

Section - A

(1) Determine  $\Theta$  bounds of the following relation

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

Masters' Theorem

$$f(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$a > 1$  = No. of subproblems

$b > 1$ ,  $\frac{n}{b}$  = size of the subproblem.

Consider  $f(n) = \Theta(n^k \log^p n)$ .

We will find 2 values

$$(1) \log_b a$$

$$(2) k$$

case-1 If  $\log_b a > k$   $\Rightarrow T.C = \Theta(n^{\log_b a})$ .

case-2 If  $\log_b a = k$  then

(a) If  $p > -1 \Rightarrow \Theta(n^k \log^{p+1} n)$

(b) If  $p = -1 \Rightarrow \Theta(n^k \log \log n)$

(c) If  $p < -1 \Rightarrow \Theta(n^k)$

case-3 If  $\log_b a < k$  then

(a) If  $p \geq 0$  then  $\Theta(n^k \log^p n)$

(b) If  $p < 0$  then  $\Theta(n^k)$ .

---


$$\therefore T(n) = 2T\left(\frac{n}{2}\right) + n \log n \quad \leftarrow \text{Question}$$

$$a = 2 \quad b = 2$$

$$(1) \log_b a = \log_2 2 = 1$$

(2)

$$n^k \log^p n = n \log n$$

$$\Rightarrow k=1 \quad p=1$$

$$\therefore \log_b a = \log_2 2 = 1 = k \Rightarrow \text{case 2}$$

And also  $p > -1$

$$T.C = \Theta(n^k \log^{p+1} n) = \Theta(n^1 \log^{1+1} n) \\ = \underline{\underline{\Theta(n \log^2 n)}}.$$

$$A_m = \cancel{\Theta(n \log^2 n)} \quad \underline{\Theta(n \log^2 n)}.$$

Sol A

(2) Evaluate the prefix expression

 $+ * 2 3 - 5 1$ 

- Scan from right to left

$\Rightarrow + * 2 3 [5-1]$

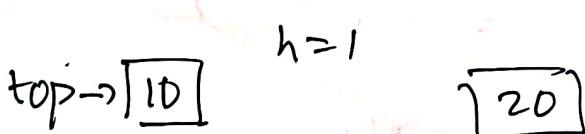
$\Rightarrow + * 2 3 [4]$

$\Rightarrow + [2 * 3] [4]$

$\Rightarrow + [6] [4] = 10.$

Sol A

(3)



def push(self, value):

    new-node = Node(value) ← Assume  
                  people may

    > new-node.next = self.top not write this

Pty assign  
    > self.top = new-node

    > self.height += 1

(3)

DSA - Ans  
Key

Sect. A  
Q. 4

Formula

Nodes Given

$$2^{nC_n}$$

$= \frac{2^{nC_n}}{(n+1)}$  = max Binary Tree

$T(n) = \text{catalan no for labelled} = n! \times \frac{2^{nC_n}}{(n+1)}$

$T(n) = \sum_{i=1}^n T(i-1) \times T(n-i)$   
recursive

$(h+1) \leq \text{Nodes} \leq 2^{h+1} - 1$   
if height is given

$\log_2(n+1) - 1 \leq \text{height} \leq (n-1)$   
If nodes are given

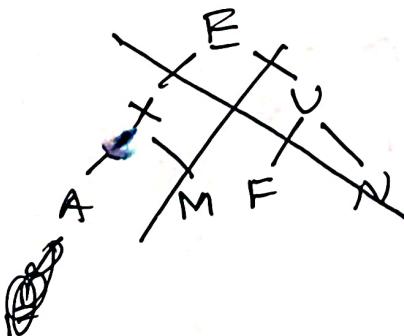
$$n = 15$$

$$h_{\min} = \log_2(15+1) - 1 = \log_2 16 - 1 = 4 - 1 = 3$$

$$h_{\max} = 15 - 1 = 14$$

$$3 \leq \text{height} \leq 14$$

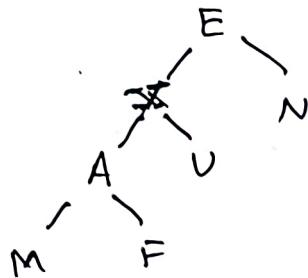
Sect A  
Q. 5



Preorder : EXAMFUV  
Inorder

Q.5)

(4)



