CSCE 500 Midterm Exam #1

10/10/2018

1. Solve the two recurrences below:

$T(n) = 2 \cdot T(n/4) + n^{1/2}$ (6%)

$T(n) = 2 \cdot T(n^{1/2}) + \lg_2 n$ (10%)

2. The algorithm for finding the maximum subarray that crosses the midpoint of Array $A[1 \ldots n]$ includes the main routine of FIND-MAXIMUM-SUBARRAY(A, *low*, *high*), which calls FIND-MAX-CROSSING-SUBARRAY(A, *low*, *mid*, *high*), as follows. Complete the six (6) missing statements in FIND-MAX-CROSSING-SUBARRAY below. (12%)

```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
// Find a maximum subarray of the form A[i .. mid].
left-sum = −∞
sum = 0
for i = mid downto low
    sum = sum + A[i]
    [                    ]
    [                ]
    [                ]
// Find a maximum subarray of the form A[mid + 1 .. j].
right-sum = −∞
sum = 0
for j = mid + 1 to high
    sum = sum + A[j]
    [                ]
    [                    ]
    [                    ]
// Return the indices and the sum of the two subarrays.
return (max-left, max-right, left-sum + right-sum)
```

3. The hash table is a widely adopted data structure. Explain briefly

(1) how perfect hashing works. (8 %)

(2) how Cuckoo hashing works under two hash functions of h_1 and h_2. (10%)

What is the situation when a new key cannot be inserted in a Cuckoo hash table successfully? (2%)

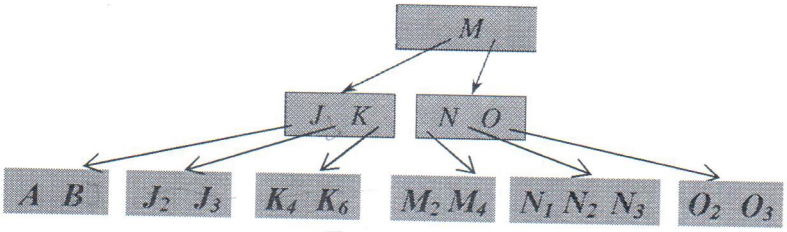
4. A binary search tree (T) is to be maintained following the in-order tree traversal order. Consider a sequence of arrival keys, {25, 23, 14, 7, 9, 21, 31, 34, 28, 24}, to T which has just the root node with its key = 20 initially.

(1) Show the resulting T after inserting all arrival keys. (5%)

(2) Show the resulting T after its root node is then deleted. (5%)

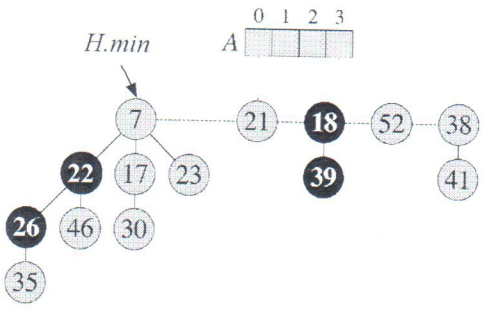
5. For any n-key B-tree of height h and with the minimum node degree of $t \geq 2$, prove that h is no larger than $\log_t \frac{n+1}{2}$. (Hint: consider the number of keys stored in each tree level.) (12%)

6. Given the initial B-tree with the minimum node degree of $t = 3$ below, show the results (a) after deleting the key of M_2 , (b) followed by inserting the key of L, (c) then by deleting the key of J_2, (d) then by inserting the key of O_1, with $O < O_1 < O_2$, and (e) then by deleting K. (Show the result after each deletion and after each insertion; 15%)



7. A Fibonacci min-heap relies on the procedure of CONSOLIDATE to merge trees in the root list upon the operation of extracting the minimum node. Given the following partially consolidated diagram, show every subsequent consolidation step till its completion. (10%)

After consolidation is completed, show the resulting Fibonacci min-heap with key '35' decreased to 4. (5%)



Good Luck!

88

C00227093
Vijay Srinivas Tide

① 16
② 12
③ 19
④ 10
⑤ 12
⑥ 5+1
⑦ 13

① $T(n) = 2T(n/4) + n^{1/2}$

i) $n^{\log_b a} = n^{\log_4 2} = n^{0.5}$

$f(n) = n^{0.5}$

According to master's theorem second rule

The Time complexity of $T(n) = \Theta(f(n) \cdot \lg n)$

$= \Theta(n^{1/2} \lg n)$

ii) $T(n) = 2T(n^{1/2}) + \lg_2 n$

we can't apply master's theorem directly.

let $n = 2^m \Rightarrow \lg_2 n = m$

$T(2^m) = 2T(2^{m/2}) + m$

$S(m) = 2S(m/2) + m$

Now apply master's theorem we will get

$m^{\log_b a} = m^{2\log_2 2} = m$

$f(m) = m$

According to second rule of master's theorem

$\Theta(m) = (f(m) \cdot \lg m) = m \lg m$

substitute $m = \lg_2 n \Rightarrow \Theta(\lg n \cdot \lg\lg n)$

$\therefore \Theta(m) = \lg n \lg\lg n$ OK

② if sum > left_sum

left_sum = sum

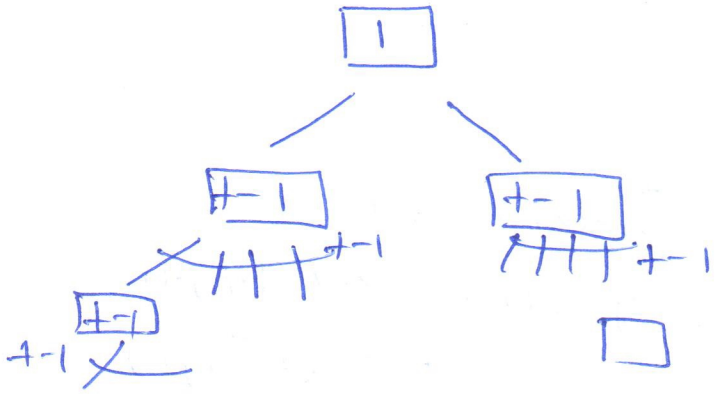
max_left = i OK

if sum > right_sum

right_sum = sum

max_right = j ok

(5)



	depth	No. of nodes
→	0	1
→	1	2
→	2	2t
→	3	$2t^2$

number of keys stored $\geq 1 + \left[2 + 2t + 2t^2 - - + 2t^{h-1}\right]$ where h is depth

$$= 1 + (t-1)\sum_{i=0}^{h-1} 2t^{i} = 1 + (t-1)\sum_{i=0}^{h-1} 2t^{h}.$$

$$\geq 1 + (t-1)\,2\left(\frac{t^h - 1}{t-1}\right). \qquad \frac{a(r^n-1)}{r-1}$$

$$\geq 1 + 2t^h - 2$$

$$\geq 1 + 2\,\frac{t^h - 1}{2(t-1)}\,(t-1)$$

$n \geq 1$

$$n \geq 2t^h - 1$$

$$t^h \leq \frac{n+1}{2}$$ OK

Apply $\log_t$ on both sides.

