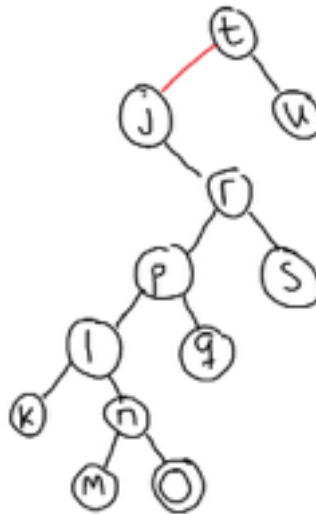(1) (15%) (Problem 3.3.16) Show the result of inserting the letter n into the red-black BST shown on page 450 (only the search path is shown, and you need to include only these nodes in your answer).

Solution:

(2): (15%) (Problem 3.3.5, shortened) Draw all structurally different 2-3 trees for N = 7, 8, and 9. (The trees for N from 1 to 6 are shown on p. 449)

Solution:

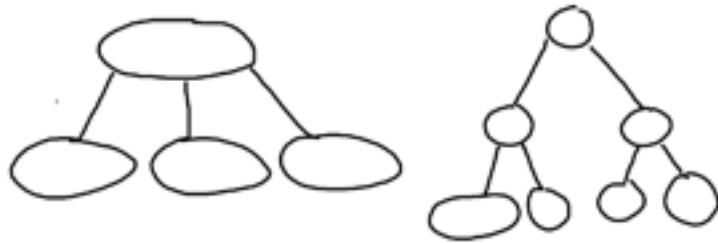N = 7:   Either all nodes in tree are 2-nodes
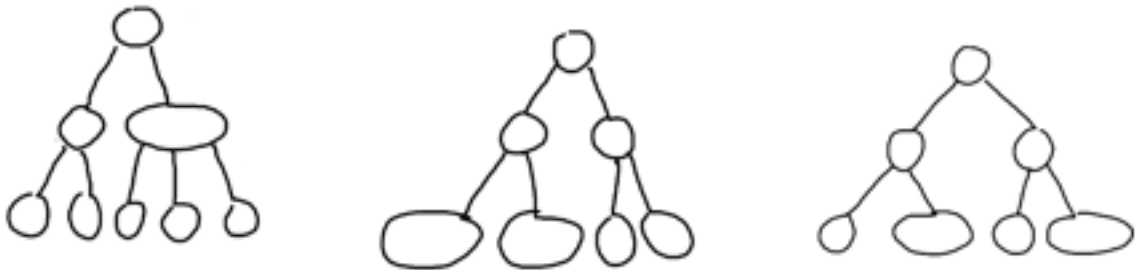   Or all nodes are 3-nodes except for one 2-node leaf

N = 8:   Either all nodes are 3-nodes or
                           All nodes are 2-nodes except for one 3-node leaf

N = 9:   Either all nodes are 2-nodes except for one 3-node at depth 1 or
                           All nodes are 2-nodes except for two 3-node leafs

(2)) (10%) (Problem 3.4.10) Insert the keys E A S Y Q U T I O N in that order into an initially empty table of size M=16 using linear probing. Use the hash function 11 k % M to transform the k-th letter of the alphabet into a table index (so, 'A' = 0). Redo this exercise for M=10.

Solution:

M = 16

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	Q	T				S		Y	I	O		E	U		N

M = 10

0	1	2	3	4	5	6	7	8	9
A	U	I	N	E	Y	Q	O	S	T

(4):) (10%) Suppose that the keys A through G, with the hash keys given below, are inserted in some order into an initially empty table of size 7 using linear probing (M=7, no resizing). key A B C D E F G hash 2 0 5 4 4 4 2 Which of the following (more than one might apply) could not possibly result from inserting these keys?
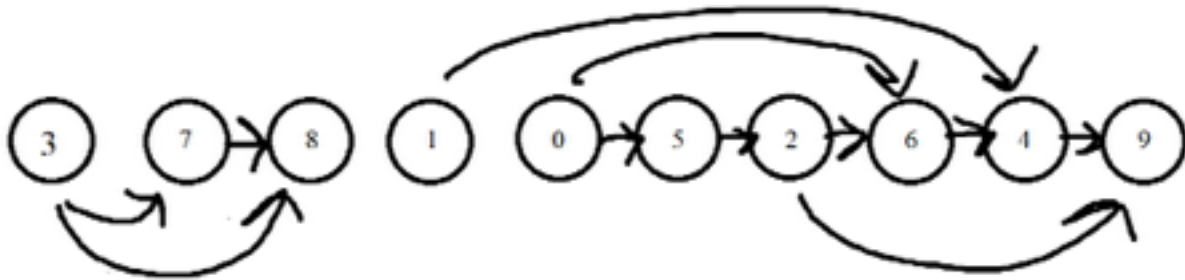
Solution:

 a. BEAGDFG: Can happen

b. CFAGDEB: Cannot happen - element B cannot possibly have index six in table in however order elements are inserted into table due to indices 1-5 never being fully occupied until last element is inserted

c. FBGAECD: Can happen

d. FCBGADE: Cannot happen - No element is in the index corresponding to it's original hash value. At least one element, the one that is inserted first into an empty table, is always bound to take the index corresponding to it's original hash value

(6): (10%) Give a topological sort of the digraph containing the edge list: 3→7 1→4 7→8 0→5
5→2 3→8 2→9 0→6 4→9 2→6 6→4.

Solution:

(7):) (15%) Run by hand DepthFirstOrder on the digraph t11.txt attached. Give the preorder,
postorder, and reverse postorder obtained

Solution:

Preorder: 0 5 1 2 4 10 6 3 7 8 9
Postorder: 10 6 4 2 1 5 0 3 9 8 7
Reverse postorder: 7 8 9 3 0 5 1 2 4 6 10