Name

CWID

Quiz 1

CS525
Advanced Database Organization
Fall 2022

Due: Saturday, Oct 22nd 11:59pm

Results

Please leave this empty! 1.1.1 1.2.1 1.2.2 1.2.3 Sum

Instructions

- **You have to hand in the assignment via blackboard**
- **This is an individual and not a group assignment. Fraud will result in 0 points**
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are **2** parts in this quiz
  1. Disk Organization
  2. Index Structures

Part 1.1 Disk Organization (Total: 10 Points)

Question 1.1.1 Disk Access (10 Points)

Consider a disk with a sector size of 4096 bytes, 1000 tracks per surface, 50 sectors per track, five double-sided platters, and average seek time of 8 msec. Suppose that the disk platters rotate ate 7200 rpm. A block of 4096 bytes is chosen. Suppose that a file containing 150,000 records of 100 bytes each is to be stored on such a disk and that no record is allowed to span two blocks. Also, no block can span two tracks.

1. How many records fit onto a block?

   Solution

   $\left\lfloor \frac{\text{Block size}}{\text{Record size}} \right\rfloor = \left\lfloor \frac{4096}{100} \right\rfloor = \lfloor 40.96 \rfloor = 40.$
   We can have at most 40 records in a block.

2. How many blocks are required to store the entire file? If the file is arranged sequentially on the disk, how many cylinders are needed?

   Solution

   1 block can hold 40 records.
   Number of blocks needed for file = $\left\lceil \frac{\text{\# Records in File}}{\text{\# Records in Block}} \right\rceil = \left\lceil \frac{150{,}000}{40} \right\rceil = 3750$

   1 block = 1 sector
   1 track holds 50 blocks
   1,000 tracks per surface
   5 double-sided platters
   1 cylinder holds 10 tracks
   1 cylinder holds 500 blocks
   File is 3750
   \# of cylinders = $\left\lceil \frac{\text{\# Blocks to store File}}{\text{\# Blocks in Cylinder}} \right\rceil = \left\lceil \frac{3{,}750}{500} \right\rceil = 7.5 \approx 8$ Cylinders.

3. How many records of 100 bytes each can be stored using this disk?

   Solution

   \# of Blocks in the Disk = \# Blocks per track $\times$ \# tracks $\times$ \# surfaces
   $= 50 \times 1{,}000 \times 10 = 500{,}000$ Blocks
   1 block can hold 40 records.
   The disk can store $40 \times 500{,}000 = 20{,}000{,}000$ records.

4. What time is required to read a file containing 100,000 records of 100 bytes each sequentially? You can assume that the time for moving from one cylinder to another is very small.

   Solution

   100,000 records of 100 bytes is stored in $\left\lfloor \frac{100{,}000}{40} \right\rfloor = 2{,}500$ blocks.
   1 cylinder holds 500 blocks.
   File required $\frac{2{,}500}{500} = 5$ cylinders
   Time to read one track (one rotation) = $\frac{60s}{7{,}200} = 0.0083$ seconds
   Time to read one cylinder = $10 \times 0{,}0083 = 0.083$ seconds
   Time to read the entire file (read 5 cylinders) = $5 \times 0.083 = 0.415$ seconds
   The average seek time is 8 msec = 0.008 seconds
   The average rotational delay is 0.00415 seconds.
   Therefore, the total time is 0.415 + 0.00415 + 0.008 = 0.42715 seconds.

Part 1.2 Index Structures (Total: 50 Points)

Question 1.2.1 B$^+$-tree Construction (10 Points)

Assume that you have the following table:

Item

SSN	name	age
11	Pete	13
24	Bob	16
25	Heinz	55
26	John	44
28	Manny	33
32	Gertrud	65
34	Alice	71
39	Lily	12
44	Sammy	11
46	Joe	17

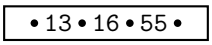
Create a B$^+$-tree for table **Item** on key *age* with $n = 3$ (up to three keys per node). Assume that the tree is initially empty and values are added in the order shown in the table above.

