GIT Department of Computer Engineering

CSE 222/505 - Spring 2021

Homework 5 Report

Muhammed Bedir ULUCAY

1901042697

1. PROBLEM SOLUTION APPROACH

Part 1:

We create an new iterator for java.util.HashMap.There are two type iterator.

- First one is not parameter it starting as normal 0 to size.
- Second one is key parameter constructor this iterator looking the key is in the hash map if it is in it start from there and end it before reach again the parameter key.To make it we use shitfting array and after that we can assume it as normal constructor.

Part 2:

Our problem is make an different hash table and with these kind of hash table make some defined method for each implementation.If we look at the method is basicly existing 4 method.

These are ;

1- Put a Value to hash map

2- Remove a Value from hash map

3- Get a Value from hash map

4- Rehash the map

Of course if the subject is hash map we need to use good hash code for keys to reach element as fast as possible.

Our first implementation is hash chain Linked List. In this implementation.When we want to add a value take hash code and add it linked list which index of hash table as last element. Removing also same process find hash code go there and remove it from linked list. Get is also same find hash code and get value from linked list but dont remove.

Our second implementation is hash chain tree set as we said in linked list implementation for add, remove and get method, first we need to find hash code of key and go the index location.

And process wanted action.

Our third implementation is some kind of different from others we are reaching element by using next1,next2,prev.

Nex1: Is working same values which has same hash code like 3->13->23 these are same hash for the rule

Next2 :In this data structure if a key kept a place dont match with index.When we want to insert normal index we are take a valid space in hash table and next 2 pointing the index and same hash code key location.

Previous: Keep who is pointing me

For example:

We add 3 and 13. 13 index position is 4 when we want to 24 or any number hash code is 4 is place is taken value not matching hash code in that case we are take a free space in hash map and next2 is pointing that adress.

Note :We show the track for next2 and prev in result

1. TEST CASES

Hash Map Iterator:

- **No Parameter Constructor moving forward and backward**
- **Parameter Constructor moving forward and backward**
- **Parameter Constructor but not include as key**
- **Moving forward and backward using iterator:**

Hashtable LinkedList Chain:

- **Put Values**
- **Get Values:**
- **Remove Values:**
- **Rehash:**
- **Size**

Hashtable TreeSet Chain:

- **Put Values**
- **Get Values:**
- **Remove Values:**
- **Rehash:**
- **Size**

Hashtable Coalesced Hash Map:

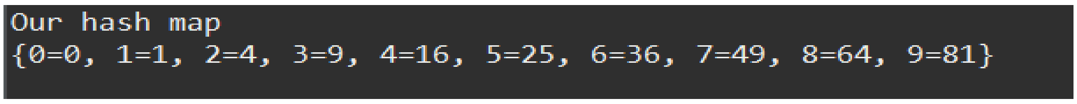
- **Put Values**
- **Get Values:**
- **Remove Values:**
- **Size**

Hashtable Coalesced Hash Map Using next2 and prev:

- **Put Values**
- **Get Values:**
- **Remove Values:**
- **Size**
- **Next2 Track**
- **Prev Track**

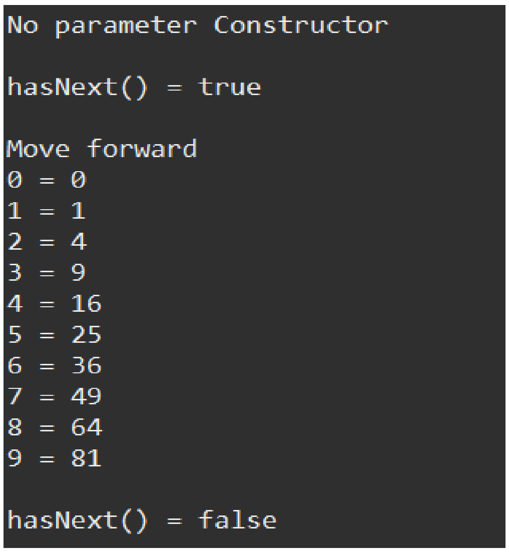
2. RUNNING AND RESULTS

> Hash Map Iterator :

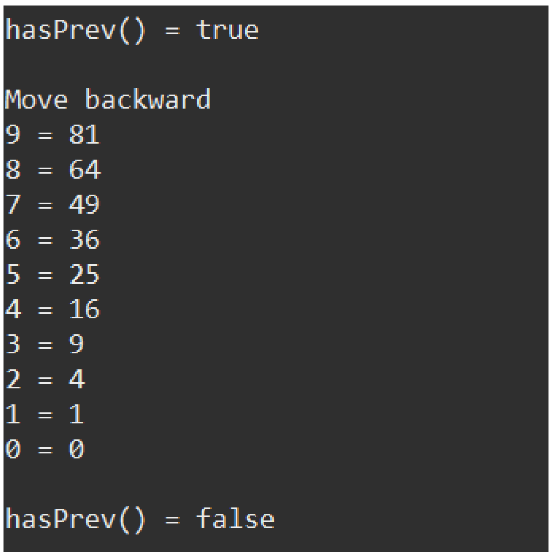


```
Our hash map
{0=0, 1=1, 2=4, 3=9, 4=16, 5=25, 6=36, 7=49, 8=64, 9=81}
```

No Parameter Constructor moving forward and backward:

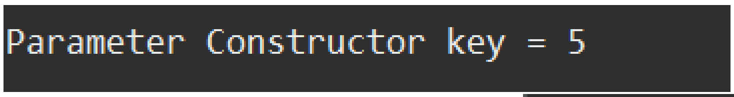


```
No parameter Constructor

hasNext() = true

Move forward
0 = 0
1 = 1
2 = 4
3 = 9
4 = 16
5 = 25
6 = 36
7 = 49
8 = 64
9 = 81

hasNext() = false
```

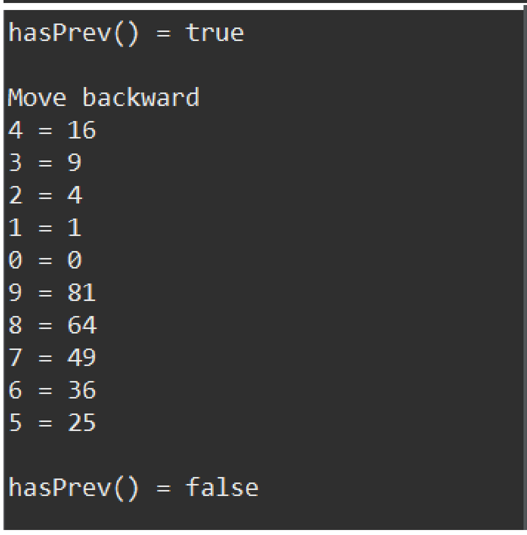


```
hasPrev() = true

Move backward
9 = 81
8 = 64
7 = 49
6 = 36
5 = 25
4 = 16
3 = 9
2 = 4
1 = 1
0 = 0

hasPrev() = false
```

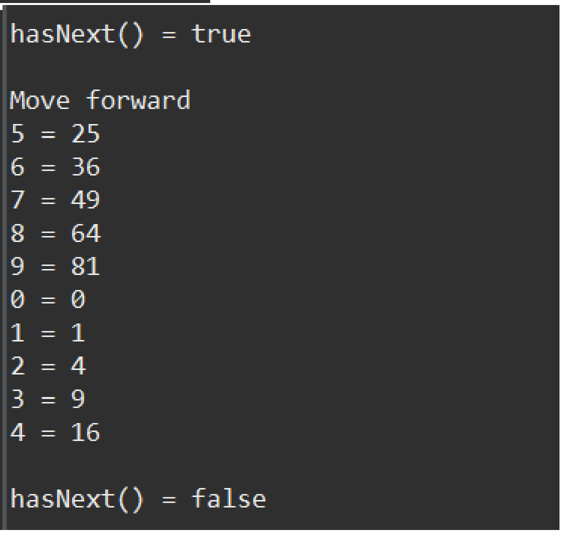
Parameter Constructor moving forward and backward:



```
Parameter Constructor key = 5
```



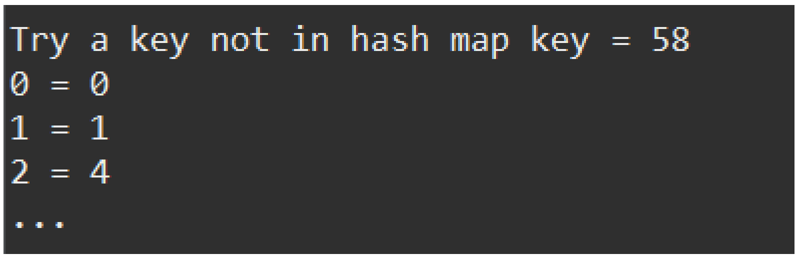
```
hasPrev() = true

Move backward
4 = 16
3 = 9
2 = 4
1 = 1
0 = 0
9 = 81
8 = 64
7 = 49
6 = 36
5 = 25

hasPrev() = false
```



```
hasNext() = true

Move forward
5 = 25
6 = 36
7 = 49
8 = 64
9 = 81
0 = 0
1 = 1
2 = 4
3 = 9
4 = 16

hasNext() = false
```