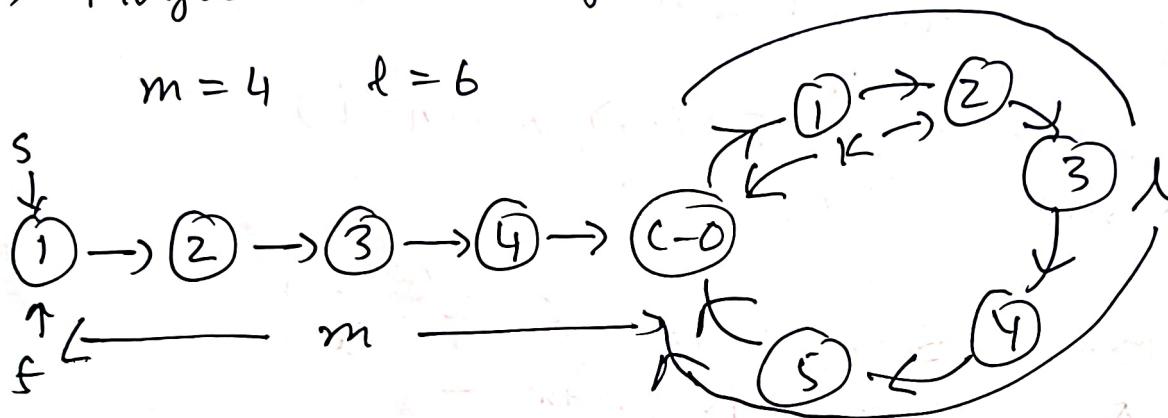
Preorder : EXAMFUN

Inorder: MAFXVEN

Section:- B

(1) Floyd detection cycle with 10 nodes

$$m = 4 \quad d = 6$$



$s$  = slow pointer  $\Rightarrow$  moves by 1

$f$  = fast pointer  $\Rightarrow$  moves by 2.

If slow ptr meets fast pointer, then the linked list has a cycle.  $(f-d - s-d)$

<u>s</u>	<u>f</u>	<u>s-d</u>	<u>f-d</u>	<u>s%d</u>	<u>f%d</u>	<u>abs-gap</u>
1	1	0	0	-	-	0
2	3	1	2	-	-	1
3	C-D	2	4	-	0	2
4	2	3	6	-	2	3
C-D	4	4	8	0	4	4
1	C-D	5	10	1	0	5
2	2	6	12	2	2	6

Presence of cycle - slow & fast pointer  
muts.

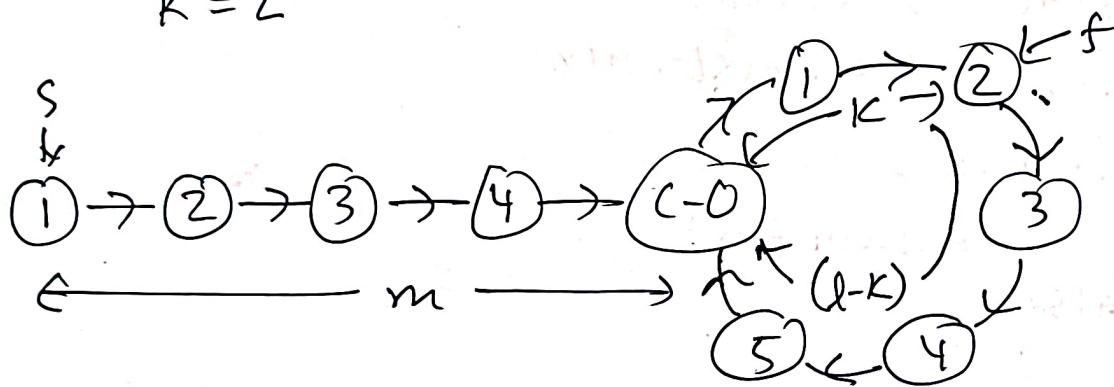
(5)

(i)  $\frac{\text{abs-gap}}{\downarrow} \% \ell = 0$   
 $\downarrow$   
 $6 \% 6 = 0$

(ii)  $s \% \ell = f \% \ell = 2$

The position where they meet =  $K$  from 1st node of the cycle

$K = 2$



Now move  $s = s \cdot \text{next}$   
 $f = f \cdot \text{next}$  ] $\rightarrow$  they should meet at  
C-O  $\rightarrow$  starting point of the cycle

slow

$$m + c_1 \times l + K$$

$$c_1 \in \mathbb{Z}$$

fast

$$m + c_2 \times l + K$$

$$(c_2 \in \mathbb{N})$$

$$\therefore m + c_2 \times l + K = 2(m + c_1 \times l + K)$$

$$\Rightarrow (c_2 - 2c_1) \times l - K = m$$

$$\boxed{(int) \times l - K = m}$$

$$\begin{aligned} m &\xrightarrow{\quad} l - K \\ &\xrightarrow{\quad} 2l - K \\ &\xrightarrow{\quad} 3l - K \\ &\vdots \end{aligned}$$

## codes

(6)

- (1) has-loop
- (2) detect-first-node-cycle

```
def has-loop (self):
```

```
    if self.head is None  
        return False
```

```
    slow = self.head
```

```
    fast = self.head
```

```
    while fast is not None and fast.next is not None:
```

```
        slow = slow.next
```

```
        fast = fast.next.next
```

```
        if (slow == fast)
```

```
            return fast
```

```
    return False
```

```
def detect-first-node-cycle (self, meeting-point):
```

```
    slow = self.head
```

```
    fast = meeting-point
```

```
    while slow != fast:
```

```
        slow = slow.next
```

```
        fast = fast.next
```

