GIT Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework 4 Report

SERDİL ANIL ÜNLÜ

1801042672

1. **SYSTEM REQUIREMENTS**

NON-FUNCTIONAL REQUIREMENTS:

User must have Java Runtime Environment to run this project.

FUNCTIONAL REQUIREMENTS:

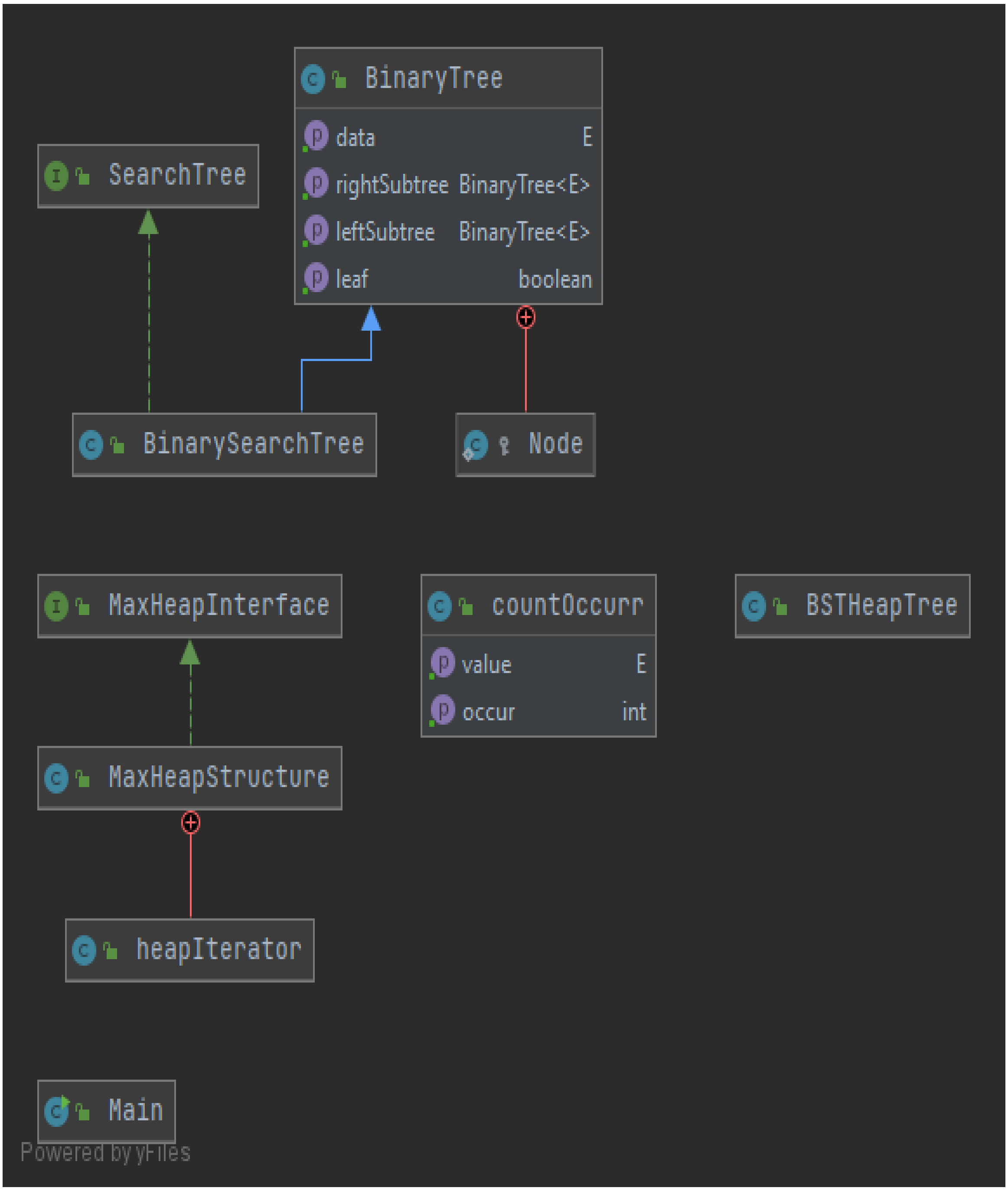
PART1:

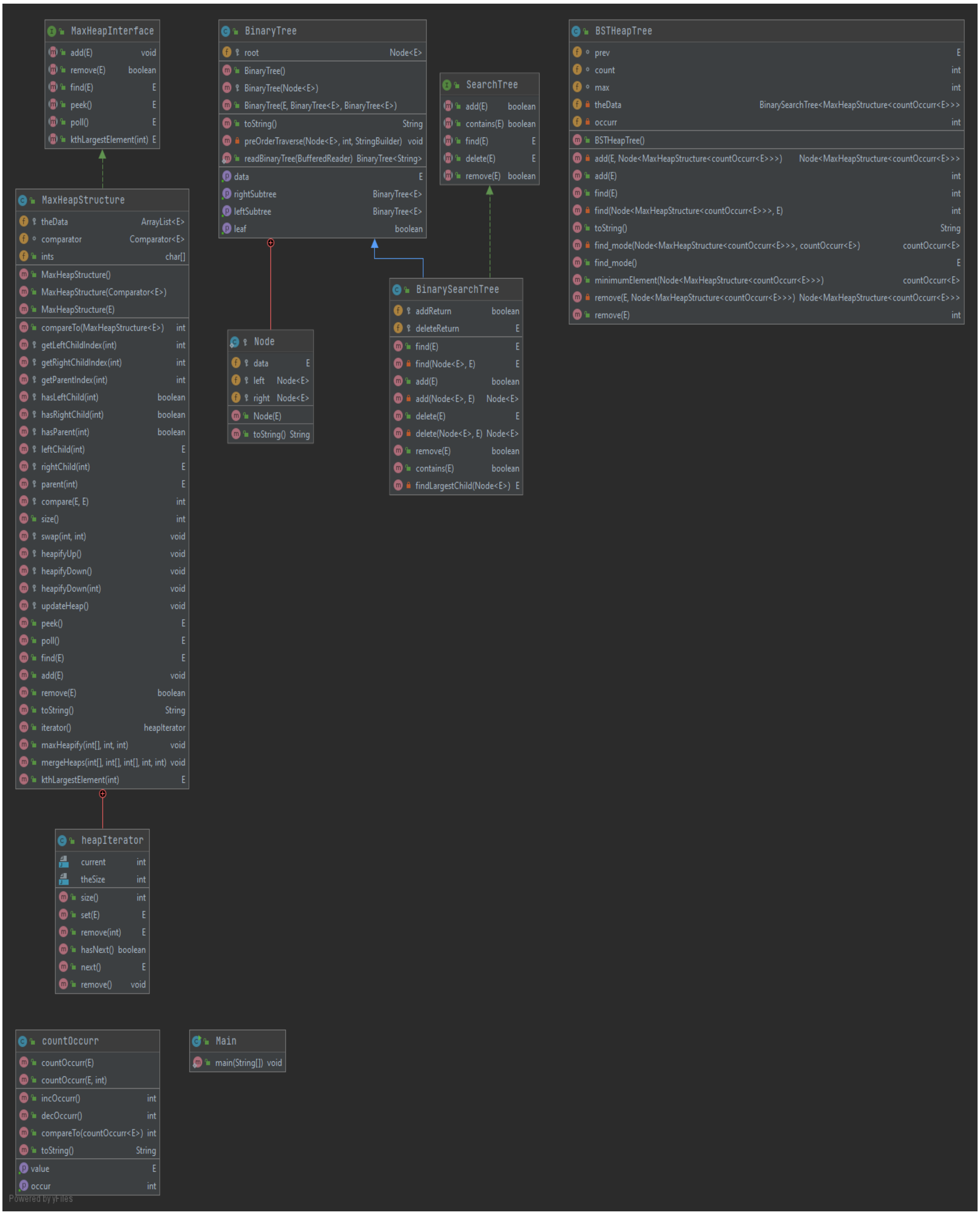
- The user can add as many elements as they want to the heap structure.
- The user can remove ith largest element from the heap.
- The user can merge the two Heap.
- The user set the value of current iteration using iterator.
- The user can be check if the determined element is in the heap.
- The user can update current status of Heap.
- The user can search item which we searched if we find we would return item
- If the user doesn't find it would return null.
- The user can get the tree representation of the heap.
- The user thanks to the iterator, the user can navigate the elements. It can also delete on the element.

PART2:

- The user can be add new elements to the tree.
- The user can be remove determined element in the tree.
- The user find the number of occurrence which of elements in tree.
- The user can find the mode of Binary Search Tree.
- The user can get the number of elements in the tree.