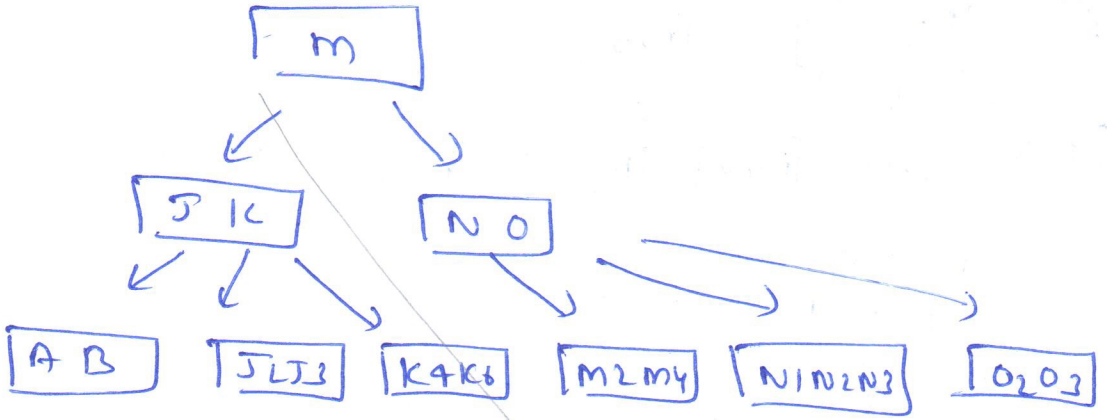
$$h \log_t t \leq \log_t \frac{n+1}{2}$$

$$h \leq \log_t \left(\frac{n+1}{2}\right)$$

$\therefore$ height of tree $\leq \log_t \left(\frac{n+1}{2}\right)$ ✓ ok

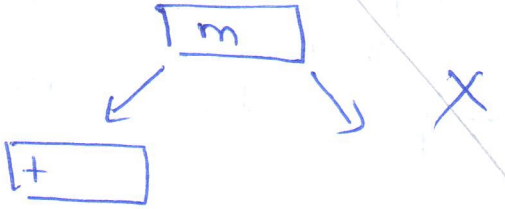
(6)

$t = 3$



(7)

