Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 6 REPORT

DENİZ BABAT
131044013

Course Assistant: Fatma Nur Esirci

1 Worst RedBlack Tree

Worst case oluşturmak için sıralı bir dizi RedBlack Tree'ye ekledim. Fakat RedBlack Tree sürekli kendini dengelediği için worstcase oluşturmak biraz zor oluyordu. Fakat ağacı sürekli sağa veya sola doğru ağırlığı arttımak worst case bulabildim. Böylece worst case elde ettim.

1.1 Problem Solution Approach

Write pseudocode and explanation about code design. Indicate what you are using that interfaces, classes, structures, etc.

Burada tam olarak yazmak mümkün değildir. Çünkü Kitabın kaynak kodu kullanılması gerektiği için ve kodda herhangi bir eksik olmadığı için ve ödevde herhangi bir metod implement etmemizi istemediğiniz için bit psuedocode yazmak mümkün değildir. Fakat worst case nasıl oluşturduğumu yukarıda yazdım. BinaryTree, BinarySearchTree, BinarySearchTreeWithRotate, RedBlackTree class'ı ve SearchTree interface'i kullanıldı.

1.2 Test Cases

`public static void test1():` foksiyonları Main class'ında yazıldı. Worst case bir RedBlack tree oluşturup ekrana basıyor.

`public static void test2():` foksiyonları Main class'ında yazıldı. Worst case bir RedBlack tree oluşturup ekrana basıyor.

1.3 Running Commands and Results

Test durumları için iki farklı RedBlack tree oluşturdum ve her durum ekrana basıldı. Fakat aşağıda enson durumları ekrana veriyorum.

- TEST 1: Array dizisi = {1,2,3,4,5,6,7,8,9,10,11,12,13,14}

```
Black: 3
  Black: 1
    Black: 0
      null
      null
    Black: 2
      null
      null
  Red: 7
    Black: 5
      Black: 4
        null
        null
      Black: 6
        null
        null
```

```
  Black: 9
    Black: 8
      null
      null
    Red: 11
      Black: 10
        null
        null
      Black: 13
        Red: 12
          null
          null
        Red: 14
          null
          null
```

- TEST 2: Array dizisi = {14,13,12,11,10,9,8,7,6,5,4,3,2,1}

```
Black: 12
  Red: 8
    Black: 6
      Red: 4
        Black: 2
          Red: 1
            null
            null
          Red: 3
            null
            null
        Black: 5
          null
          null
      Black: 7
        null
        null
    Black: 10
      Black: 9
        null
        null
      Black: 11
        null
        null
  Black: 14
    Black: 13
      null
      null
    Black: 15
      null
      null
```

2 binarySearch method

2.1 Problem Solution Approach

Kitabın kaynak kodu ve SearhTree inteface'i kullanıldı.

Psuedocode:

int binarySearch (E item, E[] data, int zero, int rootsize)

1. int leftmost = zero
2. int right = rootsize-1
3. while leftmost < right //burada 3'ten büyük durumlar için while ile dolaşılıp, index bulunuyor.
    1. index = (leftmost+right)/2 *// burada node.data'nın ortadaki index'i hesaplanıyor.*
    2. compare = item.compare(data[index]) *//hesaplanan index ile karşılaştırılıyor.*
    3. if compare == 0
        1. Return index
    4. else if compare < 0
        1. Return index-1;
    5. else lefmost = index + 1
4. end while
5. return leftmost *//Return leftmost*

2.2 Test Cases

`public static void test1():` fonksiyonu yazıldı ve ekrana eklenen ve eklenmek için herbir sayı ağaçta aranıldı.

`public static void test2():` fonksiyonu yazıldı ve ekrana eklenen ve eklenmek için herbir sayı ağaçta aranıldı.

2.3 Running Commands and Results

```
TEST 1
22
 2
  1
   null
   null


  8, 11
   null
   null
   null
```

```
  54, 65
    26, 36
      null
      null
      null


    62
      null
      null


    425
      null
      null
```

Tree de olan sayıları arayalım bulabilecek miyiz?
1 is searching: 1
22 is searching: 22
65 is searching: 65
Tree de olmayan bir sayı arayalım bulabilecek miyiz?
18 is searching: null

TEST 2

```
23
  11
    1, 5
      null
      null
      null


    12
      null
      null



  88, 226
    25, 33
      null
      null
      null


    220
      null
      null


    725
```

```
null
null


Tree de olan sayıları arayalım bulabilecek miyiz?
12 is searching: 12
33 is searching: 33
11 is searching: 11
Tree de olmayan bir sayı arayalım bulabilecek miyiz?
785 is searching: null
```

3 Project 9.5 in book

3.1 Problem Solution Approach

Write pseudocode and explanation about code design. Indicate what you are using that interfaces, classes, structures, etc.

1) BinarySearchTree, AVLTree olup olmadığını bulmak için tree'nin yükesekliğini bularak hesapladım. Sağ ve Sol ğağların yükseklikleri eşitse böylece bir AVLTree oluduğunu buldum.
   Psuedocode:

   i. AVLTree(BinarySearchTree<E> binarySearchTree)
   ii. if isBalanced(binarySearchTree.root)
      a. print Binary Search Tree is AVLTree
   iii. else print Binary Search Tree is not AVLTree

   i. boolean isBalanced(Node<E> local)
   ii. if local == null return null;
   iii. leftsubtreeheight = height(local.left)
   iv. rightsubtreeheight = height(local.right)
   v. if Math.abs(leftsubtreeheight - righsubtreeheight) ve isBalanced(local.left) ve isBalanced(local.left) return true;
   vi. else retrun false;

   i. int height(Node<E> node)
   ii. if node == null return 0;
   iii. return 1 + Math.max(height(node.left), height(node.right));

   Böylece Tree'nin yüksekliği hesaplanıp, AVLTree olup olamdığı bulunmuştur.