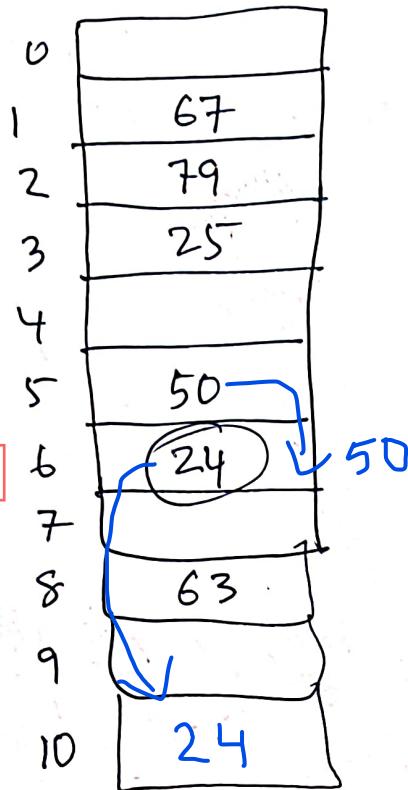
```
    return (slow)
```

→ This will be (-o) in our case

(7)

Section B

Q. 2. (a)



Keys = 63, 50, 25, 79,  
67, 24

$$h_1(K) = K \% 11$$

$$h_2(K) = 1 + K \% 7$$

$$h_3(K, i) = h_1(K) + i \cdot h_2(K)$$

$$h_1(24) = 2 \Rightarrow \text{Collision}$$

$$h_2(24) = 1 + 24 \% 7 = 1 + 3 = 4$$

$$\therefore h(24, 1) = 2 + 1 \cdot 4 = 6 \cdot X$$

$$h(24, 2) = 2 + 2 \cdot 4 = 10$$

(b) Primary Clustering  $\rightarrow$  Linear Probing

collision occurs  $\rightarrow$  algo looks for  $\rightarrow$  creates  
next slot long contiguous blocks of occupied slot

Search + Insertion  $\leftarrow$   
time

Ex:- Keys = 12, 22, 32, 42  
size (1f.T) = 10  $\uparrow$   $h(K) = K \% 10$   
Primary cluster.

Secondary Clustering  $\rightarrow$  Quadratic Probing.

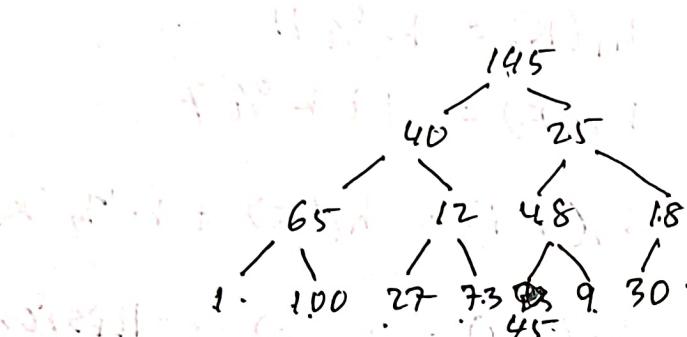
$$h(K, i) = \cancel{h(K)} + i^2$$

- while it pushes the keys distance apart, it still follows the same probe sequence

Ex:- Keys = 12, 22, 32, 42 can create secondary clustering as well.

3. (a) Construct a min heap by applying Heapsify procedure

145, 40, 25, 65, 12, 48, 18, 1, 100, 27, 73, 45, 9, 30  
using bottom-up Heapsify



$\frac{145}{1}, \frac{40}{2}, \frac{25}{3}, \frac{65}{4}, \frac{12}{5}, \frac{48}{6}, \frac{18}{7} \quad | \quad \frac{1}{8}, \frac{100}{9}, \frac{27}{10}, \frac{73}{11}, \frac{45}{12}, \frac{9}{13}, \frac{30}{14}$

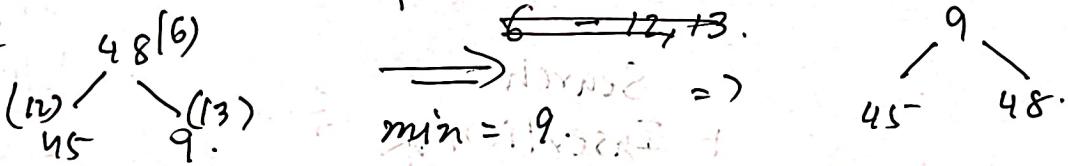
Heap  $\Rightarrow$  It is a complete B-T

(1)

$\frac{18}{18} \quad \text{what is the min?} \quad \frac{18}{30} \Rightarrow \frac{18}{30}$

<u>Element</u>	<u>Index</u>	<u>left child</u>
18	1	30

(2) min  $\frac{48}{48(6)} \rightarrow \min = 12 + 13$



swap (48, 12)

(3)

$\frac{12}{12(5)} \rightarrow \min = 12$

no chg.

(4)