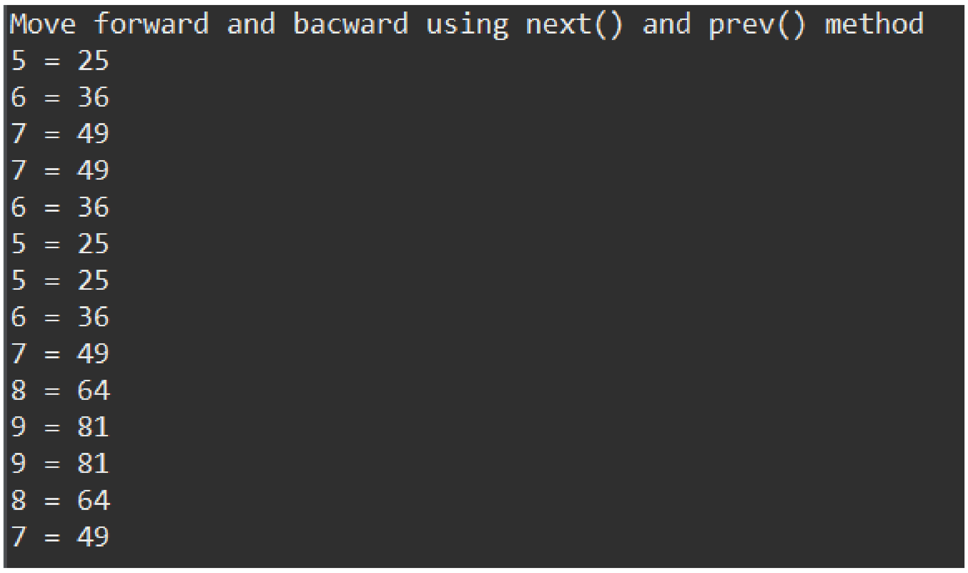
Parameter Constructor but not include as key:

It is start like no parameter constructor



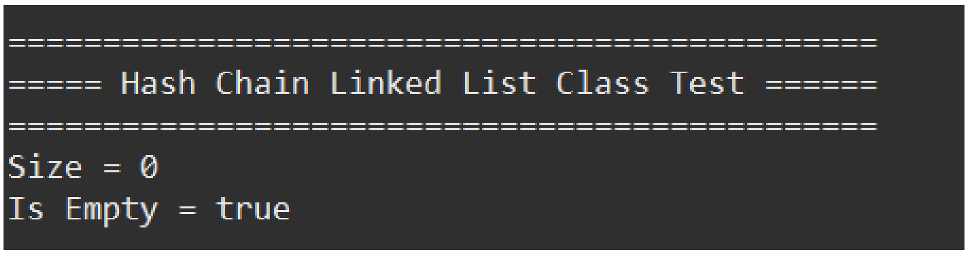
```
Try a key not in hash map key = 58
0 = 0
1 = 1
2 = 4
...
```

Moving forward and backward using iterator:



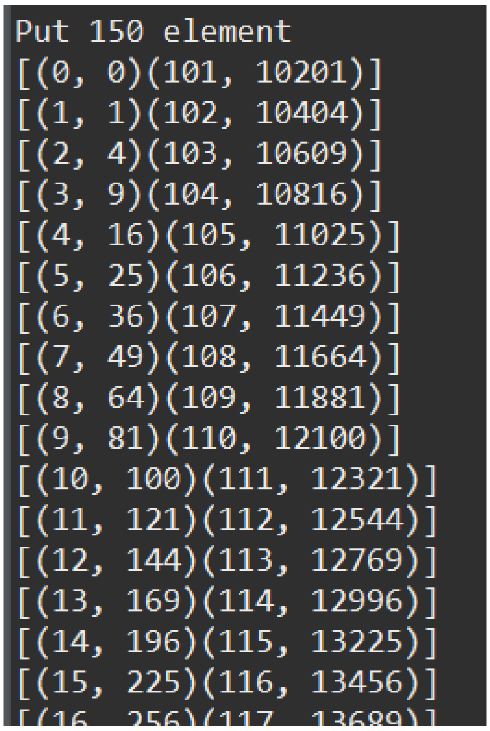
```
Move forward and bacward using next() and prev() method
5 = 25
6 = 36
7 = 49
7 = 49
6 = 36
5 = 25
5 = 25
6 = 36
7 = 49
8 = 64
9 = 81
9 = 81
8 = 64
7 = 49
```

> Hashtable LinkedList Chain:

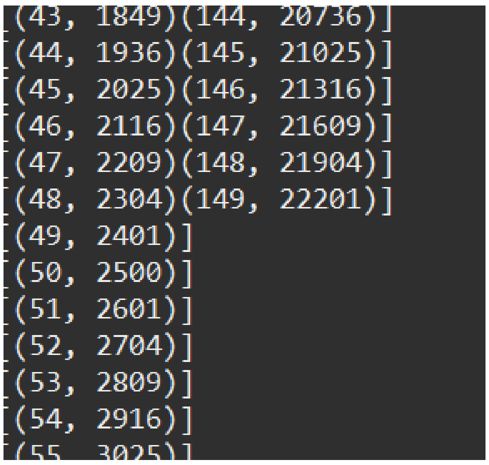


```
==============================================
===== Hash Chain Linked List Class Test ======
==============================================
Size = 0
Is Empty = true
```

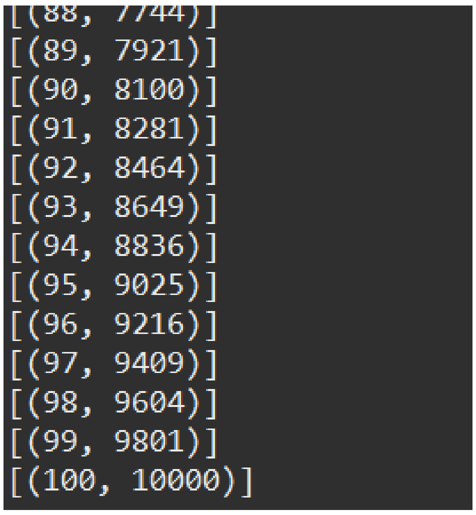
Put Values:



```
Put 150 element
[(0, 0)(101, 10201)]
[(1, 1)(102, 10404)]
[(2, 4)(103, 10609)]
[(3, 9)(104, 10816)]
[(4, 16)(105, 11025)]
[(5, 25)(106, 11236)]
[(6, 36)(107, 11449)]
[(7, 49)(108, 11664)]
[(8, 64)(109, 11881)]
[(9, 81)(110, 12100)]
[(10, 100)(111, 12321)]
[(11, 121)(112, 12544)]
[(12, 144)(113, 12769)]
[(13, 169)(114, 12996)]
[(14, 196)(115, 13225)]
[(15, 225)(116, 13456)]
[(16, 256)(117, 13689)]
```

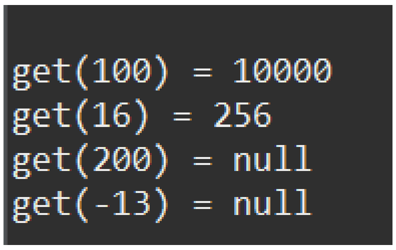


```
[(43, 1849)(144, 20736)]
[(44, 1936)(145, 21025)]
[(45, 2025)(146, 21316)]
[(46, 2116)(147, 21609)]
[(47, 2209)(148, 21904)]
[(48, 2304)(149, 22201)]
[(49, 2401)]
[(50, 2500)]
[(51, 2601)]
[(52, 2704)]
[(53, 2809)]
[(54, 2916)]
[(55, 3025)]
```



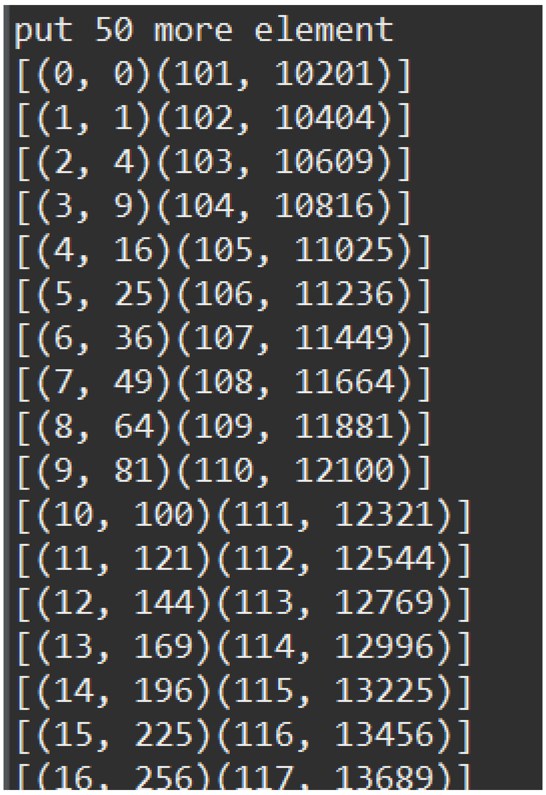
```
[(88, 7744)]
[(89, 7921)]
[(90, 8100)]
[(91, 8281)]
[(92, 8464)]
[(93, 8649)]
[(94, 8836)]
[(95, 9025)]
[(96, 9216)]
[(97, 9409)]
[(98, 9604)]
[(99, 9801)]
[(100, 10000)]
```

Get Values:

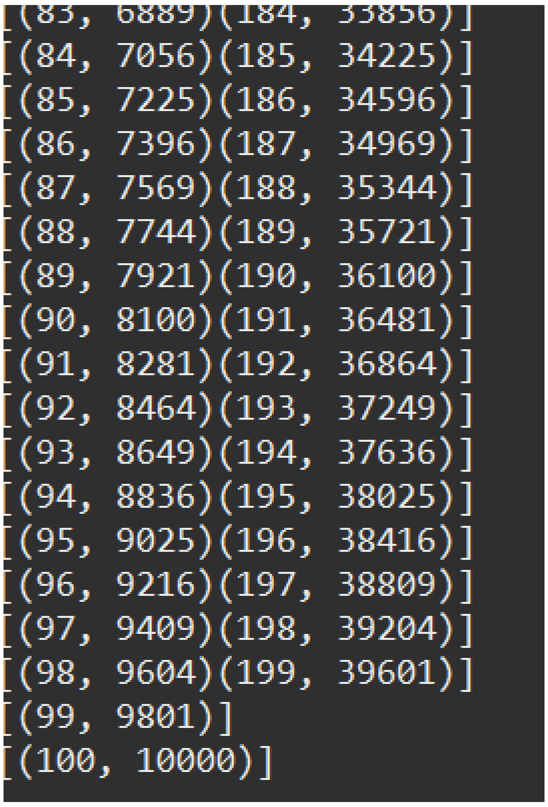


```
get(100) = 10000
get(16) = 256
get(200) = null
get(-13) = null
```

Put More Values:

