CAS CS 560: Introduction to Database Systems
Solutions to Homework #3
Spring 2003

Problem 1:

(1) The capacity of a track is $512 \times 800 = 4.096 \times 10^5$ bytes
The capacity of a surface is $512 \times 800 \times 10,000 = 4.096 \times 10^9$ bytes
The capacity of the disk is $4.096 \times 10^9 \times 5 \times 2 = 4.096 \times 10^{10}$ bytes.

(2) 10^4

(3) The maximum rotation time is $60 \div 11,000 = 5.45$ ms
The average rotation time is $5.45 \div 2 = 2.73$ ms.

(4) $4.096 \times 10^5 \div (5.45 \times 10^{-3}) = 75$ megabytes / second.

Problem 3:

1. (i) The number of blocks needed for the actual file is: $10^6/10 = 10^5$.
The height of B+ tree is: $\lceil \log 10^6 / \log 69 \rceil = 4$.
The number of keys in B+ tree is: $10^6 \times (1 + 1/69 + 1/69^2 + 1/69^3) = 1.0147 \times 10^6$.
Then the total number of blocks needed is: $10^5 + 1.0147 \times 10^6/69 = 1.147 \times 10^5$.

   (ii) You need 4 disk I/O's to traverse the tree in order to find the pointer for the record, then use another I/O to get the actual record. Therefore, given its search key, it needs 5 I/O's to retrieve a record.

   (iii) Since the records are stored sequentially, one only needs to find the pointer to the beginning record of the range query, retrieve it from the block and keep retrieving the next 999 records in the consecutive blocks.
The number of I/O's is: $5 + 1000/10 = 105$.

2. (i) The B+ tree now only stores pointers to the first record of each block. So the leaf node will store 10^5 keys instead of 10^6.
The height of tree becomes: $\lceil \log 10^5 / \log 69 \rceil = 3$.
The number of keys in B+ tree would be: $10^5 \times (1 + 1/69 + 1/69^2) = 1.0147 \times 10^5$.
The total number of blocks needed is: $10^5 + 1.0147 \times 10^5/69 = 1.0147 \times 10^5$.

   (ii) We need 4 I/O's to retrieve the correct block, and within the block, we can find the record.

   (iii) As in 1, once we find the first record, we can retrieve the next 999 records in consecutive blocks. So the total number of I/O's needed is: $4 + 1000/10 = 104$.

3. (i) Same as 1.

   (ii) Same as 1.

   (iii) Since the records are stored in no particular order, we need to find pointers for each one of the 1000 records. But fortunately, B+ tree is ordered. We use 4 to find the pointer for the

beginning record, from there, we can find the next 999 pointers in the consecutive blocks, and then get the actual records.
The number of I/O's = the number of I/O's for the pointers + the number of I/O's for the actual records = $(4 + \lceil 999/69 \rceil) + 1000 = 1019$

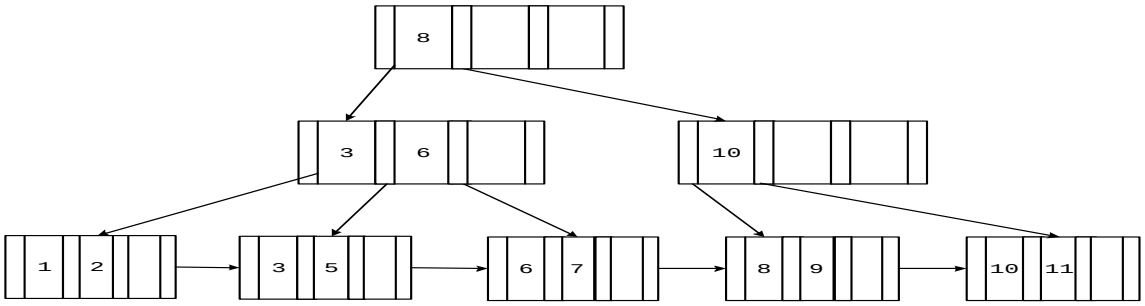
4. (i) The number of blocks needed for the leaf level is: $10^6/(10 \times 0.7) = 10^6/7$.
The number of blocks needed for other nodes in B+ tree is: $10^6 \times (1/69 + 1/69^2 + 1/69^3)/69 = 14700/69$.
The total number of blocks is: $10^6/7 + 14700/69 = 1.4307 \times 10^5$.