2. USE CASE AND CLASS DIAGRAMS




MaxHeapInterface
add(E) void
remove(E) boolean
find(E) E
peek() E
poll() E
kthLargestElement(int) E
MaxHeapStructure
theData ArrayList<E>
comparator Comparator<E>
ints char[]
MaxHeapStructure()
MaxHeapStructure(Comparator<E>)
MaxHeapStructure(E)
compareTo(MaxHeapStructure<E>) int
getLeftChildIndex(int) int
getRightChildIndex(int) int
getParentIndex(int) int
hasLeftChild(int) boolean
hasRightChild(int) boolean
hasParent(int) boolean
leftChild(int) E
rightChild(int) E
parent(int) E
compare(E, E) int
size() int
swap(int, int) void
heapifyUp() void
heapifyDown() void
heapifyDown(int) void
updateHeap() void
peek() E
poll() E
find(E) E
add(E) void
remove(E) boolean
toString() String
iterator() heapIterator
maxHeapify(int[], int, int) void
mergeHeaps(int[], int[], int[], int, int) void
kthLargestElement(int) E
heapIterator
current int
theSize int
size() int
set(E) E
remove(int) E
hasNext() boolean
next() E
remove() void
countOccurr
countOccurr(E)
countOccurr(E, int)
incOccurr() int
decOccurr() int
compareTo(countOccurr<E>) int
toString() String
value E
occurr int
Main
main(String[]) void
BinaryTree
root Node<E>
BinaryTree()
BinaryTree(Node<E>)
BinaryTree(E, BinaryTree<E>, BinaryTree<E>)
toString() String
preOrderTraverse(Node<E>, int, StringBuilder) void
readBinaryTree(BufferedReader) BinaryTree<String>
data E
rightSubtree BinaryTree<E>
leftSubtree BinaryTree<E>
leaf boolean
Node
data E
left Node<E>
right Node<E>
Node(E)
toString() String
SearchTree
add(E) boolean
contains(E) boolean
find(E) E
delete(E) E
remove(E) boolean
BinarySearchTree
addReturn boolean
deleteReturn E
find(E) E
find(Node<E>, E) E
add(E) boolean
add(Node<E>, E) Node<E>
delete(E) E
delete(Node<E>, E) Node<E>
remove(E) boolean
contains(E) boolean
findLargestChild(Node<E>) E
BSTHeapTree
prev E
count int
max int
theData BinarySearchTree<MaxHeapStructure<countOccurr<E>>>
occurr int
BSTHeapTree()
add(E, Node<MaxHeapStructure<countOccurr<E>>>) Node<MaxHeapStructure<countOccurr<E>>>
add(E) int
find(E) int
find(Node<MaxHeapStructure<countOccurr<E>>>, E) int
toString() String
find_mode(Node<MaxHeapStructure<countOccurr<E>>>, countOccurr<E>) countOccurr<E>
find_mode() E
minimumElement(Node<MaxHeapStructure<countOccurr<E>>>) countOccurr<E>
remove(E, Node<MaxHeapStructure<countOccurr<E>>>) Node<MaxHeapStructure<countOccurr<E>>>
remove(E) int
Powered by yFiles

3. PROBLEM SOLUTION APPROACH

PART1:

I implemented MaxHeapStructure class as a Max Heap class and I consist Interface for this Max Heap class.

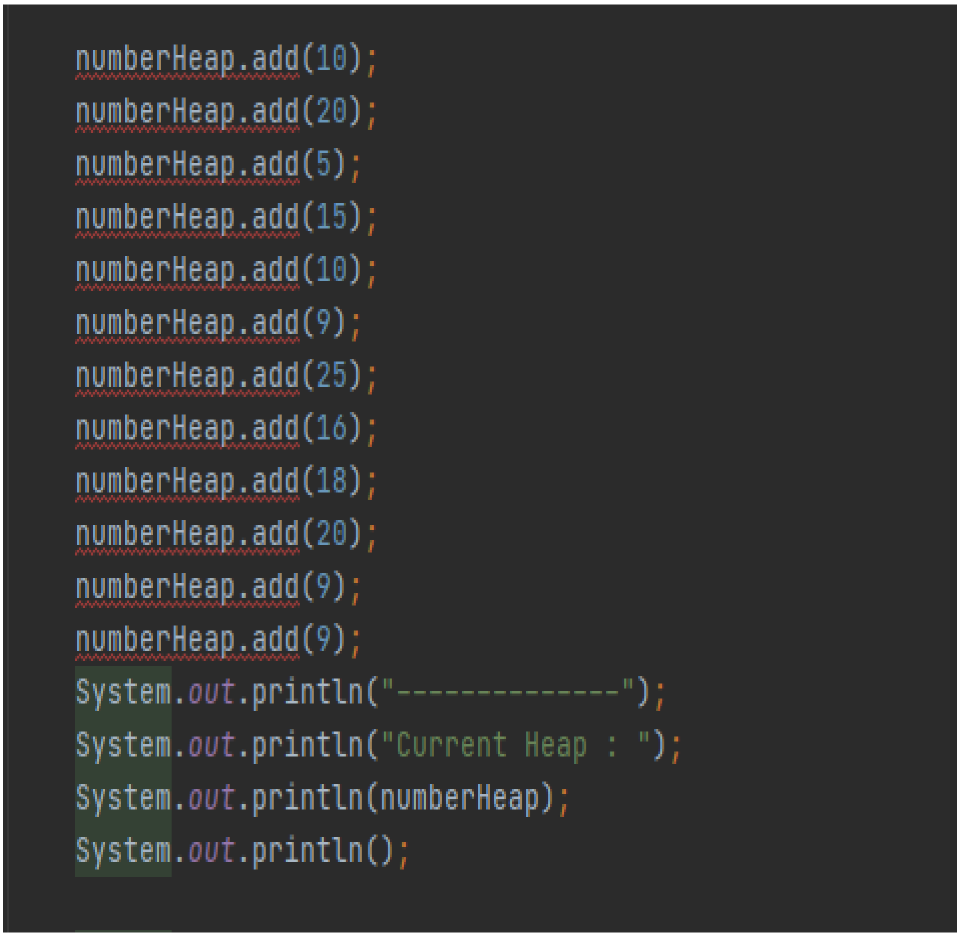
PART2:

I implemented BinaryTree, BinarySearchTree, BSTHeapTree class also I consist countOccurr class for data of items.

4. TEST CASES:

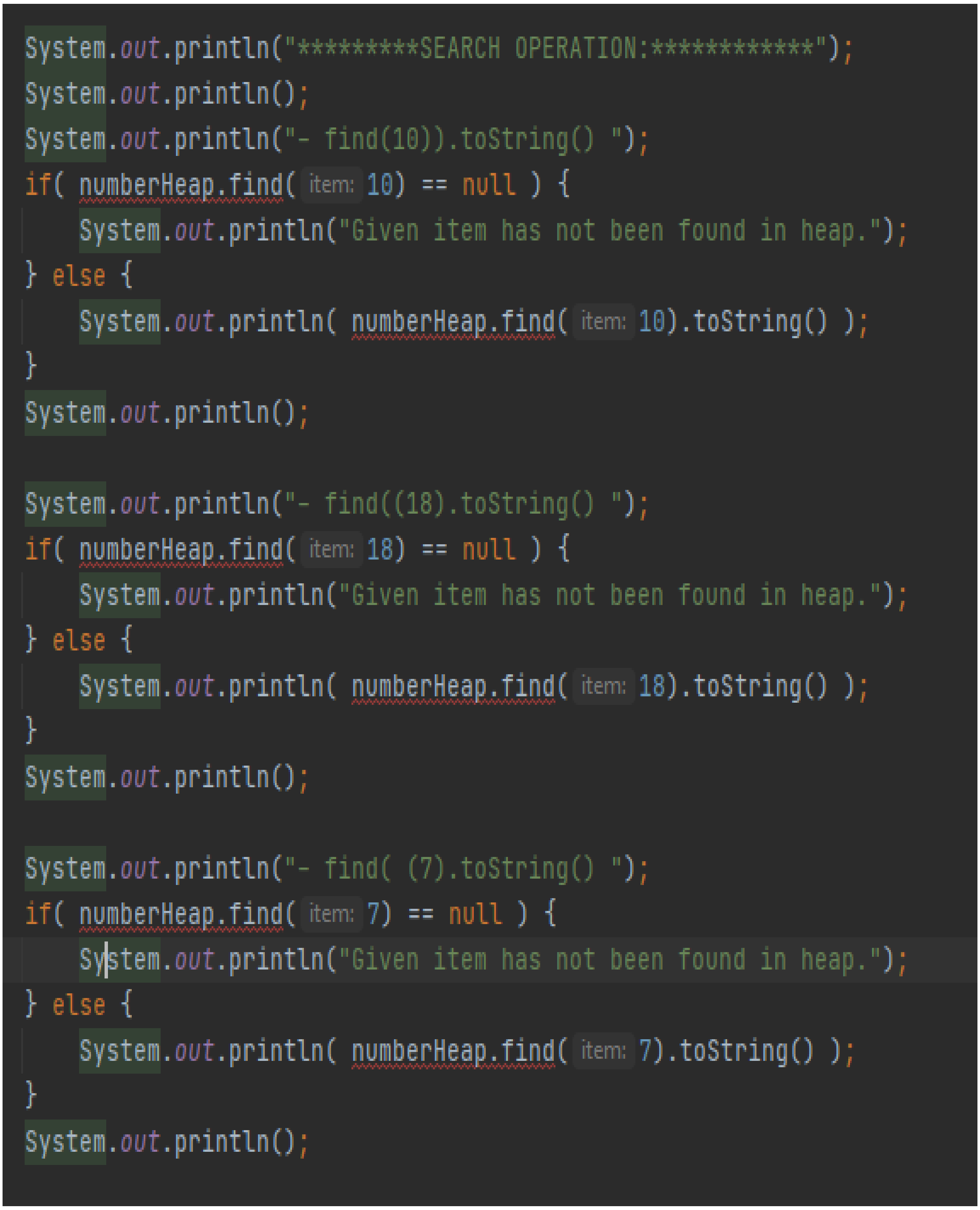
PART1:

INSERT:



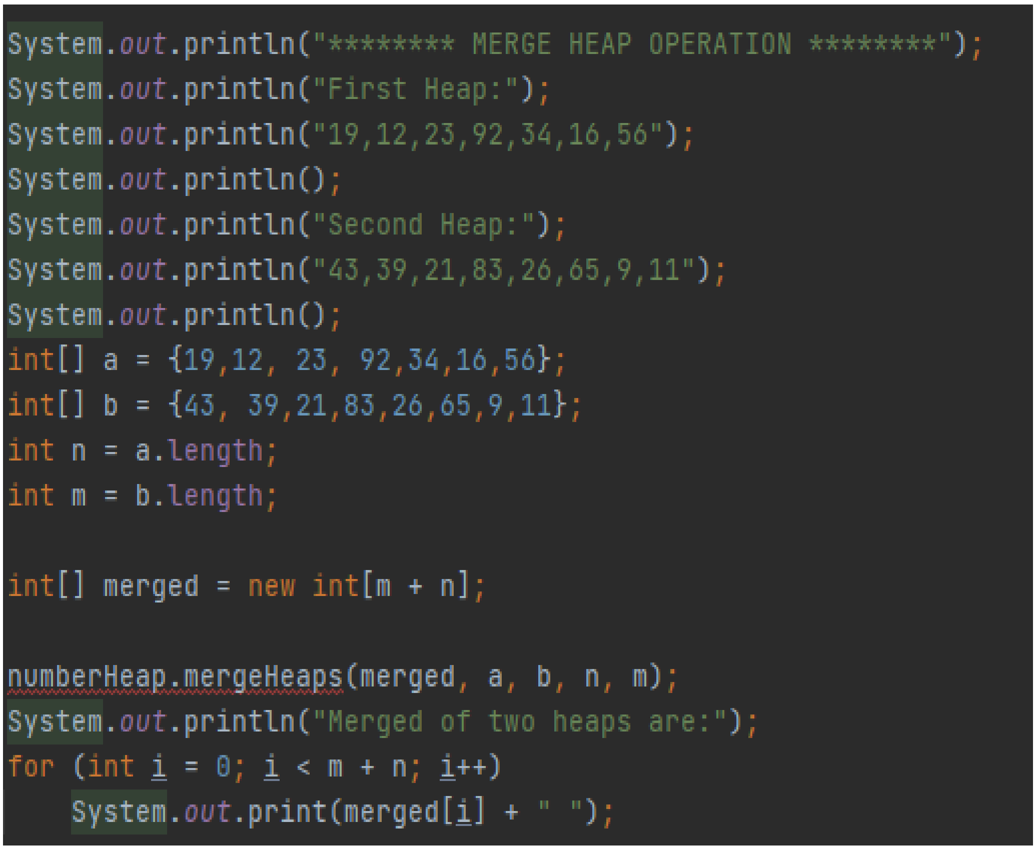
```
numberHeap.add(10);
numberHeap.add(20);
numberHeap.add(5);
numberHeap.add(15);
numberHeap.add(10);
numberHeap.add(9);
numberHeap.add(25);
numberHeap.add(16);
numberHeap.add(18);
numberHeap.add(20);
numberHeap.add(9);
numberHeap.add(9);
System.out.println("--------------");
System.out.println("Current Heap : ");
System.out.println(numberHeap);
System.out.println();
```

SEARCH:



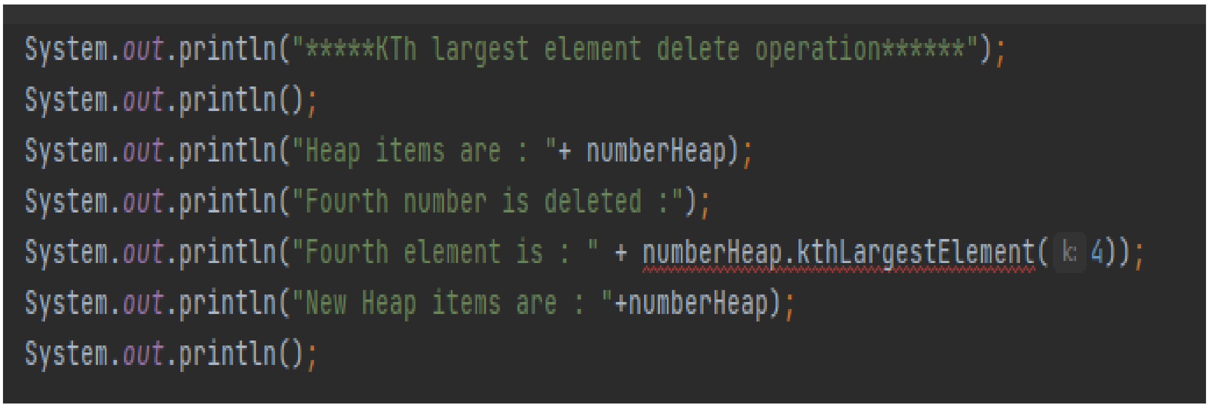
```
System.out.println("*********SEARCH OPERATION:************");
System.out.println();
System.out.println("- find(10)).toString() ");
if( numberHeap.find( item: 10) == null ) {
    System.out.println("Given item has not been found in heap.");
} else {
    System.out.println( numberHeap.find( item: 10).toString() );
}
System.out.println();

System.out.println("- find((18).toString() ");
if( numberHeap.find( item: 18) == null ) {
    System.out.println("Given item has not been found in heap.");
} else {
    System.out.println( numberHeap.find( item: 18).toString() );
}
System.out.println();

System.out.println("- find( (7).toString() ");
if( numberHeap.find( item: 7) == null ) {
    System.out.println("Given item has not been found in heap.");
} else {
    System.out.println( numberHeap.find( item: 7).toString() );
}
System.out.println();
```

MERGE:



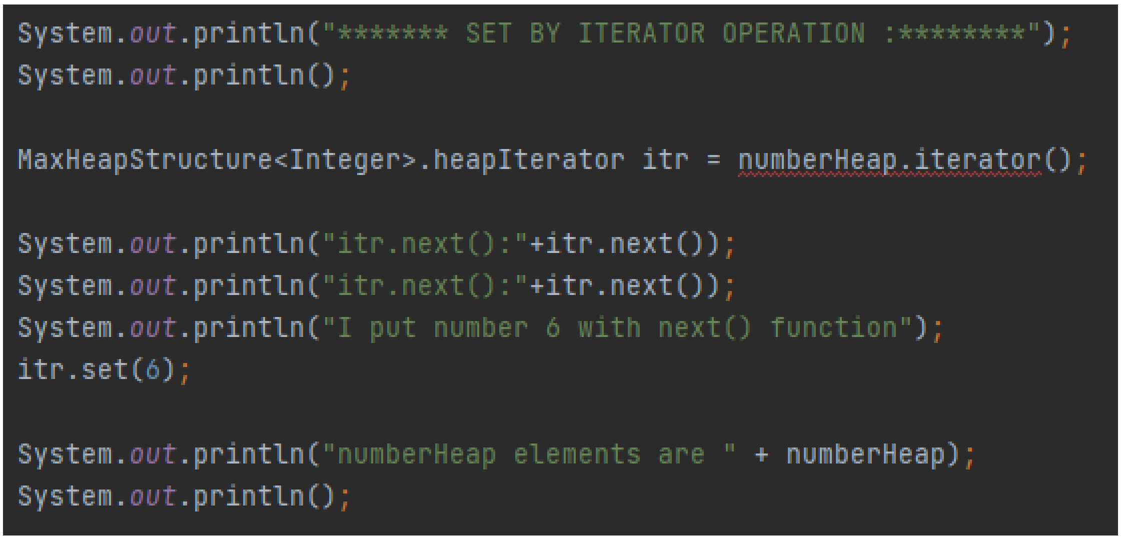
```java
System.out.println("******** MERGE HEAP OPERATION ********");
System.out.println("First Heap:");
System.out.println("19,12,23,92,34,16,56");
System.out.println();
System.out.println("Second Heap:");
System.out.println("43,39,21,83,26,65,9,11");
System.out.println();
int[] a = {19,12, 23, 92,34,16,56};
int[] b = {43, 39,21,83,26,65,9,11};
int n = a.length;
int m = b.length;

int[] merged = new int[m + n];

numberHeap.mergeHeaps(merged, a, b, n, m);
System.out.println("Merged of two heaps are:");
for (int i = 0; i < m + n; i++)
    System.out.print(merged[i] + " ");
```

KTH LARGEST ELEMENT:



```java
System.out.println("*****KTh largest element delete operation******");
System.out.println();
System.out.println("Heap items are : "+ numberHeap);
System.out.println("Fourth number is deleted :");
System.out.println("Fourth element is : " + numberHeap.kthLargestElement( k: 4));
System.out.println("New Heap items are : "+numberHeap);
System.out.println();
```