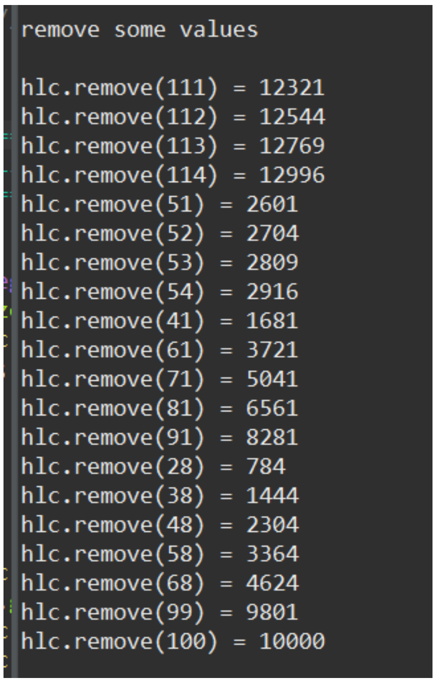


```
put 50 more element
[(0, 0)(101, 10201)]
[(1, 1)(102, 10404)]
[(2, 4)(103, 10609)]
[(3, 9)(104, 10816)]
[(4, 16)(105, 11025)]
[(5, 25)(106, 11236)]
[(6, 36)(107, 11449)]
[(7, 49)(108, 11664)]
[(8, 64)(109, 11881)]
[(9, 81)(110, 12100)]
[(10, 100)(111, 12321)]
[(11, 121)(112, 12544)]
[(12, 144)(113, 12769)]
[(13, 169)(114, 12996)]
[(14, 196)(115, 13225)]
[(15, 225)(116, 13456)]
[(16, 256)(117, 13689)]
```

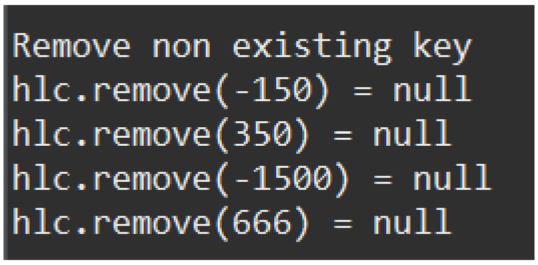


```
[(83, 6889)(184, 33856)]
[(84, 7056)(185, 34225)]
[(85, 7225)(186, 34596)]
[(86, 7396)(187, 34969)]
[(87, 7569)(188, 35344)]
[(88, 7744)(189, 35721)]
[(89, 7921)(190, 36100)]
[(90, 8100)(191, 36481)]
[(91, 8281)(192, 36864)]
[(92, 8464)(193, 37249)]
[(93, 8649)(194, 37636)]
[(94, 8836)(195, 38025)]
[(95, 9025)(196, 38416)]
[(96, 9216)(197, 38809)]
[(97, 9409)(198, 39204)]
[(98, 9604)(199, 39601)]
[(99, 9801)]
[(100, 10000)]
```

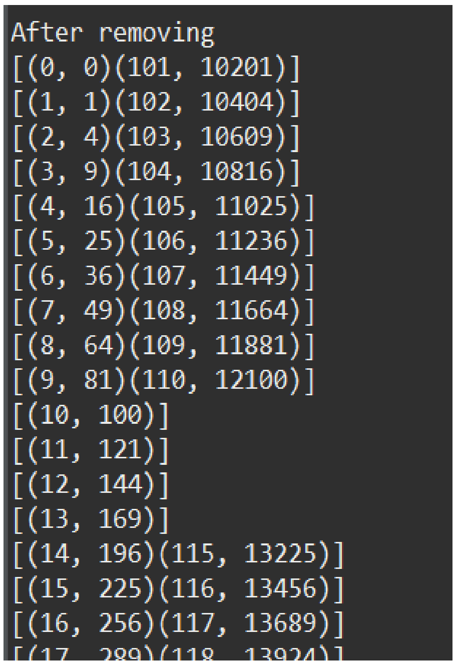
Remove Values:



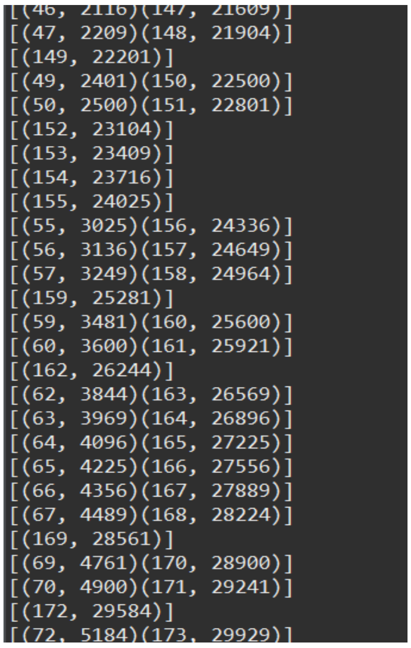
```
remove some values

hlc.remove(111) = 12321
hlc.remove(112) = 12544
hlc.remove(113) = 12769
hlc.remove(114) = 12996
hlc.remove(51) = 2601
hlc.remove(52) = 2704
hlc.remove(53) = 2809
hlc.remove(54) = 2916
hlc.remove(41) = 1681
hlc.remove(61) = 3721
hlc.remove(71) = 5041
hlc.remove(81) = 6561
hlc.remove(91) = 8281
hlc.remove(28) = 784
hlc.remove(38) = 1444
hlc.remove(48) = 2304
hlc.remove(58) = 3364
hlc.remove(68) = 4624
hlc.remove(99) = 9801
hlc.remove(100) = 10000
```



```
Remove non existing key
hlc.remove(-150) = null
hlc.remove(350) = null
hlc.remove(-1500) = null
hlc.remove(666) = null
```

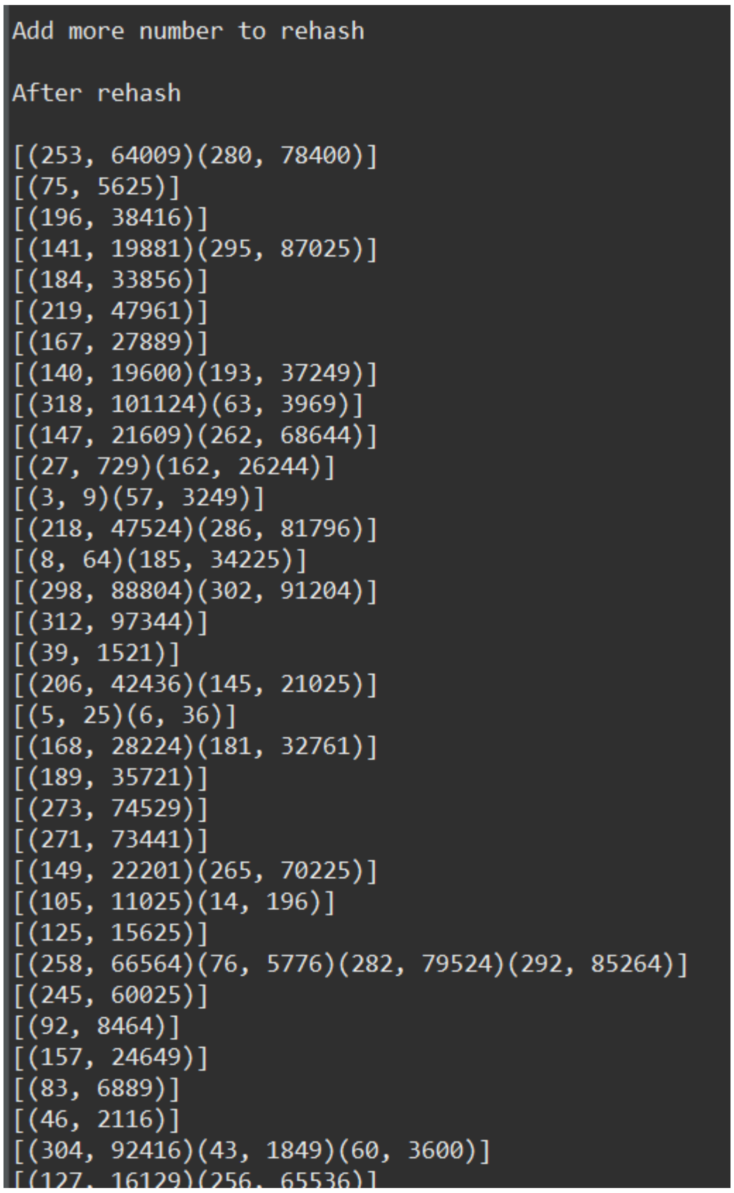


```
After removing
[(0, 0)(101, 10201)]
[(1, 1)(102, 10404)]
[(2, 4)(103, 10609)]
[(3, 9)(104, 10816)]
[(4, 16)(105, 11025)]
[(5, 25)(106, 11236)]
[(6, 36)(107, 11449)]
[(7, 49)(108, 11664)]
[(8, 64)(109, 11881)]
[(9, 81)(110, 12100)]
[(10, 100)]
[(11, 121)]
[(12, 144)]
[(13, 169)]
[(14, 196)(115, 13225)]
[(15, 225)(116, 13456)]
[(16, 256)(117, 13689)]
[(17, 289)(118, 13924)]
```



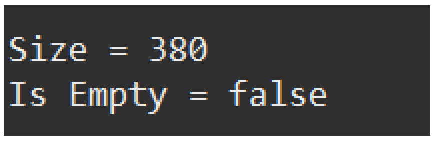
```
[(46, 2116)(147, 21609)]
[(47, 2209)(148, 21904)]
[(149, 22201)]
[(49, 2401)(150, 22500)]
[(50, 2500)(151, 22801)]
[(152, 23104)]
[(153, 23409)]
[(154, 23716)]
[(155, 24025)]
[(55, 3025)(156, 24336)]
[(56, 3136)(157, 24649)]
[(57, 3249)(158, 24964)]
[(159, 25281)]
[(59, 3481)(160, 25600)]
[(60, 3600)(161, 25921)]
[(162, 26244)]
[(62, 3844)(163, 26569)]
[(63, 3969)(164, 26896)]
[(64, 4096)(165, 27225)]
[(65, 4225)(166, 27556)]
[(66, 4356)(167, 27889)]
[(67, 4489)(168, 28224)]
[(169, 28561)]
[(69, 4761)(170, 28900)]
[(70, 4900)(171, 29241)]
[(172, 29584)]
[(72, 5184)(173, 29929)]
```