(ii) 4

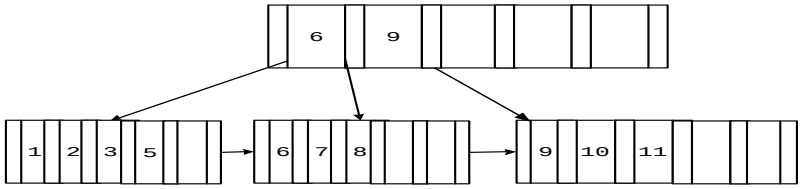
(iii) We need 5 I/O's to find the first record, then retrieve the next 999 in consecutive blocks.
$5 + 999/(10 \times 7) = 148$.

Problem 2
(a) 2 possible results

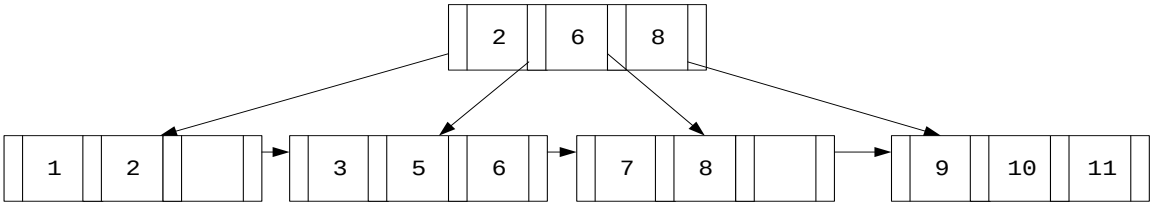
4 pointers case



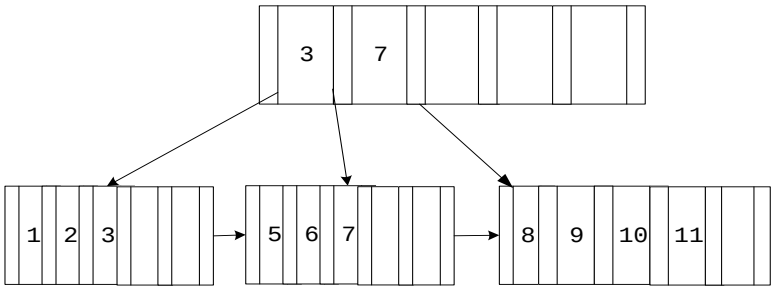
6 pointers case



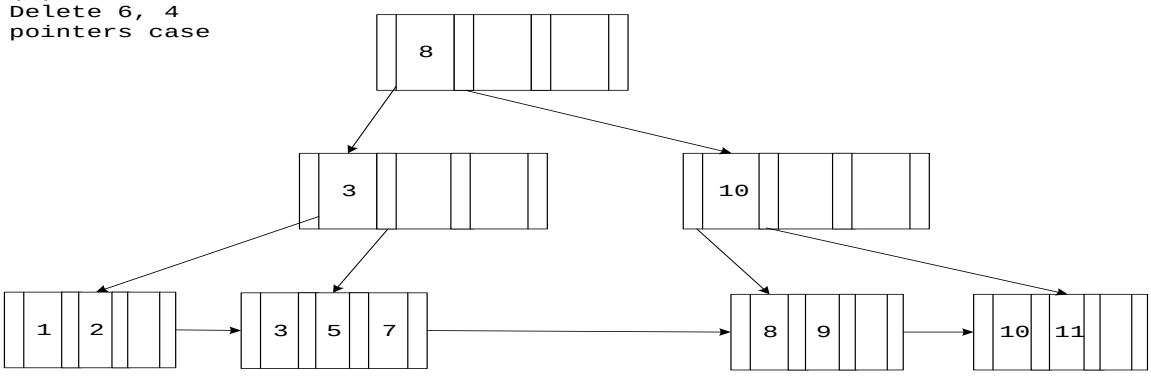
4 pointers case



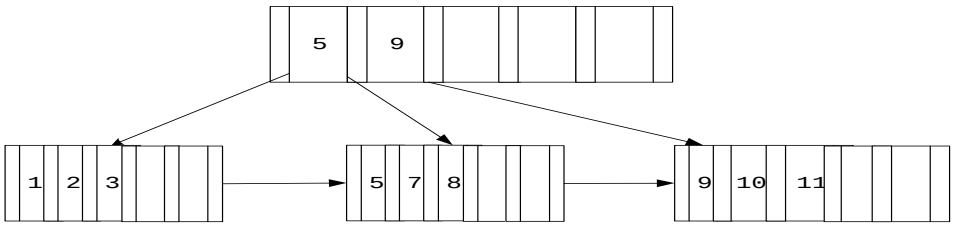
6 pointers case



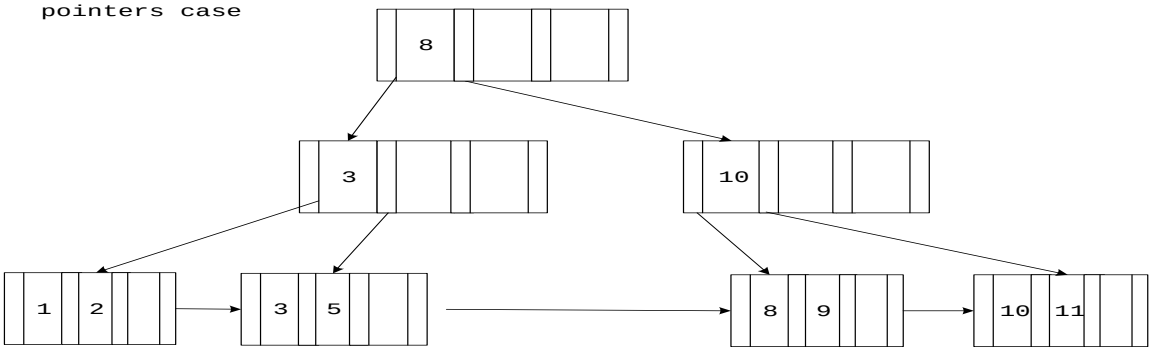
