Name: **Andrewid:**

Homework #2
15-415 (Fall 2010)

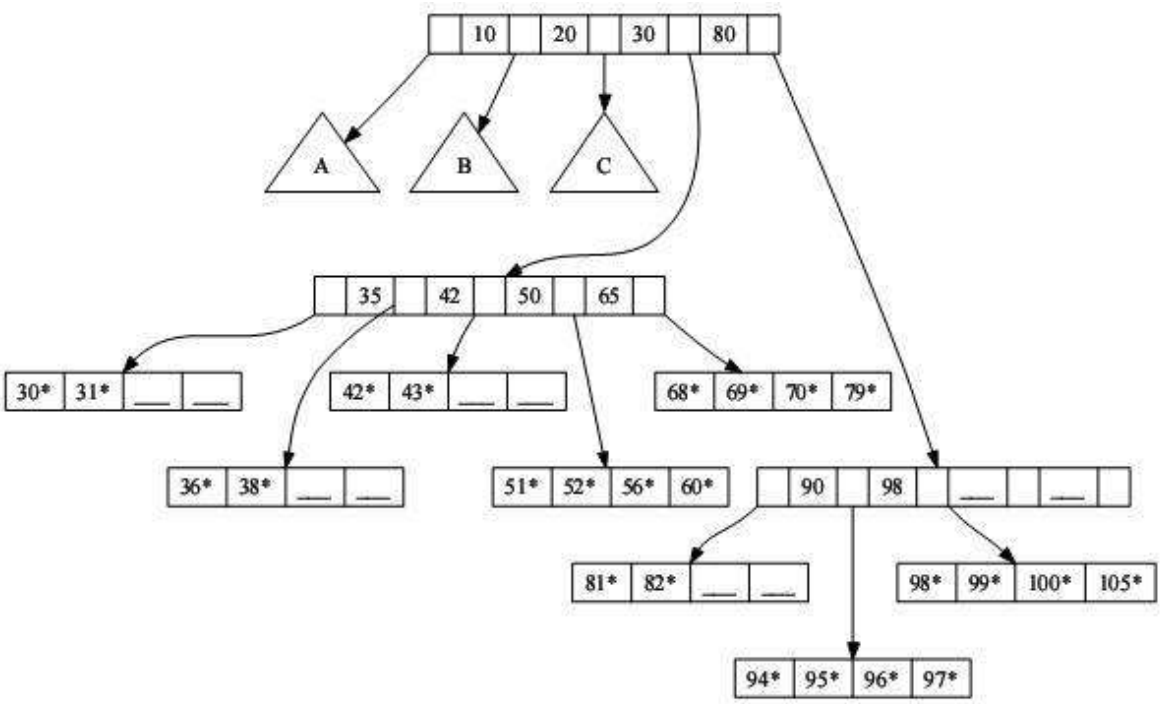
Question 1: B+ Trees [22 points]

Consider an empty B+ Tree with order d=2, i.e. there are at most 4 keys per node, and at most 5 pointers to children. Bulk load the B+ tree with data entries with even numbers from 2 to 100 (i.e. 2, 4, 6, . . . , 100) so that each leaf is full using the algorithm outlined in Section 10.8.2 (Page 360) of the textbook.

Q1.1 What is the height of the tree after inserting all the above keys? [5 points]

Q1.2 List all the keys whose insertion increased the height of the tree e.g. when key 2 was inserted, tree height increased from 0 to 1. [5 points]

For the next 2 questions consider the B+ tree shown below. The leaf key values are represented with * next to them. Unassigned values are represented with . The subtrees A, B and C are valid B+ sub-trees. Note that again, it is of order d=2.



Q1.3 Show the tree after deleting key 31. [6 points]

Q1.4 Starting from the tree in Figure 1, find a key which increases the height of the tree when inserted. [6 points]

Question 2: Extendible Hashing [12 points]

We have the following records where we indicate the hashed key in parenthesis (in binary):

a [000110]
b [111100]
c [010111]
d [010000]
e [101001]
f [010111]
g [101001]
h [011010]
i [011010]
j [001110]

Q2.1 Consider an extendible hash structure where buckets can hold up to three records. Initially the structure is empty. Show the extendible hash structure after these records (in the order shown above) have been inserted. Assume that as mentioned in the textbook, the directory doubles in size at each overflow. [6 points]

Q2.2 Consider the Extendible Hashing index shown in Figure 11.6 (Page 377) of the textbook. What is the maximum number of entries that can be inserted into the index without doubling the size of the directory? [6 points]

Question 3: Linear Hashing [12 points]

Building

Q3.1 For the same records, hash keys, and assumptions as in Q2.1, show the linear hash structure for the file. Initially the structure is empty. Assume that we split whenever the new key goes into a full bucket (which may or may not be the split bucket). [6 points]

Deletion

Q3.2 Consider the Linear Hashing index shown in Figure 11.13 (Page 384) of the textbook. Draw the index after deleting the entries with hash values 66, 22 and 32. Make sure you indicate where the Next pointer is present. Assume that the full deletion algorithm is used and we contract when a bucket becomes empty. [6 points]

Question 4: External Sorting [18 points]

Suppose we have a disk with an average seek time of 10ms, average rotational delay of 5ms, and a transfer time of 1ms for a 4K page. Assume that the cost of reading/writing a page is the sum of those values (i.e., 16ms) unless a sequence of pages is read/written. In this case, the cost is the average seek time plus the average rotational delay (to find the first page) plus 1ms per page (to transfer data). You are given 320 buffer pages and asked to sort a file with 107 pages. Make the assumptions used in Section 13.3 of the textbook. Find the cost of the following operations [6 points each = 1 point for formula used + 5 points for numerical substitutions and answer]:

Q4.1 Do 319-way merges on 319 input buffers of 1 page each, and 1 output buffer page.

Q4.2 Do 256-way merges on 256 input buffers of 1 page each, and output buffer of 64 pages.

Q4.3 Do 16-way merges on 16 input buffers of 16 pages each, and output buffer of 64 pages

Question 5: Serializability [20 points, 10 points each]

Consider the following schedules. The actions are listed in the order they are scheduled, and prefixed with the transaction name.

S1: T1:R(X), T2:R(X), T1:W(Y), T2:W(Y), T1:R(Y), T2:R(Y)
S2: T3:W(X), T1:R(X), T1:W(Y), T2:R(Z), T2:W(Z), T3:R(Z)

Q5.1 What are the serializability graphs for each for the schedules?

Q5.2 Is the schedule conflict-serializable? If so, what are all the conflict equivalent serial schedules?

Question 6: Two-Phase Locking [16 points, 8 points each]

Consider the following schedules:

S1: T1:R(X), T2:W(Y), T2:R(X), T1:W(Y), T1:Commit, T2:Commit 1
S2: T1:R(X), T2:R(Y), T1:W(Z), T1:Commit, T3:R(Y), T3:R(Z), T2:W(Y), T3:W(X), T2:Commit, T3:Commit

For each schedule, state which of the following concurrency control protocols allows it, that is, allows the actions to occur in exactly the order shown.

- 2PL
- Strict 2PL

Please provide a brief explanation for your answer...If YES, show where the lock requests could have happened; If NO, explain briefly.