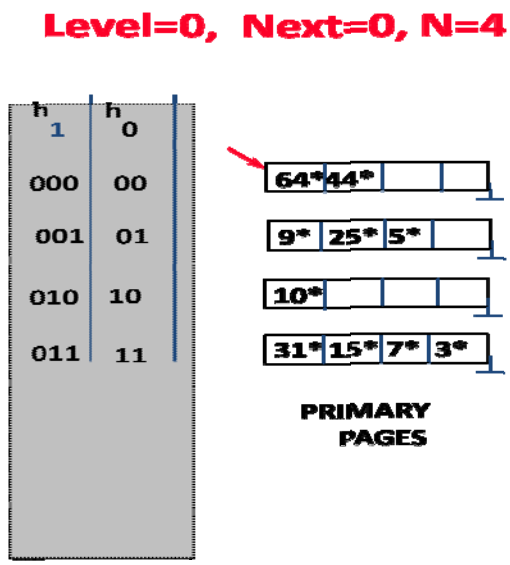
CSC540 (Fall 2009)
Written Assignment 3
Due date 11/17/2009 (at the beginning of class)

1. Consider a disk with the following parameters; block size B = 512 bytes, interblock gap size G = 128bytes (some space between blocks used to record some control info. No data is stored there), number of blocks per track = 20; number of tracks per surface = 400. A disk pack consists of 15 double sided platters.
   (a) What is the total capacity of a track and what is its useful capacity (that is excluding interblock gaps)?
   (b) How many cylinders are there?
   (c) What are the total capacity and the useful capacity of a cylinder? a disk pack?
   (d) Suppose that the disk drive rotates the disk pack at a speed of 2400rpm (rev per min); what are the transfer rate *tr* in bytes/msec and the block transfer time *btt* in msec? What is the average rotational delay *rd* in msec? What is the bulk transfer rate?
   (e) Suppose that an average seek time is 30msec. How much time does it take on the average in msec to locate and transfer a single block, given its block address?
   (f) Calculate the average time it would take to transfer 20 random blocks and compare this with the time it would take to transfer 20 consecutive blocks.

2. Consider a disk with the same characteristics as in Qu1. Further, consider a STUDENTS file with *r* = 20,000 records, fixed length format with an unspanned organization (blocks cannot span two blocks). Each record has the following fields: : Ssn, 9 bytes; Name, 30 bytes; First_name, 20 bytes; Address, 40 bytes; Phone 9 bytes; birthdate, 8 bytes; Sex, 1byte; Major-dept_code, 4 bytes ; Minor_dept_code , 4 byte; class_code, 4bytes and Degree_prog, 3bytes. An additional 1 byte was used as a *deletion marker* (these are used to marked records as deleted for organizations that do not compact at the same time as deletion occurs).
   (a) The record size *R* (including the deletion marker), the blocking factor *bfr* (number of records in a block), and the number of disks blocks *b* for the file.
   (b) Calculate the average time it takes to find a record by doing a linear search on file if (i) the file blocks are stored contiguously;(ii) if the file blocks are not stored contiguously

3. Consider the snapshot of the Linear Hashing index shown below. Assume that a bucket split occurs whenever an overflow page is created.




(a) What is the *maximum* number of data entries that can be inserted (given the best possible distribution of keys) before you have to split a bucket? Explain very briefly.

(b) Show the file after inserting a *single* record whose insertion causes a bucket split.

(c) What is the *minimum* number of record insertions that will cause a split of all four buckets? Explain very briefly.

(d) What is the value of *Next* after making these insertions? What can you say about the number of pages in the fourth bucket shown after this series of record insertions

4. Consider the data entries in the Linear Hashing index for Qu. 3. Show an Extendible Hashing index with the same data entries. Answer the questions (a) … (d) with respect to the extendible hashing index.

5. Suppose that a page can contain at most four data values and that all data values are integers. Using only B+ trees of order 2, give examples of each of the following:

( a ) A B+ tree whose height changes from 2 to 3 when the value 25 is inserted. Show your structure before and after the insertion.

( b ) A B+ tree in which the deletion of the value 25 leads to a redistribution. Show your structure before and after the deletion.

( c ) A B+ tree in which the deletion of the value 25 causes a merge of two nodes but without altering the height of the tree.

6. Consider a relation R(*a,b,c,d,e*) containing 5,000,000 records, where each data page of the relation holds 10 records. R is organized as a sorted file with secondary indexes. Assume that *R.a* is a candidate key for R, with values lying in the range 0 to 4,999,999, and that R is stored in *R.a* order. For each of the following relational algebra queries, state which of the following three approaches is most likely to be the cheapest:
   - Access the sorted file for R directly.
   - Use a (clustered) B+ tree index on attribute *R.a*.
   - Use a linear hashed index on attribute *R.a*.

   1. $\sigma_{a<50,000}(R)$
   2. $\sigma_{a=50,000}(R)$
   3. $\sigma_{a>50,000 \wedge a<50,010}(R)$
   4. $\sigma_{a\neq 50,000}(R)$