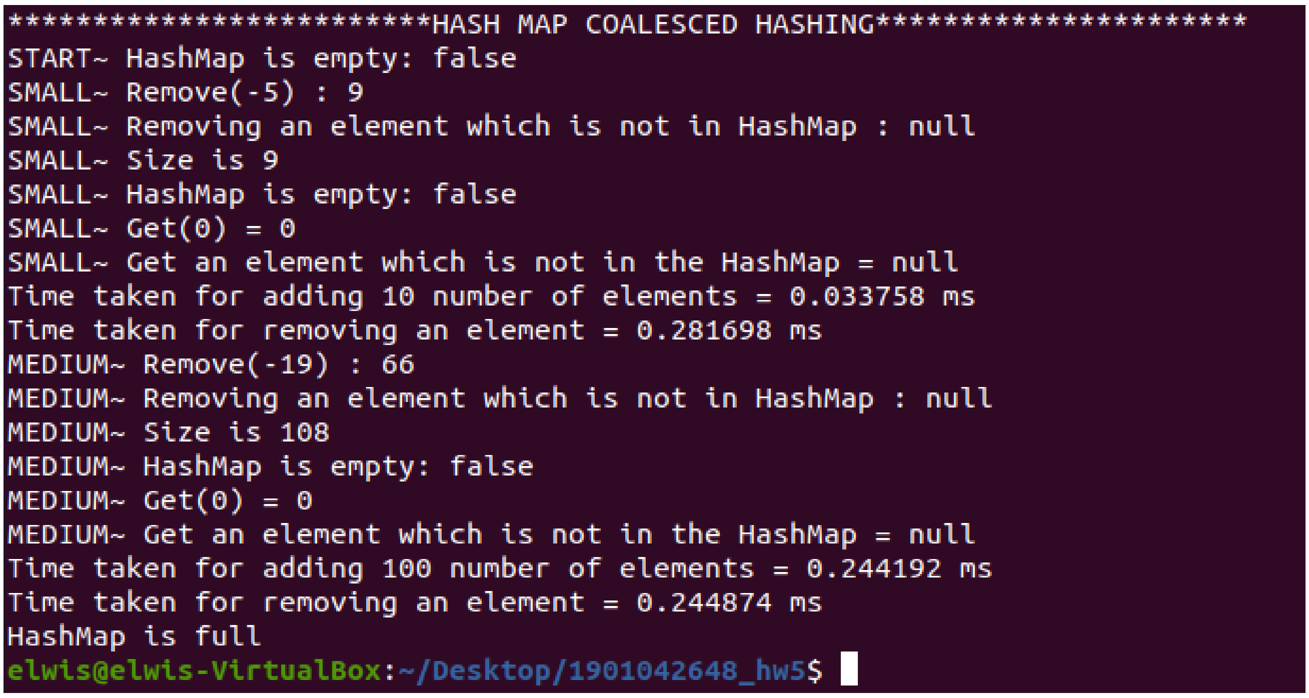
3. RUNNING AND RESULTS



```
Size after insertion = 5
map.get('4') = 4
5 is removed.
