Name	CWID

Quiz 1

CS525
Advanced Database Organization
Spring 2021

Due: March 25th 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 using your 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: 18 Points)

Question 1.1.1 Disk Access (18 Points)

Consider a disk with a sector size of 512 bytes, 2000 tracks per surface, 50 sectors per track, five double-sided platters, and average seek time of 10 msec. Suppose that a block size of 1024 bytes is chosen. Suppose that a file containing $100,000$ records of 100 bytes each is to be stored on such a disk and that no record is allowed to span two blocks.

[3 Points each]

1. What is the capacity of a track in bytes? What is the capacity of each surface? What is the capacity of the disk?

   Solution

   ```
   bytes/track= bytes/sector × sectors/track=512×50=25,600 bytes≈25K
   bytes/surface= bytes/track×tracks/surface=25,600×2000=51,200,000 bytes≈50,000K
   bytes/disk=bytes/surface×surfaces/disk=51,200,000×5×2=512,000,000 bytes≈500,000K
   ```

2. How many records fit onto a block?

   Solution

   $\left\lfloor \frac{\texttt{Block size}}{\texttt{Record size}} \right\rfloor = \left\lfloor \frac{1024}{100} \right\rfloor$ = 10. We can have at most 10 records in a block.

3. How many blocks are required to store the entire file?

   Solution

   ```
   File size = # Record × Record size
   File size = 100,000 \times 100 = 10,000,000 bytes
   ```
   We need $\left\lceil \frac{\texttt{File size}}{\texttt{Block size}} \right\rceil = \left\lceil \frac{10,000,000}{1024} \right\rceil$ = 9765.625 ≈ 10,000 blocks to store the file.

4. If the file is arranged sequentially on the disk, how many surfaces are needed?

   Solution

   ```
   # Sectors/Tracks = 50, Sector size = 512 bytes, Block size = 1024 bytes
   One track has 25 blocks,
   One cylinder has 25 × 5 × 2 = 250 blocks.
   We need 10,000 blocks to store this file.
   ```
   So, we need $\left\lceil \frac{10,000}{250} \right\rceil$ = 40 cylinders
   ```
   We  need  10 surfaces to store the file.
   ```

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

   Solution

   ```
   The capacity of the disk is 512,000,000 bytes≈500,000K, which has≈500,000blocks.
   Each block has 10 records.
   The disk can store no more than 512,000,000 ≈ 5,000,000 records.
   ```

6. If pages are stored sequentially on disk, with page 1 on block 1 of track 1, what page is stored on block 1 of track 1 on the next disk surface?

   Solution

   ```
   There are 25 blocks in each track.
   It is block 26 on block 1 of track 1 on the next disk surface.
   ```

Part 1.2 Index Structures (Total: 54 Points)

Question 1.2.1 `B+-tree` Construction (10 Points)

Assume that you have the following table:

Item

SSN	name	age
2	Pete	13
3	Bob	23
29	Heinz	14
5	John	49
19	Manny	33
23	Gertrud	29
11	Alice	77
17	Lily	3
39	Sammy	34
7	Joe	45

Create a `B+-tree` for table **Item** on key *SSN*. Assume that the tree is initially empty and values are added in the order shown in the table above. Construct `B+-tree` for the cases where the **number of pointers that will fit in one node** is as follows:

a. Three

b. Six

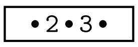
Write down the resulting `B+-tree` after each step and when splitting or merging nodes follow these conventions:

- **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.