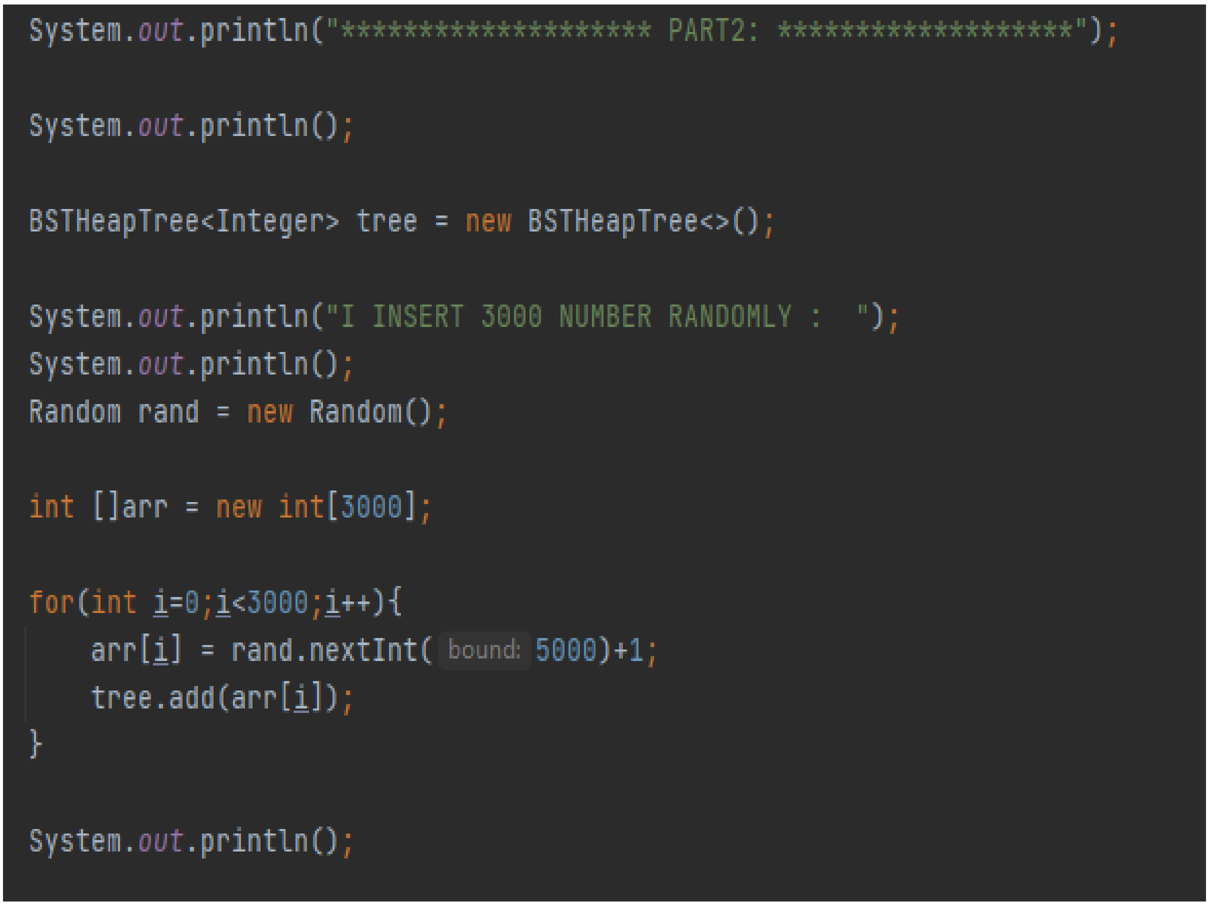
SET BY ITERATOR:



```java
System.out.println("******* SET BY ITERATOR OPERATION :********");
System.out.println();

MaxHeapStructure<Integer>.heapIterator itr = numberHeap.iterator();

System.out.println("itr.next():"+itr.next());
System.out.println("itr.next():"+itr.next());
System.out.println("I put number 6 with next() function");
itr.set(6);

System.out.println("numberHeap elements are " + numberHeap);
System.out.println();
```

PART2:

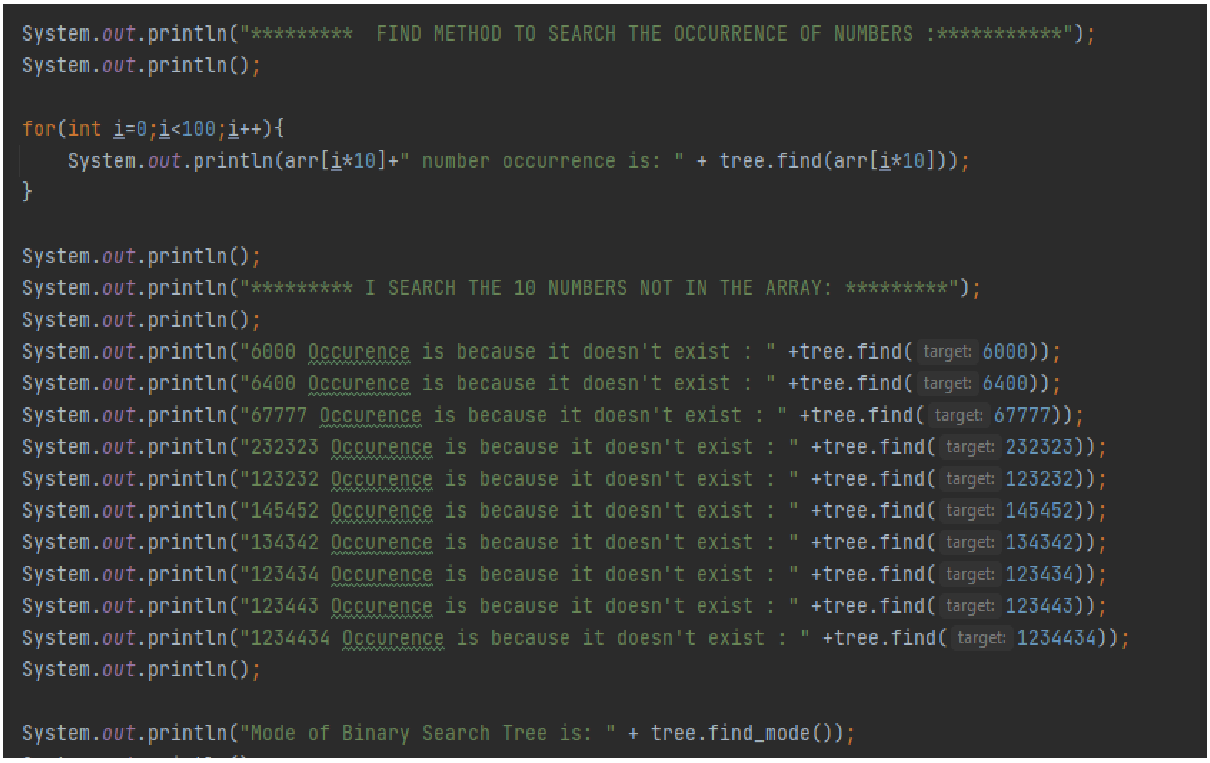
ADD AND INITIALIZE:



```java
System.out.println("******************** PART2: *******************");

System.out.println();

BSTHeapTree<Integer> tree = new BSTHeapTree<>();

System.out.println("I INSERT 3000 NUMBER RANDOMLY :  ");
System.out.println();
Random rand = new Random();

int []arr = new int[3000];

for(int i=0;i<3000;i++){
    arr[i] = rand.nextInt( bound: 5000)+1;
    tree.add(arr[i]);
}

System.out.println();
```

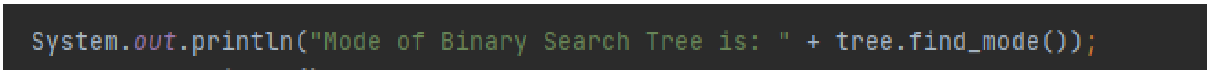
FIND THE ELEMENTS AND IT'S OCCURRENCE:

Also I search element not in the Array.



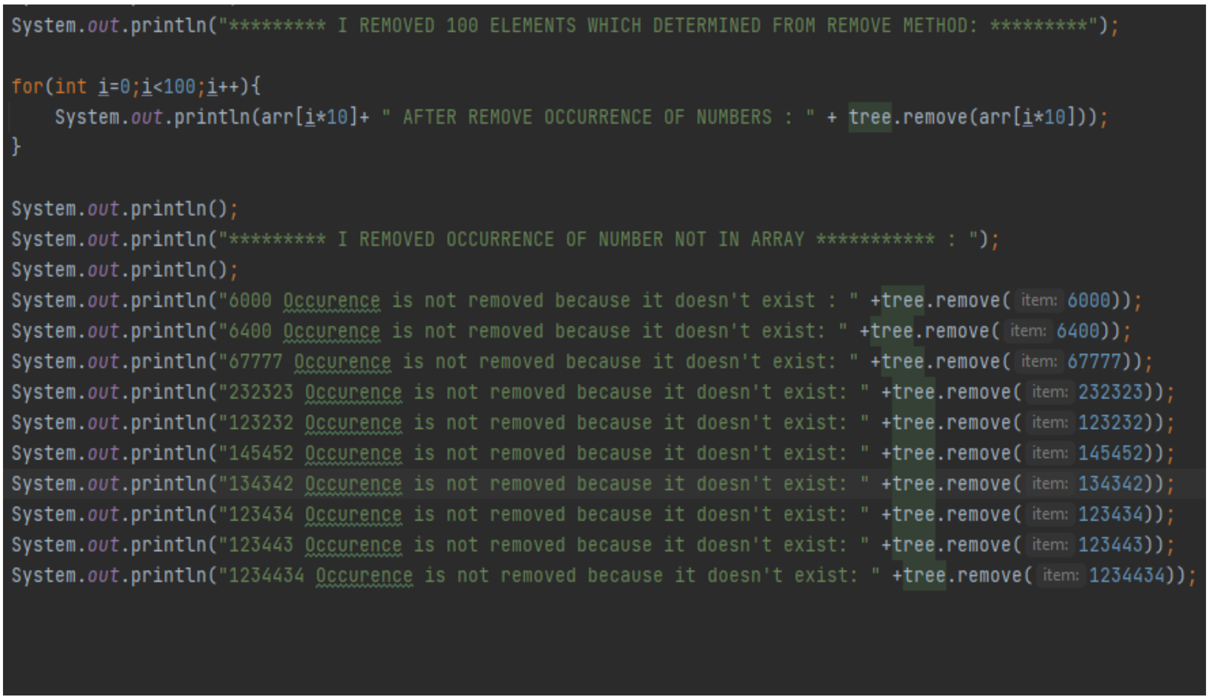
```
System.out.println("*********  FIND METHOD TO SEARCH THE OCCURRENCE OF NUMBERS :***********");
System.out.println();

for(int i=0;i<100;i++){
    System.out.println(arr[i*10]+" number occurrence is: " + tree.find(arr[i*10]));
}

System.out.println();
System.out.println("********* I SEARCH THE 10 NUMBERS NOT IN THE ARRAY: *********");
System.out.println();
System.out.println("6000 Occurence is because it doesn't exist : " +tree.find( target: 6000));
System.out.println("6400 Occurence is because it doesn't exist : " +tree.find( target: 6400));
System.out.println("67777 Occurence is because it doesn't exist : " +tree.find( target: 67777));
System.out.println("232323 Occurence is because it doesn't exist : " +tree.find( target: 232323));
System.out.println("123232 Occurence is because it doesn't exist : " +tree.find( target: 123232));
System.out.println("145452 Occurence is because it doesn't exist : " +tree.find( target: 145452));
System.out.println("134342 Occurence is because it doesn't exist : " +tree.find( target: 134342));
System.out.println("123434 Occurence is because it doesn't exist : " +tree.find( target: 123434));
System.out.println("123443 Occurence is because it doesn't exist : " +tree.find( target: 123443));
System.out.println("1234434 Occurence is because it doesn't exist : " +tree.find( target: 1234434));
System.out.println();

System.out.println("Mode of Binary Search Tree is: " + tree.find_mode());
```