(b)
Delete 6, 4
pointers case



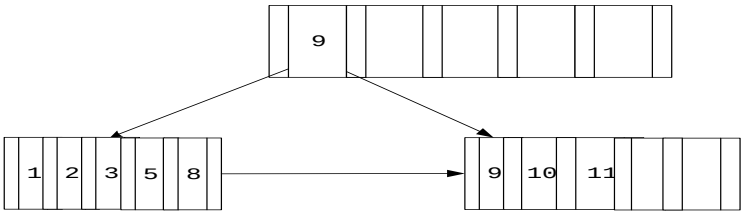
Delete 6, 6
pointers case



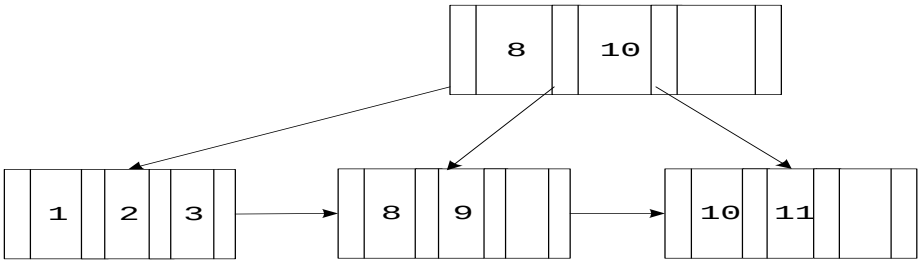
Delete 7, 4 pointers case



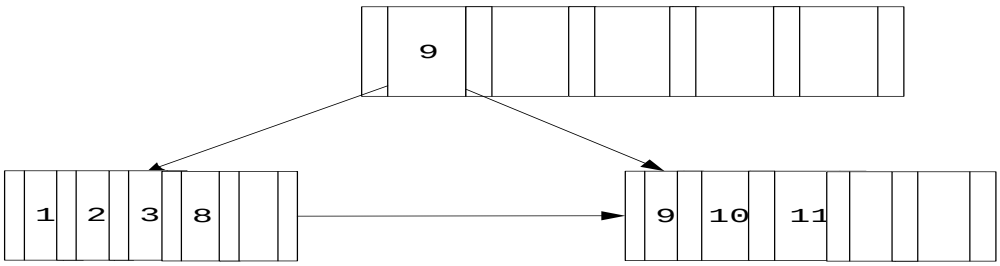
Delete 7, 6 pointers case



Delete 5, 4 pointers case



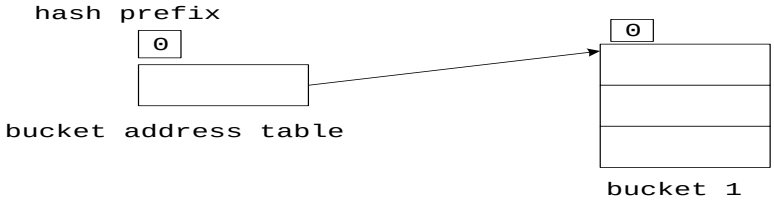
Delete 5, 6 pointers case



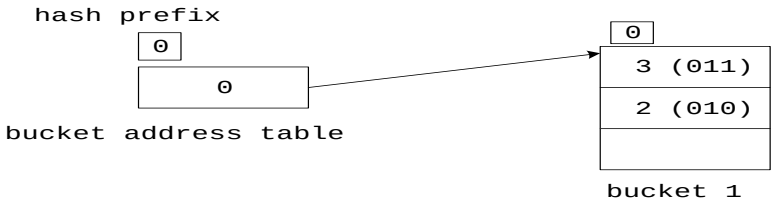
Problem 4

Values:	H(x)	bits
3	3	011
2	2	010
5	5	101
15	7	111
19	3	011
17	1	001
26	2	010
21	5	101
55	7	111

