**Gebze Technical University**

**Computer Engineering**

**CSE 222 - 2018 Spring**

**HOMEWORK 6 REPORT**

**AHMET MUZAFFER DÜLGER**

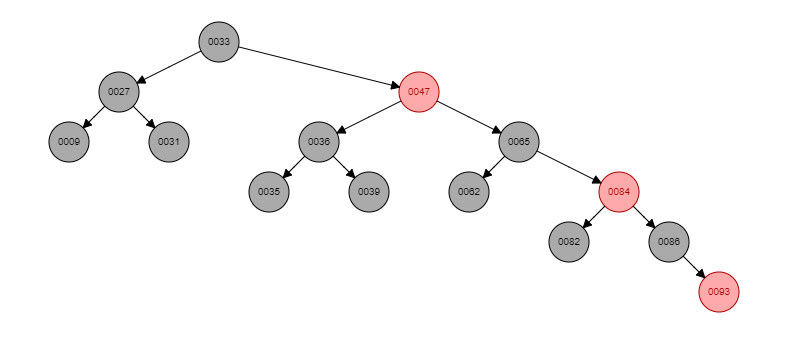
**131044082**

Course Assistant: Fatma Nur Esirci

# Worst RedBlack Tree

## Problem Solution Approach

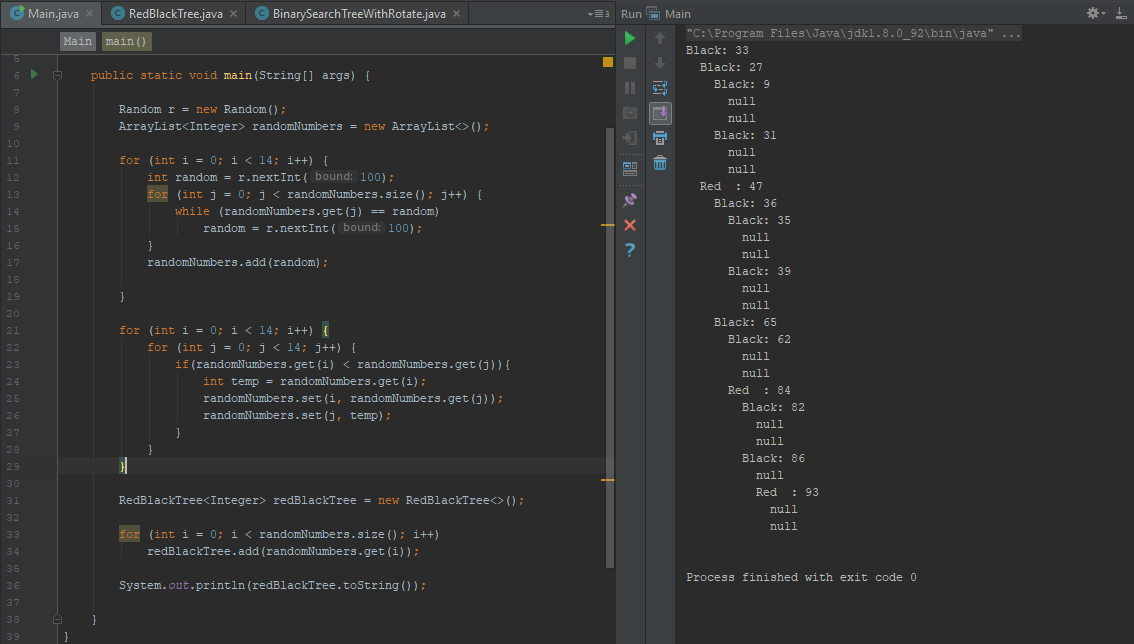
Bu partta yüksekliği 6 olan en kötü Red Black Tree oluşturmamız istenildi. Bunun için ilk önce main de 14 adet random sayı ürettim ve ardından bu random sayıları küçükten büyüğe doğru sıraladım. Bunu yapmamdaki sebep ise 6 yüksekliğindeki en kötü RedBlack Tree’nin küçükten büyüğe veya büyükten küçüğe doğru sıralama yapıldığında oluşmasıdır. Main kodu her çalıştığında farklı bir tree iplement edilecektir.



