GTU Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework 7 Report

Ömer Kaan Uslu
1801042642

1. **SYSTEM REQUIREMENTS**

1.Software Specification

Operating System : Windows 10 , macOS Catalina

Front End : Eclipse, Sublime Text

Rear End : Oracle SQL

Design Tool : UML

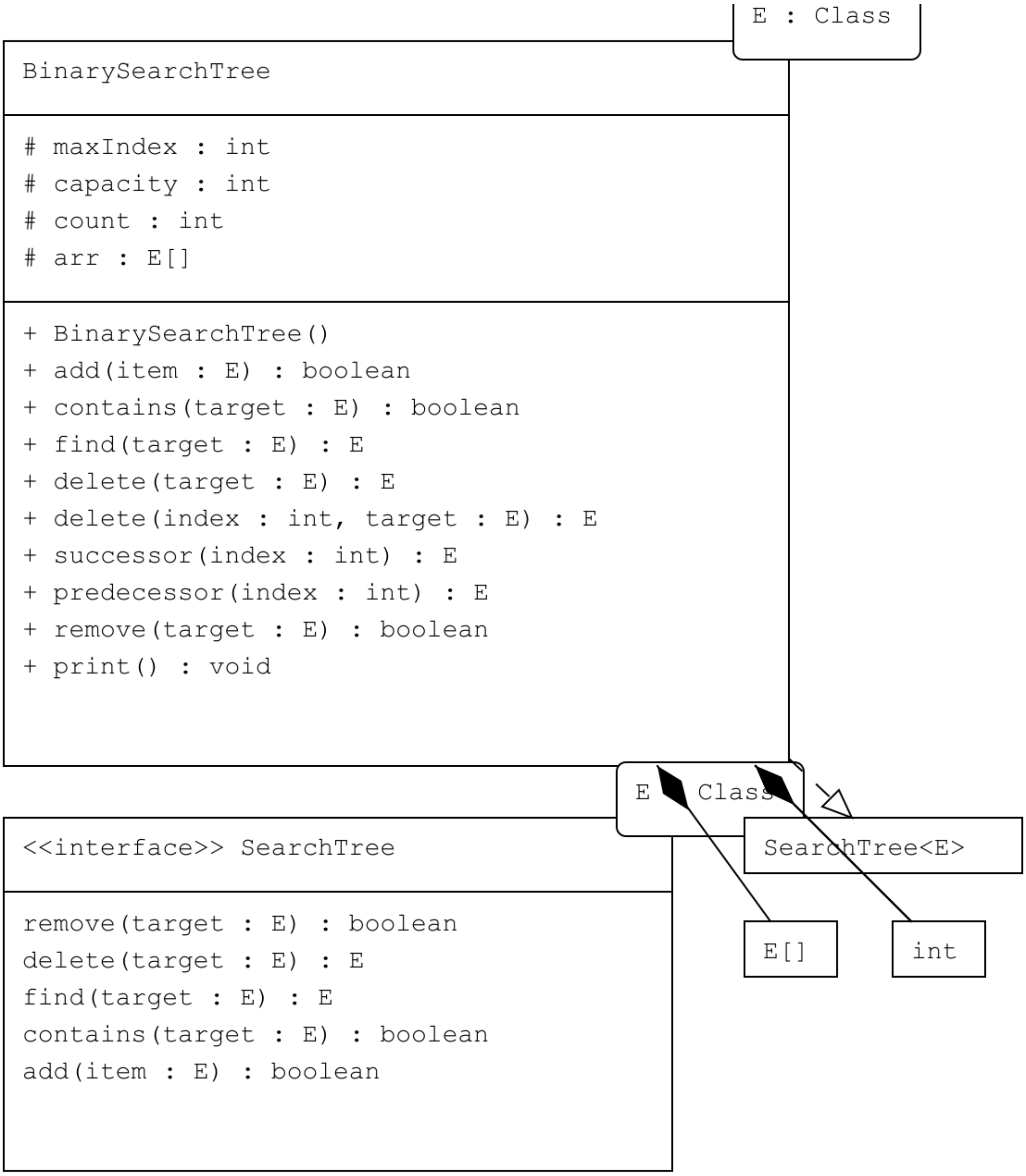
2.Hardware Specification

Processor : x86 processor

RAM : 512 MB or greater

Hard Disk : 20 GB or greater

2. **USE CASE AND CLASS DIAGRAMS**



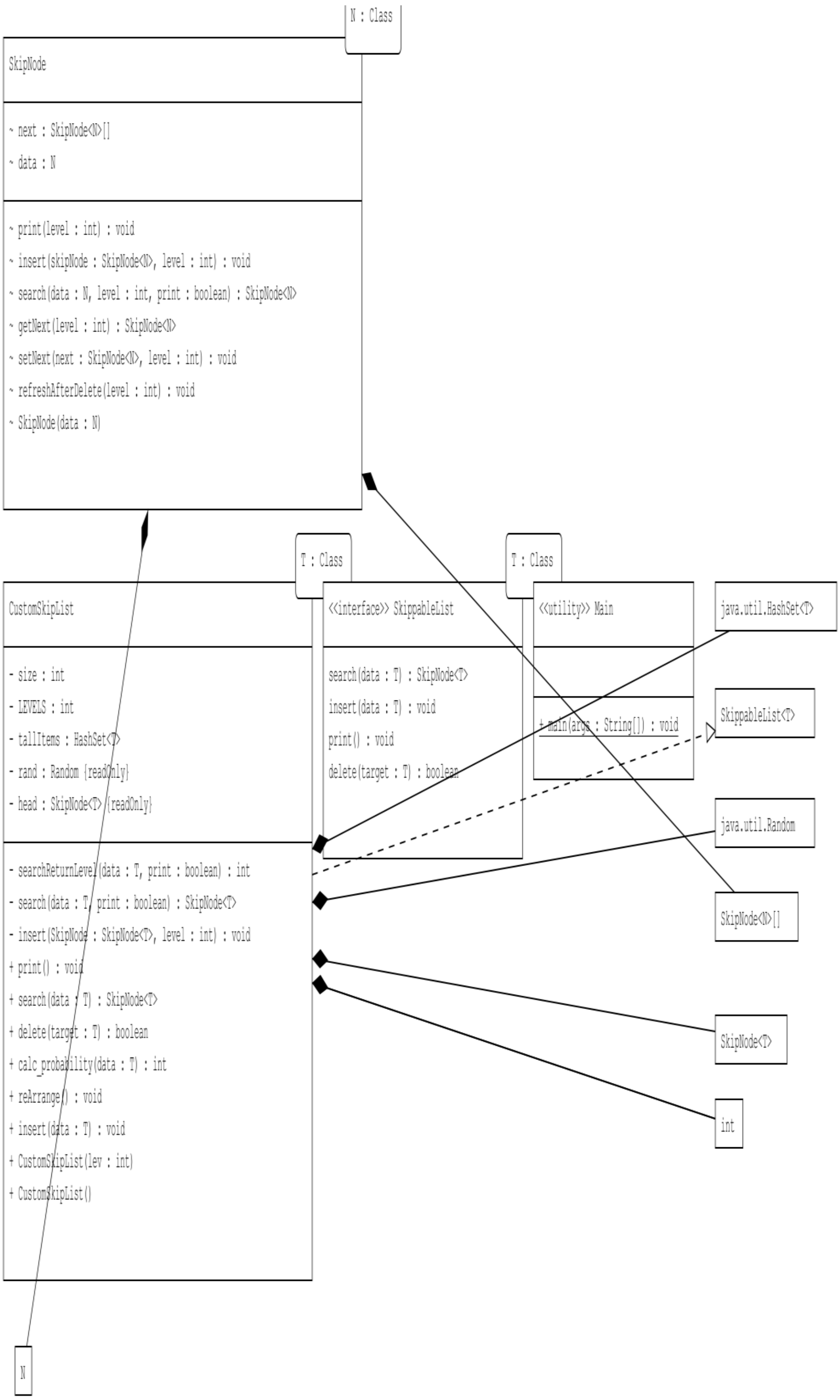
BinarySearchTree

maxIndex : int
capacity : int
count : int
arr : E[]

+ BinarySearchTree()
+ add(item : E) : boolean
+ contains(target : E) : boolean
+ find(target : E) : E
+ delete(target : E) : E
+ delete(index : int, target : E) : E
+ successor(index : int) : E
+ predecessor(index : int) : E
+ remove(target : E) : boolean
+ print() : void

<<interface>> SearchTree

remove(target : E) : boolean
delete(target : E) : E
find(target : E) : E
contains(target : E) : boolean
add(item : E) : boolean


