M T W T F S S

Assignment : 2
Course Code : DAA
SAP ID : 70142254
Name : Attiya Sajid

## Q:1 Master Method

(a) $T(n) = 4T(n/3) + n$

$a = 4, \quad b = 3 \quad f(n) = n$

$n^{\log_b a} = n^{\log_3 4}$

$f(n) = n^{\log_b a}$

$T(n) = \theta\left(n^{\log_b a}\right)$

## Recursion Tree:

$T(n) = 4T(n/3) + n$

$T(n) = 4\left(4T\left(\frac{n}{9}\right) + \frac{n}{3}\right) + n$

$= 16T\left(\frac{n}{24}\right) + 4\left(\frac{n}{3}\right) + n$

$= 64T\left(\frac{n}{81}\right) + 16\left(\frac{n}{9}\right) + \left(\frac{4}{3}\right) + n$

M T W T F S S

**Generalize:**

$$T(n) = 4^K T\left(\frac{n}{3^K}\right) + n \sum_{i=0}^{K-1} \left(\frac{4^i}{3^i}\right)$$

Recursion stops at $\frac{n}{3^K} = 1$

when $K = \log_3 n$

$$\sum_{i=0}^{K-1} \left(\frac{4}{3}\right)^i = \Theta\left(n^{\log_3 4}\right)$$

$$T(n) = \Theta\left(n^{1.2619}\right)$$

b) $T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + cn$

$T(n) = [T(n/9) + T(2n/3) + c(n/3)] +$

$[T(2n/9) + T(4n/9) + c(2n/3)] +$

$cn$

M T W T F S S

$= T(n/27) + T(2n/27) + T(2n/27) +$
$T(4n/27) + T(2n/9) + T(4n/9).$
$+ cn(1 + 1/3 + 2/3)$

$cn\left(1 + \frac{1}{3} + \frac{2}{3} + \frac{1}{9} + \frac{2}{9} + \dots\right)$

Recursion stops at $\frac{n}{3^k} = 1$

$k = \log_3 n$

- $O(n)$ at each level.
- $T(n) = \Theta(n \log n)$

For Master Method:

$a_1 = 1$, $a_2 = 1$

$b_1 = 3$, $b_2 = 3$

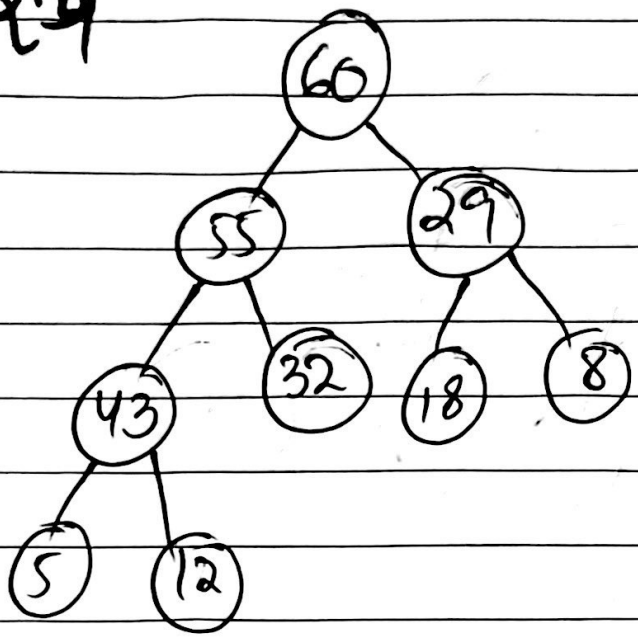
$f(n) = O(n)$

$n^{\log_b^1} = n \log_3 1$

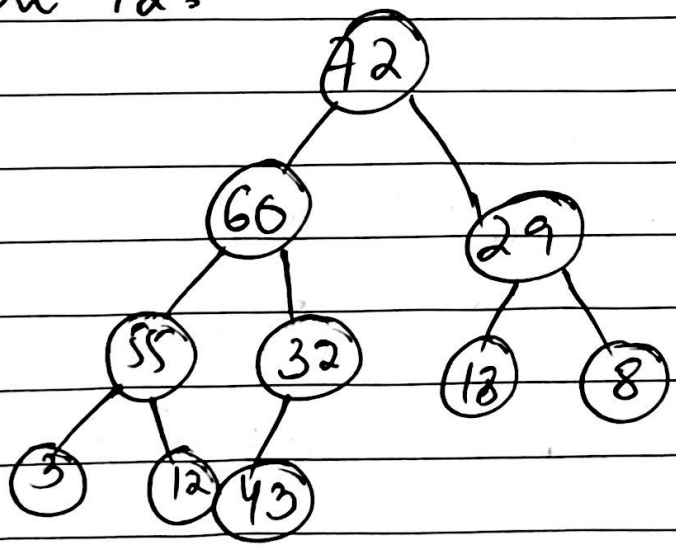
- 2 recursive with d/f sizes, so, Master Method not apply.