## Test Cases

14 adet random sayı (9,27,31,33,35,36,39,47,62,65,82,84,86,93) üretiliyor. Bu random sayılar küçükten büyüğe doğru sıralanıp AVLTree objesine tek tek ekleniyor. Oluşturulan ağacın son hali bölüm 1.3’deki gibidir.

## Running Commands and Results



# binarySearch method

## Problem Solution Approach

Bu partta insert metodu içerisinde kullanılan binarySearch() metodunu implement etmemiz isteniyordu. Bu metot Binary Search Tree’de kullandığımız search metoduna çok benziyor ve yine aynı şekilde recursive mantık ile çalışıyor. Mantık olarak aynılar. Tek farkları burada first ve last olarak verdiğimiz aralıklarda arama yapması.

Bu metot 4 adet parametre alıyor. İlk parametre E tipinde item; aranacak öğe. İkinci parametre E[] tipinde data; item bu sıralanmış biçimde duran datalar içerisinde aranır. Üçüncü parametre int tipinde first; aramanın nereden başlayacağını gösterir. Son parametre ise int tipinde last; aramanın nerede biteceğini gösterir.

İlk olarak first ve last’ın birbirine eşitlik durumu kontrol edilir. Eğer eşit iseler first return edilir. Bu, eleman olmadığını gösterir. Eğer ki last, first’en 1 büyük ise item ve array’in first indexinde bulunan elaman karşılaştırılır ve gelen değer 0’dan küçükse first değilse last return edilir. Küçükse aradığımız item ağacın sol kısmında, aksi durumda da sağ kısmında kalıyor demektir.

Daha sonra middle isminde bir integer değer tanımladım ve bu değer ile ortada bulunan kısmı temsil ettim. Ardından data’nın middle indexinde bulunan değer ile item’i karşılaştırdım ve bu karşılaştırmanın sonucunda 0 değeri geliyorsa ise bu ikisi eşit demektir ve middle’ı return ettim. Eğer karşılaştırma sonucunda 0’dan küçük bir değer geliyor ise ağacın solunda arama yapmam gerekiyor demektir. Bu yüzden tekrar binarySearch metodunu çağırdım. Bu çağırmaya parametre olarak sırasıyla item, data, first ve middle’ı gönderdim. Eğer karşılaştırma sonucunda 0’dan büyük bir değer geliyor ise ağacın sağında arama yapmam gerekiyor demektir. Bu yüzden tekrar binarySearch metodunu çağırdım. Bu çağırmaya parametre olarak sırasıyla item, data, middle + 1 ve last’ı gönderdim.

## Test Cases

Elle eklediğim integer değerlerden 3 order’a sahip bir BTree oluşturuluyor.

## Running Commands and Results

# Project 9.5 in book

## Problem Solution Approach

Bu partta decrementBalance, incrementBalance, rebalanceleft ve rebalanceRight metotlarının tekrar yazılmasının yanı sıra parametre olarak BinaryTree alan constructor yazmamız isteniyordu. Bu constructor parametre olarak gelen tree’nin AVL Tree olup olmadığının kontrolünü yapıyor.

Constructor’ı yaparken maxDepth ve minDepth olmak üzere iki adet yardımcı metot kullandım. Her iki metotda recursion mantıkta çalışmaktadır. maxDepth, yüksekliği en büyük olan node’un yüksekliğini, minDepth ise yüksekliği en düşük olan node’un yüksekliğini return eder.

Constructor içerisinde ilk önce bu iki metodu çağırıp farklarını aldım. Bunu yapmamdaki amaç ise AVLTree olması için en düşük node ile en yüksek node’un yüksekliklerinin farkını 1 veya 0 olmasıdır. Eğer farkları 1 veya 0 ise AVLTree olduğuna dair ekrana bilgi mesajı bastırıyorum. Eğer bu değerlerden başka bir şey geliyor ise AVLTree olmadığına dair ekrana bilgi mesajı bastırıyorum.

## Test Cases

Birinci test de AVLTree olan bir BinaryTree test edilmiştir.

İkinci test de ise AVLTree olmayan bir BinaryTree test edilmiştir.

## Running Commands and Results