N : Class
SkipNode
~ next : SkipNode<N>[]
~ data : N
~ print(level : int) : void
~ insert(skipNode : SkipNode<N>, level : int) : void
~ search(data : N, level : int, print : boolean) : SkipNode<N>
~ getNext(level : int) : SkipNode<N>
~ setNext(next : SkipNode<N>, level : int) : void
~ refreshAfterDelete(level : int) : void
~ SkipNode(data : N)
T : Class
CustomSkipList
- size : int
- LEVELS : int
- tallItems : HashSet<T>
- rand : Random {readOnly}
- head : SkipNode<T> {readOnly}
- searchReturnLevel(data : T, print : boolean) : int
- search(data : T, print : boolean) : SkipNode<T>
- insert(SkipNode : SkipNode<T>, level : int) : void
+ print() : void
+ search(data : T) : SkipNode<T>
+ delete(target : T) : boolean
+ calc_probability(data : T) : int
+ reArrange() : void
+ insert(data : T) : void
+ CustomSkipList(lev : int)
+ CustomSkipList()
<<interface>> SkippableList
search(data : T) : SkipNode<T>
insert(data : T) : void
print() : void
delete(target : T) : boolean
T : Class
<<utility>> Main
+ main(args : String[]) : void
java.util.HashSet<T>
SkippableList<T>
java.util.Random
SkipNode<N>[]
SkipNode<T>
int
N

3. PROBLEM SOLUTION APPROACH AND THEORETICAL RUN TIME ANALYSIS

In the first question, I wrote a method createBST. Method takes a structured binary tree and array. Method first searches for how many nodes on the right, and on the left of current
node. After it decides how many elements are bigger and smaller, it searches for element in array. In example if element is 'x', if array has 2 elements smaller than 'x' and it is total nodes of head's left side, it decides the element is 'x'. After putting 'x' in BST, array divides by 2 parts. One call been done for smallers, other call been done for biggers.
And right side and left side goes with it as a parameter.

Theoretical run time analysis: T(n) = 2T(n/2) + n^2*logN

In the second question, I wrote a method convert_avl. Method decides if head nodes' right and
left heights have difference bigger than 1. If it is, it decides the way tree should rotate.
If it right imbalanced, it does right rotation. Otherwise, it does left rotation.

Therotical run time analysis: T(n) = T(n-1) + O(n)
=O(n^2)

In the third question, it is based skiplist except 2 rule, First is upper level insertions
is done by tall distances of element. Left distance is starts with 0. Loop starts with head and gets next in every iteration, and if there is taller element, left distance set by 0,otherwise incremented by one. Right distance have same issue,if it finds taller element, breaks loop. and probability decided with this way.
Second rule is in every 10 element level incremented by one and taller elements get taller by 1 level. The array in SkipNode is reallocated , and taller elements' last level incremented by 1.

Thoretical run time analysis(Insertion): O(log(n)) in average case
Thoretical run time analysis(Level Up): O(n^2)

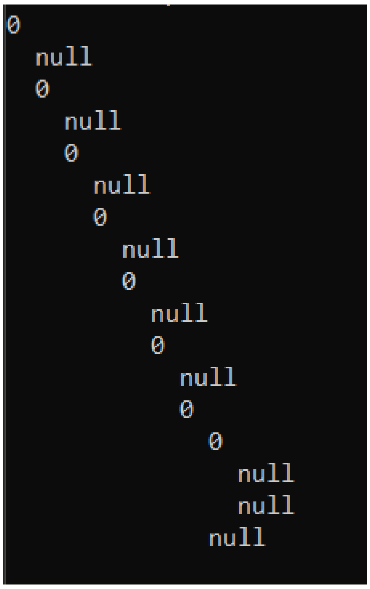
4. TEST CASES

Test Case No	Test Scenario	Test Steps	Test Data	Expected Result	Actual result	Pass/Fail
1	Return BST Of structured BT	Run driver code	Binary Tree, Array	Array’s elements is properly added.	As expected	Pass
2	Return AVL Version of BST	Run driver code	Binary Search Tree	BST converted to AVL	As expected	Pass

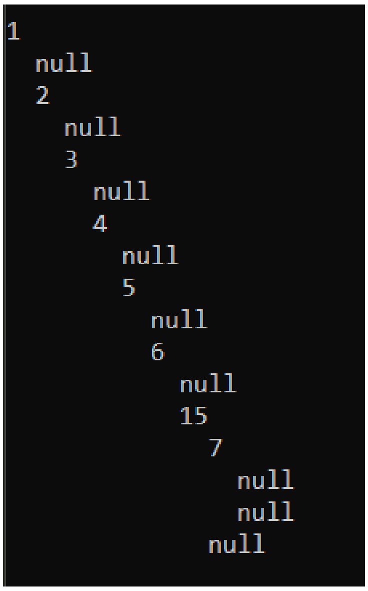
3	Add item to SkipList	Run driver code	4	4 is added	As expected	Pass
4	Remove item from SkipList	Run driver code	4	4 is added	As expected	Pass