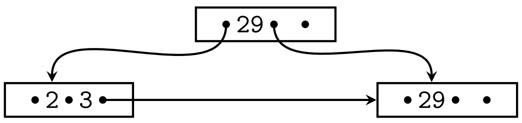
Solution

a. Three

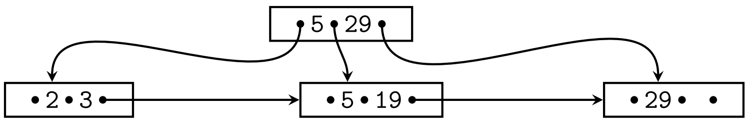
Insert 2, 3



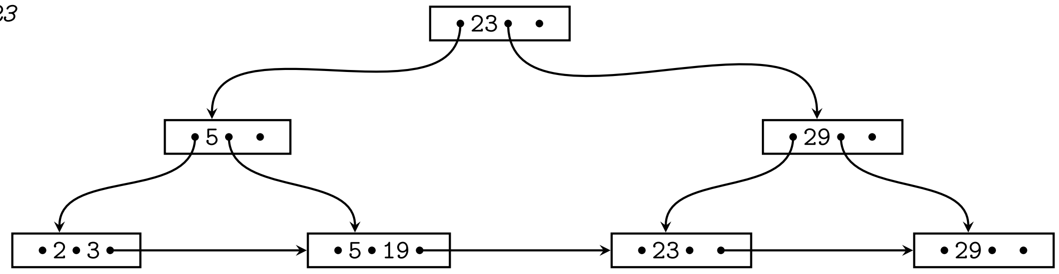
Insert 29, 5



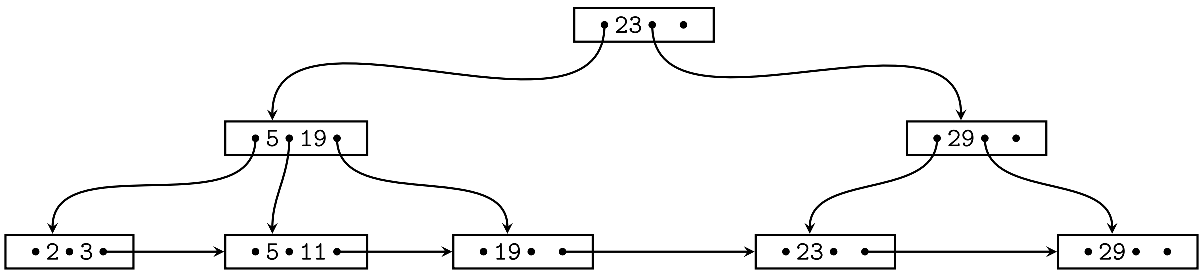
Insert 5, 19



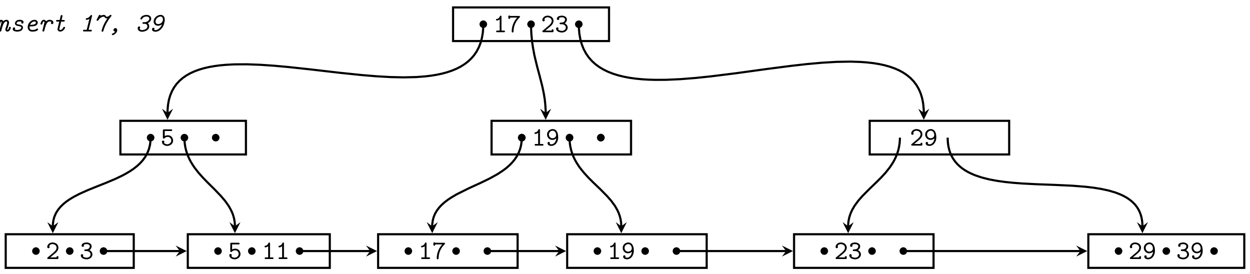
Insert 23



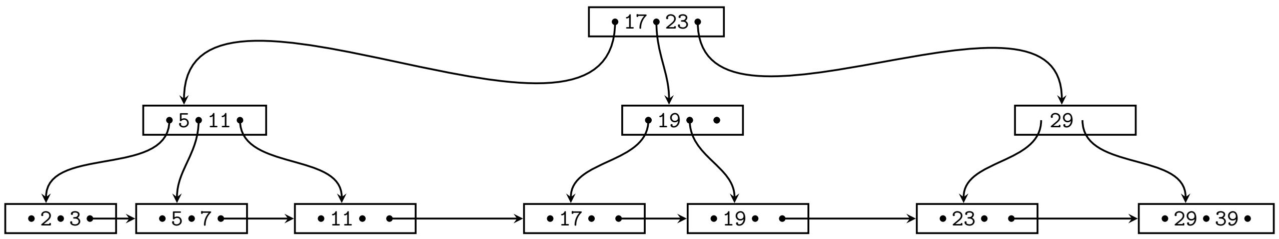
