Virginia Tech. **CS 4604 – Introduction to DBMS**
Computer Science **Spring 2015, Prakash**

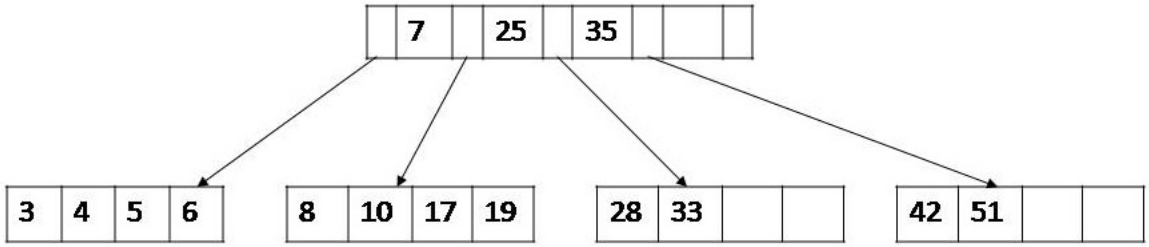
Homework 3: B+ Trees, Hashing, Bulk Loading
(due March 2nd, 2015, 4:00pm, in class—hard-copy please)

Reminders:

a. Out of 100 points. Contains 4 pages.
b. Rough time-estimates: 2~4 hours.
c. Please type your answers. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.
d. There could be more than one correct answer. We shall accept them all.
e. Whenever you are making an assumption, please state it clearly.
f. Lead TA for this HW: Elaheh Raisi.

Q1. B+ Tree [24 points]

Assume the following B+ tree exists with d = 2:



Sketch the state of the B+ tree after each step in the following sequence of insertions and deletions, maintaining at least 50% occupancy at each step and overflow triggered split. In the diagram above we have not shown pointers in the leaf nodes for simplicity but remember that the leaf nodes are linked lists.

Note: Use the insertion and deletion algorithms given in the textbook section 10.5 (page 349) and 10.6 (page 353) respectively. Root node can have 1 to 2d keys. During deletion redistribute the leaf pages wherever possible.

Q1.1. (4 points) Insert 30

Q1.2. (4 points) Insert 12

Q1.3. (4 points) Insert 60

Q1.4. (4 points) Insert 1

Q1.5. (4 points) Delete 33

Q1.6. (4 points) Delete 5

Q2. Bulk Loading a B+Tree [20 points]

Suppose we are bulk-loading a B+-Tree. Pages have 72 bytes to store information. A key value takes 8 bytes, and a pointer to a tree node or row takes 8 bytes. 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 **at least** half full using the algorithm outlined in Section 10.8.2 (Page 360) of the textbook.

Q2.1. (5 points) What is the order of the B+ tree? How many keys and pointers per node can it hold? (Note: Recall that each node in a B+ tree is a page)

Q2.2. (5 points) What is the height of the tree after inserting all the above keys?

Q2.3. (5 points) Sketch the final B+ tree after bulk loading. No need to show each node, just enough for us to be convinced you have the right tree.

Q2.4. (5 points) What is the minimum number of keys that must be deleted so that the height of the tree decreases by 1? List these keys. Note: Please use the algorithm outlined in section 10.6 (page 353) of the textbook.

Q3. Linear Hashing [15 points]

We have the following records with the given hash values.

8,4,9,10,13,18,3,30,7,19,24,36,21,15

Q3.1. (10 points) For the given hash keys, sketch the linear hash structure for the file. There are 4 buckets in the start and each bucket can hold at most 2 records. Initially the structure is empty. Assume that we split whenever a new key triggers the creation of a new overflow page.

Note: Use the linear hashing algorithm outlined in Section 11.3 (page 379) of the textbook.

Q3.2. (5 points) How many overflow pages were created by the time the last record was inserted (which were subsequently discarded)?

Q4. Extendible Hashing [16 points]

Q4.1. (10 points) Consider an extendible hash structure in Table 1 below where buckets can hold up to three records. Initially the structure is empty and global depth is 2. Sketch the extendible hash structure after the records given in Table 1 (in the

same order shown) have been inserted. Assume that as mentioned in the textbook, the directory doubles in size at each overflow.

Record	Hash value
a	00100
b	00010
c	00001
d	01000
e	01100
f	10000
g	10111
h	10110
i	10010
j	11111
k	01000
l	00011
m	10011
n	10100

Note: Use the extendible hashing algorithm outlined in section 11.2 of the textbook.

Q4 .2 (3 points) After sketching the extendible hashing, what is the final global depth and final depth of all the buckets?

Q4.3 (3 points) After sketching the extendible hashing, can you write a binary value record that increase the global depth by one?

Q5. External Sorting [25 points]

Suppose you have a file with $N = 5 \times 10^7$ pages.

Q5.1. (10 points) What is the total I/O cost of sorting the file using the two-way merge sort (with three buffer pages)?

Now suppose you have $B = 129$ buffer pages. Answer the following questions using the general external sorting algorithm outlined in section 13.3 of the textbook (page 424). Please write the formula you used in calculating the answers.

Q5.2. (5 points) How many runs will you produce in the first pass?

Q5.2. (5 points) Now assume that we have a disk with an average seek time of 10ms, average rotation delay of 5ms and a transfer time of 1ms for each page. Assuming the cost of reading/writing a page is the sum of those values (i.e. 16ms)

and do not distinguish between sequential and random disk-access – *any access* is 16ms, what is the total running time to sort the file?

Q5.3. (5 points) With 129 buffer pages, what's the maximum size of the file (in number of pages) that we can sort with 3 passes?