FIND MODE:



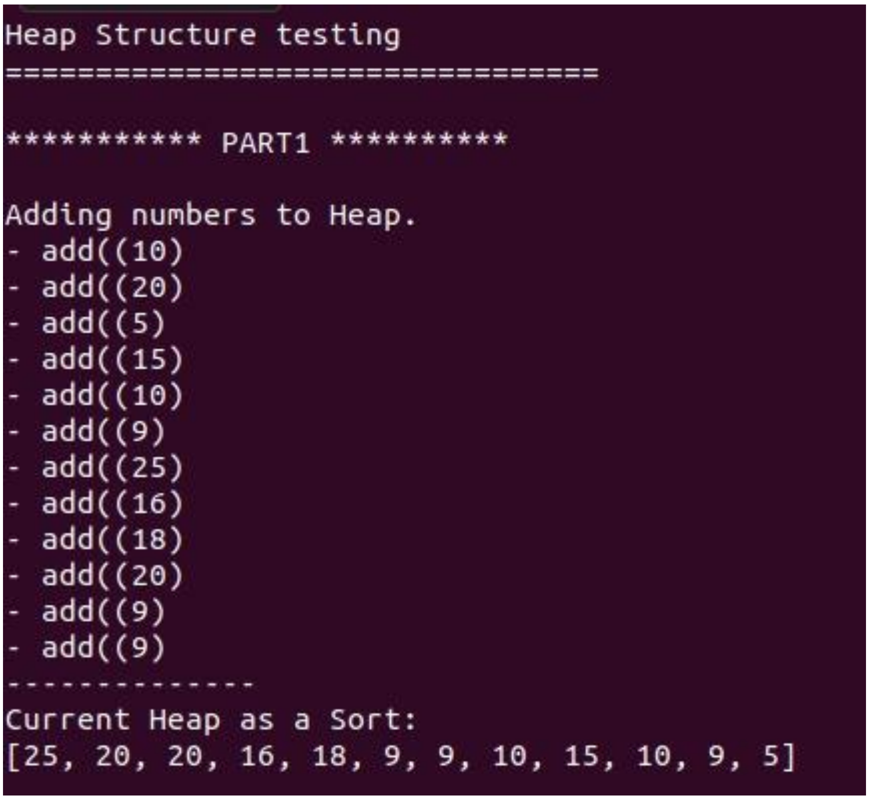
```
System.out.println("Mode of Binary Search Tree is: " + tree.find_mode());
```

REMOVE: (Also I search element which not in the array.)

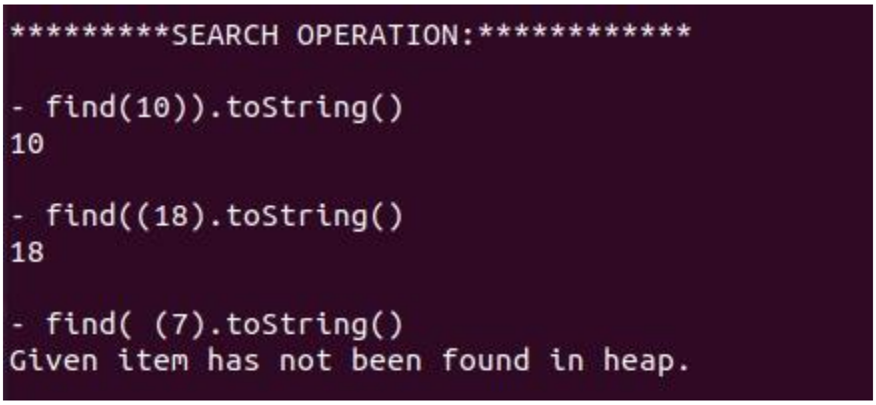


```
System.out.println("********* I REMOVED 100 ELEMENTS WHICH DETERMINED FROM REMOVE METHOD: *********");

for(int i=0;i<100;i++){
    System.out.println(arr[i*10]+ " AFTER REMOVE OCCURRENCE OF NUMBERS : " + tree.remove(arr[i*10]));
}

System.out.println();
System.out.println("********* I REMOVED OCCURRENCE OF NUMBER NOT IN ARRAY *********** : ");
System.out.println();
System.out.println("6000 Occurence is not removed because it doesn't exist : " +tree.remove( item: 6000));
System.out.println("6400 Occurence is not removed because it doesn't exist: " +tree.remove( item: 6400));
System.out.println("67777 Occurence is not removed because it doesn't exist: " +tree.remove( item: 67777));
System.out.println("232323 Occurence is not removed because it doesn't exist: " +tree.remove( item: 232323));
System.out.println("123232 Occurence is not removed because it doesn't exist: " +tree.remove( item: 123232));
System.out.println("145452 Occurence is not removed because it doesn't exist: " +tree.remove( item: 145452));
System.out.println("134342 Occurence is not removed because it doesn't exist: " +tree.remove( item: 134342));
System.out.println("123434 Occurence is not removed because it doesn't exist: " +tree.remove( item: 123434));
System.out.println("123443 Occurence is not removed because it doesn't exist: " +tree.remove( item: 123443));
System.out.println("1234434 Occurence is not removed because it doesn't exist: " +tree.remove( item: 1234434));
```

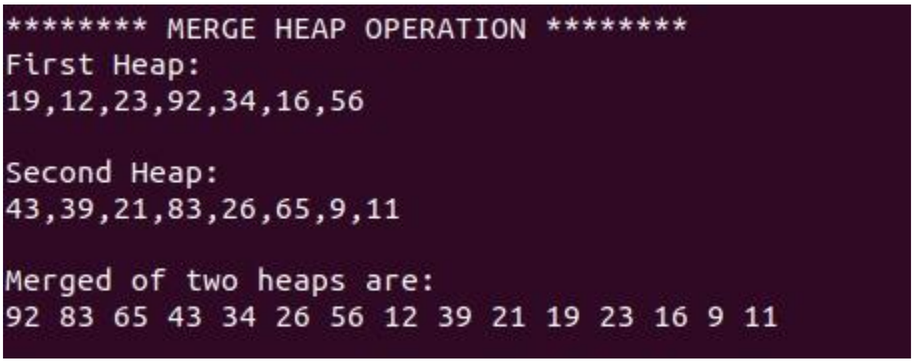
5. RUNNING AND RESULTS:



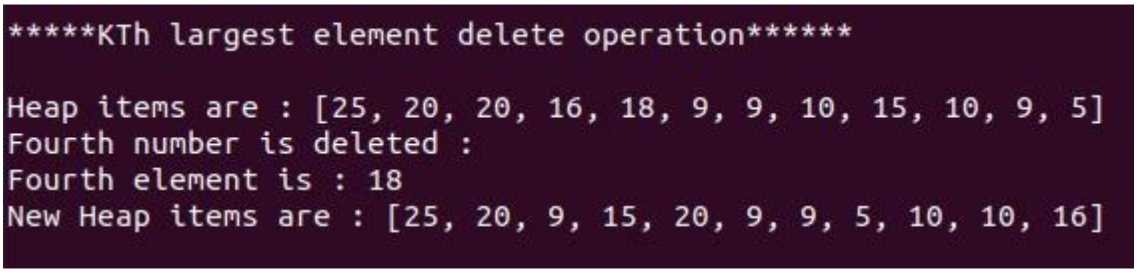
```
Heap Structure testing
=================================

*********** PART1 **********

Adding numbers to Heap.
- add((10)
- add((20)
- add((5)
- add((15)
- add((10)
- add((9)
- add((25)
- add((16)
- add((18)
- add((20)
- add((9)
- add((9)
--------------
Current Heap as a Sort:
[25, 20, 20, 16, 18, 9, 9, 10, 15, 10, 9, 5]
```



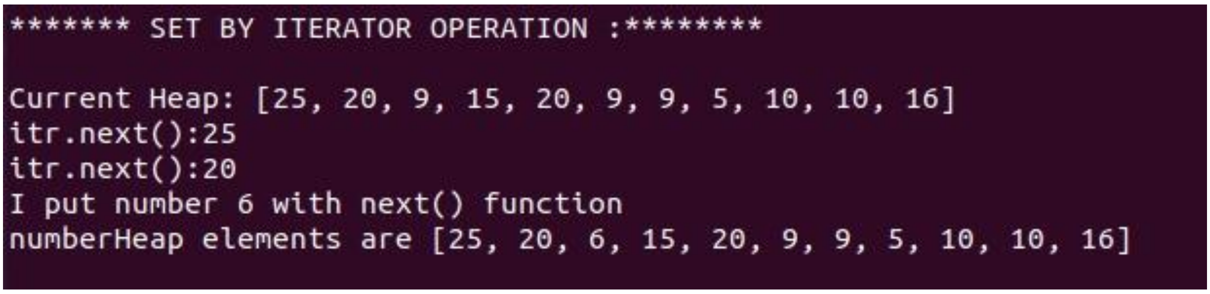
```
*********SEARCH OPERATION:************

- find(10)).toString()
10

- find((18).toString()
18

- find( (7).toString()
Given item has not been found in heap.
```