Rehash:



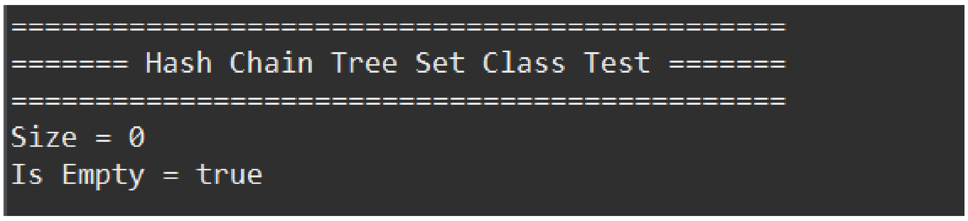
```
Add more number to rehash

After rehash

[(253, 64009)(280, 78400)]
[(75, 5625)]
[(196, 38416)]
[(141, 19881)(295, 87025)]
[(184, 33856)]
[(219, 47961)]
[(167, 27889)]
[(140, 19600)(193, 37249)]
[(318, 101124)(63, 3969)]
[(147, 21609)(262, 68644)]
[(27, 729)(162, 26244)]
[(3, 9)(57, 3249)]
[(218, 47524)(286, 81796)]
[(8, 64)(185, 34225)]
[(298, 88804)(302, 91204)]
[(312, 97344)]
[(39, 1521)]
[(206, 42436)(145, 21025)]
[(5, 25)(6, 36)]
[(168, 28224)(181, 32761)]
[(189, 35721)]
[(273, 74529)]
[(271, 73441)]
[(149, 22201)(265, 70225)]
[(105, 11025)(14, 196)]
[(125, 15625)]
[(258, 66564)(76, 5776)(282, 79524)(292, 85264)]
[(245, 60025)]
[(92, 8464)]
[(157, 24649)]
[(83, 6889)]
[(46, 2116)]
[(304, 92416)(43, 1849)(60, 3600)]
[(127, 16129)(256, 65536)]
```

Last Size:



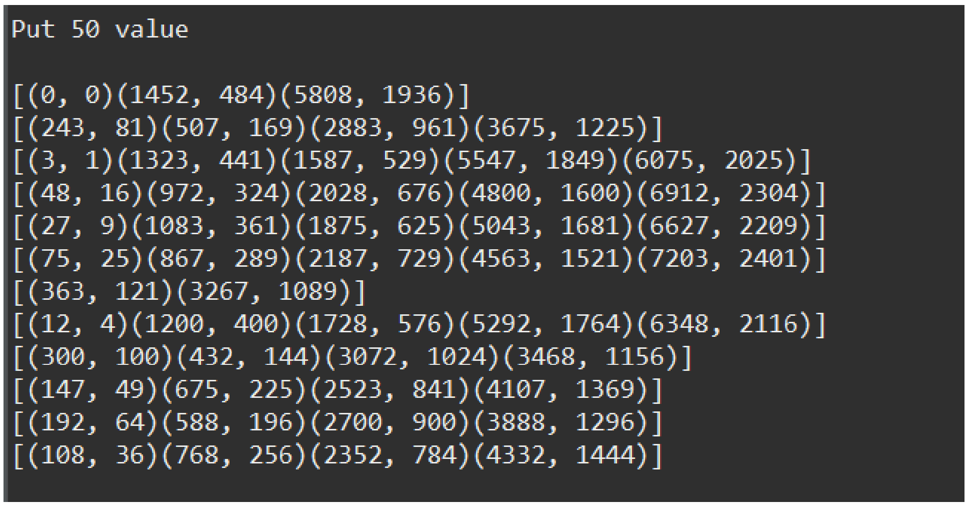
```
Size = 380
Is Empty = false
```

> Hashtable TreeSet Chain:



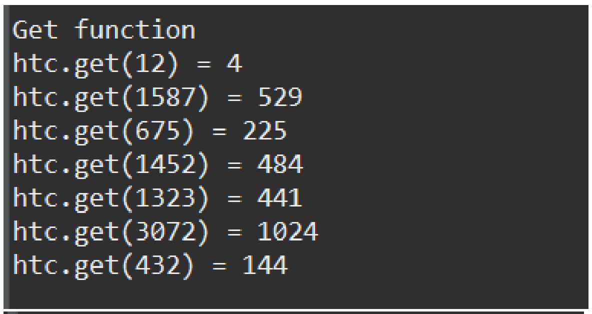
```
==============================================
======= Hash Chain Tree Set Class Test =======
==============================================
Size = 0
Is Empty = true
```

Put Values:

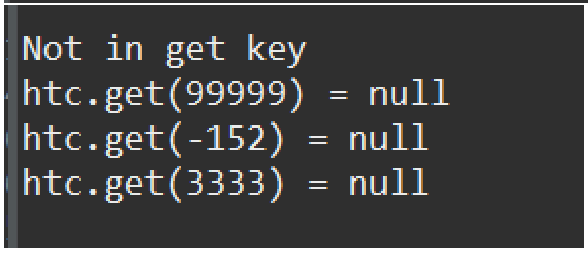


```
Put 50 value

[(0, 0)(1452, 484)(5808, 1936)]
[(243, 81)(507, 169)(2883, 961)(3675, 1225)]
[(3, 1)(1323, 441)(1587, 529)(5547, 1849)(6075, 2025)]
[(48, 16)(972, 324)(2028, 676)(4800, 1600)(6912, 2304)]
[(27, 9)(1083, 361)(1875, 625)(5043, 1681)(6627, 2209)]
[(75, 25)(867, 289)(2187, 729)(4563, 1521)(7203, 2401)]
[(363, 121)(3267, 1089)]
[(12, 4)(1200, 400)(1728, 576)(5292, 1764)(6348, 2116)]
[(300, 100)(432, 144)(3072, 1024)(3468, 1156)]
[(147, 49)(675, 225)(2523, 841)(4107, 1369)]
[(192, 64)(588, 196)(2700, 900)(3888, 1296)]
[(108, 36)(768, 256)(2352, 784)(4332, 1444)]
```

Get Values:

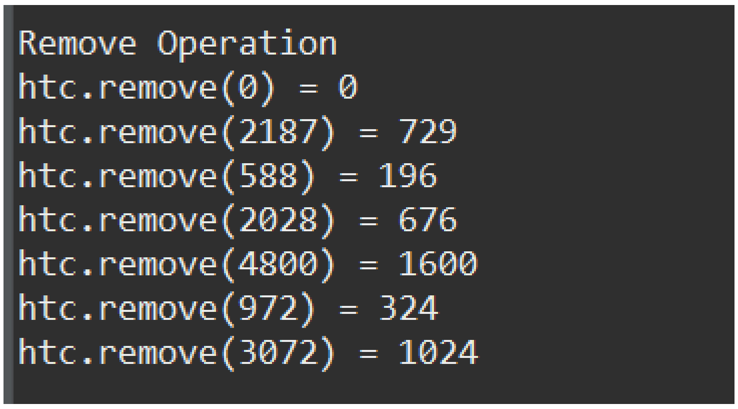


```
Get function
htc.get(12) = 4
htc.get(1587) = 529
htc.get(675) = 225
htc.get(1452) = 484
htc.get(1323) = 441
htc.get(3072) = 1024
htc.get(432) = 144
```

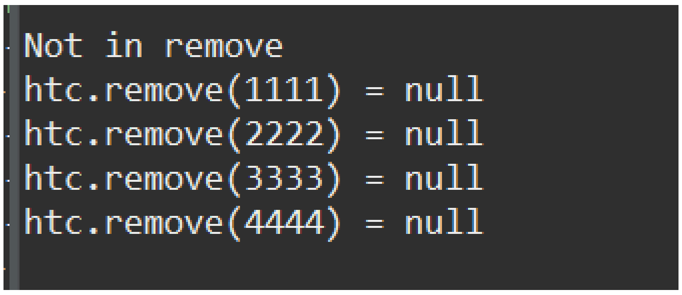


```
Not in get key
htc.get(99999) = null
htc.get(-152) = null
htc.get(3333) = null
```

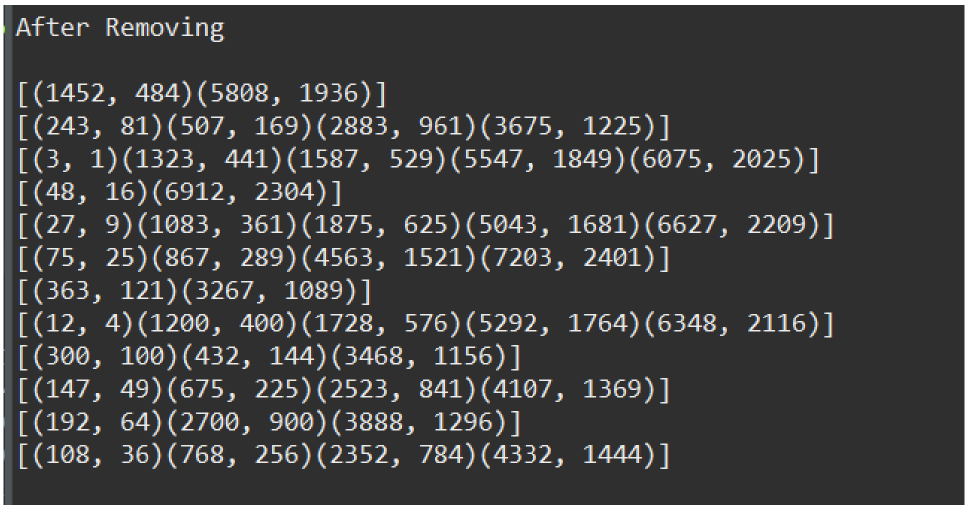
Remove Values:



```
Remove Operation
htc.remove(0) = 0
htc.remove(2187) = 729
htc.remove(588) = 196
htc.remove(2028) = 676
htc.remove(4800) = 1600
htc.remove(972) = 324
htc.remove(3072) = 1024
```