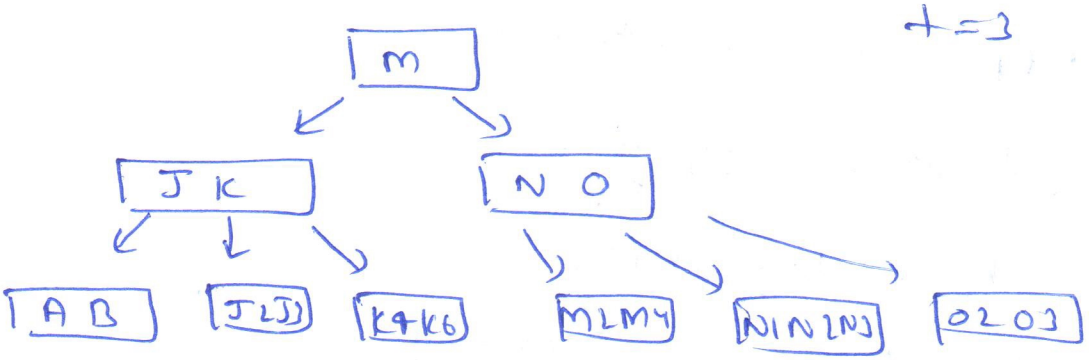
delete M2



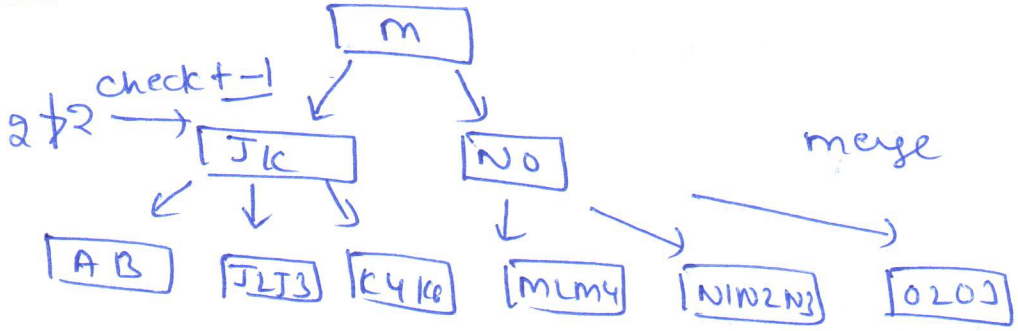
Rough

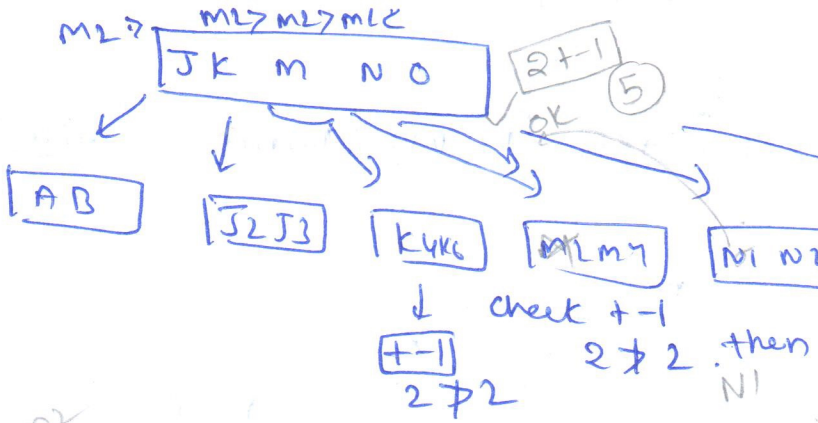
t=3

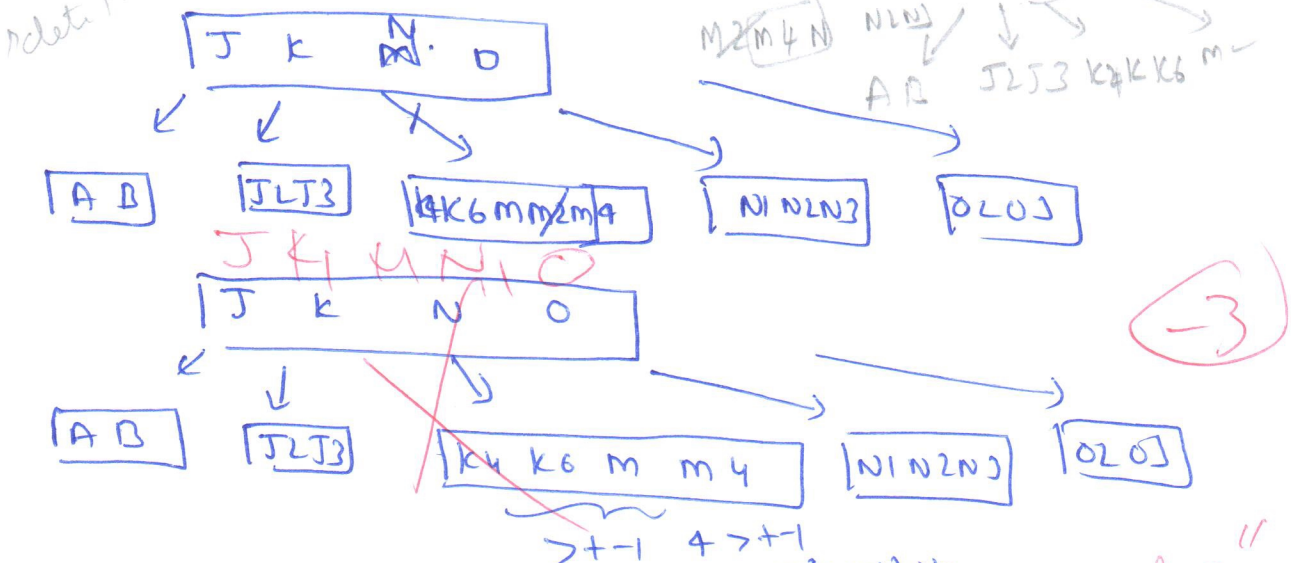
(6)



a) delete M2

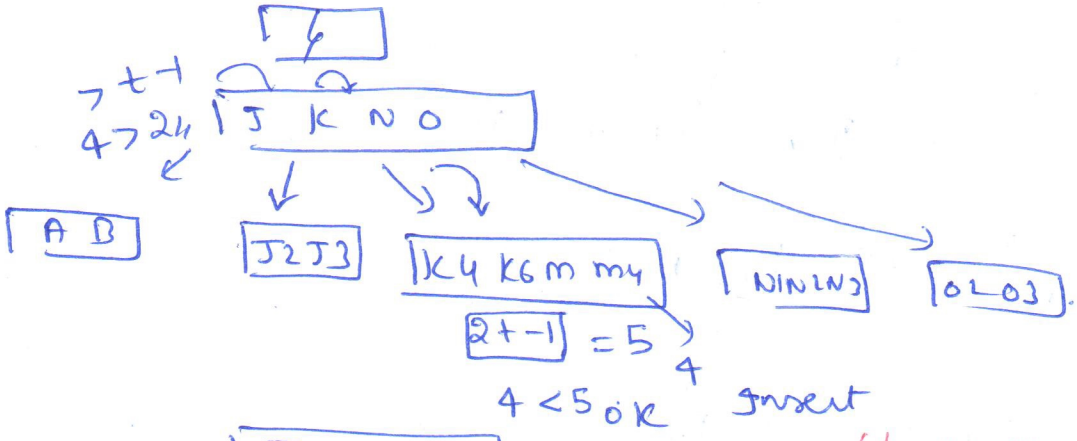


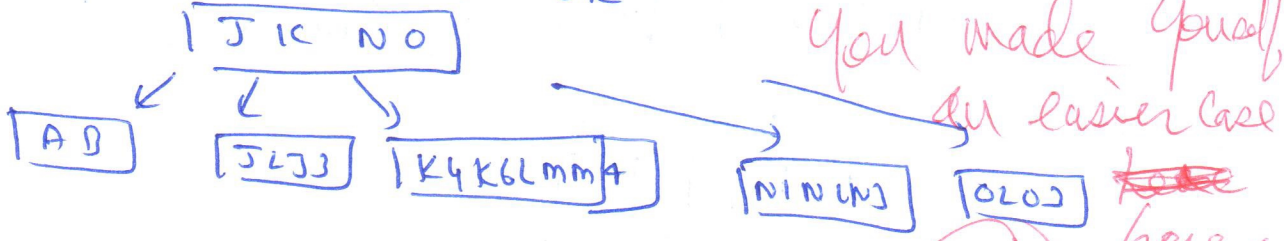




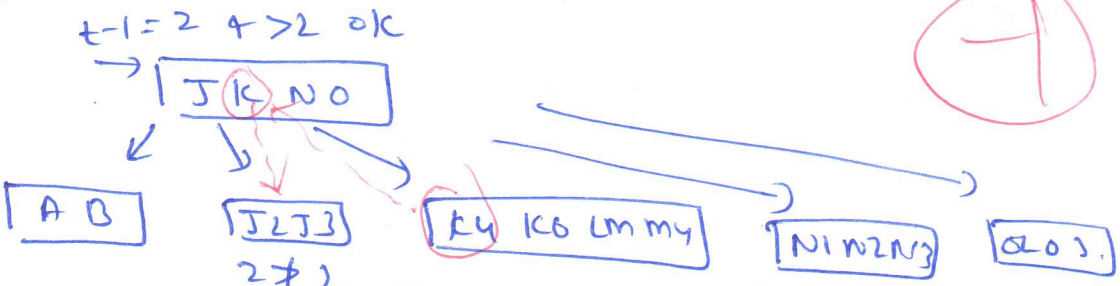
Cannot do "merging nodes" at random!!