5. **RUNNING AND RESULTS**

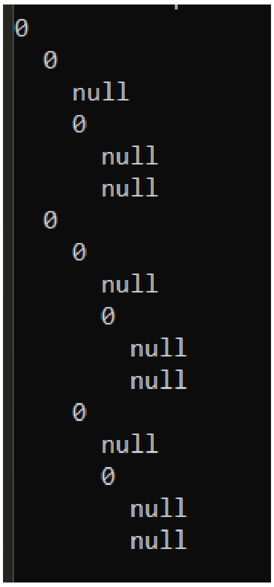
Q1) BT structure to BST



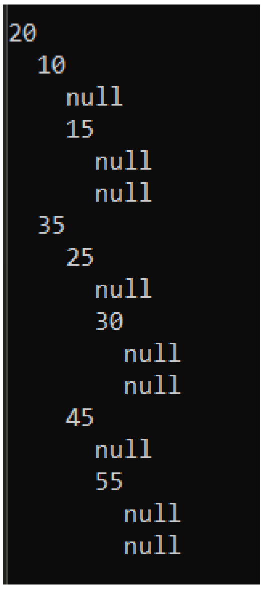
```
0
  null
  0
    null
    0
      null
      0
        null
        0
          null
          0
            null
            0
              0
                null
                null
              null
```



```
1
  null
  2
    null
    3
      null
      4
        null
        5
          null
          6
            null
            15
              7
                null
                null
              null
```

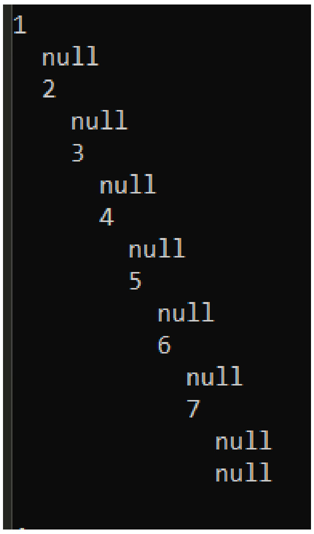


```
0
  0
    null
    0
      null
      null
  0
    0
      null
      0
        null
        null
    0
      null
      0
        null
        null
```

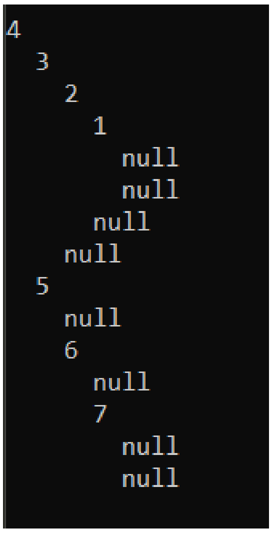


```
20
  10
    null
    15
      null
      null
  35
    25
      null
      30
        null
        null
    45
      null
      55
        null
        null
```