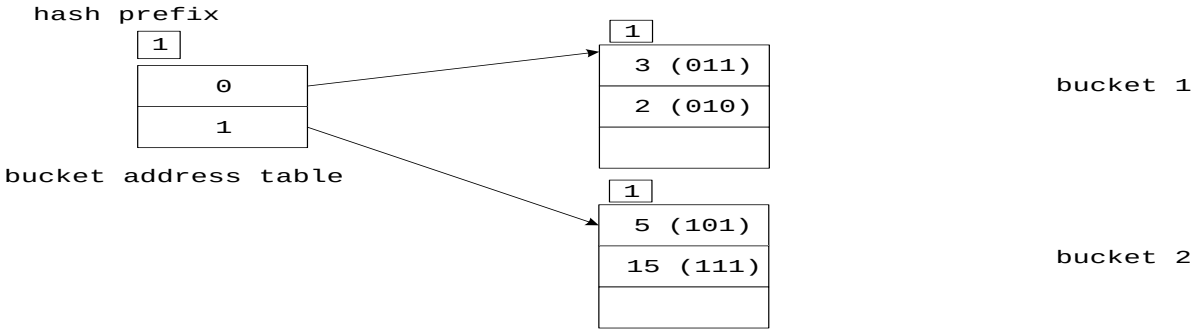
1. Initial extendable hash structure



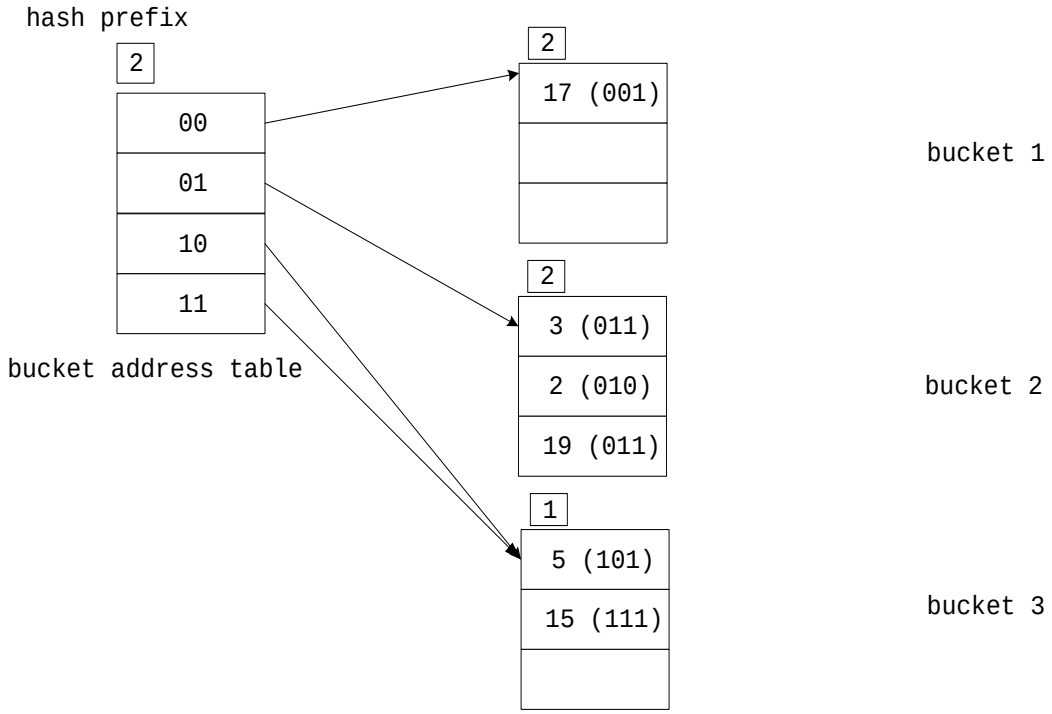
2. insert 3 and 2



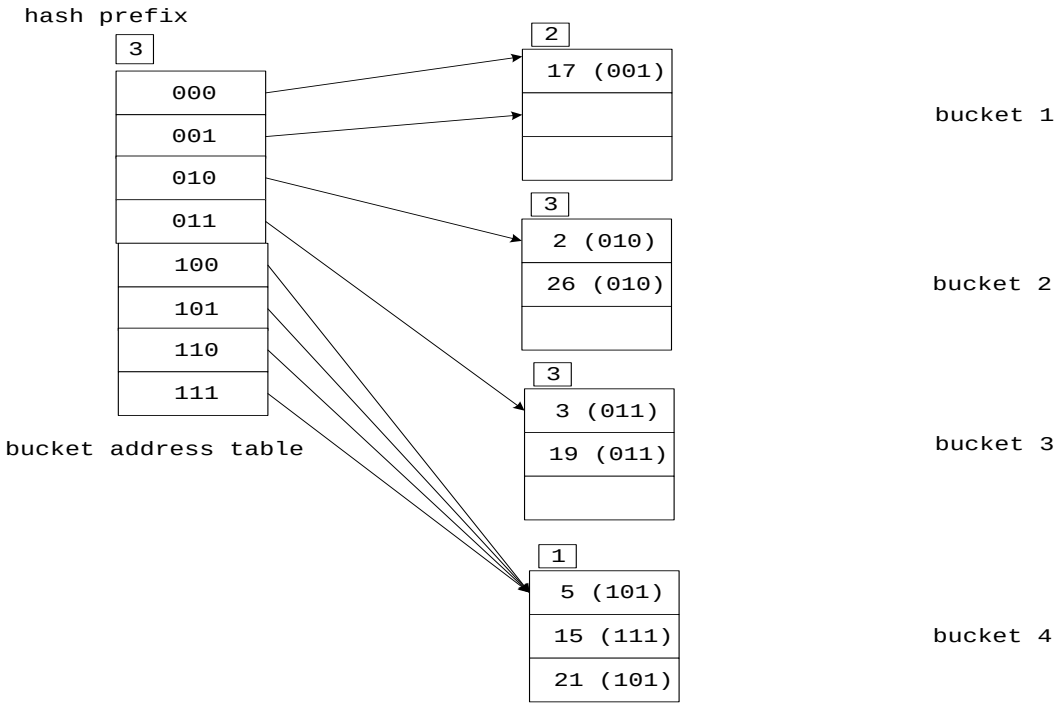
