[30 points] Problem 1. Disk Storage

(a) Describe the process of reading one sector from the disk.

1. **Seek**: The read/write head of the disk is moved to the correct track where the desired sector is located. This step involves physically moving the head over the spinning disk.
2. **Rotational Delay**: Once the head is over the correct track, it waits for the disk to rotate until the desired sector is under the head.
3. **Reading**: The read/write head reads the data from the sector as it spins under the head.

(b) In the procedure described in (a), which step is the most time-consuming (on average)?

The most time-consuming step is typically the Seek step. Moving the read/write head physically to the correct track takes more time than waiting for the disk to rotate to the correct position (rotational delay) and the actual reading of the data.

(c) What is the average rotational delay for a 15000 RPM disk? Express your answer in milliseconds. (RPM: revolutions per minute, which means the number of rotations in one minute)

Convert RPM to RPS:
RPS= 15000 / 60 = 250 RPS

Calculate Time for One Full Rotation:
Time for One Full Rotation: 1/ RPS = 0.004 seconds

Calculate Average Rotational Delay: Average Rotational Delay = time for One Full Rotation / 2 = 0.004/2 X 1000 = 2.0 milliseconds

(d) What can we do to reduce the rotational delay?

1. **Increasing the Rotational Speed**: Faster spinning disks have lower rotational delay.
2. **Disk Scheduling Algorithms**: Algorithms like SCAN or CSCAN can optimize the order of reading/writing to reduce the average wait time.
3. **Using SSDs**: Solid State Drives have no moving parts and therefore no rotational delay.

[20 points] Problem 2. File Organization

(a) Consider the following record: <120, 'Jeffrey'>. Its data type is <int, varchar>.

Now we want to store this record as a variable length record in the file. Draw a diagram that shows what content will be stored in which bytes.

Byte 1: [Int(120) 1st byte]
Byte 2: [Int(120) 2nd byte]
Byte 3: [Int(120) 3rd byte]
Byte 4: [Int(120) 4th byte]
Byte 5: [Varchar length: 7]
Byte 6: ['J']
Byte 7: ['e']
Byte 8: ['f']
Byte 9: ['f']
Byte 10: ['r']
Byte 11: ['e']
Byte 12: ['y']

(b) Consider the following record: <0, 'student', 'book', 10>. Its data type is <int, varchar, varchar, int>. Now we want to store this record as a variable length record in the file. Draw a diagram that shows what content will be stored in which bytes.

Byte 1: [Int(0) 1st byte]
Byte 2: [Int(0) 2nd byte]
Byte 3: [Int(0) 3rd byte]
Byte 4: [Int(0) 4th byte]
Byte 5: [Varchar length: 7]
Byte 6: ['s']
Byte 7: ['t']
Byte 8: ['u']
Byte 9: ['d']
Byte 10: ['e']
Byte 11: ['n']
Byte 12: ['t']
Byte 13: [Varchar length: 4]
Byte 14: ['b']
Byte 15: ['o']
Byte 16: ['o']
Byte 17: ['k']
Byte 18: [Int(10) 1st byte]
Byte 19: [Int(10) 2nd byte]
Byte 20: [Int(10) 3rd byte]
Byte 21: [Int(10) 4th byte]

[50 points] Problem 3. B+ tree

a) (10 points) The B+ tree in figure 1 has order=4 and height=2.
What would the tree's height be after executing the following operations?
(Assume each operation is executed independently of each other, i.e., before "insert 5, then insert 7", the tree is exactly as shown in figure 1)

- Insert 17, height of the tree = ____2_____
- Insert 5, then insert 7, height of the tree = _____2_____
- Insert 32, height of the tree = ___3_____
- Insert 32, then delete 32, height of the tree = ___3_____
- Insert 32, then delete 32, then delete 31, height of the tree = ____3_____

b) (10 points) The B+ tree in figure 2 has an order of 4. Assume when storing the B- tree, a disk error happened, causing search-key 5 in the root node to be replaced by 12 (see the right tree in figure 2). The other part of the tree remains unchanged. Obviously, after this error, the tree is in an invalid state.
If we perform searches on this tree, searching which of the following values will return the wrong result (i.e., the search algorithm will say value doesn't exist, but it actually exists)? List of values: 4, 5, 9, 11, 17.

Due to the disk error in the B+ tree, the search key 5 was replaced with 12, leading to an incorrect search path for the value 11. Although 4, 5, 9, and 17 would still be found in their correct positions, searches for 11 will be misdirected to the left subtree due to the wrong root key, causing the search algorithm to falsely indicate that 11 does not exist in the tree.

c) In what situation, deleting one value is guaranteed to decrease the height of the tree by one? Make sure your explanation applies to B+ tree of any order and any height. (15 points)

Height decreases when the root node of a B+ tree is merged or redistributed with its child. This occurs when a deletion causes the number of keys in the root to fall below the minimum, and it only has one child left.

d) The following B+ tree has an order of 4 (see figure 3). Now let's delete random values from the tree until the height of the tree decreases from 3 to 2. For example, we delete 3, 5, 7, 9, 17. And only after 17 is deleted, the tree decreases height from 3 to 2. Let X denote the number of values deleted (in this example, X=5). Let S denote the list of deleted values (in this example, S=3,5,7,9,17).

If we start over (restore the tree) and delete different values (meaning S is different), X may still be 5, or a different value.

- **Give two more examples in which after the last value is deleted, the tree will decrease height from 3 to 2 (you only need to write down S and X). (5 points)**

Example 1:

- S = {1, 2, 3, 4, 5, 17}
- X = 6

Deleting the values 1, 2, 3, 4, and 5, the nodes containing these values would underflow, and removing the value 17 would cause the root to underflow, which would decrease the tree's height from 3 to 2.

Example 2:

- S = {12, 18, 33, 41, 5, 7}
- X = 6

Deleting 12, 18, 33, and 41 would lead to underflow in the leaf nodes and their parent nodes. Deleting 5 and 7 would eventually cause the root to underflow, thus decreasing the height.

- What is the minimum possible value of X? Explain your reasoning. (5 points; no points if no reasoning is presented)

The minimum possible value of X, or the fewest deletions needed to decrease the B+ tree's height from 3 to 2, is determined by causing enough underflows in the nodes at the second level so that they merge or redistribute, ultimately leading to the root node underflowing. This would typically be the deletion of all but two values in each of the second-level nodes.

- **What is the maximum possible value of X? Explain your reasoning. (5 points; no points if no reasoning is presented)**

The maximum value of X represents the largest number of deletions without affecting the tree's height. This entails deleting values right to the point where each node is on the brink of underflow, which is one less than the minimum number of values required to maintain the tree's structure. The specific value depends on the tree's initial value distribution and could involve removing nearly all values in each node just before it causes a height decrease.