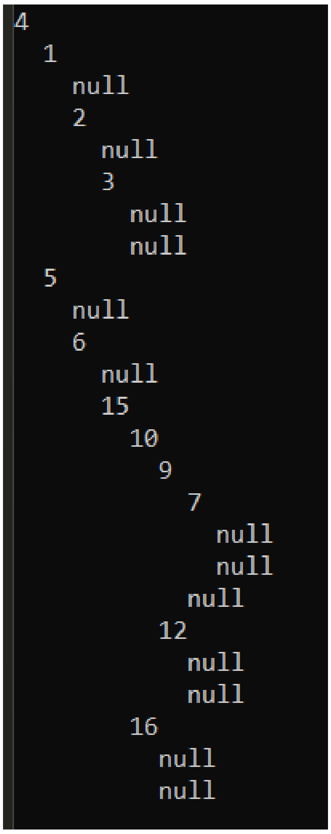
Q2)Convert AVL



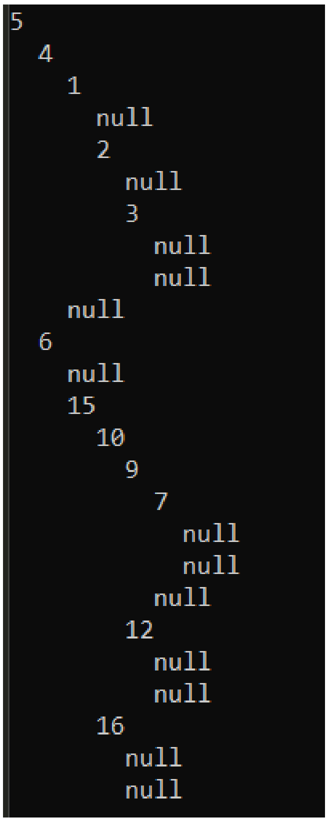
```
1
  null
  2
    null
    3
      null
      4
        null
        5
          null
          6
            null
            7
              null
              null
```



```
4
  3
    2
      1
        null
        null
      null
    null
  5
    null
    6
      null
      7
        null
        null
```

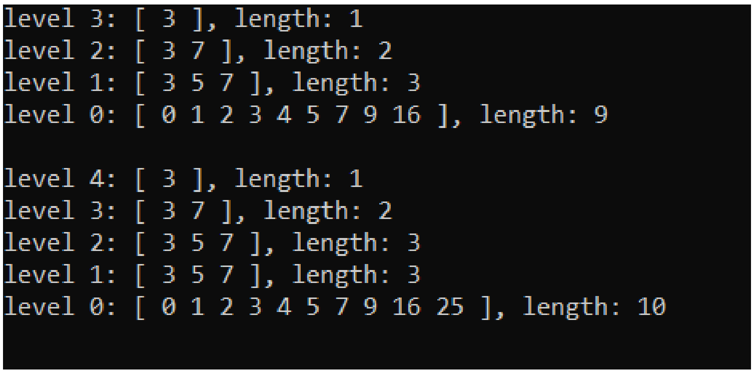


```
4
  1
    null
    2
      null
      3
        null
        null
  5
    null
    6
      null
      15
        10
          9
            7
              null
              null
            null
          12
            null
            null
        16
          null
          null
```



```
5
  4
    1
      null
      2
        null
        3
          null
          null
    null
  6
    null
    15
      10
        9
          7
            null
            null
          null
        12
          null
          null
      16
        null
        null
```

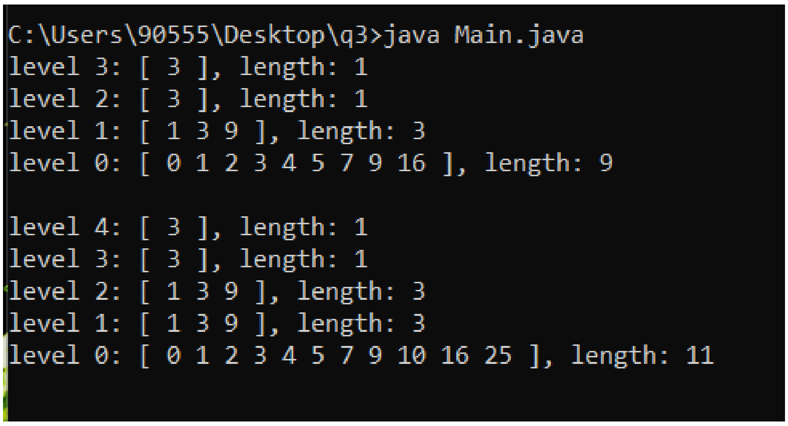
Q3)SkipList



```
level 3: [ 3 ], length: 1
level 2: [ 3 7 ], length: 2
level 1: [ 3 5 7 ], length: 3
level 0: [ 0 1 2 3 4 5 7 9 16 ], length: 9

level 4: [ 3 ], length: 1
level 3: [ 3 7 ], length: 2
level 2: [ 3 5 7 ], length: 3
level 1: [ 3 5 7 ], length: 3
level 0: [ 0 1 2 3 4 5 7 9 16 25 ], length: 10
```

After 10th element(25) is added, level is increased by 1.



```
C:\Users\90555\Desktop\q3>java Main.java
level 3: [ 3 ], length: 1
level 2: [ 3 ], length: 1
level 1: [ 1 3 9 ], length: 3
level 0: [ 0 1 2 3 4 5 7 9 16 ], length: 9

level 4: [ 3 ], length: 1
level 3: [ 3 ], length: 1
level 2: [ 1 3 9 ], length: 3
level 1: [ 1 3 9 ], length: 3
level 0: [ 0 1 2 3 4 5 7 9 10 16 25 ], length: 11
```

After 10th element(25) and 11th element(10) is added, level is increased by 1.