Insert 11



Insert 17, 39

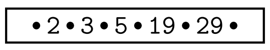


Insert 7

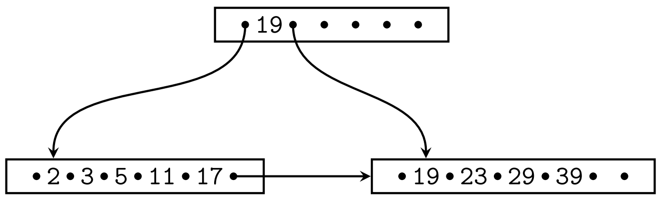


Solution

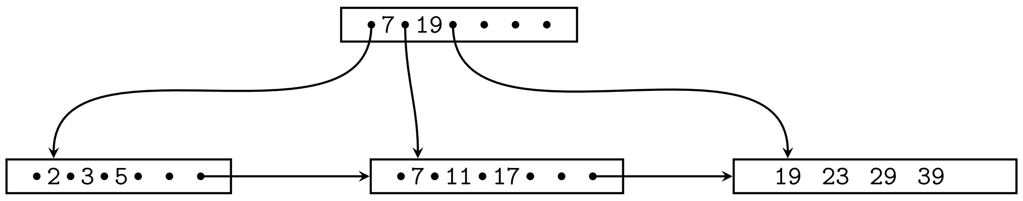
b. Six
Insert 2, 3, 29, 5, 19



Insert 23, 11, 17, 39



Insert 7

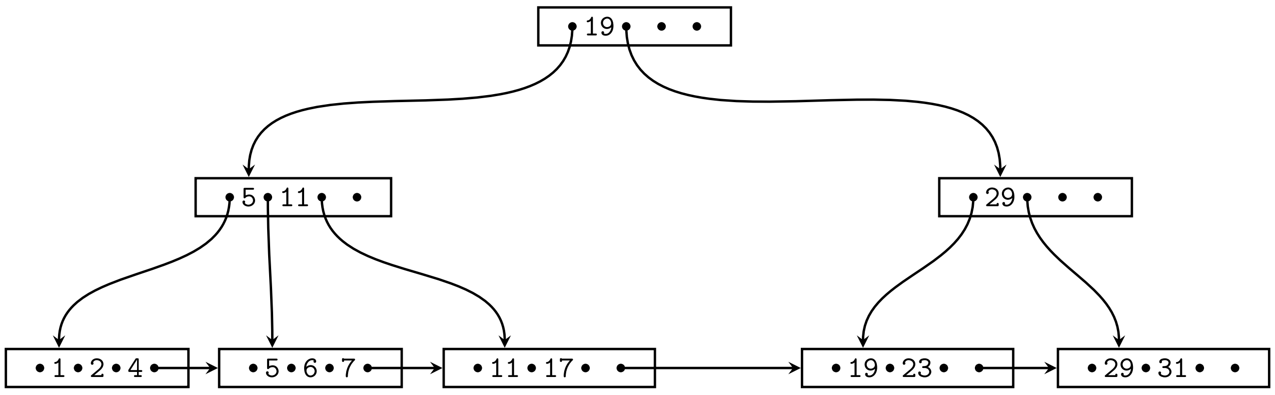


Question 1.2.2 Operations (20 Points)