b) Insert key (

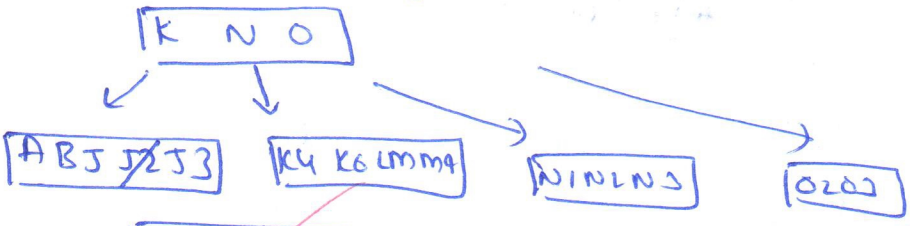


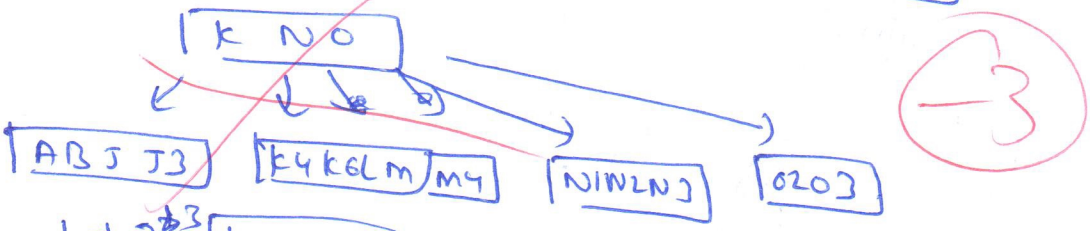


c) Delete J2

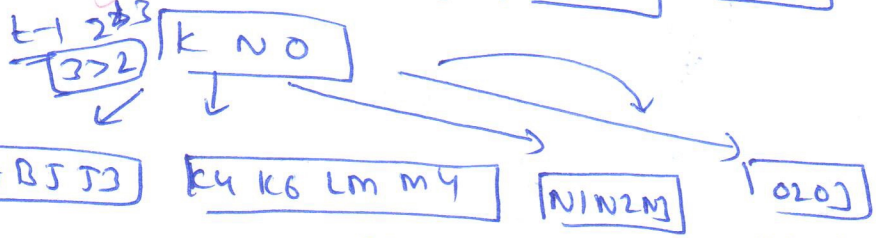


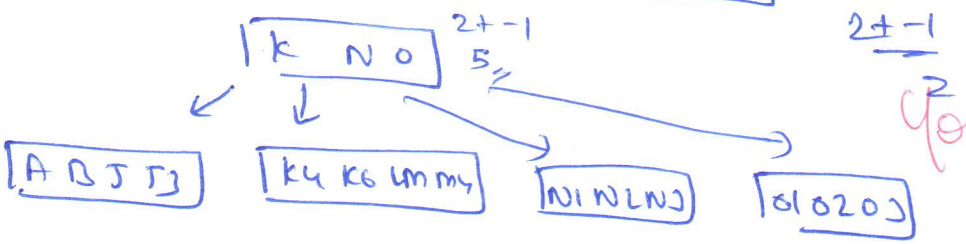
Merge



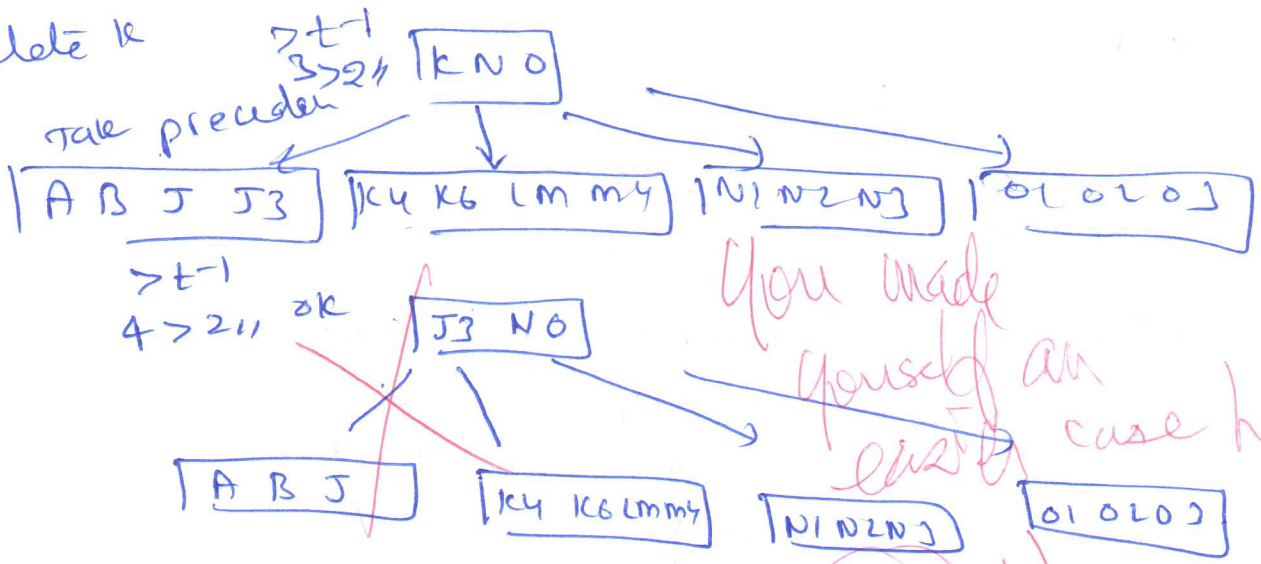


d) Insert 01



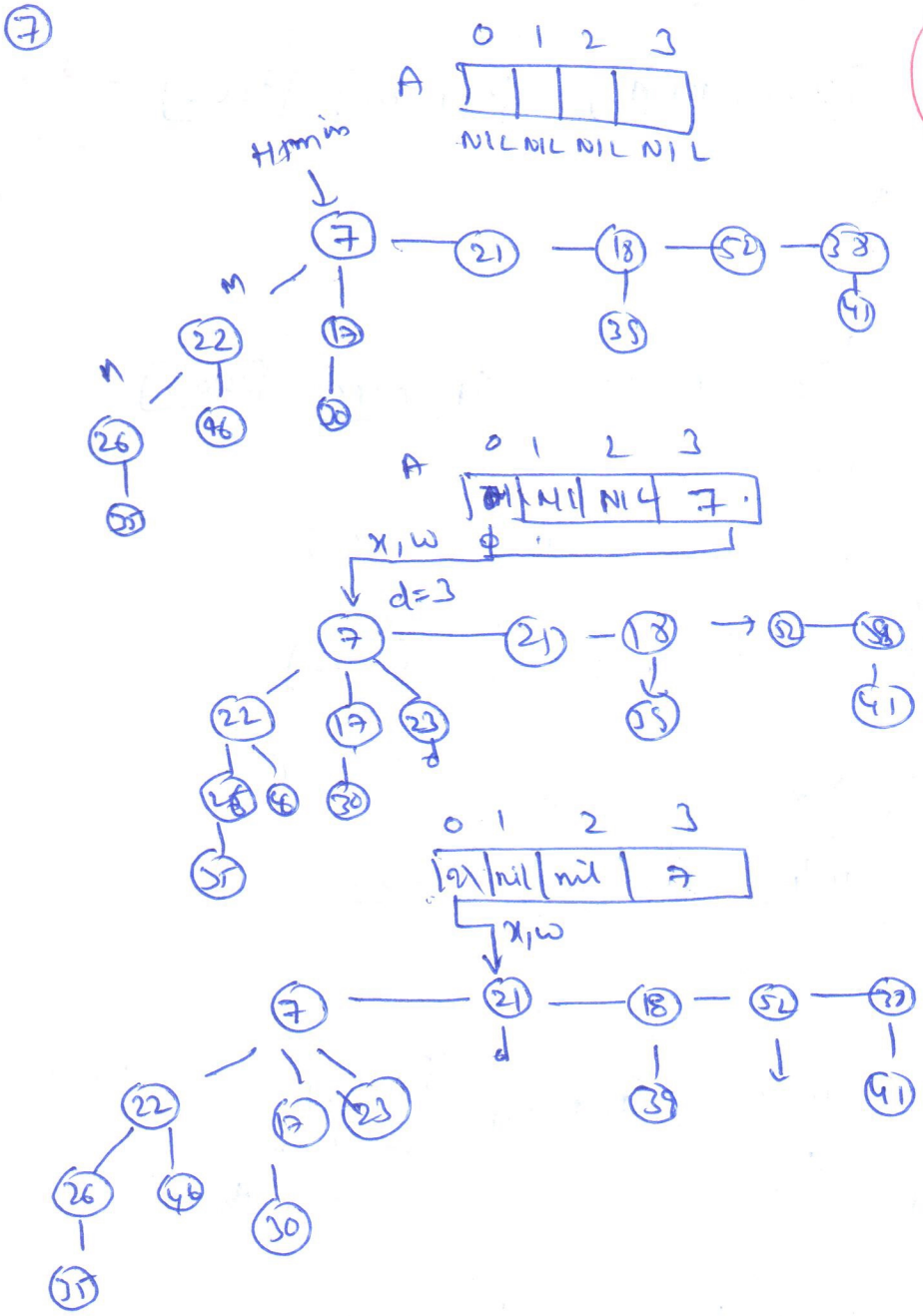


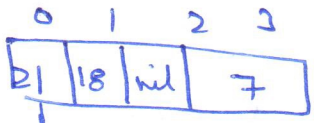
(e) delete k

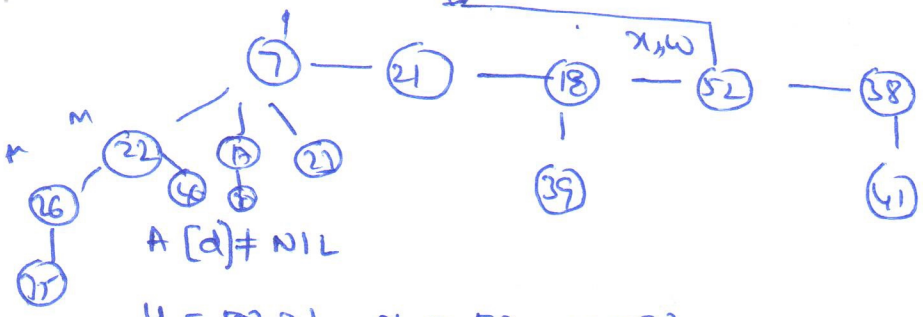