3. insert 5 and 15



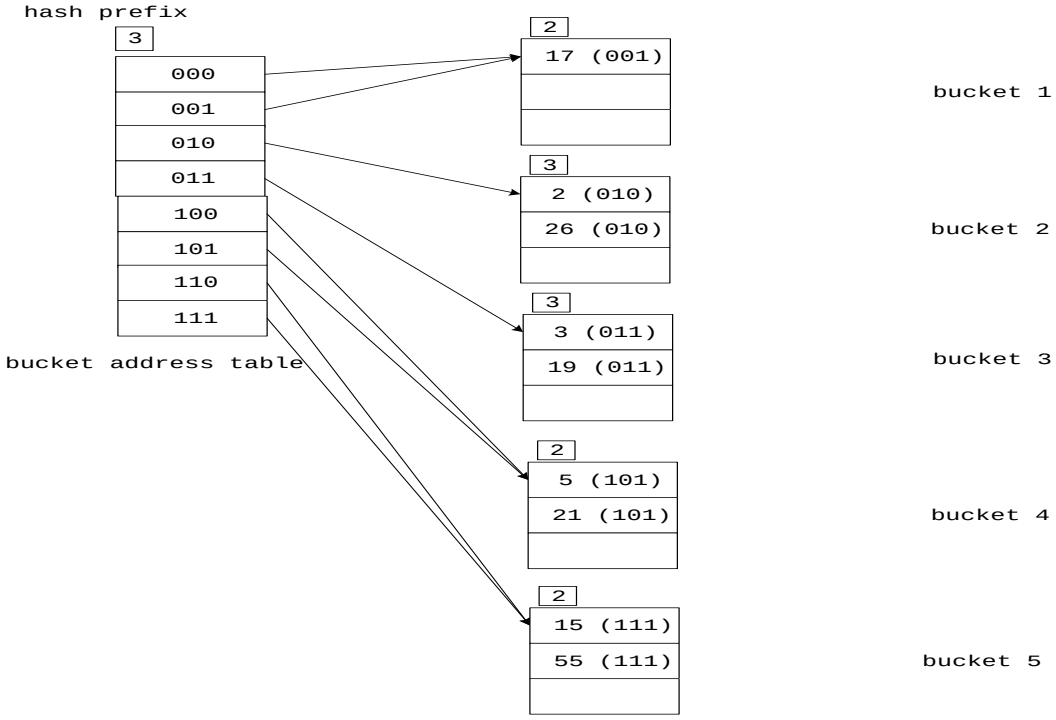
4. insert 19 and 17



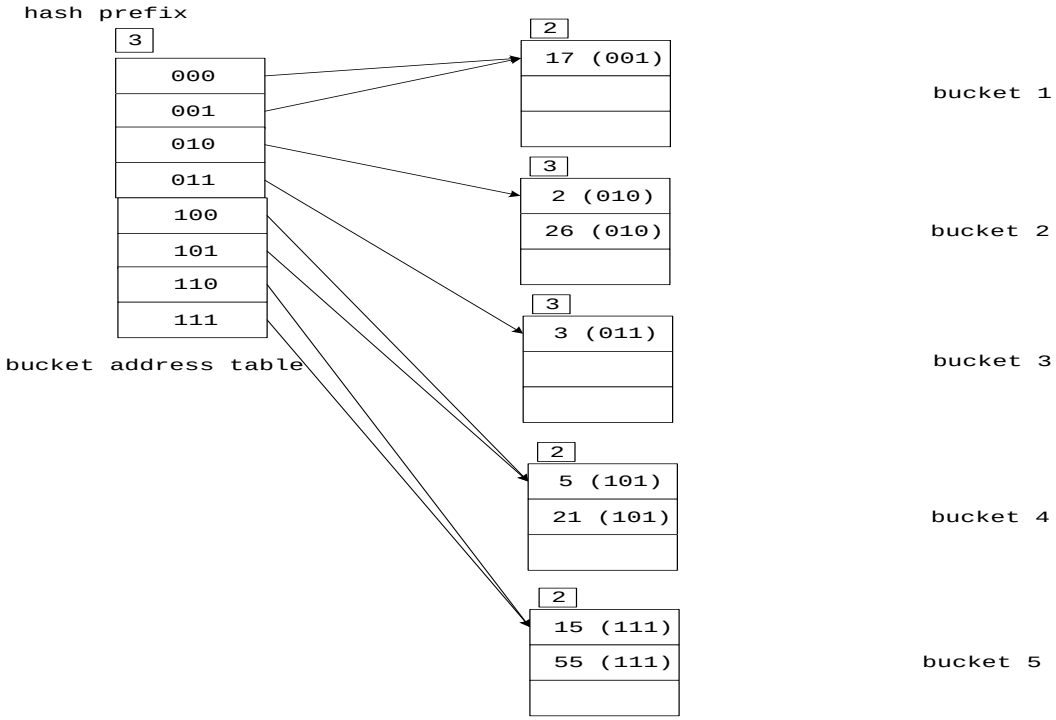
5. insert 26 and 21