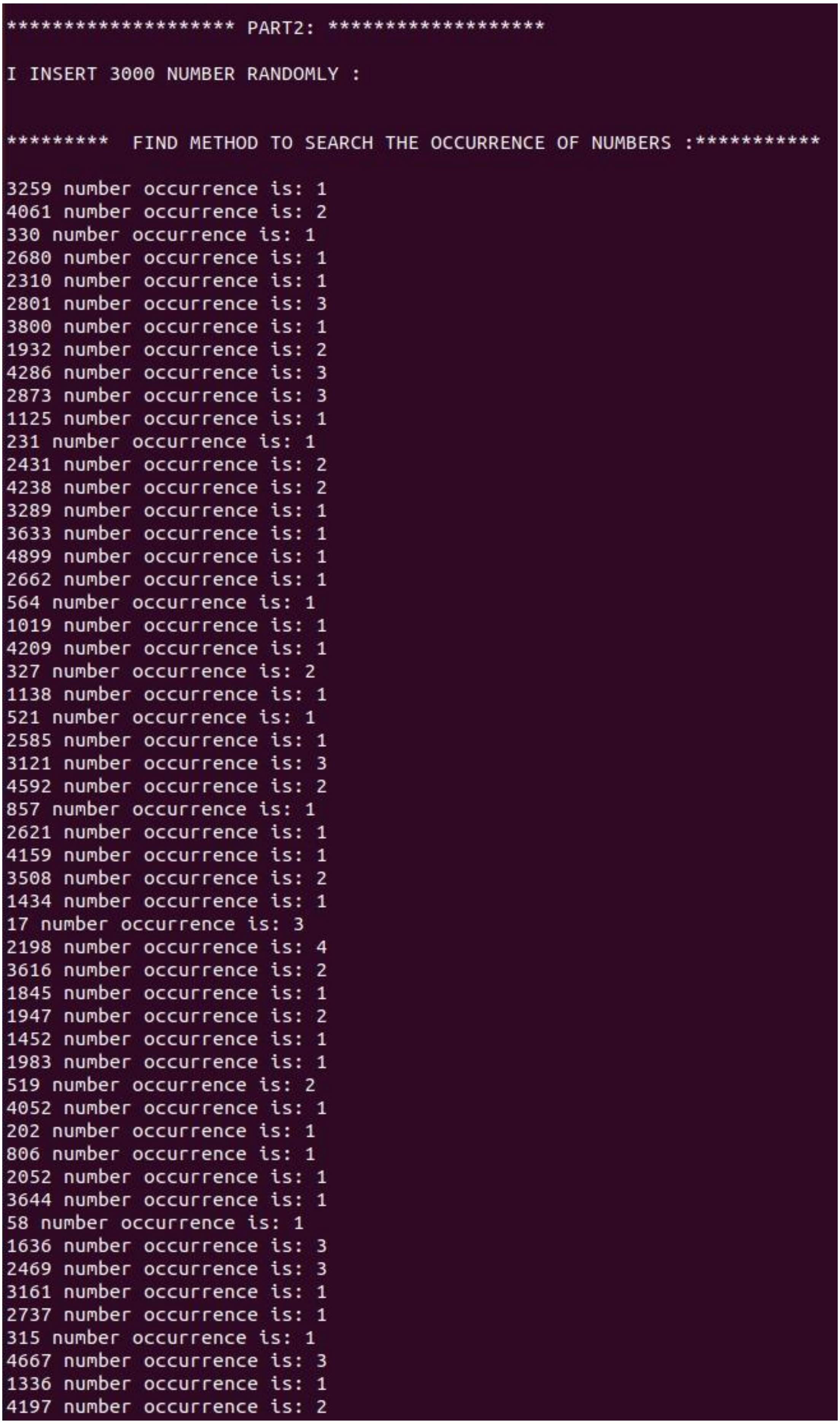
```
******** MERGE HEAP OPERATION ********
First Heap:
19,12,23,92,34,16,56

Second Heap:
43,39,21,83,26,65,9,11

Merged of two heaps are:
92 83 65 43 34 26 56 12 39 21 19 23 16 9 11
```



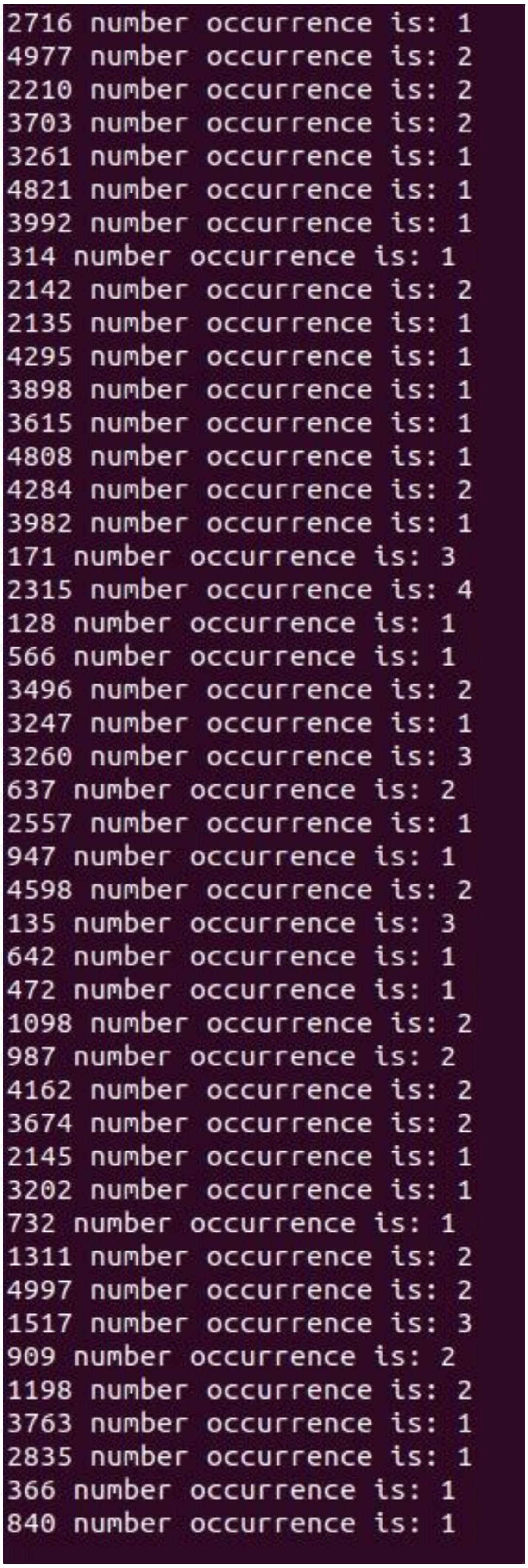
```
*****KTh largest element delete operation******

Heap items are : [25, 20, 20, 16, 18, 9, 9, 10, 15, 10, 9, 5]
Fourth number is deleted :
Fourth element is : 18
New Heap items are : [25, 20, 9, 15, 20, 9, 9, 5, 10, 10, 16]
```



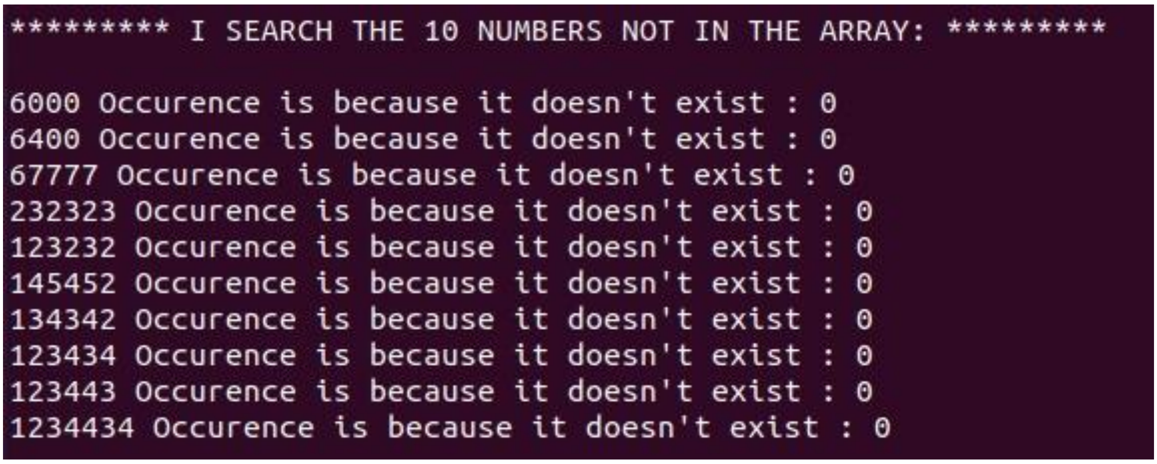
```
******* SET BY ITERATOR OPERATION :********

Current Heap: [25, 20, 9, 15, 20, 9, 9, 5, 10, 10, 16]
itr.next():25
itr.next():20
I put number 6 with next() function
numberHeap elements are [25, 20, 6, 15, 20, 9, 9, 5, 10, 10, 16]
```



