1. [5pts] Explain two components of the disk access time.
   - Seek time – time it takes to reposition the arm over the correct track
   - Rotational latency – time it takes for the sector to be accessed to appear under the head

2. [5pts] Describe two reasons why RAID systems are used.
   - To provide a view of a single disk of high capacity and high speed by using multiple disks in parallel
   - High reliability by storing data redundantly, so that data can be recovered even if a disk fails

3. a) [5pts] What would be the problems if byte string representation or fixed length representation is used for variable-length records?
   - Byte string representation – difficulty with deletion / growth
   - Fixed length representation – waste of unused space in shorter records, filled with a null or end-of-record symbol

   b) [10pts] Explain slotted page structure with a diagram.

4. [5pts] Describe the pros and cons of sequential file organization.
   - Pros: suitable for applications that require sequential processing of the entire file
   - Cons: insertion/deletion overhead, need to reorganize periodically

5. a) [5pts] Why does hash bucket skew occur?
   - Multiple records have same search-key value
   - Chosen hash function produces non-uniform distribution of key values

   b) [5pts] What is the deficiencies of static hashing?
   - If initial number of buckets is too small, and file grows, performance will degrade due to too much overflows
   - If initial number of buckets is too large, or database shrinks, a significant amount of space will be wasted.

6. [10pts] Describe the pros and cons of using balanced tree (B-tree and its variants) and binary tree as an index in database systems.
   - Balanced tree (B-tree and its variants)
     - Pros: Average query performance is guaranteed because every path from the root to a leaf of the tree has the same length.
       - Overall reorganization is not required because it maintains the balanced form.
     - Cons: Additional performance overhead and implementation complexity for balancing is imposed on insertion and deletion.
   - Binary tree
     - Pros: Insertion and deletion is straightforward.
     - Cons: Query performance is unpredictable if the tree is unbalanced.
       - Overall reorganization may be required for performance.
7. [5pts: 1pt each / 0pt for no answer / -1pt for a wrong answer] Fill in the blanks.

A B*-tree is a rooted tree satisfying the following properties.
- Each branch node (that is not a root or a leaf) has between __4__ and 7 children.
- A leaf node has between __3__ and 6 values.
- If the root is not a leaf, it has at least __2__ children.
- If the root is a leaf, it can have between 0 and __6__ values.

8. For the following B*-tree, show the form of the tree after each of the following series of operations.

a) [5pts] Insert 7.

b) [5pts] Delete 23.

c) [5pts] Delete 15.
9. [5pts] Suppose you decide to use a primary index or secondary index, to select from a relation with an equality condition. The selection condition can be on a key or a non-key attribute. Does the choice of index affect the selection cost? Justify your answer.

Both index result in the same cost for the selection on a key attribute. However, cost may differ for the selection on a non-key attribute if it retrieves multiple matching records which may be on a different block.

10. [10pts] Suppose you need to sort a relation of 40 gigabytes, with 4 kilobyte blocks, using a memory size of 40 megabytes. Suppose the cost of a seek is 5 milliseconds, while the disk transfer rate is 40 megabytes per second. Find the cost of sorting the relation, in seconds, with $b_b = 1$ and $b_b = 100$.

$b_r = \frac{40GB}{4KB} = 10,000,000$ blocks
$M = \frac{40MB}{4KB} = 10,000$ blocks

The initial number of runs = $(b_r / M) = 1,000$

The number of merge passes required = $\lceil \log_{M^{-1}}(b_r / M) \rceil = \lceil \log_{9999}(1000) \rceil = 1$

Block transfers = $b_r(2*1 + 1) = 30,000,000$ blocks

Seeks = $2 \ lceil b_r / M \rceil + \lceil b_r / b_b \rceil (2*1 - 1)$

∴ if $b_b = 1$, $2000 + 10,000,000 = 10,002,000$ seeks

$b_b = 100$, $30,000,000 * 4KB / 40MB + 102,000 * 5/1000 = 5310$ sec.

11. Let relations $r$ and $s$ have the following properties: $r$ has 10,000 tuples, $s$ has 27,000 tuples, 25 tuples of $r$ fit on one block, and 30 tuples of $s$ fit on one block. Estimate the number of block transfers and seeks required in the worst case, using each of the following join strategies for the natural join between $r$ and $s$.

$b_r = 10,000 / 25 = 400$
$b_s = 27,000 / 30 = 900$

a) [5pts] Nested-loop join, with $r$ as the outer relation.

Block transfers = $n_r * b_s + b_r = 10,000 * 900 + 400 = 9,000,400$ blocks

Seeks = $n_r + b_r = 10,000 + 400 = 10,400$ seeks

b) [5pts] Block nested-loop join, with $r$ as the outer relation.

Block transfers = $b_r * b_s + b_r = 400 * 900 + 400 = 360,400$ blocks

Seeks = $2 * b_r = 2 * 400 = 800$ seeks

C) [5pts] Hash join, using 20 buffers for the input and each 4 output partitions.

$b_b = 20$, $n_h = 4$

Block transfers = $3(b_r + b_s) + 4 * n_h = 3(400 + 900) + 4 * 4 = 3916$ blocks

Seeks = $2(\lceil b_r / b_b \rceil + \lceil b_s / b_b \rceil) + 2 * n_h = 2(20 + 45) + 2 * 4 = 138$ seeks