Write down the resulting B$^+$-tree after each step and when splitting or merging nodes follow these conventions:

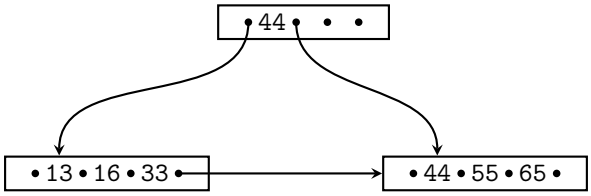
- **Leaf Split**: In case a leaf node needs to be split during insertion and n is even, the left node should get the extra key. E.g, if $n = 2$ and we insert a key 4 into a node [1,5], then the resulting nodes should be [1,4] and [5]. For odd values of n we can always evenly split the keys between the two nodes. In both cases the value inserted into the parent is the smallest value of the right node.
- **Non-Leaf Split**: In case a non-leaf node needs to be split and n is odd, we cannot split the node evenly (one of the new nodes will have one more key). In this case the "middle" value inserted into the parent should be taken from the right node. E.g., if $n = 3$ and we have to split a non-leaf node [1,3,4,5], the resulting nodes would be [1,3] and [5]. The value inserted into the parent would be 4.
- **Node Underflow**: In case of a node underflow you should first try to redistribute values from a sibling and only if this fails merge the node with one of its siblings. Both approaches should prefer the left sibling. E.g., if we can borrow values from both the left and right sibling, you should borrow from the left one.

Solution

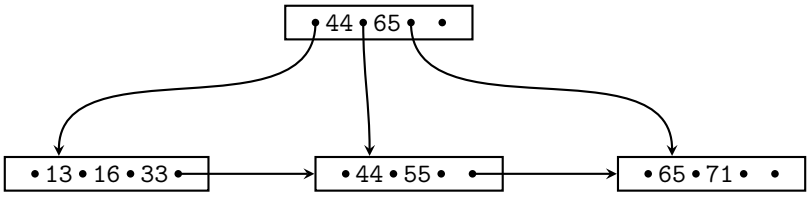
Insert 13, 16, 55



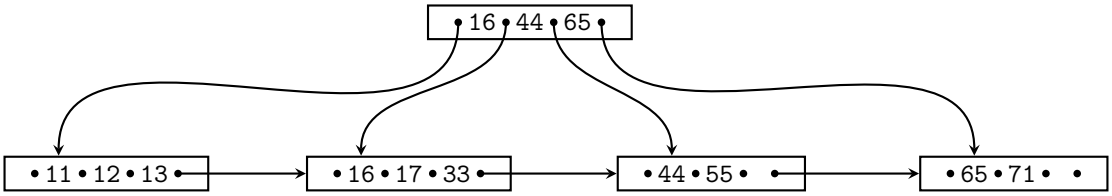
Insert 44, 33, 65



Insert 71



Insert 12, 11, 17

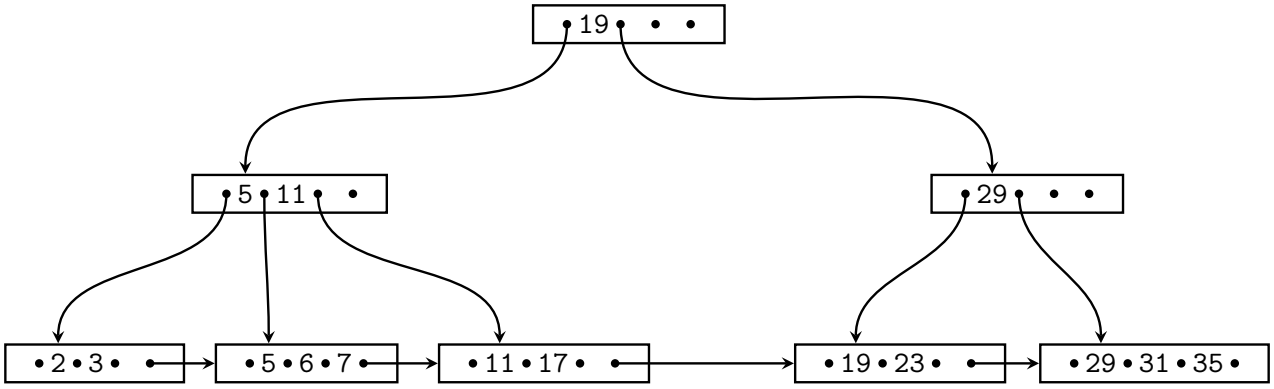


Question 1.2.2 Operations (20 Points)