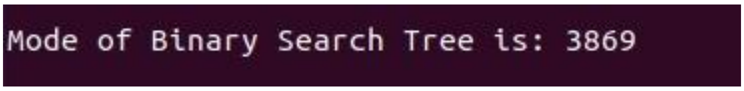
```
******************** PART2: *******************

I INSERT 3000 NUMBER RANDOMLY :


*********  FIND METHOD TO SEARCH THE OCCURRENCE OF NUMBERS :***********

3259 number occurrence is: 1
4061 number occurrence is: 2
330 number occurrence is: 1
2680 number occurrence is: 1
2310 number occurrence is: 1
2801 number occurrence is: 3
3800 number occurrence is: 1
1932 number occurrence is: 2
4286 number occurrence is: 3
2873 number occurrence is: 3
1125 number occurrence is: 1
231 number occurrence is: 1
2431 number occurrence is: 2
4238 number occurrence is: 2
3289 number occurrence is: 1
3633 number occurrence is: 1
4899 number occurrence is: 1
2662 number occurrence is: 1
564 number occurrence is: 1
1019 number occurrence is: 1
4209 number occurrence is: 1
327 number occurrence is: 2
1138 number occurrence is: 1
521 number occurrence is: 1
2585 number occurrence is: 1
3121 number occurrence is: 3
4592 number occurrence is: 2
857 number occurrence is: 1
2621 number occurrence is: 1
4159 number occurrence is: 1
3508 number occurrence is: 2
1434 number occurrence is: 1
17 number occurrence is: 3
2198 number occurrence is: 4
3616 number occurrence is: 2
1845 number occurrence is: 1
1947 number occurrence is: 2
1452 number occurrence is: 1
1983 number occurrence is: 1
519 number occurrence is: 2
4052 number occurrence is: 1
202 number occurrence is: 1
806 number occurrence is: 1
2052 number occurrence is: 1
3644 number occurrence is: 1
58 number occurrence is: 1
1636 number occurrence is: 3
2469 number occurrence is: 3
3161 number occurrence is: 1
2737 number occurrence is: 1
315 number occurrence is: 1
4667 number occurrence is: 3
1336 number occurrence is: 1
4197 number occurrence is: 2
```



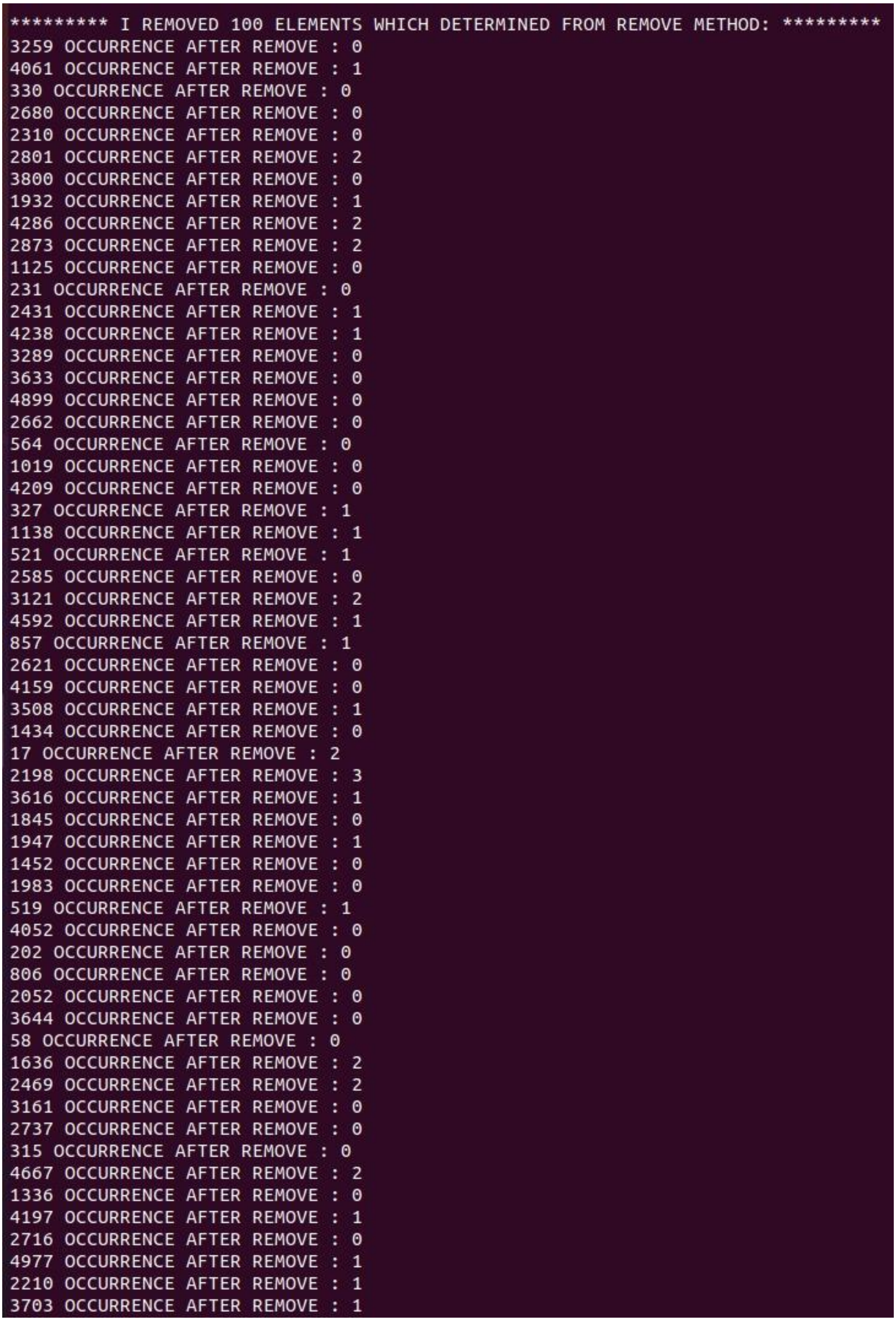
```
2716 number occurrence is: 1
4977 number occurrence is: 2
2210 number occurrence is: 2
3703 number occurrence is: 2
3261 number occurrence is: 1
4821 number occurrence is: 1
3992 number occurrence is: 1
314 number occurrence is: 1
2142 number occurrence is: 2
2135 number occurrence is: 1
4295 number occurrence is: 1
3898 number occurrence is: 1
3615 number occurrence is: 1
4808 number occurrence is: 1
4284 number occurrence is: 2
3982 number occurrence is: 1
171 number occurrence is: 3
2315 number occurrence is: 4
128 number occurrence is: 1
566 number occurrence is: 1
3496 number occurrence is: 2
3247 number occurrence is: 1
3260 number occurrence is: 3
637 number occurrence is: 2
2557 number occurrence is: 1
947 number occurrence is: 1
4598 number occurrence is: 2
135 number occurrence is: 3
642 number occurrence is: 1
472 number occurrence is: 1
1098 number occurrence is: 2
987 number occurrence is: 2
4162 number occurrence is: 2
3674 number occurrence is: 2
2145 number occurrence is: 1
3202 number occurrence is: 1
732 number occurrence is: 1
1311 number occurrence is: 2
4997 number occurrence is: 2
1517 number occurrence is: 3
909 number occurrence is: 2
1198 number occurrence is: 2
3763 number occurrence is: 1
2835 number occurrence is: 1
366 number occurrence is: 1
840 number occurrence is: 1
```



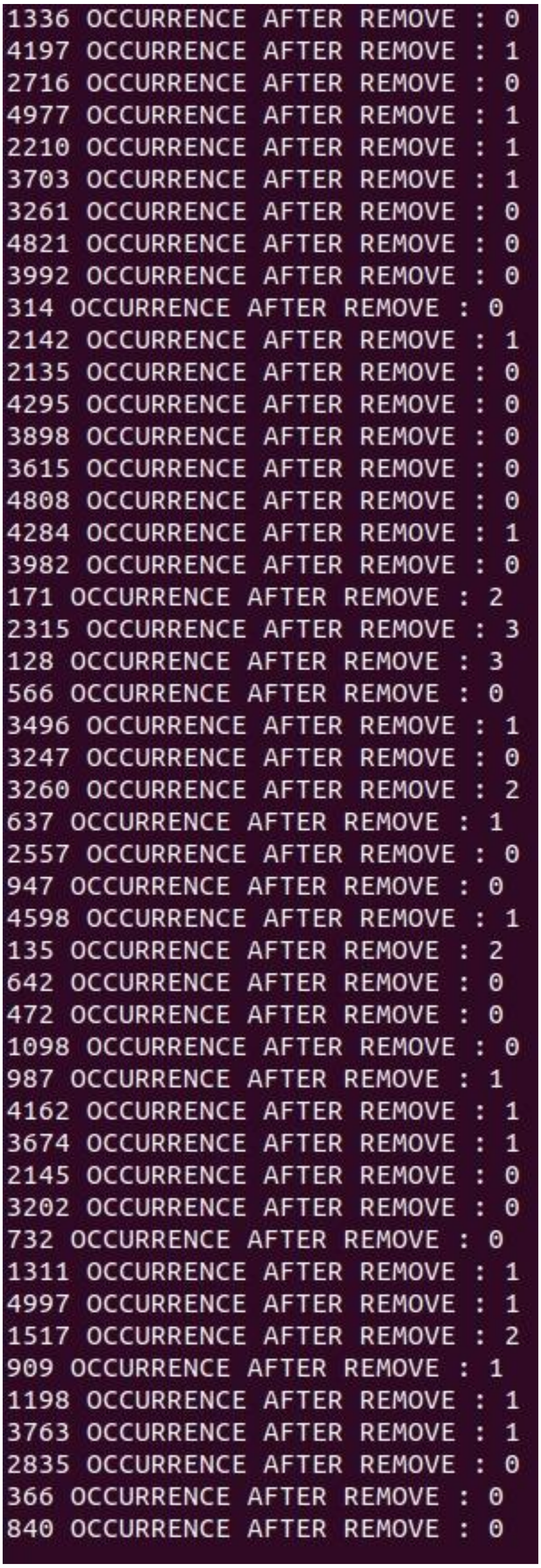
```
********* I SEARCH THE 10 NUMBERS NOT IN THE ARRAY: *********

6000 Occurence is because it doesn't exist : 0
6400 Occurence is because it doesn't exist : 0
67777 Occurence is because it doesn't exist : 0
232323 Occurence is because it doesn't exist : 0
123232 Occurence is because it doesn't exist : 0
145452 Occurence is because it doesn't exist : 0
134342 Occurence is because it doesn't exist : 0
123434 Occurence is because it doesn't exist : 0
123443 Occurence is because it doesn't exist : 0
1234434 Occurence is because it doesn't exist : 0
```