You made yourself an extra case here !

(7)







$y = 21$ $x = 52, w = 52$

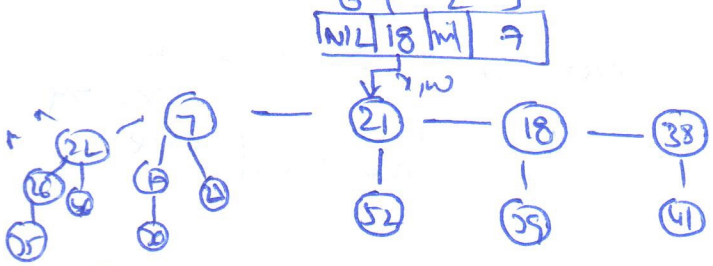
if $x.key > y.key$.

Swap (x, y)

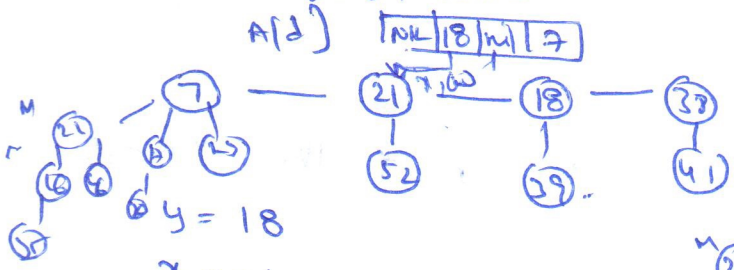
$x = 21$ $y = 52$ $d = 0$

FibHeap (H, x, y) $d = d + 1$

$= 1$

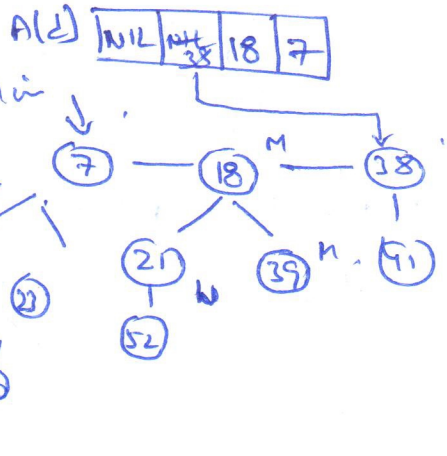


Again $A[d] \neq NIL$



$y = 18$

$x = 21$



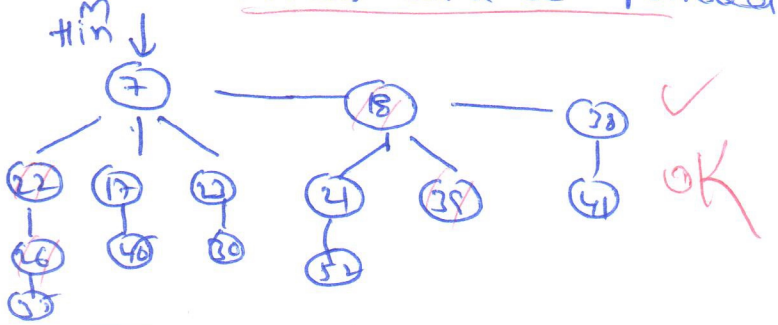
$x.key > y.key$.