Given is the B$^+$-tree shown below ($n = 3$). Execute the following operations and write down the resulting B$^+$-tree after each operation:

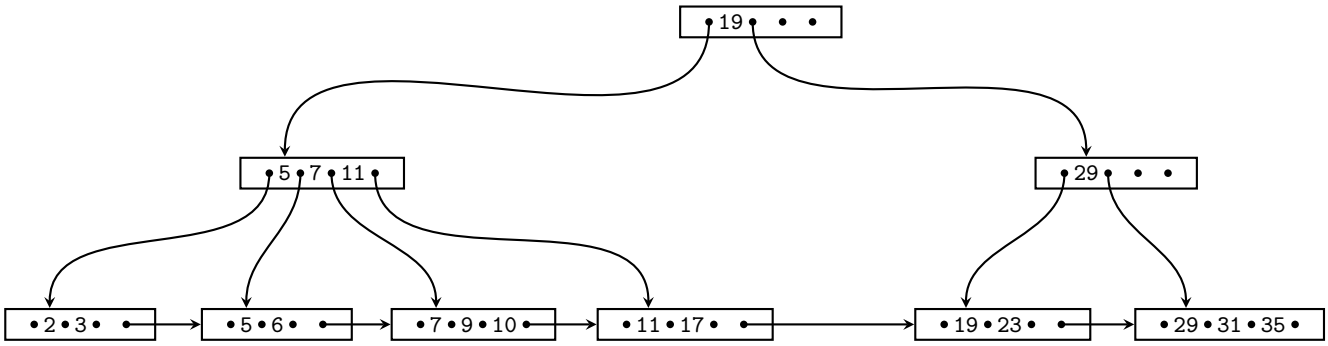
insert(9), insert(10), delete(2), insert(40), delete(29), delete(19)

Use the conventions for splitting and merging introduced in the previous question.