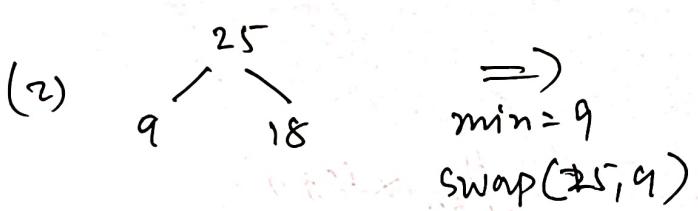
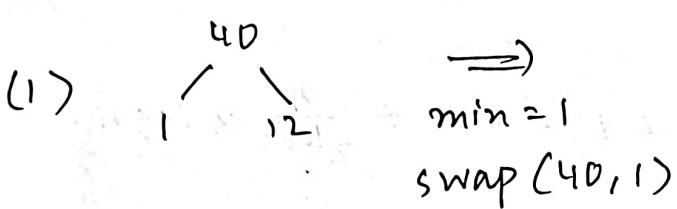
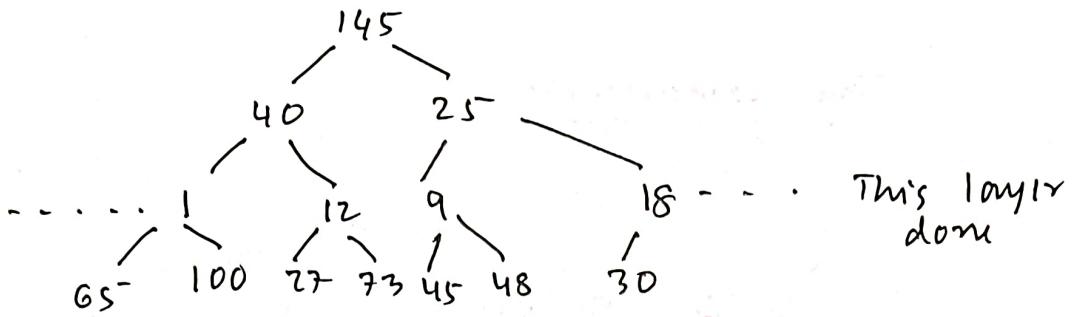
$\frac{65}{65(4)} \rightarrow \min = 1$

swap (65, 1)

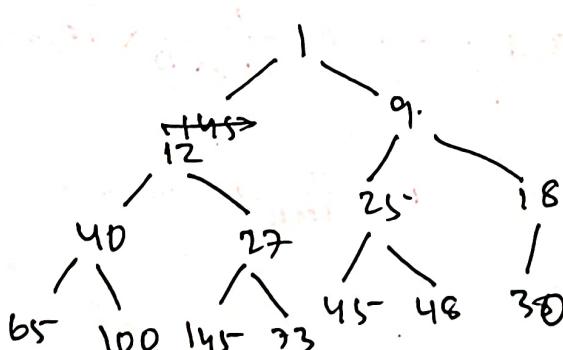
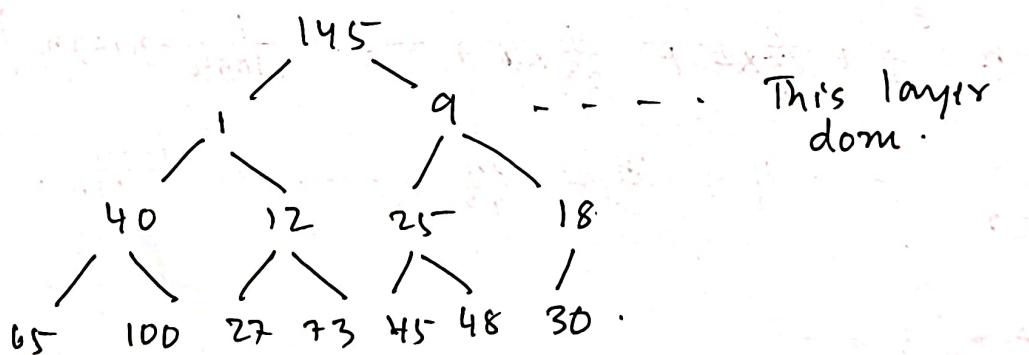


(9)

Intermediate



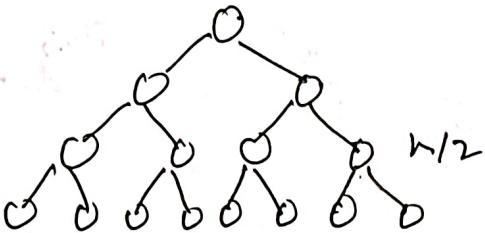
Intermediate



Final : 1, 12, 9, 40, 27, 25, 18, 65, 100, 145, 73, 45, 48, 30.  $\rightarrow (\text{P.T.O})$

Final:  $\frac{1}{1}, \frac{12}{2}, \frac{9}{3}, \frac{40}{4}, \frac{27}{5}, \frac{25}{6}, \frac{18}{7}, \frac{65}{8}, \frac{100}{9}, \frac{145}{10}, \frac{73}{11}, \frac{45}{12}, \frac{48}{13}, \frac{30}{14}$ .

### Time complexity



### Total swap

$$S = \frac{n}{2^0} \times 0 + \frac{n}{2^1} \times 1 + \frac{n}{2^2} \times 2 + \dots + \frac{n}{2^{\log n}} \times \log n.$$

no. of elements at that level      no. of swaps.

$$S = n \left[ \frac{1}{2} + \frac{1}{2^2} \times 2 + \dots + \frac{1}{2^{\log n}} \times \log n \right].$$

~~total no. of total~~

$$S = n \left[ \frac{1}{2} + \frac{1}{2^2} \times 2 + \frac{1}{2^3} \times 3 + \dots + \frac{1}{2^{\log n}} \times \log n \right]$$