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

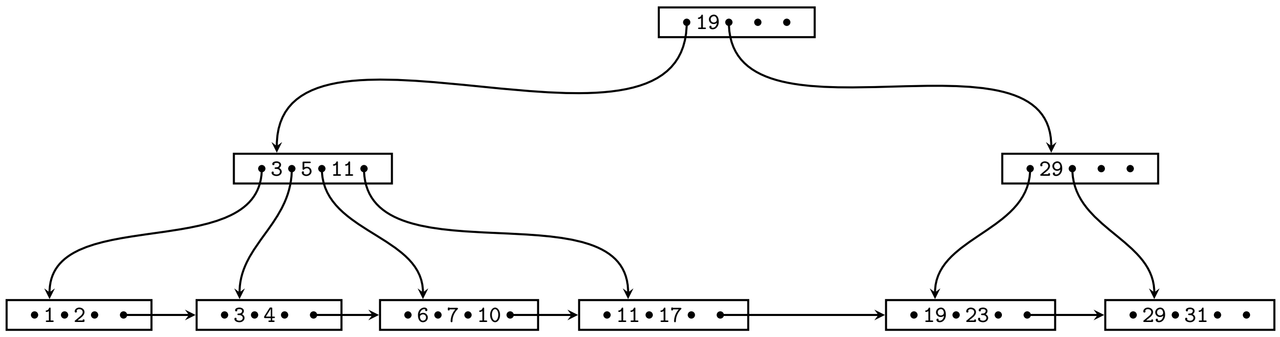
insert(3), delete(5), insert(10), insert(8), delete(29), delete(19), delete(11)

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

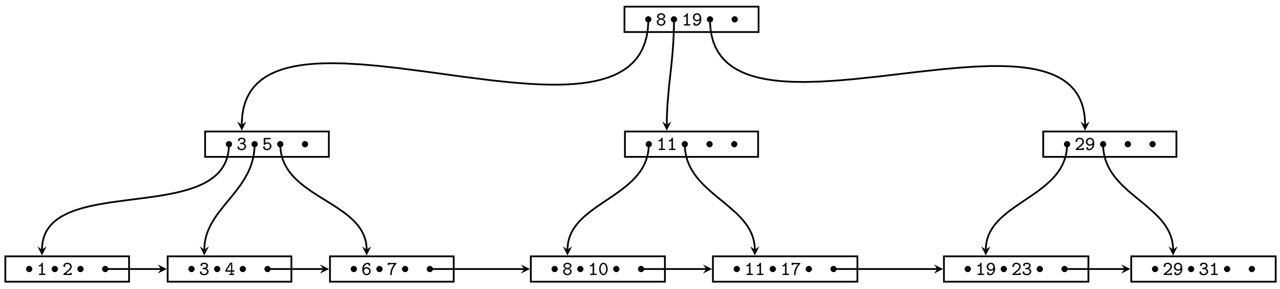


Solution

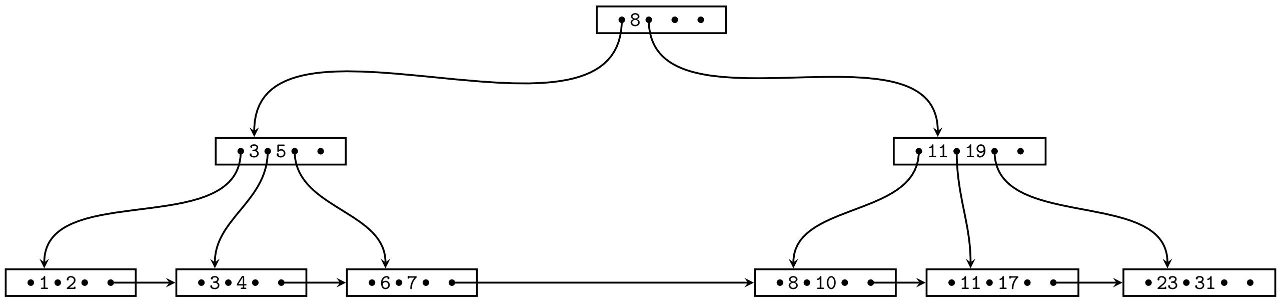
insert(3), delete (5), insert (10)



