

### Example Questions

1. A database uses a Linear Hashing based index. Answer the following questions if the Next pointer is pointing to bucket 25 (buckets are indexed starting from 0).

- a) What is the minimum number of primary buckets in this Linear Hash?
- b) If the number of primary buckets is as calculated in part (a) how many bucket splits are needed to split the 3<sup>rd</sup> bucket.
- c) If Next points to 25 and the number of primary buckets is 89 which buckets are accessed for searching the following two keys  $k_1$  and  $k_2$ , clearly show how you calculate the bucket index from the hash values indicating the bits you are using (Use the least significant bits for addressing).

$$H(k_1) = 175$$

$$H(k_2) = 83$$

2. An extendible hash is used to index a string field storing names of students. A bucket stores 100 keys and the global depth of the directory is 8. There are 256 buckets in this extendible hash. If the directory is buffered in the memory, how many disk accesses are needed for the following queries:

- a) Finding if there is a student with name "Ahmet" exists or not?

- b) Finding the number of students whose name starts with the letter 'B'?

3. You have got 1000 records that are fixed length and 100 bytes. The records will be stored on a disk with a sector size of 102 bytes. Each cluster involves 4 sectors and each track involves 60 sectors. The rotational time is 4000 rpm and the average seek time is 6 msec. Answer the following questions.

- a) Blocking factor of the files:

- a) Number of blocks to store the file.

- b) Average time to find a record if the read/write head is at a random position initially and the file is sorted (cylinder based approach is not used):

- c) Time to add a new record into the file if the read/write head is at a random position initially and the file is sorted. Consider the worst case in your solution and assume that the number of blocks in the sorted file will remain the same once the new record is inserted:
  
- d) We want to index the given file with a dense index. Assume that index stores a search key of 4 bytes and a pointer of 4 bytes. How many disk accesses do we need in order to find a record?
  
- e) We want to index the given file with a sparse index. Assume that index stores a search key of 4 bytes and a pointer of 4 bytes. How many disk accesses do we need in order to find a record?
  
- f) Imagine we have 5 available buffer pages and our file is a heap file. We want to sort the file. How many runs will be produced in the first pass?
  
- g) How many passes will it take to sort the file completely?
  
- h) What is the total IO cost of sorting the file?