~~18=0~~

$$\Rightarrow S = \frac{n}{2} + \frac{2n}{2^2} + \frac{3n}{2^3} + \dots + \frac{\log n}{2^{\log n}}$$

$$\Rightarrow \frac{1}{2} S =$$

$$S = \frac{n}{2^1} \times 1 + \frac{n}{2^2} \times 2 + \frac{n}{2^3} \times 3 + \dots + \frac{n}{2^{(\log n-1)}} \times (\log n - 1) + \frac{n}{2^{\log n}} \times \log n \quad (1)$$

$$\Rightarrow \frac{1}{2} S = \frac{n}{2^2} \times 1 + \frac{n}{2^3} \times 2 + \frac{n}{2^4} \times 3 + \dots + \frac{n}{2^{\log n}} \times (\log n - 1) + \frac{n \log n}{2 \cdot 2^{\log n}} \quad (2)$$

$$(1) - (2)$$

$$\Rightarrow \frac{1}{2} S = \left[ \frac{n}{2} + \frac{n}{2^2} + \frac{n}{2^3} + \dots + \frac{n}{2^{\log n}} \right] - \frac{n \log n}{2 \cdot 2^{\log n}} \times 2$$

$$\Rightarrow \frac{1}{2} S = \frac{n}{2} \left( 1 + \frac{1}{2} + \frac{1}{2^2} + \dots + \frac{1}{2^{\log n-1}} \right) - \frac{n \log n}{2 \cdot 2^{\log n}}.$$

(11)

$$\Rightarrow \frac{1}{2}s = \frac{n}{2} \underbrace{\left(1 + \frac{1}{2} + \frac{1}{2^2} + \dots + \frac{1}{2^{\lceil \log_2 n \rceil - 1}}\right)}_{x} - \frac{n \log n}{2 \cdot 2^{\lceil \log_2 n \rceil}}.$$

$$x \Rightarrow \frac{a(1-r^n)}{1-r} = \frac{1(1 - (\frac{1}{2})^{\log n})}{1 - \frac{1}{2}} = 2 \left[1 - \left(\frac{1}{2}\right)^{\log n}\right]$$

$$a=1 \quad r=\frac{1}{2}$$

$$n = \log n \quad = 2 \left(1 - \frac{1}{2^{\log n}}\right)$$

Note  $2^{\log n} = n$

$$= 2 \left(1 - \frac{1}{n}\right).$$

$$\Rightarrow \frac{1}{2}s = \frac{n}{2} \times 2 \left(1 - \frac{1}{n}\right) - \frac{n \log n}{2n}$$

$$\Rightarrow s = 2n \left(1 - \frac{1}{n}\right) - \log n = 2n - 2 - \log n.$$

$$\therefore s = O(n).$$

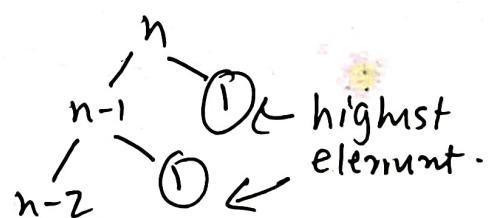
Total Swap = Time complexity =  $O(n)$ .

4.(a) Q.S algorithm applied on  $n$  distinct elements.

$$T(n) = \begin{cases} T(n-1) & \text{if } n > 1 \\ T(n-1) & \text{if } n = 1 \end{cases}$$

$$T(n) = T(n-1) + cn$$

work done  
to break  
the tree.



Master's theorem

$$T(n) = aT(n-b) + f(n).$$

$$a < 1$$

$$T(n) = O(f(n))$$

$$a = 1$$

$$T(n) = O(n \times f(n))$$

$$a > 1$$

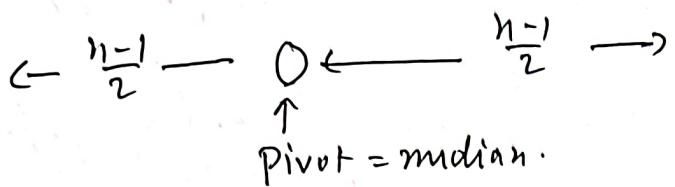
~~$$O(n^{a/b} \cdot f(n))$$~~

Here  $a = 1$

$$T(n) = \Theta(n \times \Theta(n)) = \Theta(c \cdot n^2) = \Theta(n^2).$$

(12)

4.(b) suppose the pivot is always chosen as the median.



R.R

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

$\underbrace{\quad\quad\quad}_{\text{scan the elements}}$

$$\Rightarrow T(n) = 2T\left(\frac{n}{2}\right) + cn.$$

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) \rightarrow \underbrace{\text{work}}_{a > 1} = \underbrace{\text{divide}}_{\text{the problem}} + \underbrace{\text{combine}}_{\text{the result}}$$

$\underbrace{\quad\quad\quad}_{f(n) = n \rightarrow \text{scan } n \text{ elements.}}$

$$\text{consider } f(n) = \Theta(n^k \log^p n).$$

we will find 2 values

- i)  $\log_b a$       case 1      if  $\log_b a > k$        $\Theta(n^{\log_b a})$   
ii)  $k$                   case 2      if  $\log_b a = k$        $\Theta(n^k \log^{p+1} n)$ .