M T W T F S S

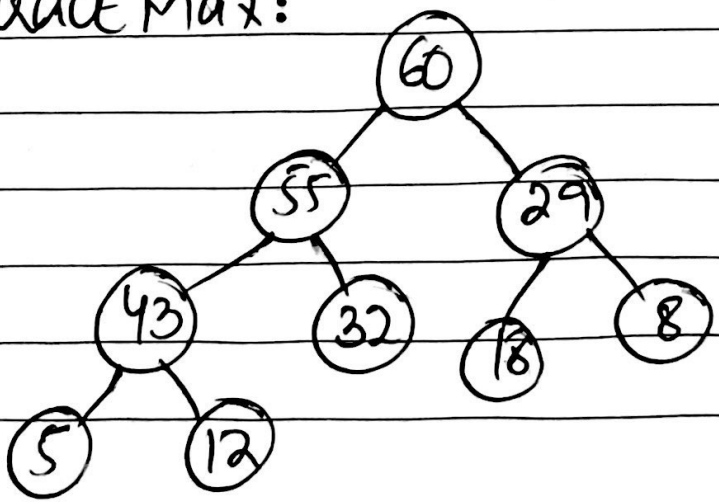
Q:4




Insert 72:



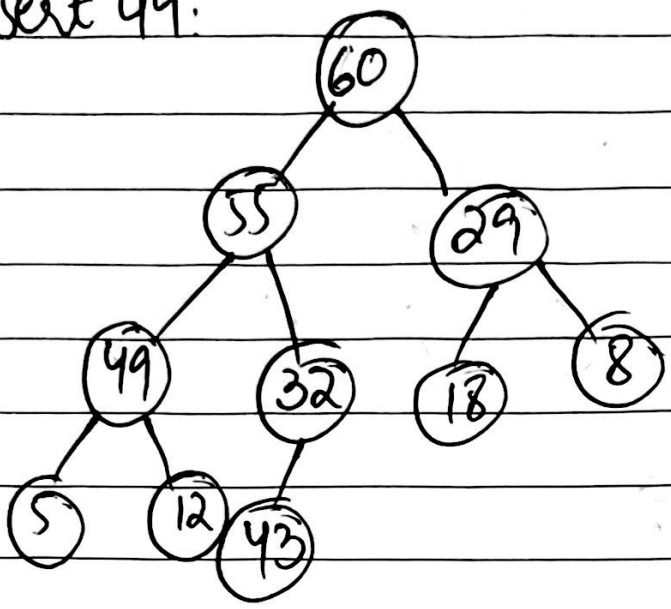

Extract max:




Lucrum erp

M T W T F S S

Insert 49:




```
void minHeapify(int arr[], int n, int i)
{
    int small = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

if( left < n && arr[left] < arr[small])
{
    small = left;

if( right < n && arr[right] < arr[small])
{
    small = right;
```

M T W T F S S

```
if (small != i) {
swap (arr[i], arr[small]);
minHeapify(arr, n, small);}
void buildMinHeap(int arr[], int n)
for(int i = n/2 - 1; i >= 0; i--) {
    minHeapify(arr, n);}
```

- Heapify takes $O(\log n)$ per node.

Total nodes to Heapify = $\frac{n}{2}$

$T(n) = O(n)$

M T W T F S S

Q:3

```
Node* insert (Node* root, int Key){
    if (root == NULL)
    return new Node(Key);

if (Key < root->key){
root->left = insert(root->left, Key);
}
  else{
root->right = insert (root->right, Key);
}
  return root;
}
void InOrder(Node* root){

if (root){
      InOrder (left->key);
      cout << root->key;
  InOrder(right->key);
    }
}
```

Insertion takes:

M T W T F S S

$$T(n) = T\left(\frac{n}{2}\right) + O(1)$$

Inorder takes:

$$T(n) = 2T\left(\frac{n}{2}\right) + O(1)$$

Master Method.

$$a = 2, \quad b = 2, \quad f(n) = O(1)$$

$$n^{\log_2 2} = n$$

$$T(n) = \Theta\left(n^{\log_b a}\right)$$

$$= \Theta(n)$$

Best case $= O(\log n)$

Worst case $= O(n)$

InOrder $= O(n)$

$$O(n \log n) + O(n) = O(n \log n).$$

M T W T F S S

Question: 2

height = 2.

$2^{3+1} - 1 = 7$

10 nodes. So height 2 not possible.

height = 3

$2^4 - 1 = 15$

15 nodes. So height 3 is possible.

height = 4, 5, 6, 7 ⇒ possible