**Red-Black trees**

**They are a specific type of balanced tree**

1. **How to tell if a tree is red-black tree?**A red-black tree (or RB-tree) is a binary search tree with the following properties.
   1. Each node is either black or red.
   2. The root node is black.
   3. Every red node has only black children.
   4. For each node k, each path from k to a leaf contains the same number of black nodes ("black depth" of all leaves must be equal)

A black and white diagram with white text

Description automatically generated

1. Is a red-black tree
2. Not a binary tree:
   1. red node 7 has a red child (6)
   2. “Black depth” of all the leaves is not the same
3. Not a binary tree, as root rode must be black

Three Main Operations:

Search

Insert

Remove

Open Questions

* 1. How to calculate the black height of a tree or even of a specific node
  2. Are there standard ways of rotating left and doing rotation on right
  3. A diagram of a line with red circles and a red circle with black text

     Description automatically generated with medium confidence
     1. Why can’t i directly change then the color of u to red and the color of p x to black, would this solve the problem, or is the issue is that p should have two blacks, if that is true, then i can change any node’s color at anytime or first i need to do the rotation
     2. Also at the same time in to make sure that the black height rule is been followed with also, so how can i make sure that, Prof. slides he has someways/cases of solving these problems are the steps like an algorithm that will solve most of the cases

**Binary Trees**

A binary tree is a tree, where each node can have at most two children. We consider here ordered trees, i.e. we distinguish between the left and can right child of a node.

Binary Trees can be viewed as a recursive data structure in the following way:

1. An empty tree (Without nodes) is a binary tree
2. If t1 and t2 are binary trees and if r is a new node, then also the following is a binary tree (with root r )

A diagram of a triangle with a circle and a circle

Description automatically generated

**Implementation**

In a binary tree, each node stores a specific “payload”, e.g. an int value (as in the following example), as well as two references to the left and right subtree

The empty tree is represented by the null reference

**A screenshot of a computer code

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

The height of a binary tree can be calculated recursively according to the formal definition 2.4.

Make sure to check the implementation of the Traversal of Binary trees, and check if thery are in the past exam

**1. Pre-Order**

**2. In-Order**

**3. Post-Order**

**Definition 7.9 –**

**Binary Search Tree - Now we will talk about the specifics of a Binary Search tree to achieve efficiency**

A binary search tree (BST) is an ordered binary tree where for each node p of the tree holds:

1. All keys in the left subtree of p are smaller than the key of p

2. All keys in the right subtree of p are greater than the key of p

Generally, in search-tree, there is a property that each key can be contained only once in a search tree. The search tree properties should not only full fill the root node, but also for the other nodes of the tree

Example of a Binary Search Tree   
A diagram of a tree with numbers and circles

Description automatically generated

A screenshot of a computer game

Description automatically generated

1. No because of 4 its on the left sub tree of 3  
2. Yes  
3. Yes  
4. Yes

Exercise 7.12 - Traversal of binary search trees In which order (preorder, inorder, or postorder) must a search tree be traversed to obtain a sorted sequence of the key values??

Searching in the tree for a node

If the tree is empty, then naturally the search was unsuccessful  
If tree is not empty: then we go over these cases:  
 1. Case k<w: Continue search for k recursively in the left subtree  
 2. Case k>w: Continue search for k recursively in the right subtree  
 3. Case w=l: Key found, root node of the (sub) tree is searched node

A screenshot of a computer code

Description automatically generated

Question: How does the return function serahcNodeRek(r.left, v ) or the one below is going over different nodes as we are not adding or subtracting the values of r.left to iterate basically

Exercise 7.17 - Minimum and maximum of BST How can the minimum or maximum of the values stored in a binary search tree be efficiently computed ?

**Insertion into the Binary Search Trees**

The implementation makes sense, but the main question is if its part of the exam so make sure from the Prof.

A screenshot of a computer

Description automatically generated

Verify above solution

The insertion method does not ensure that a “Balanced” binary tree is created, i.e. a tree where the values are evenly distributed between left and right subtrees and the tree then has a low height. In extreme cases, the tree can degenerate into a linear list where each node has only one successor and the other sub tree is empty in each case  
  
Personal Analysis:

1. Part 2 apparently looks like the extreme case or is wrong to interpret in that way.

**Deletion in the Binary Search Trees** It must be ensured that the search tree property is preserved. A simple method for deleted is indicated below:

Algo 7.21:

The key value x is what we want to delete

1. Search for the node with key value x  
 2. Further procedure depends on the number of children of that node:

Case 1: if x is the leaf:  
 The leaf node is deleted, i.e. replaced by the empty tree  
 Case 2: if x has one child:  
 Node x is removed. Parent node of x now refers to child of x

A diagram of a triangle

Description automatically generated

Does it ensure that w is greater or lesser than v, as we must

follow the search tree properties

Case 3: if x has two children: let tl and tr be the non empty subtrees of x

A diagram of a diagram

Description automatically generated with medium confidence

* The parent node of x now references to right child of x
  + As y is smaller than z anyway so it stays on the left side of the z sub tree.   
      
    Also, we select y instead of w because y is bigger than w.
    - Apparently this statement is not correct because on page 251 under the headline “Remark” it says that the symmetrical variant could also be used where the (left instead of the right) subtree takes the internal node position of the deleted node and then the right subtree is hooked at the maximum of the left subtree  
      A diagram of a diagram

      Description automatically generated

A diagram of a diagram

Description automatically generated

* + Now we need to fix the hanging link of left node including its subtree (Subtree doesn’t get affected but below W the Binary search tree properties already follows)
    - So now in the right subtree tr, we need to find the node q as its the minimum node in the sub tree that has no further leaves attached to it and naturally it would go on the left side of q because anyway everything on left subtree of x before deleting was smaller than everything on the right

**Exercise 7.23 - Deletion in binary search tree**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A yellow smiley face with black text

Description automatically generated

Not important   
A diagram of a tree

Description automatically generated

All questions related to Binary trees in past exams and assignments

A black board with white text and circles

Description automatically generated

A screenshot of a tablet

Description automatically generated

A screenshot of a tablet

Description automatically generated

A screenshot of a blackboard with white lines

Description automatically generated

Also don’t forget to show the way instead of connecting with 15 as the replacement of the deleted node, using 7 as the replacement node. Follow notes.   
Solution is basically 5 connects 7 and under the right side 8 we will connect 15 and its sub tree

A black board with writing on it

Description automatically generated

**Assignment 7 SS21**

A tree with numbers and a few words

Description automatically generated with medium confidence

1. No because 62 is on the left side of 58  
2. No because 21 on the right side of 22 subtree  
3. Yes

A screenshot of a computer

Description automatically generated

Datteln

Brot Gurke

Ananas Chili Eis Marmelade

Axt Dosenmilch Ente

A screenshot of a tablet

Description automatically generated

A screenshot of a computer

Description automatically generated

A black board with writing on it

Description automatically generated

A