2) AVL Tree tamamlandı. Test case de tüm durumların test edildiği açıklanmıştır.

3.2 Test Cases

doesBinaryTreeisAvlTree(); Main de yazılmış bu fonksiyon Binarysearhtree'nin avl tree olup olmadığını buluyor.

avltreetestfunc(); Main'de yazılmış bu fonksiyonda AVLTree'nin 4 durumunu (**Left Left Case, Left Right Case, Right Right Case, Right Left Case**) test etmektedir. Aşağıda ekran görüntüleri mevcuttur.

3.3 Running Commands and Results

Show that test case results using screenshots.

1) BinarySearhTree'nin AVLTree olup olmadığını gösteren ekran alıntıları;

```
"C:\Program Files\Java\jdk1.8.0_131\bin\java" ...
Dengeli bir BinarySearchTree oluşturup AVLTree'nin Constructurına verip AVLTreeolup olmadığını bulacağım
10
  5
    4
      null
      null
    6
      null
      null
  15
    12
      null
      null
    17
      null
      null

Dengeli bir Tree oluşturuldu.Şimdi AVLTRee'nin Constructorına veriyorum: SONUÇ;
Binary Search Tree is AVLTree
================================================================================================================
----------------------------------------------------------------------------------------------------------------
Şimdi de dengesiz bir binarySearhTree oluşturup AVLTree olup olmadığına bakalım.
1
  null
  2
    null
    3
      null
      4
        null
        5
          null
          6
            null
            7
              null
              8
                null
                9
                  null
                  null

Dengesiz bir Tree oluşturuldu.Şimdi AVLTRee'nin Constructorına veriyorum: SONUÇ;
Binary Search Tree is not AVLTree
```

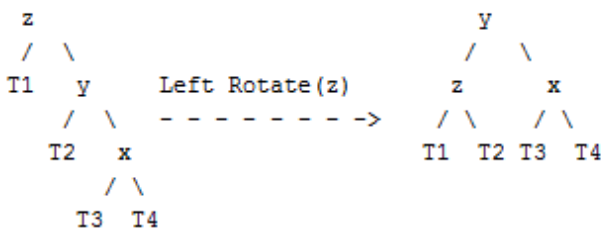
2) AVLTree'nin case'lerinin ekran alıntıları;

```
"C:\Program Files\Java\jdk1.8.0_131\bin\java" ...
A)  Left Left Case
T1, T2, T3 and T4 are subtrees.
         z                                      y
        / \                                    /   \
       y   T4      Right Rotate (z)           x      z
      / \          - - - - - - - - - ->      /  \    /  \
     x   T3                                T1  T2  T3  T4
    / \
  T1   T2
Şimdi böyle bir Tree oluşturacam Ama sonucu ekranda gösterebilirim. Hocam anlayın. (*_*)
0: 7
  0: 4
    0: 2
      null
      null
    0: 5
      null
      null
  0: 10
    0: 8
      null
      null
    0: 11
      null
      null

Görüldüğü gibi dengededir.
================================================================================================
B) Left Right Case
    z                               z                           x
    / \                            /   \                       /  \
   y   T4  Left Rotate (y)        x    T4  Right Rotate(z)    y      z
  / \      - - - - - - - - - ->   /  \      - - - - - - - - ->  / \    / \
T1   x                          y    T3                    T1  T2 T3  T4
    / \                        / \
  T2   T3                    T1   T2
Şimdi böyle bir Tree oluşturacam Ama sonucu ekranda gösterebilirim.
0: 8
  0: 5
    0: 2
      null
      null
    0: 7
      null
      null
  0: 10
    0: 9
      null
      null
    0: 11
      null
      null

Görüldüğü gibi dengededir.
================================================================================================
```

C) Right Right Case



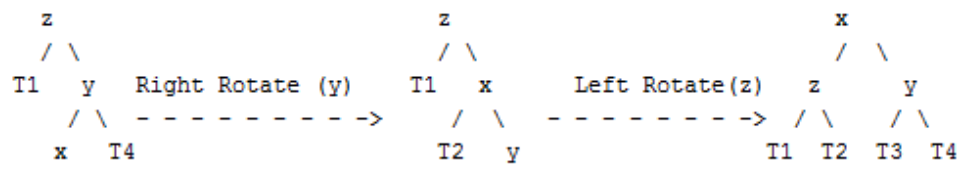
Şimdi böyle bir Tree oluşturacam Ama sonucu ekranda gösterebilirim.

```
0: 15
  0: 5
    0: 4
      null
      null
    0: 13
      null
      null
  0: 18
    0: 17
      null
      null
    0: 19
      null
      null
```

Görüldüğü gibi dengededir.

==========

D) Right Left Case



Şimdi böyle bir Tree oluşturacam Ama sonucu ekranda gösterebilirim.

```
0: 10
  0: 5
    0: 4
      null
      null
    0: 8
      null
      null
  0: 15
    0: 12
      null
      null
    0: 17
      null
      null
```

Görüldüğü gibi bu tree de dengededir. Yani DURUMLAR BAŞARILIDIR.

==========

```
Process finished with exit code 0
```

Yukarıda ekran alıntıları verilmiştir.