Swap $(18, 21)$

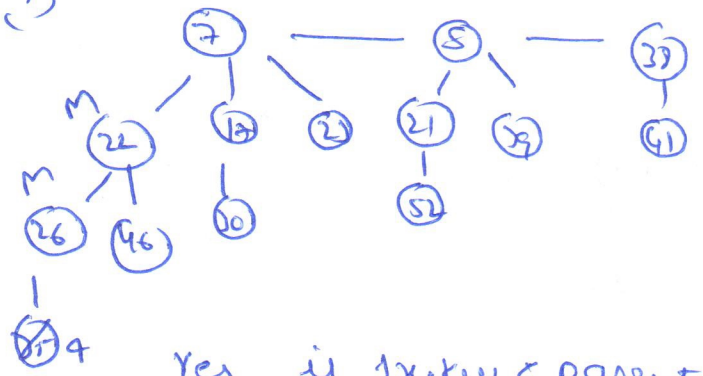
$x = 18$ $y = 21$

FibHeap $d = d + 1$

After Hmin will be pointed to min A[d] if A[d] has min than Hmin since both are same

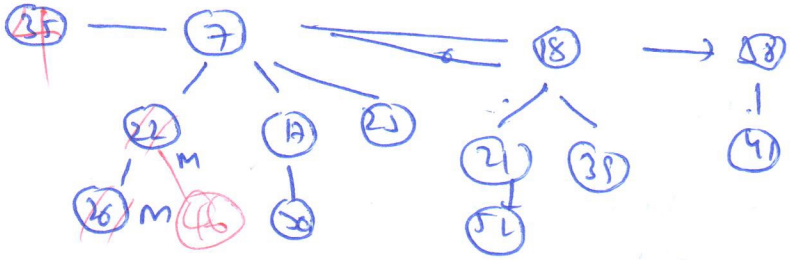


✓ OK

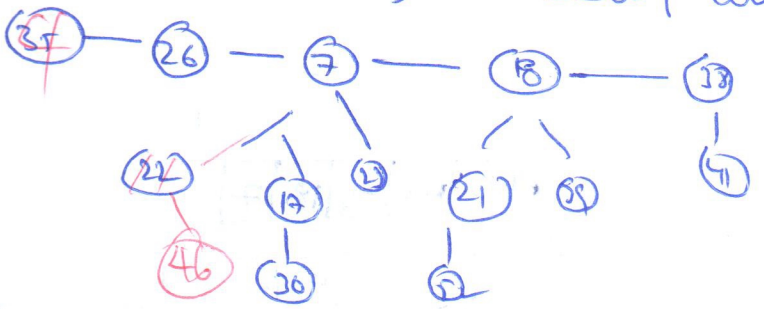


decreased 35 to 4

Yes if 1x.key < parent add to root list by means of cut.

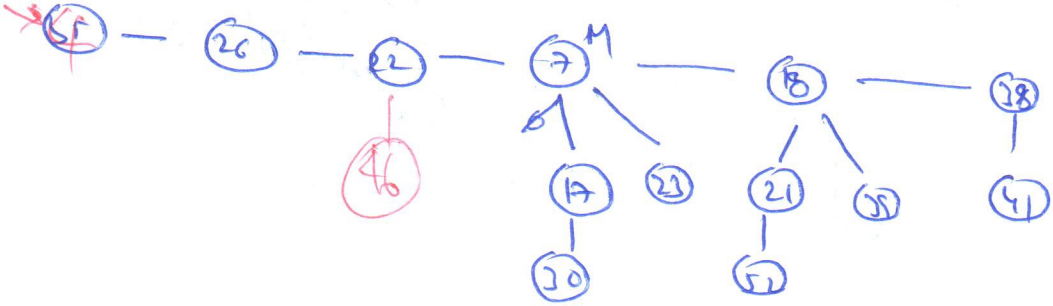


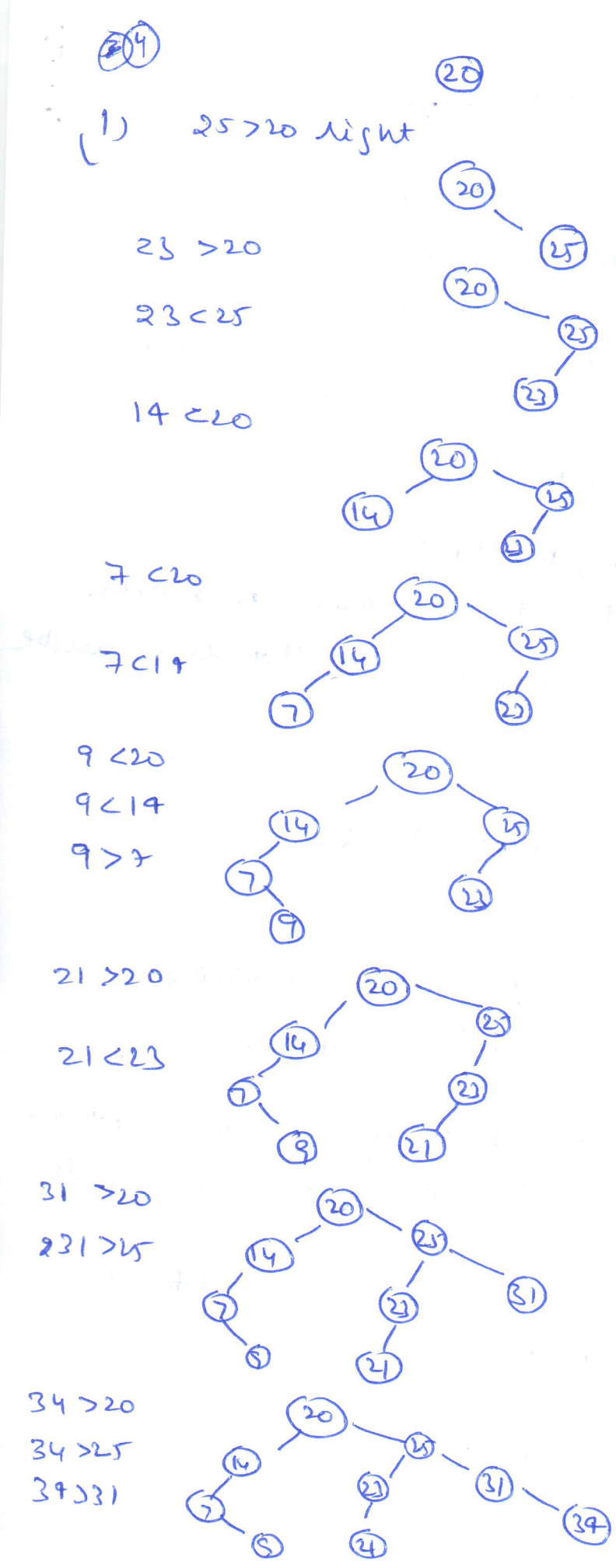
Next marked nodes are also added to root list by means of cascading cut if parent is marked one by one

