Date: 18/05/2019

Title: Balanced Search Trees , Hashing and Graphs

Author: Zeynep Cankara

ID: 21703381

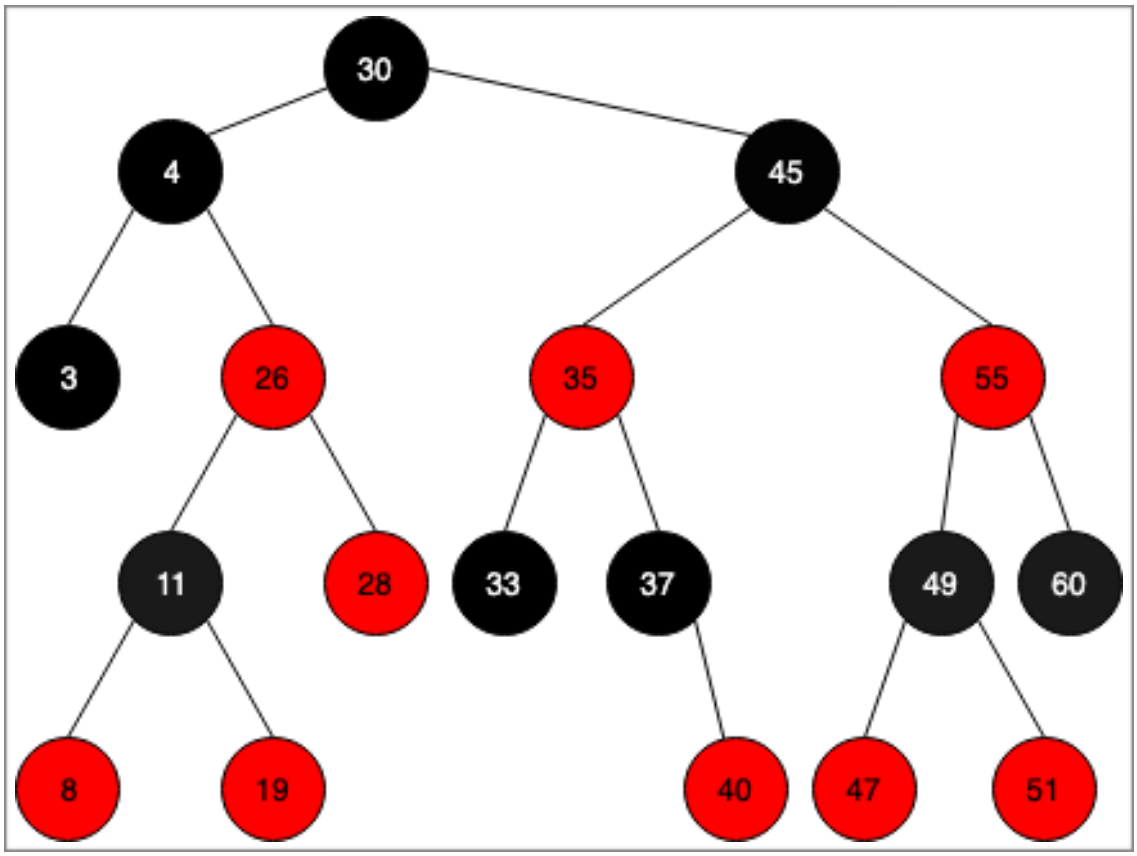
Section: 2

Assignment: 4

Description: My Solutions for Question 1 Part (a) & (b), Question 2 and Question 3 (a) & (b) & (c)

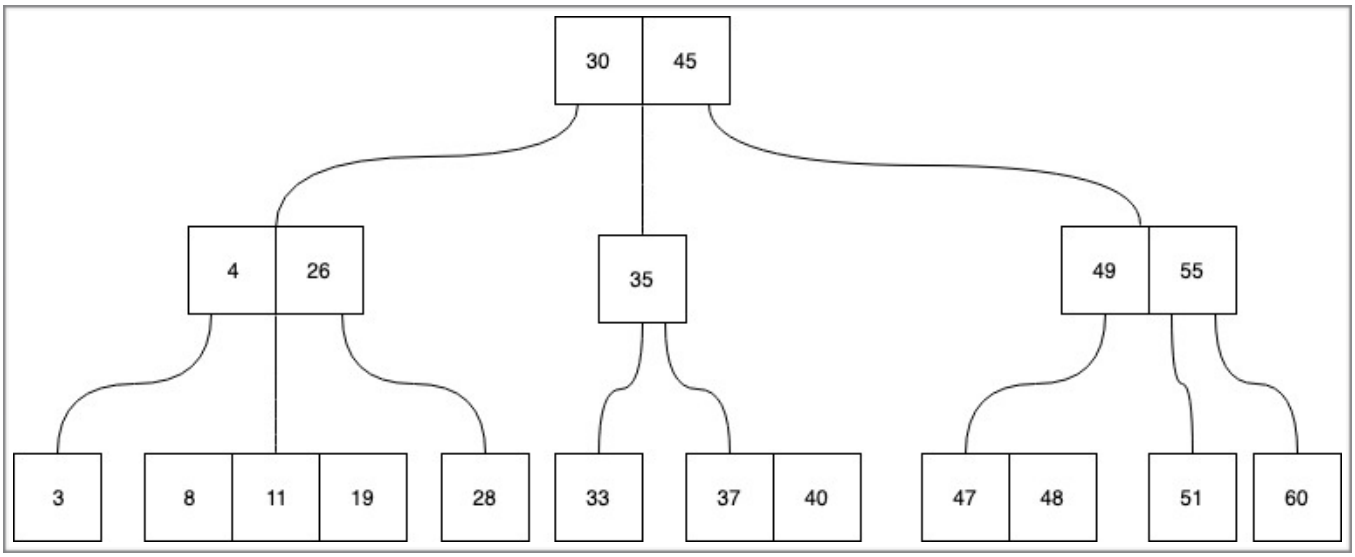
Question1: Part (a)

Representing 2-3-4 tree as a Red-Black Tree



Question1: Part (b)

2-3-4 Tree after inserting 48



Question2:

Data Structure	Insert	ExtractMin
Unsorted array	O(1)	O(n)
Red-black tree	O(log(n))	O(log(n))
Hashing	O(1)	O(n)
Min-heap	O(log(n))	O(log(n))
Sorted linked list (ascending)	O(n)	O(1)

Question3: Part (a)

- $3^h - 1$ is the maximum number of elements 2-3 tree with height h can contain. The result obtained from the balance condition 2-3 tree has and maximum number of elements a node can contain is 2 with 3 children.
- Proof: $\sum_{i=0}^{h} (2 * 3^h) = 3^h - 1$ (from sums of geometric sequences and series)

Question3: Part (b)

- **No**, First of all from the definition of Red-Black Trees, a **red-black tree must have a black root** node and all external nodes should have the **same number of black pointers on the path from the root to the external node.**

Question3: Part (c)

- Initialize an empty hash-table.
- Start iterating through the array. Check whether (target - current item) in the hash-table. If the complement value (target - current item) not in the hash table add current element in the hash table.
- If (target - current item) in the hash-table you can return pair of elements.
- Searching a value in a proper hash-table O(1) time and iterating the array takes O(n) time taking total time complexity O(n)