Assignment 2
[80 points in total]

Professor Yanlei Diao

Question 1 [28 points] Disks and Access Time

Consider a disk with a sector size of 512 bytes, 63 sectors per track, 16,383 tracks per surface, 10 double-sided platters (i.e., 20 surfaces). The disk platters rotate at 7,200 rpm (revolutions per minute). The average seek time is 9 msec, whereas the track-to-track seek time is 1 msec (use these numbers in appropriate places in your calculation).

Suppose that a page size of 4096 bytes is chosen, and a page can span sectors on difference tracks. Suppose that a file containing 1,000,000 records of 256 bytes each is to be stored on such a disk. No record is allowed to span two pages.

(1) What is the capacity of a track (in number of bytes)?
(2) What is the capacity of the disk (in number of bytes)?
(3) How many records fit in a page?
(4) How many records fit in a cylinder?
(5) If the file is arranged sequentially on the disk, how many sectors are needed? How many pages are needed? And how many cylinders are needed?
(6) How much time is required to read this file **sequentially**? Please show your calculation clearly for (a) seek time, where a random seek takes 9 msec and then the track-to-track seek takes 1 msec; (b) rotational delay, which occurs only in the first seek operation in sequential I/O; (c) transfer time, and (d) total time.
(7) How much time is needed to read 50% of the pages in the file **randomly**, that is, one random I/O per page as observed through index lookups? Please calculate the seek time, rotational delay and transfer time for each page first, and then calculate the total cost of 50% of the pages in the file.

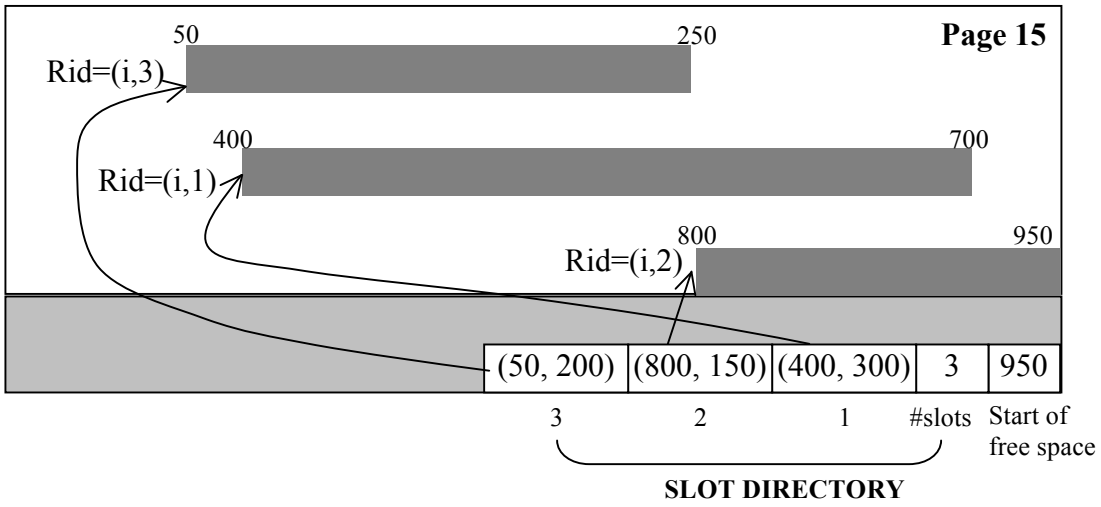
To answer each question, please write a clean formula and your final answer.

(1) What is the capacity of a track (in number of bytes)?	
(2) What is the capacity of the disk (in number of bytes)?	
(3) How many records fit in a page?	
(4) How many records fit in a cylinder?	
(5) If the file is arranged sequentially on the disk, how many sectors are needed? How many pages are needed? And how many cylinders are needed?	(a) Num. of sectors: _____ (b) Num. of pages: _____ (c) Num. of cylinders: _____

(6) How much time is needed to read this file **sequentially**?	(a) Seek time: _____ (b) Rotational delay: _____ (c) Transfer time: _____ (d) Total: _____
(7) How much time is needed to read 50% of the pages in the file **randomly**, that is, one random I/O per page as observed through index lookups?	(a) Cost per page: _____ (b) Total cost _____

Question 2: Disk Page Layout [10 points]

The figure below shows a page containing variable length records. The page size is 1KB (1024 bytes). It contains 3 records, some free space, and a slot directory in that order. Each record has its record id, in the form of Rid=(page id, slot number), as well as its start and end addresses in the page, as shown in the figure.



(1) Now a new record of size 200 bytes needs to be inserted into this page. Apply the record insertion algorithm (with page compaction, if necessary) that we learned in class to this page. Show the **content of the slot directory after the new record is inserted**.

(2) The next question proceeds after the operation in Part (1). Now, the record with Rid =(15,3) needs to be deleted. Afterwards, another record of size 300 bytes needs to be inserted. Show the **content of the slot directory after the deletion and new insertion.**

Question 3 [26 points]: B+ Trees

(1) [8 points] Show the results of entering the keys 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 (in that order) to an initially empty B+ tree. Assume that every non-leaf node can hold up to 3 index entries and every leaf node can hold up to 3 data entries. In case of overflow, split the node (do not re-distribute keys to neighbors).

(2) [8 points] Now demonstrate a different insertion order that leads to a tree of different depth from the one in Part (a).

(3) [10 points] Assume that you have just built a B+ tree index using Alternative (2) on a heap file containing 1,000,000 records. The key field for this B+ tree index is a 40-byte string, and it is a candidate key of the associated relation. Pointers (i.e., record ids and page ids) are (at most) 10-byte values. The size of one disk page is 1024 bytes. The index was built in a bottom-up fashion (using the bulk-loading algorithm), and the nodes at each level were filled up as much as possible.

(a) How many levels does the resulting tree have? _____

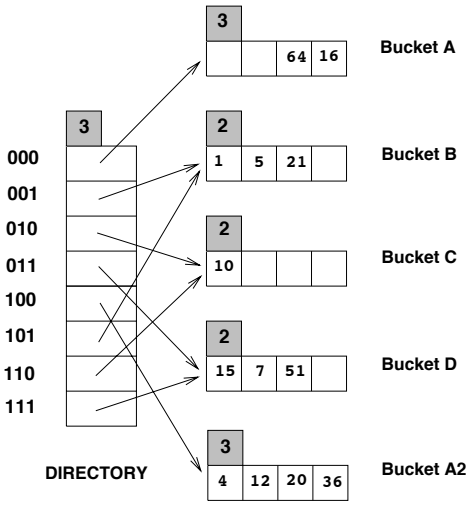
(b) For each level of the tree, how many nodes are at that level?

Level 0: _____ , Level 1: _____ , Level 2: _____ , …

Please add necessary explanation.

Question 4 [16 points]: Extendible Hashing

Consider the Extendible Hashing index shown below. Answer the following questions about this index:



(1) Show the index after inserting an entry with hash value 68.

(2) Show the index after inserting entries with hash values 17 and 69 into the original index.