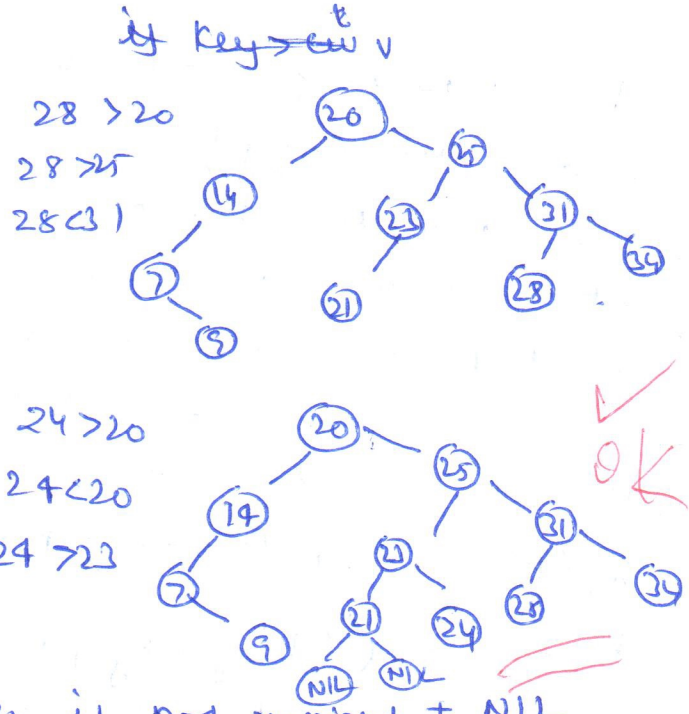


-2

If, All marked parent nodes are added no further change



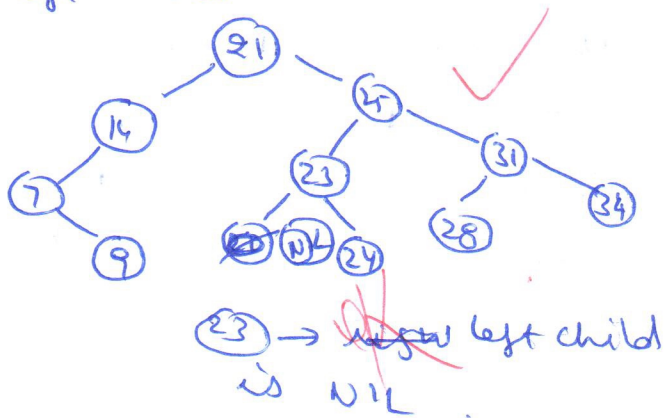




2) if x.right ≠ NIL
Then find Min of right subtree

Min = 21

* Min of right subtree is not next to 20 then we have to make the children of 21 if it has to its parent as it doesn't contains left child since its not there then NIL will be updated with left child of 23 and then 21 will be placed such that right child of 20 is made as right child of 21 and parent of 25 will be 21. Similarly for left child



(3)

1) 1. Perfect hashing works under static keys.

2. Its two level hash

2. It has two functions one for inserting the keys into the m slot and another each keys will be inserted according to the next n_j slots.

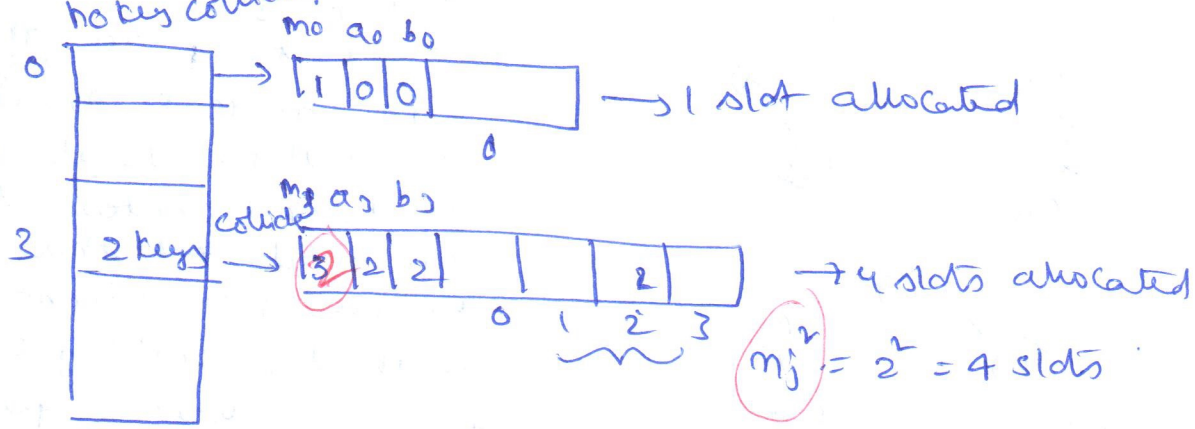
3. If two keys are colliding then 4 slots under each m slot will be created ← Cannot read your writing here!

$$h_1(k) = ((ka + b) \bmod p) \bmod m$$

$$h_2(k) = ((ka_j + b_j) \bmod p) \bmod m_j$$

where j is the slot related to the main hash table.

if j keys are collidings then for each slot there will be n_j^2 slots will be allocated



Each slot in sub hash table will have three values at first m_0 a_0 b_0 to specify/define the hash function for searching for a key which will require.

the value of $P > m$ and $P > m_j$ in case of second function

$a \in \{0, P-1\}$ $b \in \{0, P-1\}$.

in case of second hash function

$a \in \{0, \ldots P-1\}$ $b \in \{0, \ldots P-1\}$.