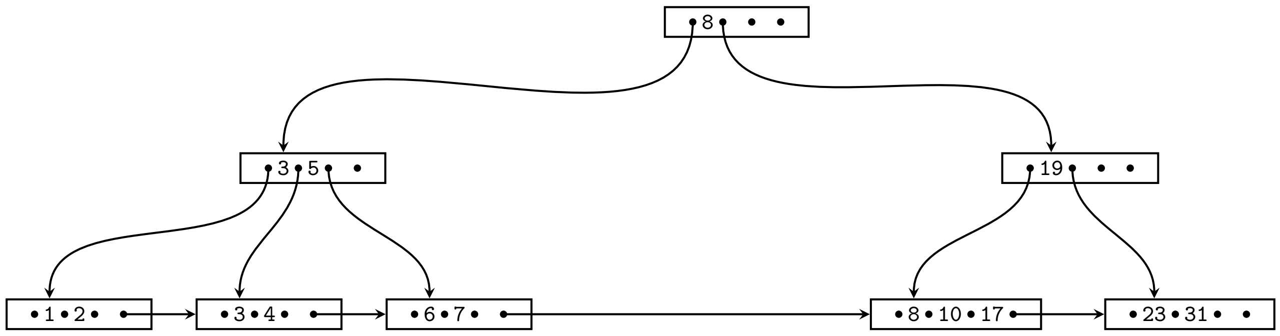
insert(8)



delete(29), then delete(19)



delete(11)



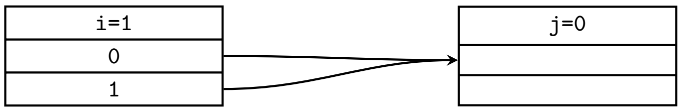
Question 1.2.3 Extendable Hashing-tree Construction (24 Points)

Suppose that we are using extendable hashing on a file that contains records with the following search key values: `7, 15, 44, 11, 19, 3, 33, 17, 14, 43, 18, 20`

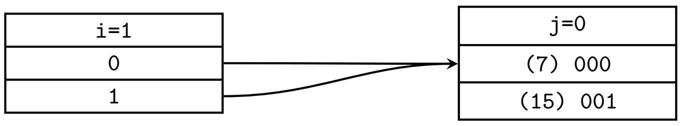
Show the extendable hash structure for this file if the hash function is *h(x)=x mod 7* and buckets can hold 2 records