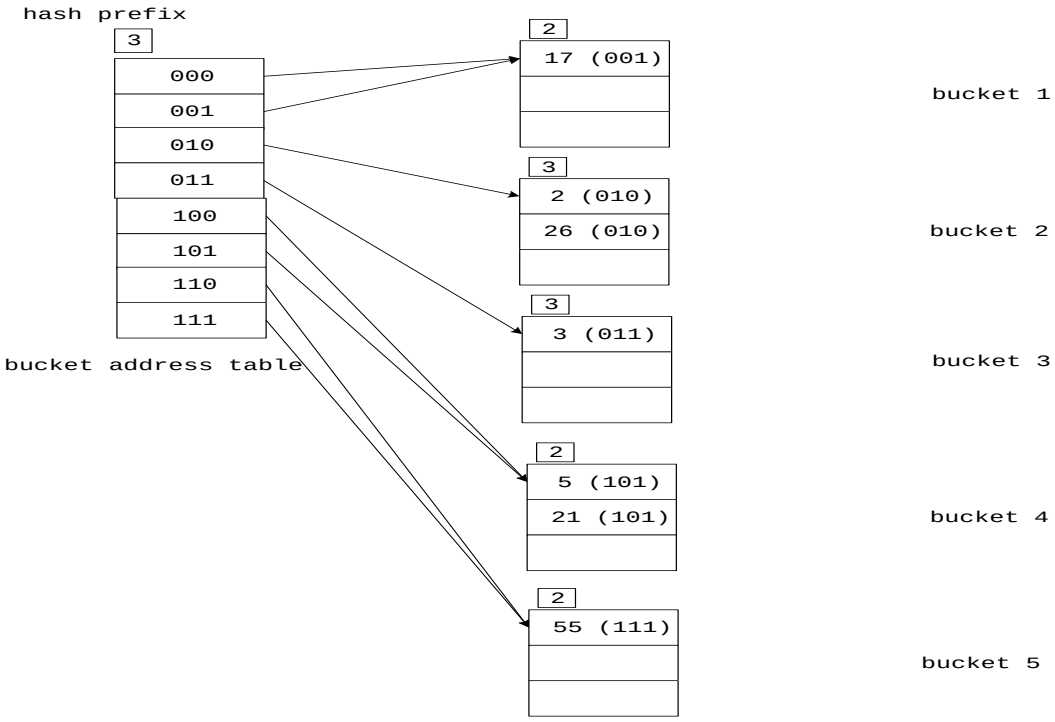
5. insert 55



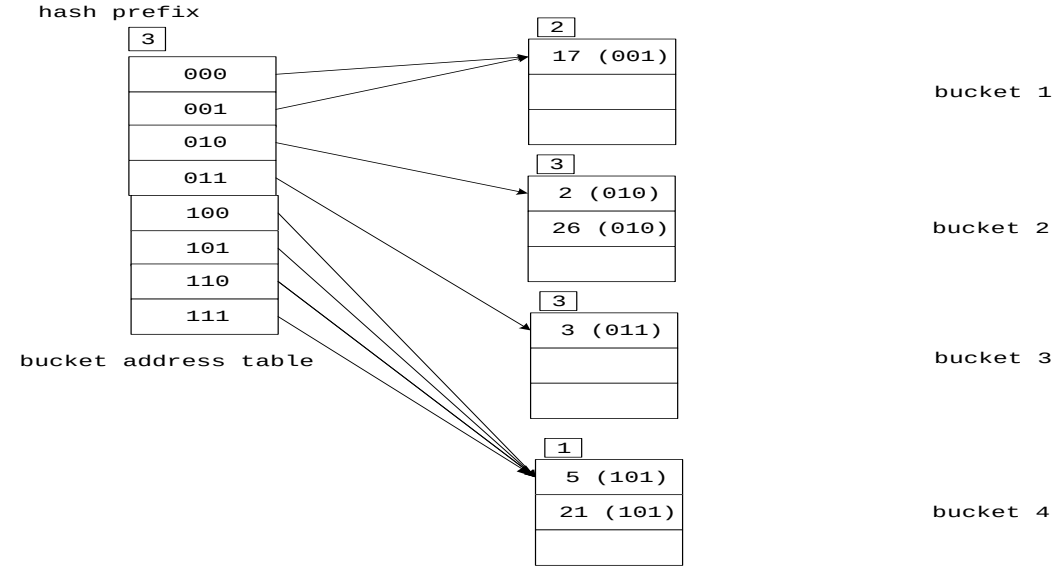
1. delete 19



2. delete 15



3. delete 55



Note: You can also merge the buckets in order to reduce the directory size. In practice, people set an underload threshold which specifies when the buckets will be merged, for merging is very costly.