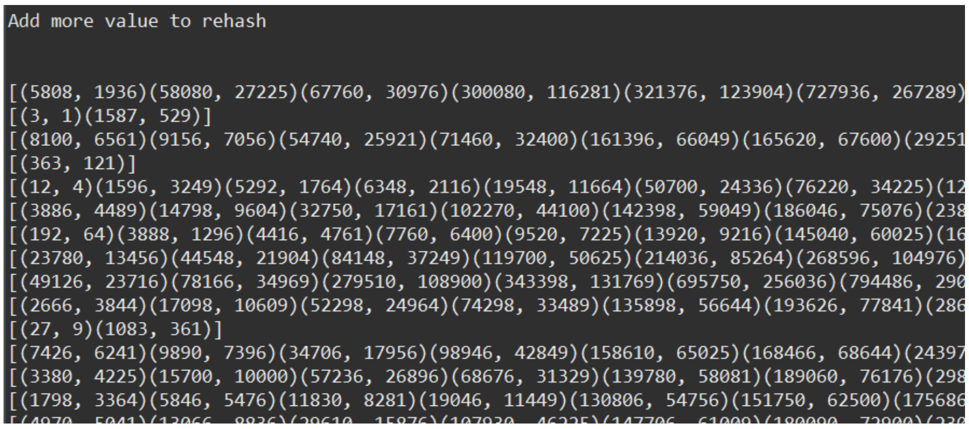


```
Not in remove
htc.remove(1111) = null
htc.remove(2222) = null
htc.remove(3333) = null
htc.remove(4444) = null
```



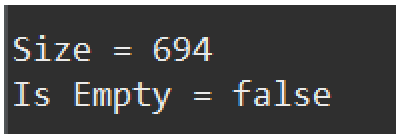
```
After Removing

[(1452, 484)(5808, 1936)]
[(243, 81)(507, 169)(2883, 961)(3675, 1225)]
[(3, 1)(1323, 441)(1587, 529)(5547, 1849)(6075, 2025)]
[(48, 16)(6912, 2304)]
[(27, 9)(1083, 361)(1875, 625)(5043, 1681)(6627, 2209)]
[(75, 25)(867, 289)(4563, 1521)(7203, 2401)]
[(363, 121)(3267, 1089)]
[(12, 4)(1200, 400)(1728, 576)(5292, 1764)(6348, 2116)]
[(300, 100)(432, 144)(3468, 1156)]
[(147, 49)(675, 225)(2523, 841)(4107, 1369)]
[(192, 64)(2700, 900)(3888, 1296)]
[(108, 36)(768, 256)(2352, 784)(4332, 1444)]
```

Rehash:



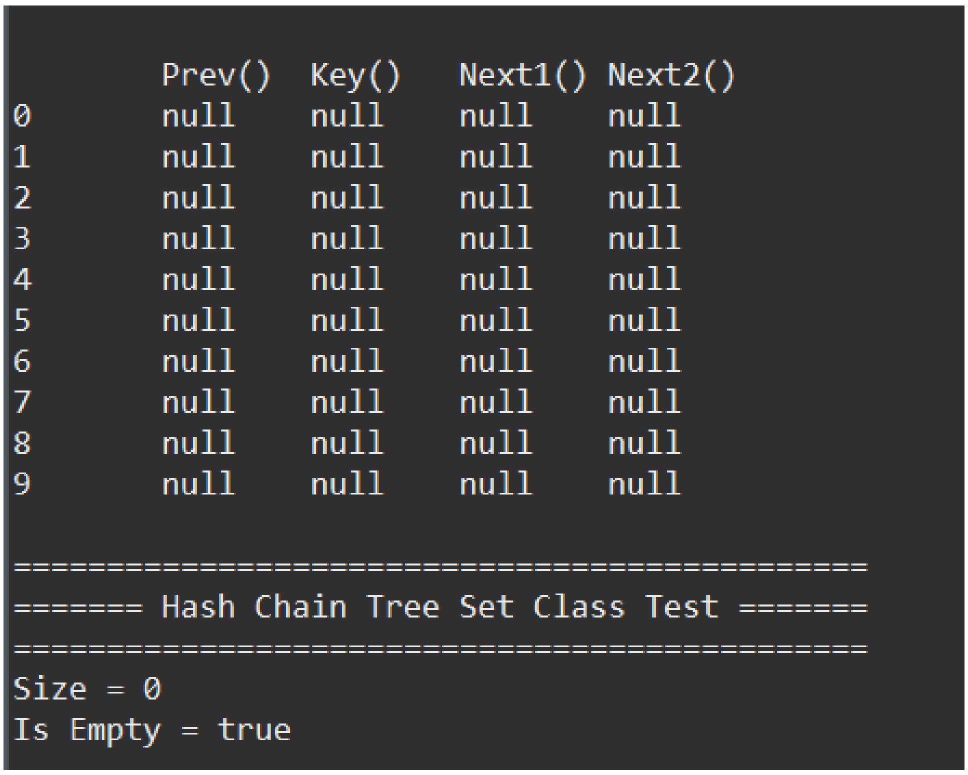
```
Add more value to rehash


[(5808, 1936)(58080, 27225)(67760, 30976)(300080, 116281)(321376, 123904)(727936, 267289)
[(3, 1)(1587, 529)]
[(8100, 6561)(9156, 7056)(54740, 25921)(71460, 32400)(161396, 66049)(165620, 67600)(29251
[(363, 121)]
[(12, 4)(1596, 3249)(5292, 1764)(6348, 2116)(19548, 11664)(50700, 24336)(76220, 34225)(12
[(3886, 4489)(14798, 9604)(32750, 17161)(102270, 44100)(142398, 59049)(186046, 75076)(238
[(192, 64)(3888, 1296)(4416, 4761)(7760, 6400)(9520, 7225)(13920, 9216)(145040, 60025)(16
[(23780, 13456)(44548, 21904)(84148, 37249)(119700, 50625)(214036, 85264)(268596, 104976)
[(49126, 23716)(78166, 34969)(279510, 108900)(343398, 131769)(695750, 256036)(794486, 290
[(2666, 3844)(17098, 10609)(52298, 24964)(74298, 33489)(135898, 56644)(193626, 77841)(286
[(27, 9)(1083, 361)]
[(7426, 6241)(9890, 7396)(34706, 17956)(98946, 42849)(158610, 65025)(168466, 68644)(24397
[(3380, 4225)(15700, 10000)(57236, 26896)(68676, 31329)(139780, 58081)(189060, 76176)(298
[(1798, 3364)(5846, 5476)(11830, 8281)(19046, 11449)(130806, 54756)(151750, 62500)(175686
[(4970, 5041)(13066, 8836)(29610, 15876)(107930, 46225)(147706, 61009)(180090, 72900)(230
```

Last Size:



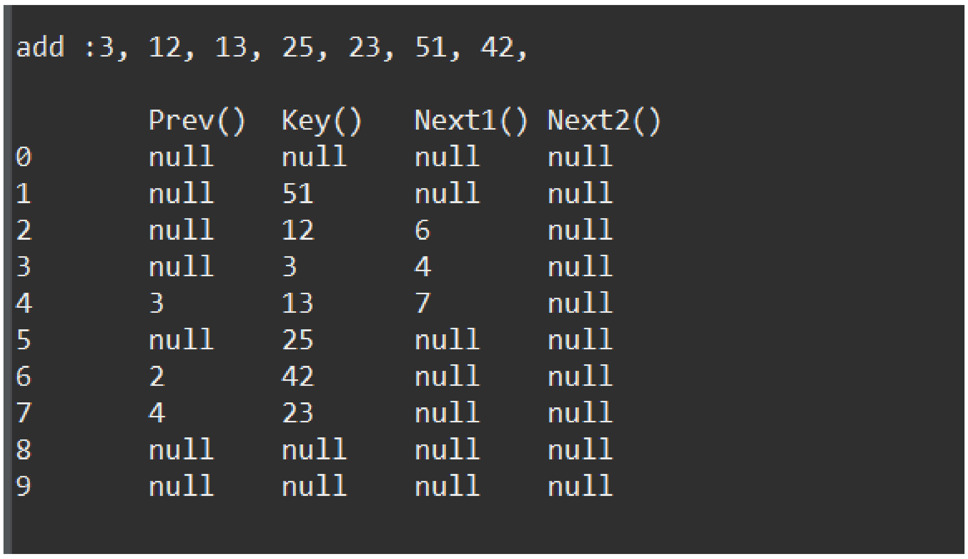
```
Size = 694
Is Empty = false
```

> Hashtable Coalesced Hash Map:



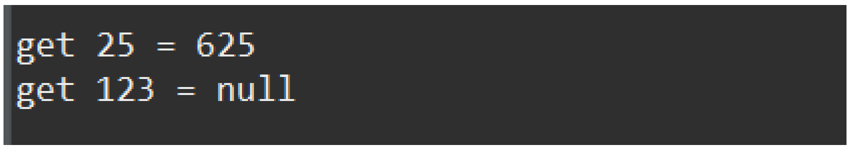
```
        Prev()  Key()   Next1() Next2()
0       null    null    null    null
1       null    null    null    null
2       null    null    null    null
3       null    null    null    null
4       null    null    null    null
5       null    null    null    null
6       null    null    null    null
7       null    null    null    null
8       null    null    null    null
9       null    null    null    null

==============================================
======= Hash Chain Tree Set Class Test =======
==============================================
Size = 0
Is Empty = true
```

Put Values:



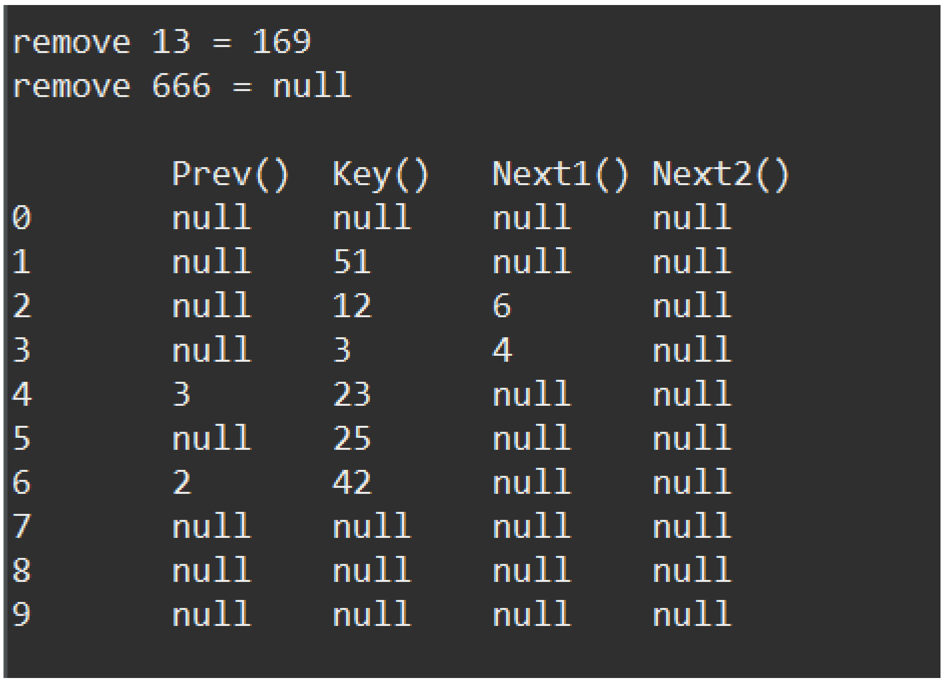
```
add :3, 12, 13, 25, 23, 51, 42,

        Prev()  Key()   Next1() Next2()
0       null    null    null    null
1       null    51      null    null
2       null    12      6       null
3       null    3       4       null
4       3       13      7       null
5       null    25      null    null
6       2       42      null    null
7       4       23      null    null
8       null    null    null    null
9       null    null    null    null
```

Get Values:



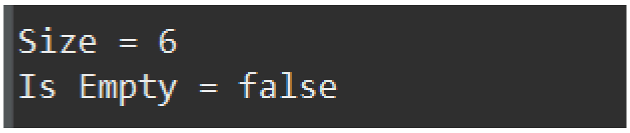
```
get 25 = 625
get 123 = null
```

Remove Values:



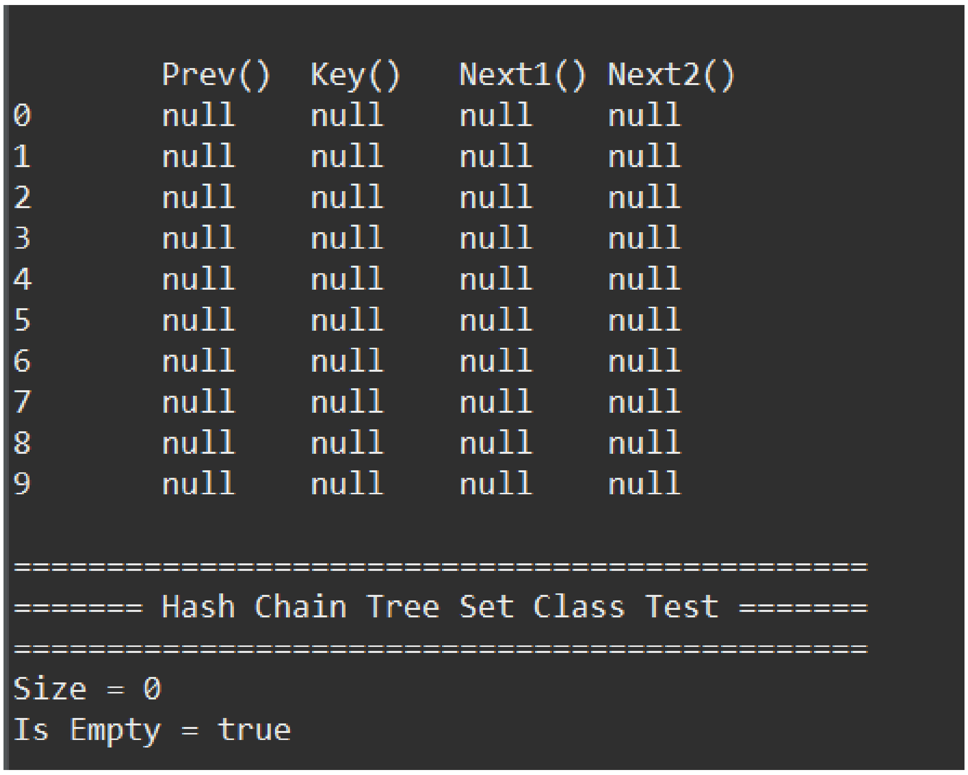
```
remove 13 = 169
remove 666 = null

        Prev()  Key()   Next1() Next2()
0       null    null    null    null
1       null    51      null    null
2       null    12      6       null
3       null    3       4       null
4       3       23      null    null
5       null    25      null    null
6       2       42      null    null
7       null    null    null    null
8       null    null    null    null
9       null    null    null    null
```

Last Size:



```
Size = 6
Is Empty = false
```

> Hashtable Coalesced Hash Map Using next2 and prev:



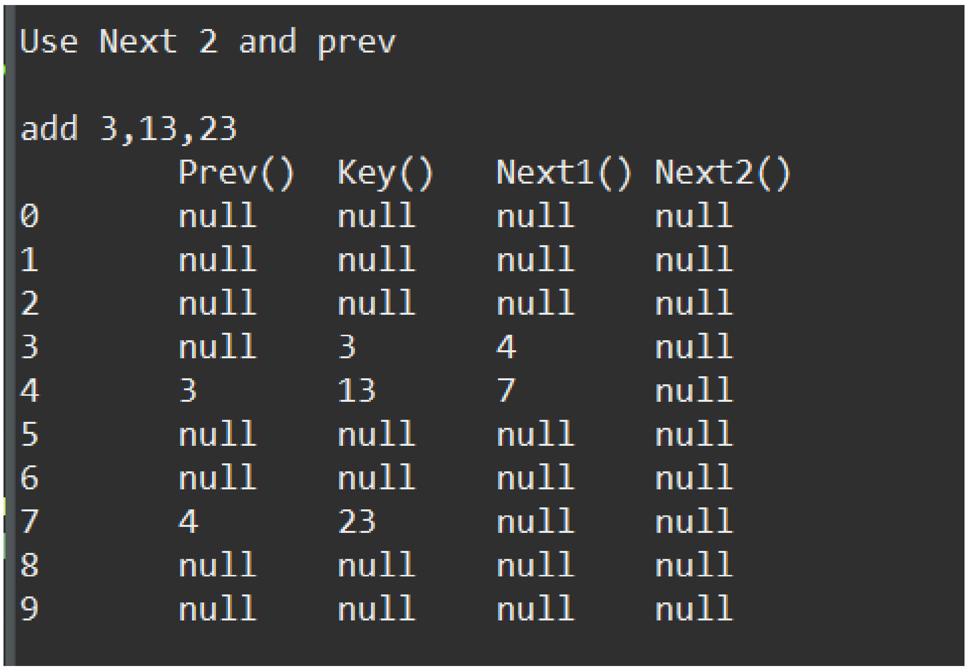
```
        Prev()  Key()   Next1() Next2()
0       null    null    null    null
1       null    null    null    null
2       null    null    null    null
3       null    null    null    null
4       null    null    null    null
5       null    null    null    null
6       null    null    null    null
7       null    null    null    null
8       null    null    null    null
9       null    null    null    null

===============================================
======= Hash Chain Tree Set Class Test =======
===============================================
Size = 0
Is Empty = true
```

- **Next2 , In this data structure if a key kept a place dont match with index.When we want to insert normal index we are take a valid space in hash table and next 2 pointing the index and same hash code key location.**
- **Previous keep who is pointing me**

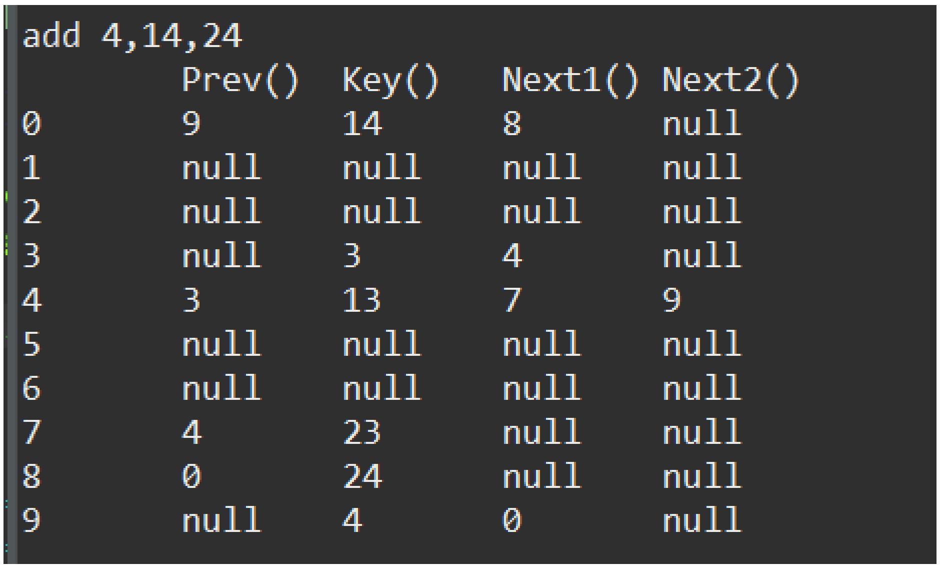
For example:

We add 3 and 13. 13 index position is 4 when we want to 24 or any number hash code is 4 is place is taken value not matching hash code in that case we are take a free space in hash map and next2 is pointing that adress.