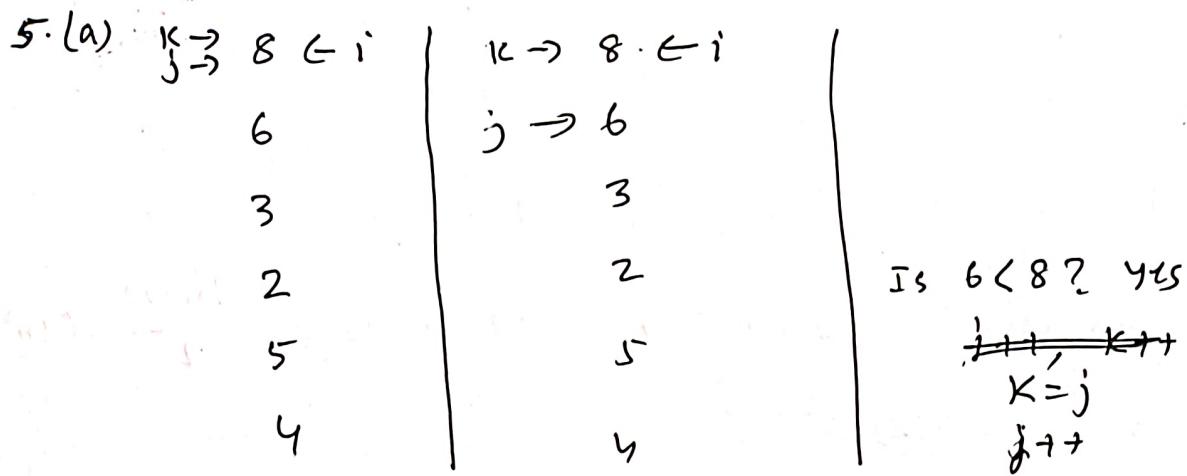
$$a = 2 \quad b = 2$$

$$n^k \log^p n = n$$

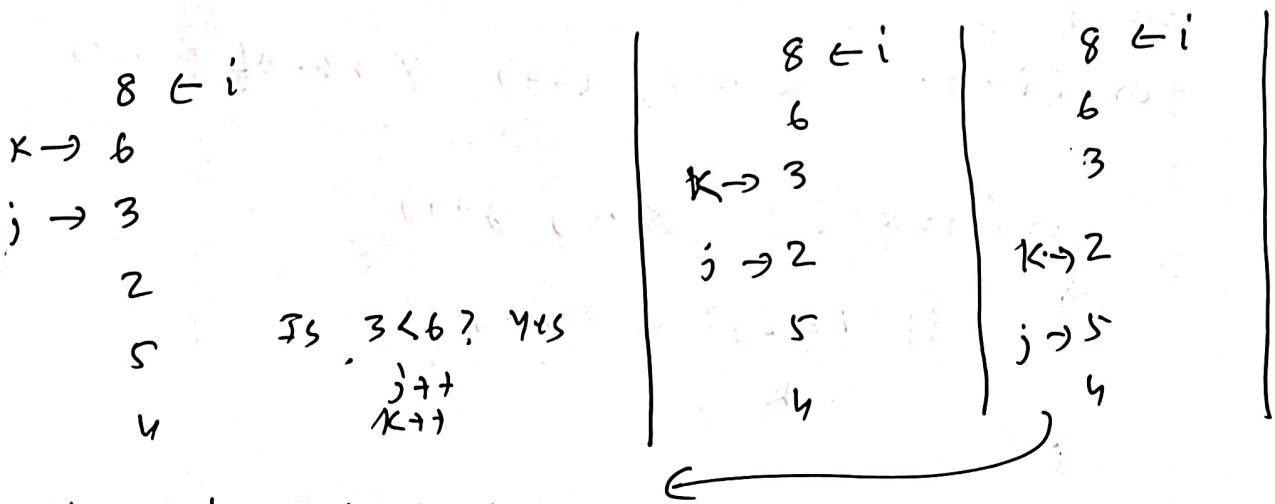
$$\therefore \log_2 2 = 1$$

$$k=1 \quad p=0$$

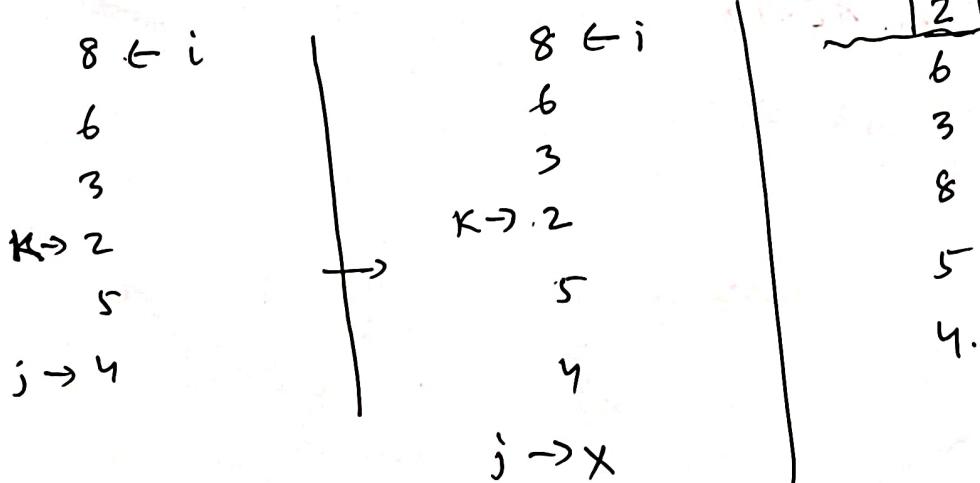
$$T(n) = \Theta(n^1 \log^0 n) = \Theta(n \log n).$$



- (1) keep  $i$  fixed
  - (2) use  $j$  &  $k$  to find the min.
  - (3) swap ( $i, k$ )
- If  $\cancel{A[j]} < A[k]$   
 ~~$j++$~~   $\cancel{k = j}$   
 ~~$i++$~~   $\cancel{j++}$   
else  $j++$  until array exists.



just  $j++$        $\leftarrow$  min found on top.



