PRINCETON UNIVERSITY

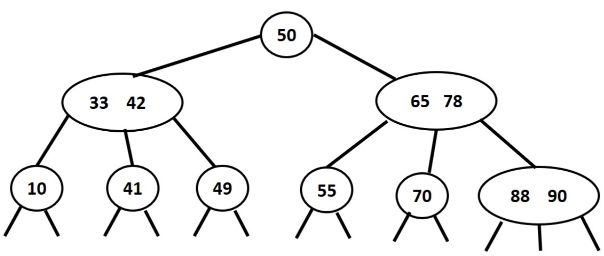
COS 226–Algorithms and Data Structures

Week 6: *Balanced Trees & Hashing* *(Videos §10.A,B §11.A,B,C & Algorithms §3.4)*

Version: October 19, 2017

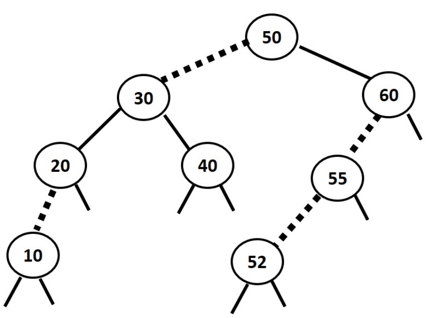
Exercise 1 – Left Leaning Red-Black BST

A. Consider the following 2-3 Tree

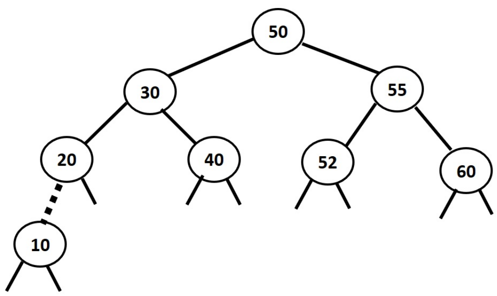


Draw the corresponding red-black tree. Use dotted lines to represent red links.

B. Consider the following red-black tree. Does this tree satisfy the LLRB invariant? If not, apply the elementary red-black BST operations, rotations and color flips, to convert this tree to a LLRB. The red links are shown as dotted lines.



C. The answer to Part B is shown below. Draw the resulting red-black tree after inserting key 62 and then draw it again after inserting key 61. The new tree must be a valid LLRB. Draw dotted lines to indicate red links.



D. (Optional homework question) Construct a LLRB by inserting the given set of comparable keys $A < B < C < D < E < F < G$ in that order. Show the rotations and/or color flips after each insertion. Draw a new tree after each insertion.

Exercise 2 – Hashing Short Questions

Answer each questions by indicating whether it is true, false, or by choosing among the multiple choices that may apply.

A. With *separate chaining* for collision resolution, it is unnecessary to resize the hash table to obtain good efficiency.

B. With *linear probing*, each key will be placed in some cell if the number of keys is less or equal to table size.

C. Deletions in *linear probing* are easier to implement than in *separate chaining*.

D. Recall that a hash table is a symbol table which associates some key to a value (and we call entry the key-value pair). Then, the best definition of a collision in a hash table is when:

   (i) two entries are identical except for their keys;
   (ii) two entries with different values have the exact same key;
   (iii) two entries with different keys have the same hash value;
   (iv) two entries with the exact same key have different hash values.

E. Choosing a bad hash function can result in

   (i) too many collisions;
   (ii) slow performance;
   (iii) no expected constant runtime guarantees;
   (iv) too much clustering when using linear probing.

F. If a linear probing hash table is 50% full, the average number of probes required for a search hit is 1.5.

G. Which of the following scenarios leads to a linear running time for a random search hit in a linear probing hash table containing n keys?

   (i) All keys hash to different values.
   (ii) All keys hash to the same value.
   (iii) All keys hash to different even-numbered values.
   (iv) The table has size larger than n^2.

Exercise 3 – Algorithm Design Question (Bonus)

An array b is called a circular shift of some sorted array a, if b is obtained by rotating array a clockwise by m positions $(0 < m < n)$ where n is the size of the array a. A sample array b with $n = 10$ and $m = 3$ is as shown below. The array a is not given and m is not known. The circular shifted array b satisfies the following properties.

- All elements of array b are distinct.
- Array b consists of two sorted (ascending order) subarrays.
- $b[0] > b[n-1]$

circular shift b[]

0	1	2	3	4	5	6	7	8	9
34	**55**	**89**	**1**	**2**	**3**	**5**	**6**	**8**	**9**

A. Assume that the array b consists of n comparable distinct keys. Design an efficient algorithm to determine the index of the minimum value of array b. Briefly describe your algorithm using crisp and concise prose or clean code.

B. Design an efficient algorithm to find any given key in array b. You can use your algorithm in part A to help solve this problem. Briefly describe your algorithm using crisp and concise prose.