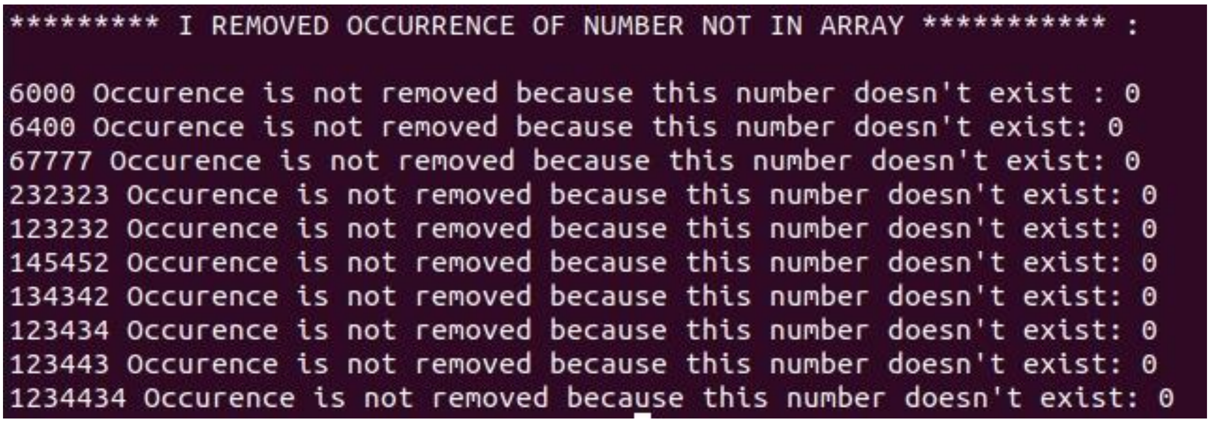
```
Mode of Binary Search Tree is: 3869
```



```
********* I REMOVED 100 ELEMENTS WHICH DETERMINED FROM REMOVE METHOD: *********
3259 OCCURRENCE AFTER REMOVE : 0
4061 OCCURRENCE AFTER REMOVE : 1
330 OCCURRENCE AFTER REMOVE : 0
2680 OCCURRENCE AFTER REMOVE : 0
2310 OCCURRENCE AFTER REMOVE : 0
2801 OCCURRENCE AFTER REMOVE : 2
3800 OCCURRENCE AFTER REMOVE : 0
1932 OCCURRENCE AFTER REMOVE : 1
4286 OCCURRENCE AFTER REMOVE : 2
2873 OCCURRENCE AFTER REMOVE : 2
1125 OCCURRENCE AFTER REMOVE : 0
231 OCCURRENCE AFTER REMOVE : 0
2431 OCCURRENCE AFTER REMOVE : 1
4238 OCCURRENCE AFTER REMOVE : 1
3289 OCCURRENCE AFTER REMOVE : 0
3633 OCCURRENCE AFTER REMOVE : 0
4899 OCCURRENCE AFTER REMOVE : 0
2662 OCCURRENCE AFTER REMOVE : 0
564 OCCURRENCE AFTER REMOVE : 0
1019 OCCURRENCE AFTER REMOVE : 0
4209 OCCURRENCE AFTER REMOVE : 0
327 OCCURRENCE AFTER REMOVE : 1
1138 OCCURRENCE AFTER REMOVE : 1
521 OCCURRENCE AFTER REMOVE : 1
2585 OCCURRENCE AFTER REMOVE : 0
3121 OCCURRENCE AFTER REMOVE : 2
4592 OCCURRENCE AFTER REMOVE : 1
857 OCCURRENCE AFTER REMOVE : 1
2621 OCCURRENCE AFTER REMOVE : 0
4159 OCCURRENCE AFTER REMOVE : 0
3508 OCCURRENCE AFTER REMOVE : 1
1434 OCCURRENCE AFTER REMOVE : 0
17 OCCURRENCE AFTER REMOVE : 2
2198 OCCURRENCE AFTER REMOVE : 3
3616 OCCURRENCE AFTER REMOVE : 1
1845 OCCURRENCE AFTER REMOVE : 0
1947 OCCURRENCE AFTER REMOVE : 1
1452 OCCURRENCE AFTER REMOVE : 0
1983 OCCURRENCE AFTER REMOVE : 0
519 OCCURRENCE AFTER REMOVE : 1
4052 OCCURRENCE AFTER REMOVE : 0
202 OCCURRENCE AFTER REMOVE : 0
806 OCCURRENCE AFTER REMOVE : 0
2052 OCCURRENCE AFTER REMOVE : 0
3644 OCCURRENCE AFTER REMOVE : 0
58 OCCURRENCE AFTER REMOVE : 0
1636 OCCURRENCE AFTER REMOVE : 2
2469 OCCURRENCE AFTER REMOVE : 2
3161 OCCURRENCE AFTER REMOVE : 0
2737 OCCURRENCE AFTER REMOVE : 0
315 OCCURRENCE AFTER REMOVE : 0
4667 OCCURRENCE AFTER REMOVE : 2
1336 OCCURRENCE AFTER REMOVE : 0
4197 OCCURRENCE AFTER REMOVE : 1
2716 OCCURRENCE AFTER REMOVE : 0
4977 OCCURRENCE AFTER REMOVE : 1
2210 OCCURRENCE AFTER REMOVE : 1
3703 OCCURRENCE AFTER REMOVE : 1
```



```
1336 OCCURRENCE AFTER REMOVE : 0
4197 OCCURRENCE AFTER REMOVE : 1
2716 OCCURRENCE AFTER REMOVE : 0
4977 OCCURRENCE AFTER REMOVE : 1
2210 OCCURRENCE AFTER REMOVE : 1
3703 OCCURRENCE AFTER REMOVE : 1
3261 OCCURRENCE AFTER REMOVE : 0
4821 OCCURRENCE AFTER REMOVE : 0
3992 OCCURRENCE AFTER REMOVE : 0
314 OCCURRENCE AFTER REMOVE : 0
2142 OCCURRENCE AFTER REMOVE : 1
2135 OCCURRENCE AFTER REMOVE : 0
4295 OCCURRENCE AFTER REMOVE : 0
3898 OCCURRENCE AFTER REMOVE : 0
3615 OCCURRENCE AFTER REMOVE : 0
4808 OCCURRENCE AFTER REMOVE : 0
4284 OCCURRENCE AFTER REMOVE : 1
3982 OCCURRENCE AFTER REMOVE : 0
171 OCCURRENCE AFTER REMOVE : 2
2315 OCCURRENCE AFTER REMOVE : 3
128 OCCURRENCE AFTER REMOVE : 3
566 OCCURRENCE AFTER REMOVE : 0
3496 OCCURRENCE AFTER REMOVE : 1
3247 OCCURRENCE AFTER REMOVE : 0
3260 OCCURRENCE AFTER REMOVE : 2
637 OCCURRENCE AFTER REMOVE : 1
2557 OCCURRENCE AFTER REMOVE : 0
947 OCCURRENCE AFTER REMOVE : 0
4598 OCCURRENCE AFTER REMOVE : 1
135 OCCURRENCE AFTER REMOVE : 2
642 OCCURRENCE AFTER REMOVE : 0
472 OCCURRENCE AFTER REMOVE : 0
1098 OCCURRENCE AFTER REMOVE : 0
987 OCCURRENCE AFTER REMOVE : 1
4162 OCCURRENCE AFTER REMOVE : 1
3674 OCCURRENCE AFTER REMOVE : 1
2145 OCCURRENCE AFTER REMOVE : 0
3202 OCCURRENCE AFTER REMOVE : 0
732 OCCURRENCE AFTER REMOVE : 0
1311 OCCURRENCE AFTER REMOVE : 1
4997 OCCURRENCE AFTER REMOVE : 1
1517 OCCURRENCE AFTER REMOVE : 2
909 OCCURRENCE AFTER REMOVE : 1
1198 OCCURRENCE AFTER REMOVE : 1
3763 OCCURRENCE AFTER REMOVE : 1
2835 OCCURRENCE AFTER REMOVE : 0
366 OCCURRENCE AFTER REMOVE : 0
840 OCCURRENCE AFTER REMOVE : 0
```



```
********* I REMOVED OCCURRENCE OF NUMBER NOT IN ARRAY *********** :

6000 Occurence is not removed because this number doesn't exist : 0
6400 Occurence is not removed because this number doesn't exist: 0
67777 Occurence is not removed because this number doesn't exist: 0
232323 Occurence is not removed because this number doesn't exist: 0
123232 Occurence is not removed because this number doesn't exist: 0
145452 Occurence is not removed because this number doesn't exist: 0
134342 Occurence is not removed because this number doesn't exist: 0
123434 Occurence is not removed because this number doesn't exist: 0
123443 Occurence is not removed because this number doesn't exist: 0
1234434 Occurence is not removed because this number doesn't exist: 0
```