Pass ends here.

swap ( $A[i], A[k]$ )

Passes : 1 2 3 4 5

Comparisons : 5 4 3 2 1

swaps : 1 1 1 1

$$\text{Total comparison} = 1 + 2 + 3 + \dots + (n-1) = \frac{n(n-1)}{2} = O(n^2)$$

$$\text{swaps} = (n-1) = O(n)$$

$$T.C = O(n^2)$$

5-(b)

### Algo

void selection - sort ( int A[], int n )

{ int i;

for ( i=0; i < n-1; ++ ) // no. of passes

{

for ( j=k=i; j < n; j++ )

{

if ( A[j] < A[k] )

    k=j;

}

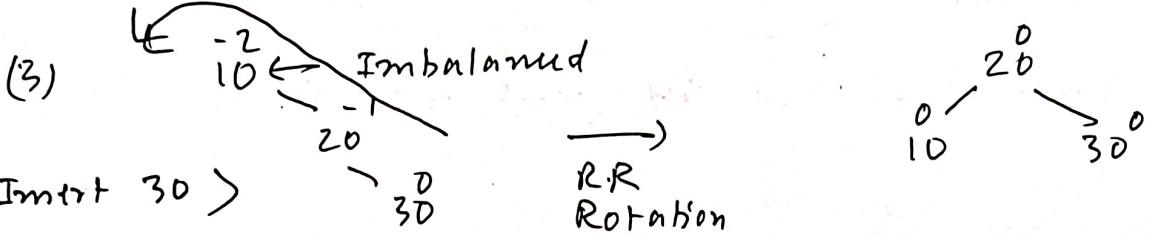
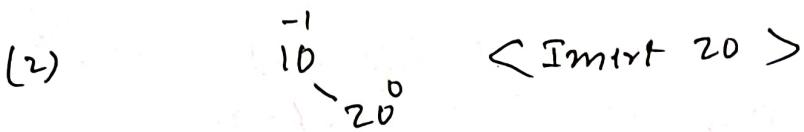
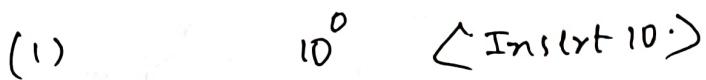
# now k will be on the min element

swap ( A[i], A[k] ).

}

}

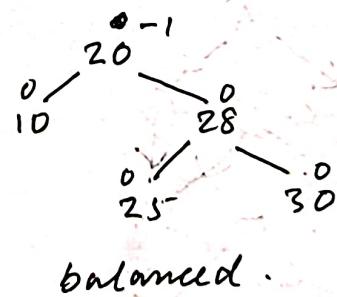
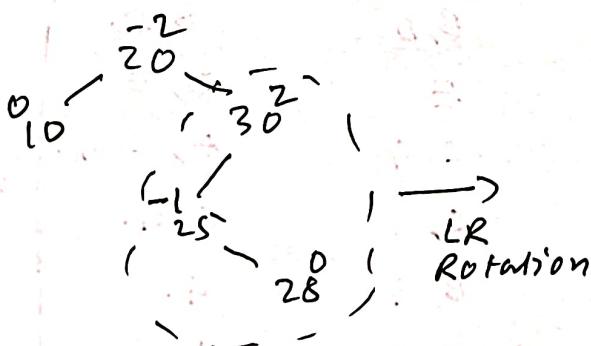
6. (a) AVL Tree  
with keys = 10, 20, 30, 25, 28, 27, 5



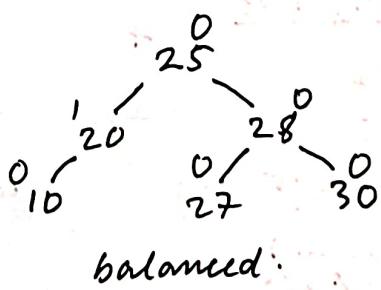
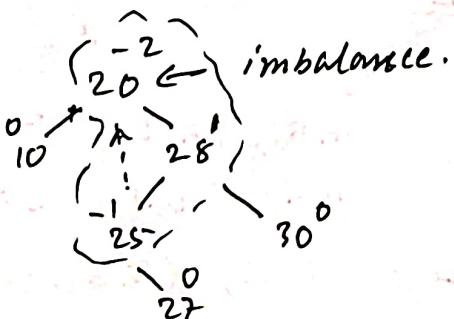
(4)  $\begin{array}{c} 20^{-1} \\ \swarrow \quad \searrow \\ 10^0 \quad 30^0 \\ \swarrow \quad \searrow \\ 25^0 \end{array}$  Insert 25

Balanced.