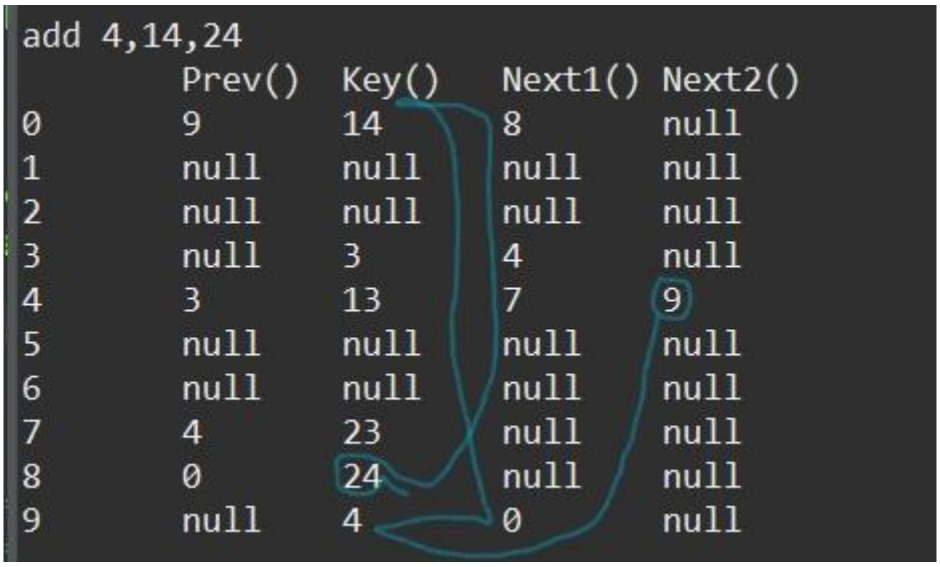
Put Values:



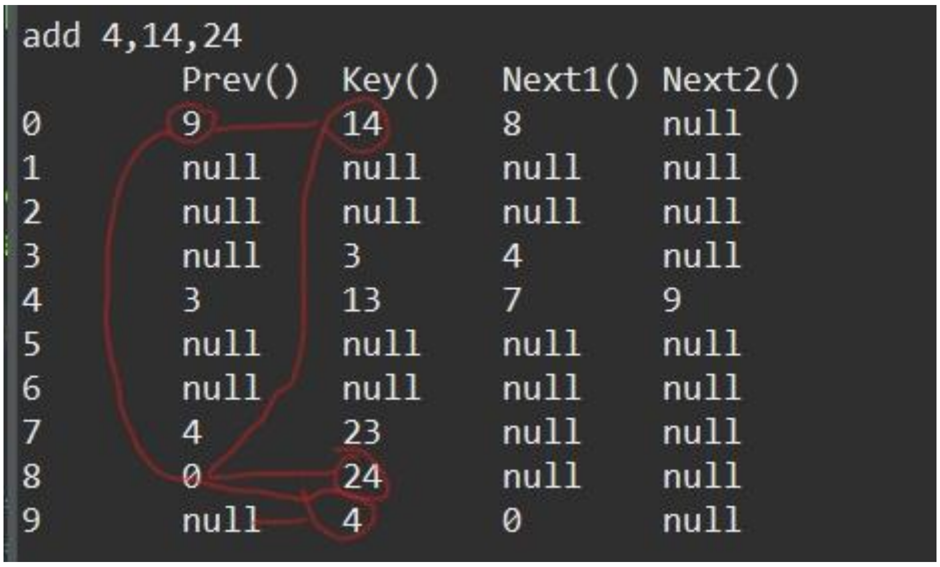
```
Use Next 2 and prev

add 3,13,23
        Prev()  Key()   Next1() Next2()
0       null    null    null    null
1       null    null    null    null
2       null    null    null    null
3       null    3       4       null
4       3       13      7       null
5       null    null    null    null
6       null    null    null    null
7       4       23      null    null
8       null    null    null    null
9       null    null    null    null
```



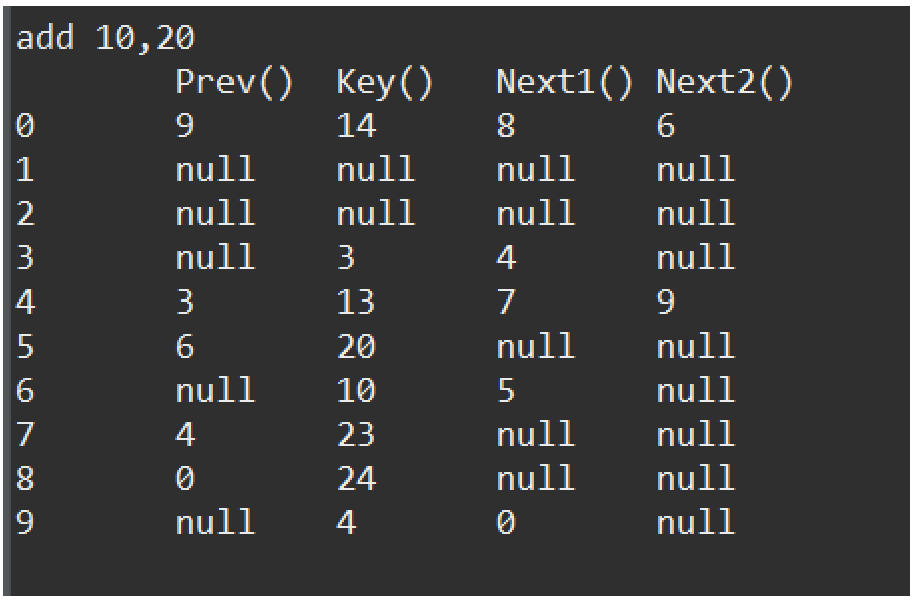
```
add 4,14,24
        Prev()  Key()   Next1() Next2()
0       9       14      8       null
1       null    null    null    null
2       null    null    null    null
3       null    3       4       null
4       3       13      7       9
5       null    null    null    null
6       null    null    null    null
7       4       23      null    null
8       0       24      null    null
9       null    4       0       null
```



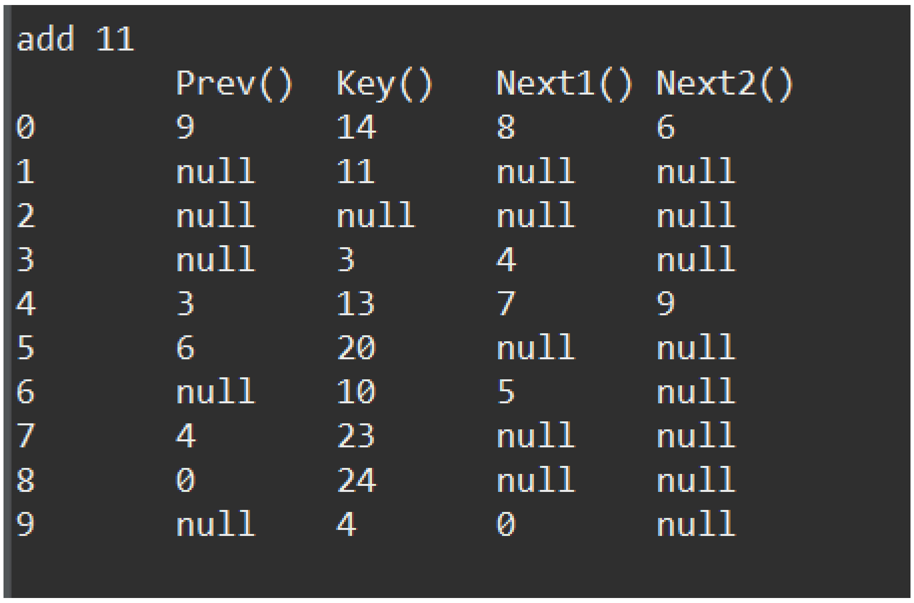
```
add 4,14,24
          Prev()  Key()   Next1() Next2()
0         9       14      8       null
1         null    null    null    null
2         null    null    null    null
3         null    3       4       null
4         3       13      7       9
5         null    null    null    null
6         null    null    null    null
7         4       23      null    null
8         0       24      null    null
9         null    4       0       null
```



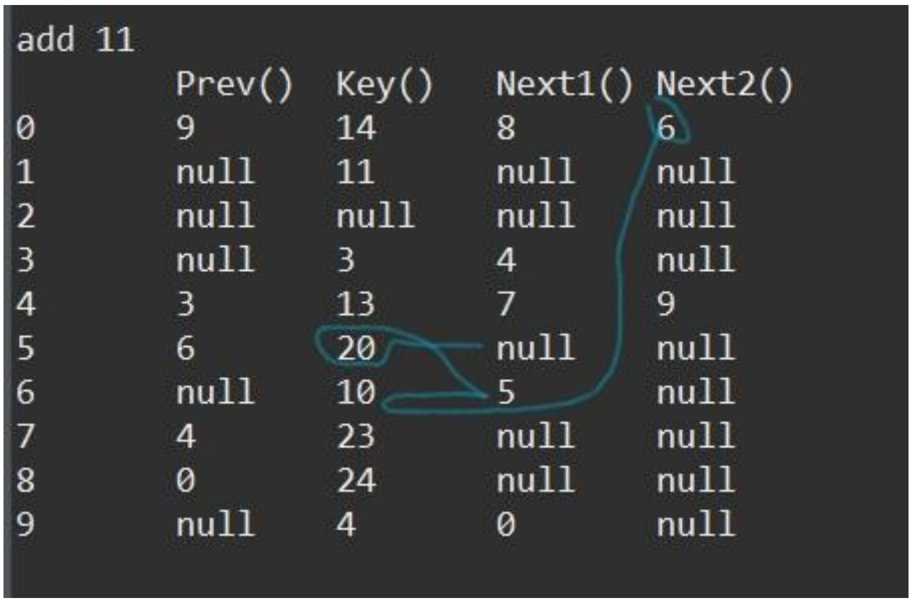
```
add 4,14,24
          Prev()  Key()   Next1() Next2()
0         9       14      8       null
1         null    null    null    null
2         null    null    null    null
3         null    3       4       null
4         3       13      7       9
5         null    null    null    null
6         null    null    null    null
7         4       23      null    null
8         0       24      null    null
9         null    4       0       null
```