'Altı' is removed but it is not in the HashMap so remove funtion returned null
Size after remove = 4
*************************ITERATOR (WITH ZERO-PARAMETER CONSTRUCTOR)**********************
Values in HashMap started from first: 4 1 2 3
Previous item is 3
*************************ITERATOR (WITH PARAMETER CONSTRUCTOR)**********************
Values in HashMap started from the given key: 3 4 1 2
Previous item is 2
*************************HASH MAP CHAIN (LINKED LIST)**********************
START~ HashMap is empty: true
SMALL~ Remove(8) : A9
SMALL~ Removing an element which is not in HashMap : null
SMALL~ Size is 6
SMALL~ HashMap is empty: false
SMALL~ Get(0) = null
SMALL~ Get an element which is not in the HashMap = null
Time taken for adding 10 number of elements = 0.584381 ms
Time taken for removing an element = 15.294051 ms
MEDIUM~ Remove(-88) : A24
MEDIUM~ Removing an element which is not in HashMap : null
MEDIUM~ Size is 79
MEDIUM~ HashMap is empty: false
MEDIUM~ Get(0) = null
MEDIUM~ Get an element which is not in the HashMap = null
Time taken for adding 100 number of elements = 0.930729 ms
Time taken for removing an element = 0.46954 ms
LARGE~ Remove(-520) : A24
LARGE~ Removing an element which is not in HashMap : null
LARGE~ Size is 857
LARGE~ HashMap is empty: false
LARGE~ Get(0) = A21
LARGE~ Get an element which is not in the HashMap = null
Time taken for adding 1000 number of elements = 5.667394 ms
Time taken for removing an element = 0.463626 ms
*************************HASH MAP CHAIN (TREE SET)**********************
START~ HashMap is empty: true
SMALL~ Remove(-7) : 9
SMALL~ Removing an element which is not in HashMap : null
SMALL~ Size is 8
SMALL~ HashMap is empty: false
SMALL~ Get(0) = null
SMALL~ Get an element which is not in the HashMap = null
Time taken for adding 10 number of elements = 0.770501 ms
Time taken for removing an element = 0.390814 ms
MEDIUM~ Remove(-80) : 99
MEDIUM~ Removing an element which is not in HashMap : null
```



```
MEDIUM~ Size is 89
MEDIUM~ HashMap is empty: false
MEDIUM~ Get(0) = 66
MEDIUM~ Get an element which is not in the HashMap = null
Time taken for adding 100 number of elements = 0.301994 ms
Time taken for removing an element = 0.372781 ms
LARGE~ Remove(-429) : 999
LARGE~ Removing an element which is not in HashMap : null
LARGE~ Size is 844
LARGE~ HashMap is empty: false
LARGE~ Get(0) = 87
LARGE~ Get an element which is not in the HashMap = null
Time taken for adding 1000 number of elements = 5.076297 ms
Time taken for removing an element = 0.624687 ms
*************************HASH MAP COALESCED HASHING**********************
START~ HashMap is empty: false
SMALL~ Remove(1) : 2
SMALL~ Removing an element which is not in HashMap : null
SMALL~ Size is 9
SMALL~ HashMap is empty: false
SMALL~ Get(0) = null
SMALL~ Get an element which is not in the HashMap = null
Time taken for adding 10 number of elements = 0.029961 ms
Time taken for removing an element = 0.292963 ms
MEDIUM~ Remove(-9) : 99
MEDIUM~ Removing an element which is not in HashMap : null
MEDIUM~ Size is 108
MEDIUM~ HashMap is empty: false
MEDIUM~ Get(0) = 89
MEDIUM~ Get an element which is not in the HashMap = null
Time taken for adding 100 number of elements = 0.266129 ms
Time taken for removing an element = 0.246743 ms
LARGE~ Remove(151) : 477
LARGE~ Removing an element which is not in HashMap : null
LARGE~ Size is 1107
LARGE~ HashMap is empty: false
LARGE~ Get(0) = 89
LARGE~ Get an element which is not in the HashMap = null
Time taken for adding 1000 number of elements = 6.422244 ms
Time taken for removing an element = 0.555393 ms
elwis@elwis-VirtualBox:~/Desktop/1901042648_hw5$
```



```
*************************HASH MAP COALESCED HASHING**********************
START~ HashMap is empty: false
SMALL~ Remove(-5) : 9
SMALL~ Removing an element which is not in HashMap : null
SMALL~ Size is 9
SMALL~ HashMap is empty: false
SMALL~ Get(0) = 0
SMALL~ Get an element which is not in the HashMap = null
Time taken for adding 10 number of elements = 0.033758 ms
Time taken for removing an element = 0.281698 ms
MEDIUM~ Remove(-19) : 66
MEDIUM~ Removing an element which is not in HashMap : null
MEDIUM~ Size is 108
MEDIUM~ HashMap is empty: false
MEDIUM~ Get(0) = 0
MEDIUM~ Get an element which is not in the HashMap = null
Time taken for adding 100 number of elements = 0.244192 ms
Time taken for removing an element = 0.244874 ms
HashMap is full
elwis@elwis-VirtualBox:~/Desktop/1901042648_hw5$
```

NOTE: When the capacity of the HashMap (Coalesced Method) is full then put method throws NoSuchFieldException and terminates the program as shown above.