- The number of collisions in a perfect hashing will be $< 1/2$
- The total storage requirement for the perfect hashing technique would be $O(m)$ order of n. if table has n slots.
- perfect hashing uses two universal hashing functions.

Number of collisions $< 1/2$

if two keys are colliding to the same slot then

$$\frac{n_{C_2}}{m} = \frac{\frac{n(n-1)}{2}}{n_j^2} = \frac{1}{2} - \frac{1}{2n_j}$$

$$\leq 1/2.$$

Expected

- Maximum storage will be order of n is not n^2

linear

$$E[S] = E[m] + E\left[\sum_{j=0}^{m-1} n_j^2\right].$$

$$\sum_{j=0}^{m-1} E[n_j^2] = \sum_{j=0}^{m-1} E(n_j) + 2\sum_{j=0}^{m-1} E\left[\frac{n_j}{2}\right]$$

$$= \sum_{j=0}^{m-1} n_j + \frac{1}{m} \sum_{j=0}^{m-1} n_j C_2.$$

$$= n.$$

$$\frac{n_j C_2}{m} = \frac{n_j(n-1)}{2n} \quad \text{let } m = n$$

$$E\,E(n_j^2) = n + \frac{n-1}{2}$$

$$= 2n - 1$$

$$E(S) = O(m + 2n - 1)$$

$$= O(m) \quad \text{is linear}$$

(3)

C00 2270913
Vijay Srinivas Tida

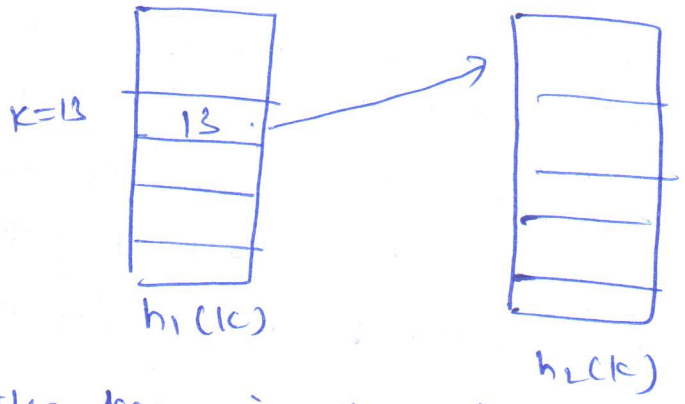
(2) Cuckoo hashing

General cuckoo hashing consists of two hash tables.

$h_1(k)$ & $h_2(k)$ are the functions allocated space for each key inserted.

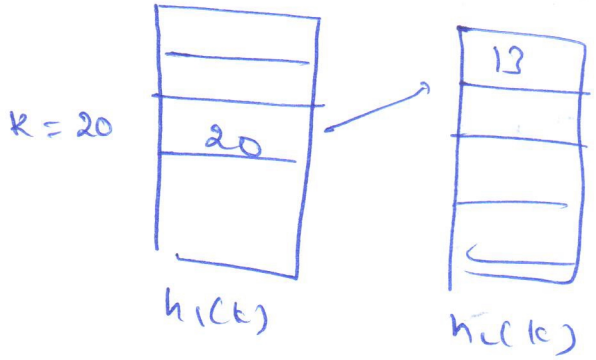
k will be the

if an element comes then $h_1(k)$ and $h_2(k)$ are calculated

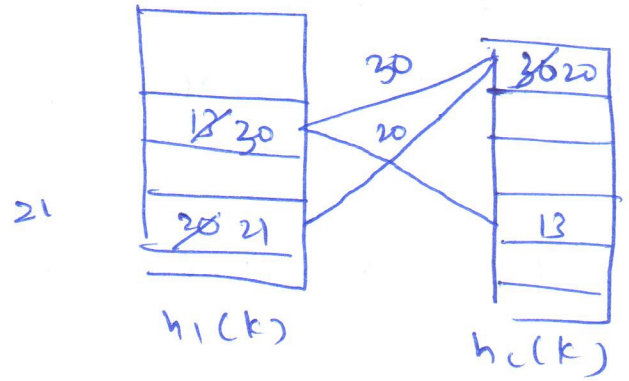


Then the key will be inserted to the corresponding slot allocated in table 1.

if ~~suppose~~ the new key ~~which~~ wants the same slot as $h_1(k)$ & $h_2(k)$ are ~~allocated~~ calculated and then the ~~new key~~ now if any key exists then the key in the table will be displaced to $h_2(k)$ and the key is has the space to enter



suppose if the key wants the other slot then the process of rearranging will be continued until the new key contains the slot.

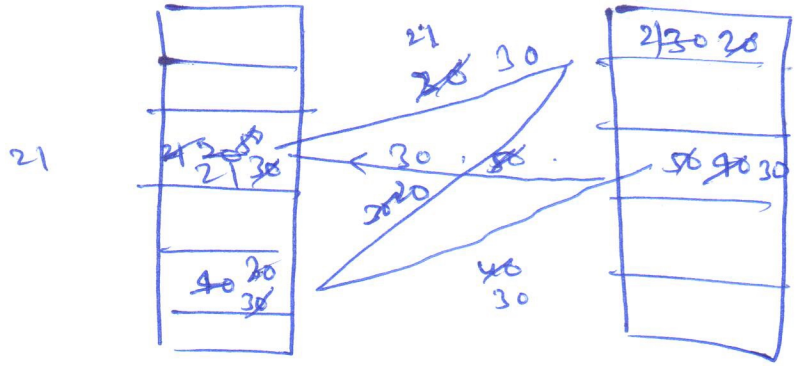


The maximum efficiency of cuckoo hashing will be 50%. when it has two hash tables. and >90%. if it has 3 hash tables.. Insertion will be successful atmost one cycle

2) The key won't be inserted when the corresponding displacements in between the hash key tables formed into cycles. The cuckoo hashing will check for the number of loops the new upcoming key made after certain amount the key will be kicked out of the table and rehashing of functions would be required.

one cycle can be handled

* Also the case would be when we have ideal hash function if all keys are stored in two tables.



C00227093

we can say that insertion fails when the key visited the same slot again at (same key with same slot).
Insertion takes $\Theta(1)$ time if inserted successfully
Searching takes $\Theta(1)$ time if =