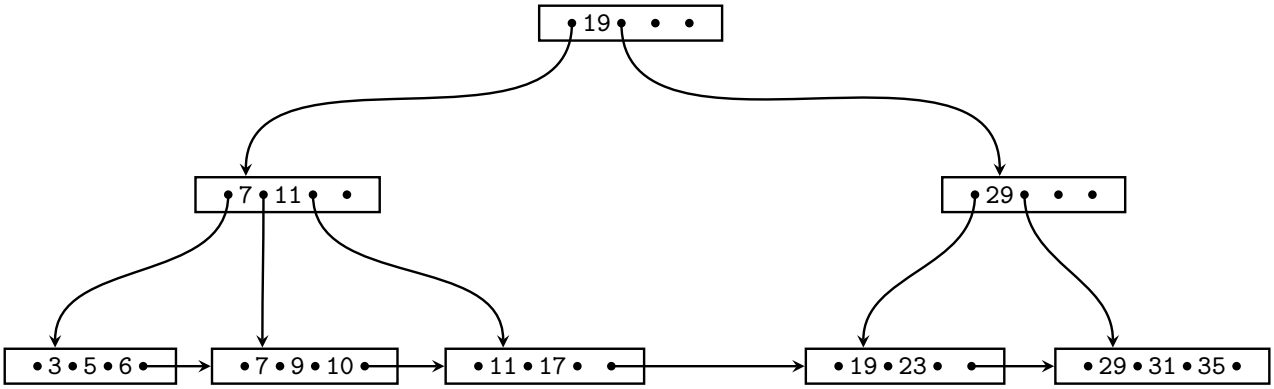


Solution

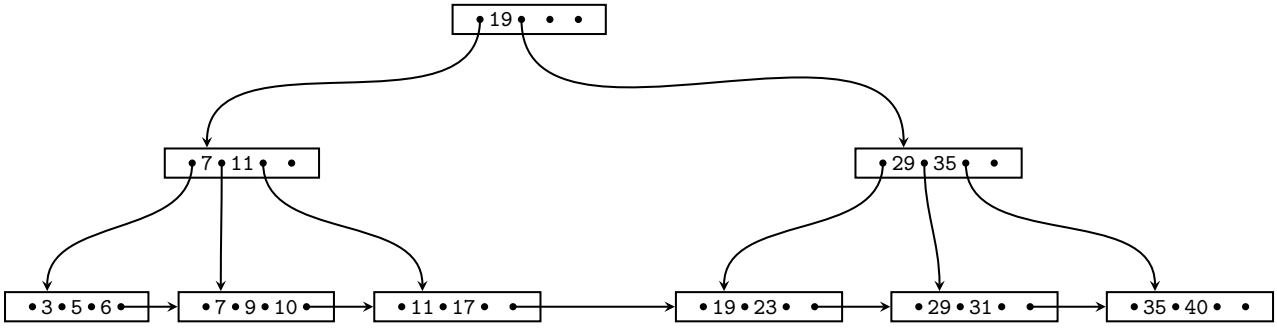
insert(9), then insert(10)



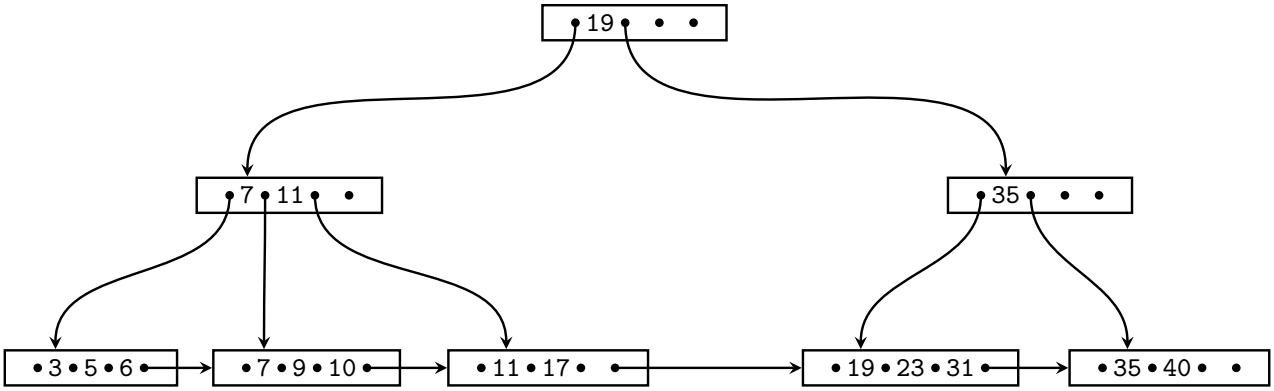
delete(2)



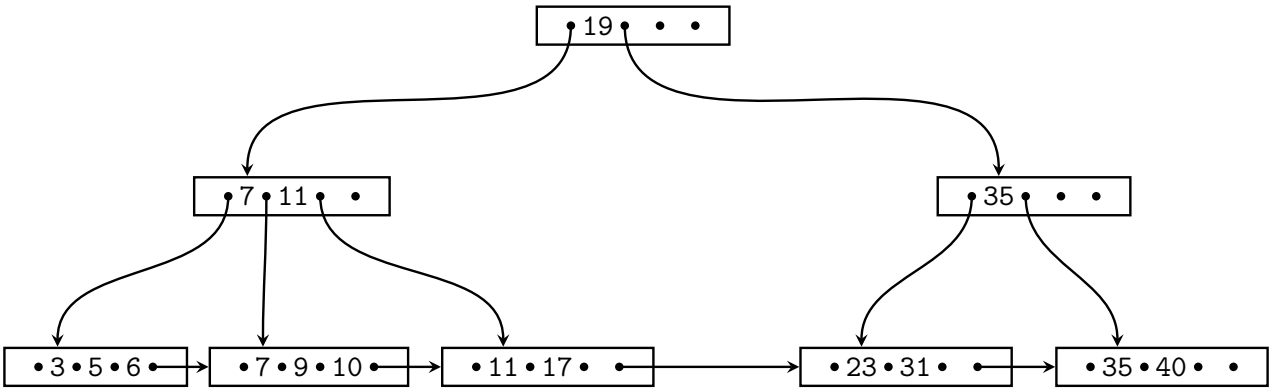
insert(40)



delete(29)