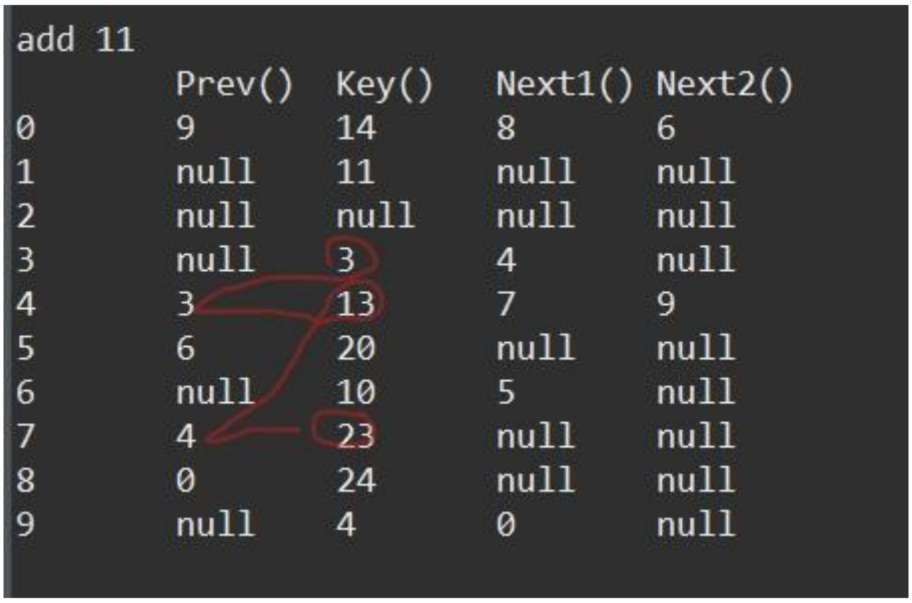
```
add 10,20
          Prev()  Key()   Next1() Next2()
0         9       14      8       6
1         null    null    null    null
2         null    null    null    null
3         null    3       4       null
4         3       13      7       9
5         6       20      null    null
6         null    10      5       null
7         4       23      null    null
8         0       24      null    null
9         null    4       0       null
```



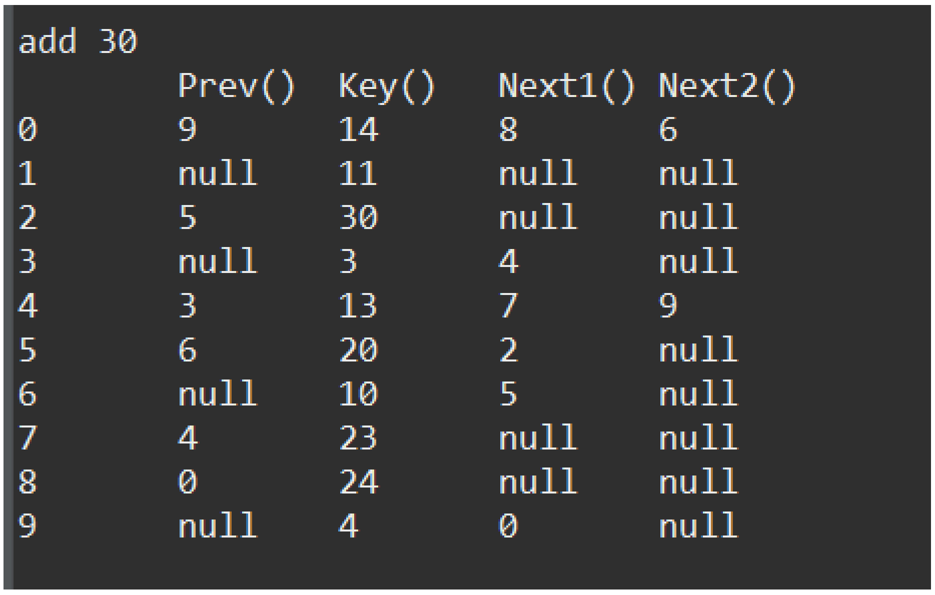
```
add 11
        Prev()  Key()   Next1() Next2()
0       9       14      8       6
1       null    11      null    null
2       null    null    null    null
3       null    3       4       null
4       3       13      7       9
5       6       20      null    null
6       null    10      5       null
7       4       23      null    null
8       0       24      null    null
9       null    4       0       null
```



```
add 11
        Prev()  Key()   Next1() Next2()
0       9       14      8       6
1       null    11      null    null
2       null    null    null    null
3       null    3       4       null
4       3       13      7       9
5       6       20      null    null
6       null    10      5       null
7       4       23      null    null
8       0       24      null    null
9       null    4       0       null
```

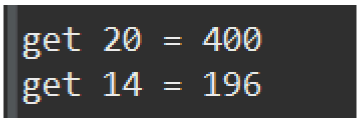


```
add 11
        Prev()  Key()   Next1() Next2()
0       9       14      8       6
1       null    11      null    null
2       null    null    null    null
3       null    3       4       null
4       3       13      7       9
5       6       20      null    null
6       null    10      5       null
7       4       23      null    null
8       0       24      null    null
9       null    4       0       null
```



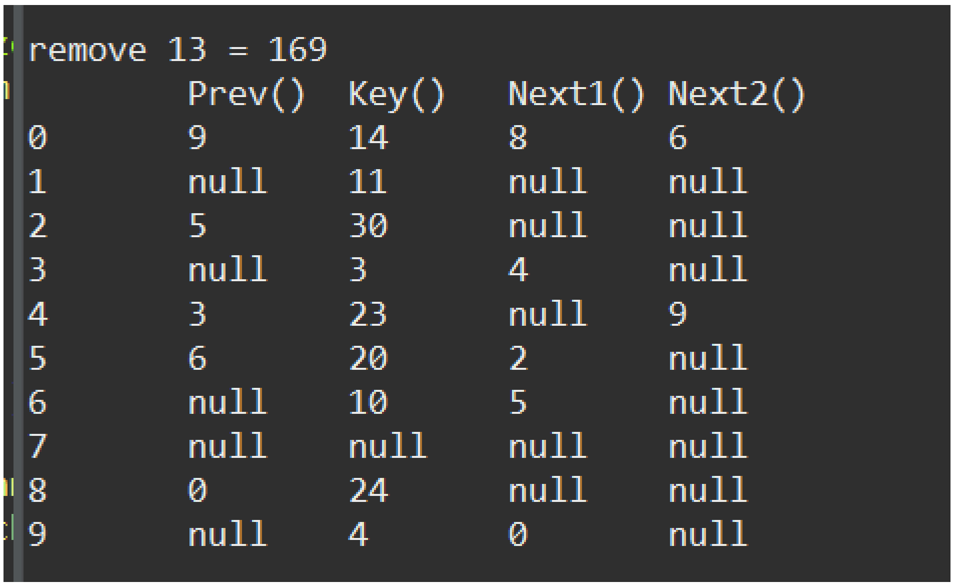
```
add 30
        Prev()  Key()   Next1() Next2()
0       9       14      8       6
1       null    11      null    null
2       5       30      null    null
3       null    3       4       null
4       3       13      7       9
5       6       20      2       null
6       null    10      5       null
7       4       23      null    null
8       0       24      null    null
9       null    4       0       null
```

Get Values:



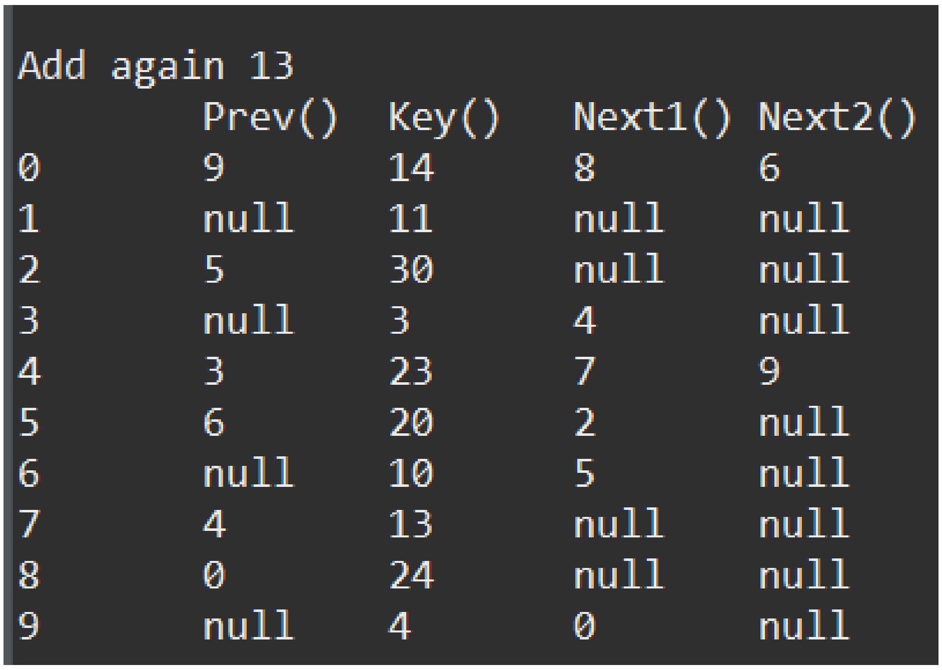
```
get 20 = 400
get 14 = 196
```

Remove Values:



```
remove 13 = 169
          Prev()   Key()    Next1() Next2()
0         9        14       8       6
1         null     11       null    null
2         5        30       null    null
3         null     3        4       null
4         3        23       null    9
5         6        20       2       null
6         null     10       5       null
7         null     null     null    null
8         0        24       null    null
9         null     4        0       null
```

Add Again:



```
Add again 13
          Prev()   Key()    Next1() Next2()
0         9        14       8       6
1         null     11       null    null
2         5        30       null    null
3         null     3        4       null
4         3        23       7       9
5         6        20       2       null
6         null     10       5       null
7         4        13       null    null
8         0        24       null    null
9         null     4        0       null
```