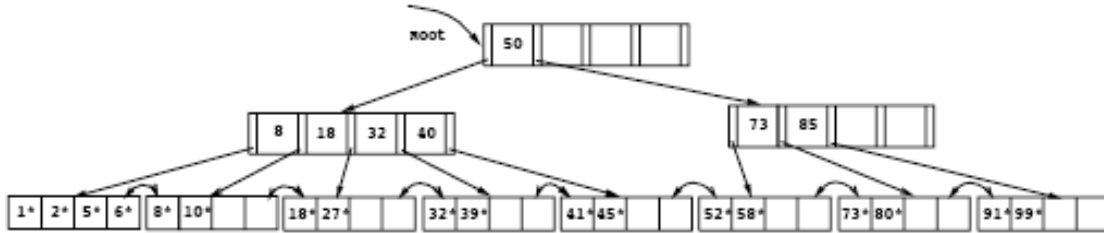


**COP 4710 Fall 2009 Homework 4.**  
**Due: 11:55pm, 12/02/2009**

**Problem I.** Consider the B+ tree index of order  $d = 2$  shown in the following figure.



1. Show the tree that would result from inserting a data entry with key 9 into this tree.
2. Show the B+ tree that would result from inserting a data entry with key 3 into the original tree. How many page reads and page writes does the insertion require?
3. Show the B+ tree that would result from deleting the data entry with key 8 from the original tree, assuming that the left sibling is checked for possible redistribution.
4. Show the B+ tree that would result from deleting the data entry with key 8 from the original tree, assuming that the right sibling is checked for possible redistribution.
5. Show the B+ tree that would result from starting with the original tree, inserting a data entry with key 46 and then deleting the data entry with key 52.
6. Show the B+ tree that would result from deleting the data entry with key 91 from the original tree.

**Problem II.** A PARTS table with Part# as key field includes records with the following Part# values: 23, 65, 37, 60, 46, 92, 48, 71, 56, 59, 18, 21, 10, 74, 78, 15, 16, 20, 24, 28, 39, 43, 47, 50, 69, 75, 8, 49. Suppose the search key values are inserted in the given order in a B+ -tree of order  $d=2$ . Show how the tree would expand and what the final tree looks like.

**Problem III.** Consider the join  $R \bowtie_{R.a=S.b} S$ , given the following information about the relations to be joined. The cost metric is the number of page I/Os unless otherwise noted, and the cost of writing out the result should be uniformly ignored.

Relation R contains 10,000 tuples and has 10 tuples per page.  
 Relation S contains 2000 tuples and also has 10 tuples per page.  
 Attribute  $b$  of relation S is the primary key for S.  
 Both relations are stored as simple heap files.  
 Neither relation has any indexes built on it.  
 52 buffer pages are available.

Answer the following question with your justifications.

1. What is the cost of joining R and S using a page-oriented simple nested loops join?
2. What is the cost of joining R and S using a block nested loops join?
3. What is the cost of joining R and S using a sort-merge join?

For a 5% bonus, answer the following question for each item in the above list.  
 What is the minimum number of buffer pages required for this cost to remain unchanged?