Solution
delete(19)



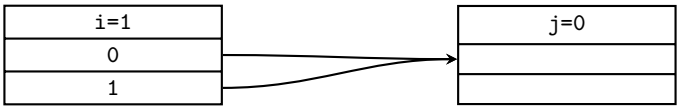
Question 1.2.3 Extensible Hash Construction (20 Points)

Suppose that we are using Extensible hashing on a file. Each bucket holds two keys. Execute the following operations
`insert(4), insert(1), insert(7), insert(8), insert(6), insert(0), insert(2), insert(3)`
and write down the resulting index after each operation. Assume the hash function is defined as:

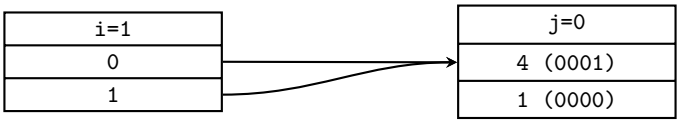
x	$h(x)$
0	1101
1	0000
2	1010
3	1100
4	0001
5	0000
6	1010
7	0111
8	1110

Solution

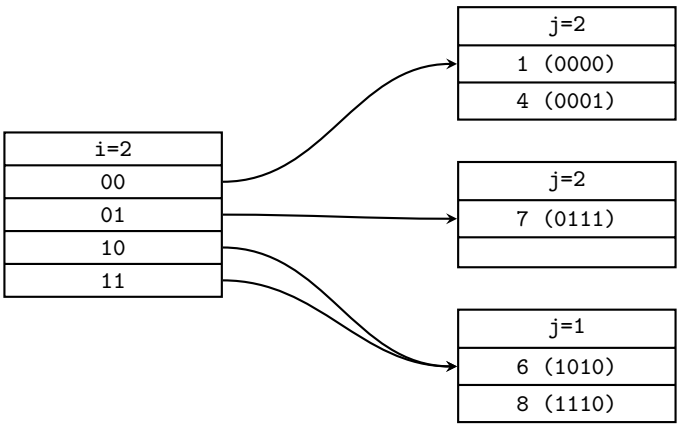
`Initial Index Table`



`Insert 4 (0001), 1 (0000)`

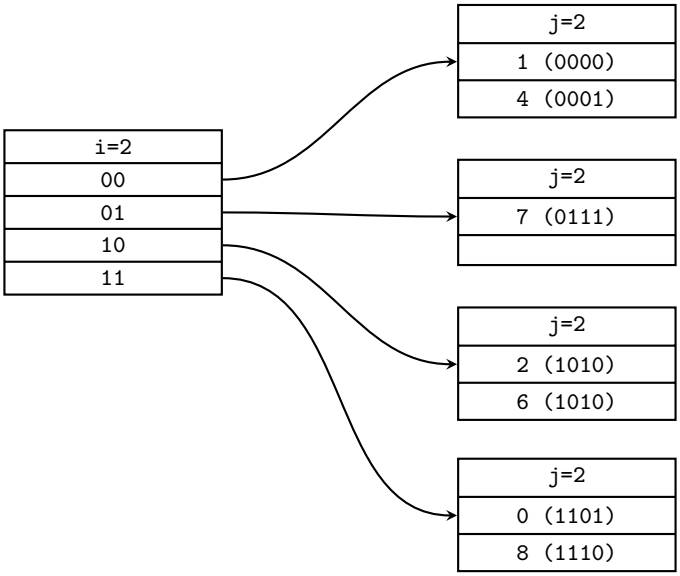


`Insert 7 (0111), then 8 (1110), 6(1010)`



Solution

Insert 0 (1101), then 2 (1010)



Insert 3 (1100)