Solution

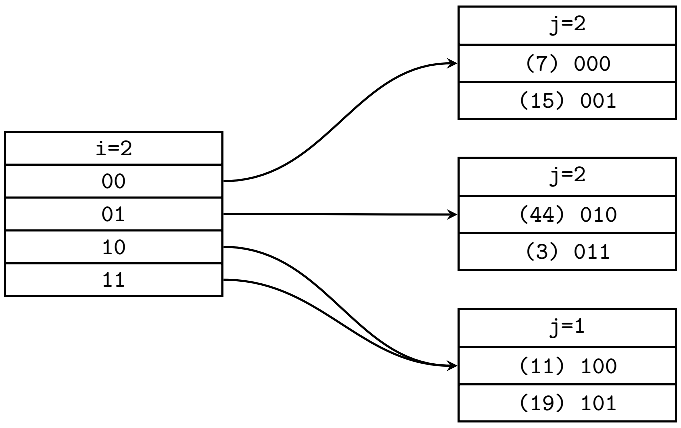
`Initial Index Table`



`Insert 7, 15`

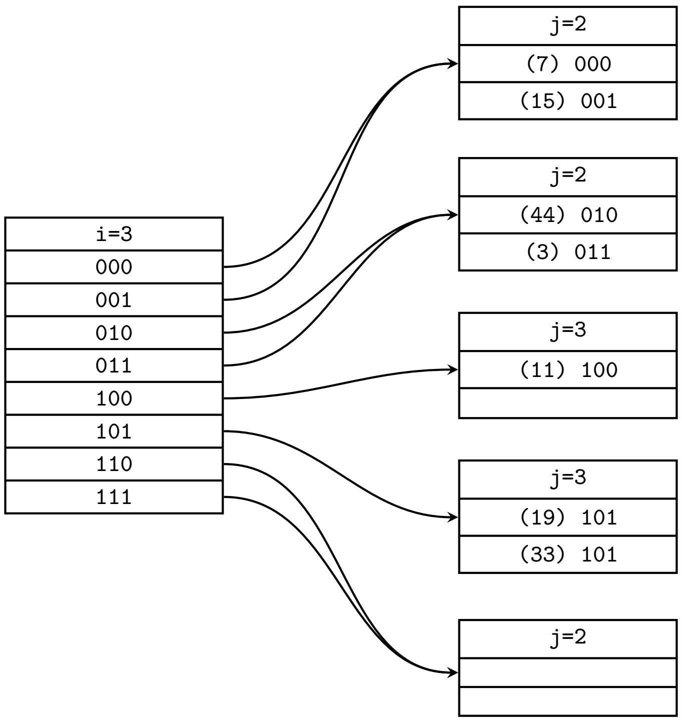


`Insert 44, 11, 19, 3`

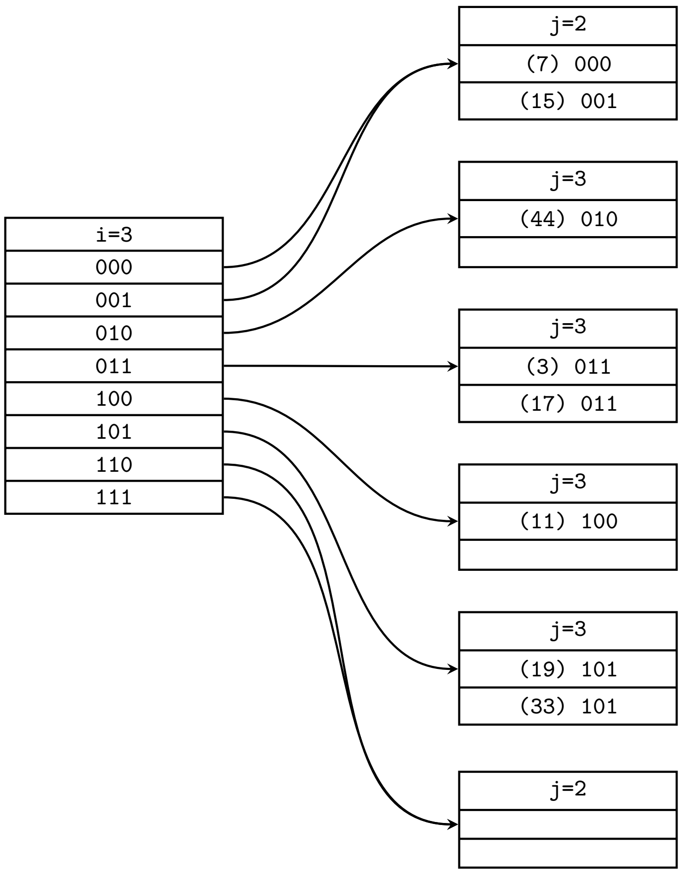


Solution

Insert 33

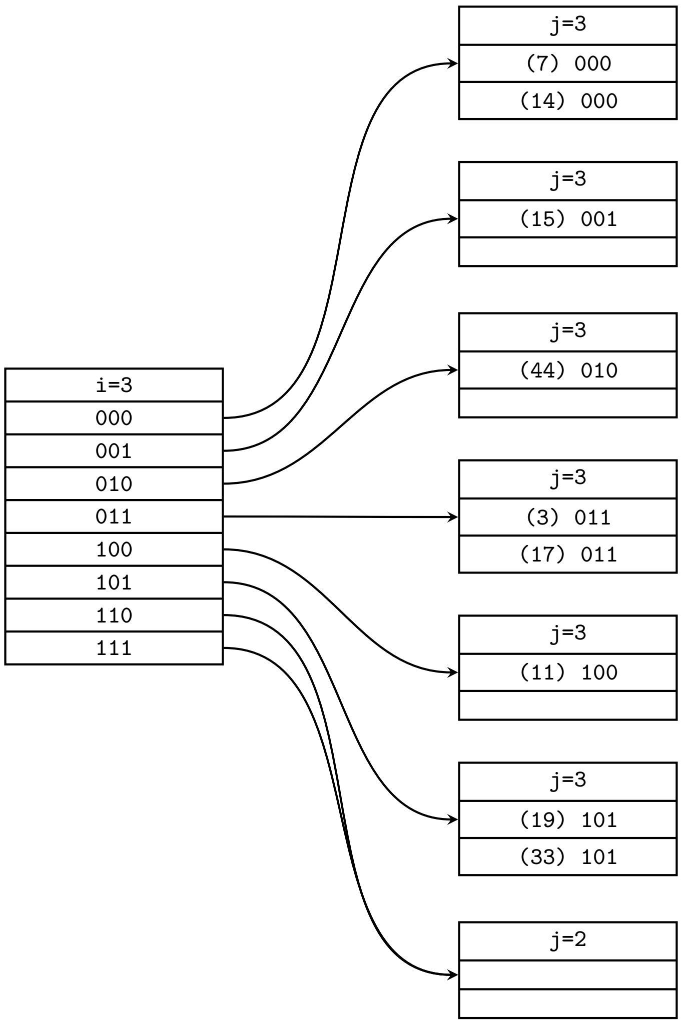


Insert 17



Solution

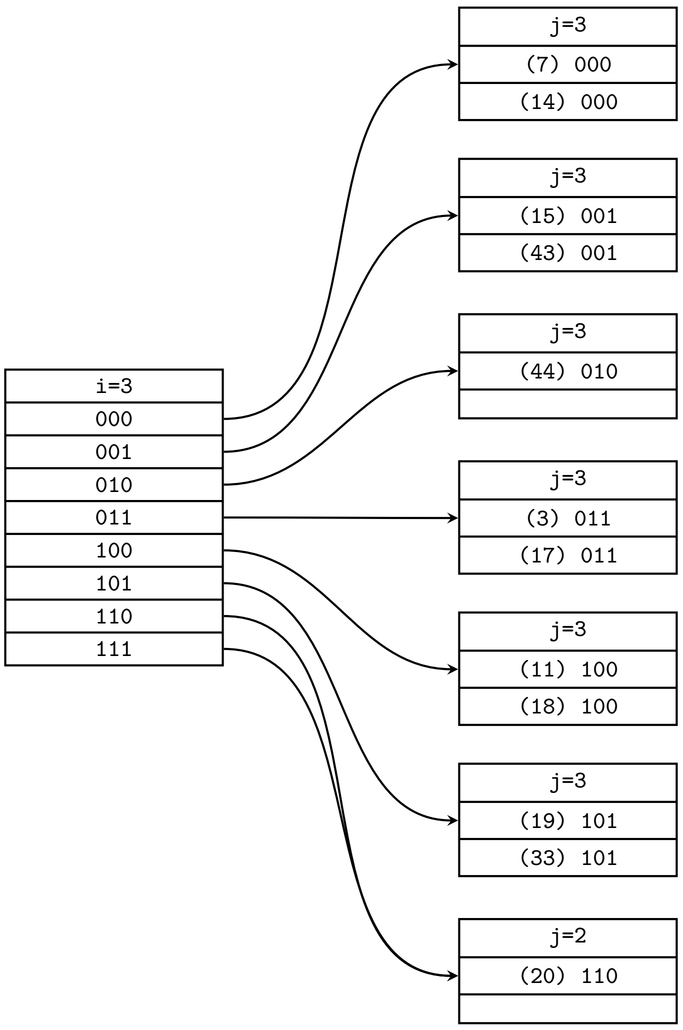
Insert 14



i=3
000
001
010
011
100
101
110
111
j=3
(7) 000
(14) 000
j=3
(15) 001
j=3
(44) 010
j=3
(3) 011
(17) 011
j=3
(11) 100
j=3
(19) 101
(33) 101
j=2

Solution

`Insert 43, 18, 20`



i=3
000
001
010
011
100
101
110
111
j=3
(7) 000
(14) 000
j=3
(15) 001
(43) 001
j=3
(44) 010
j=3
(3) 011
(17) 011
j=3
(11) 100
(18) 100
j=3
(19) 101
(33) 101
j=2
(20) 110