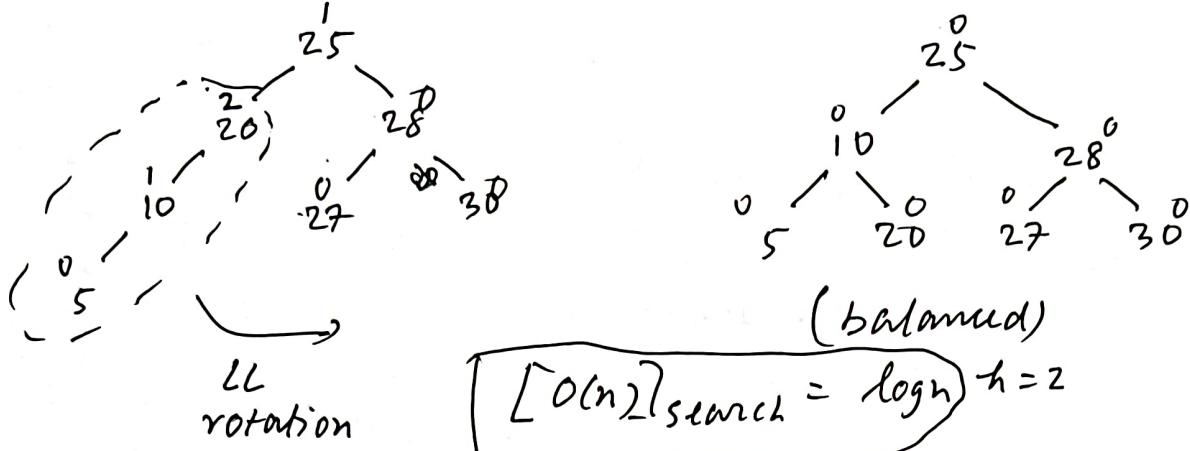
(5) Insert 28



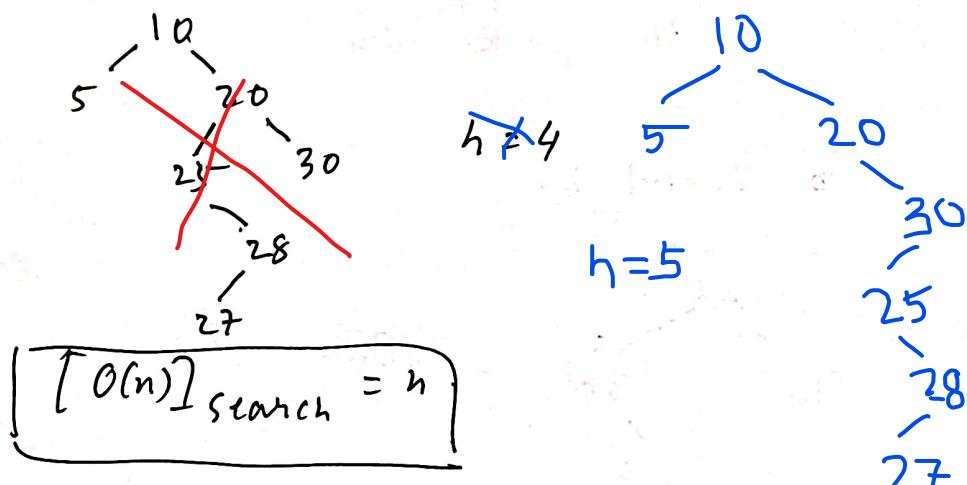
(6) Insert 27



(7) Insert 5

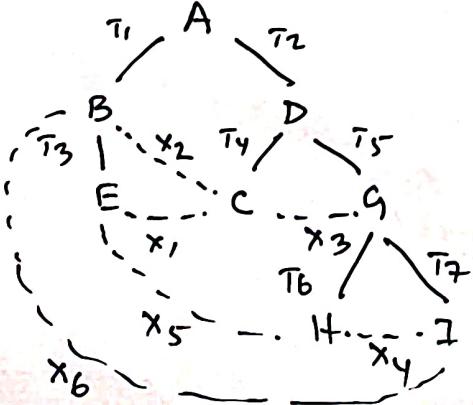


6-(b) Without AVL Tree the tree would be

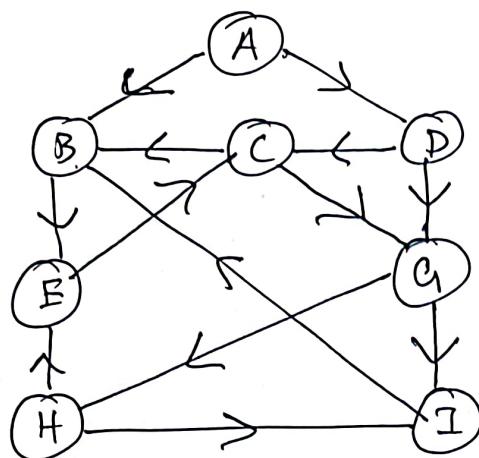


(7)	BFS	visit	neighbour	queue	BFS path
<del>a ≈ b</del>		A	B, D	B, D.	= A, B, D, E, C, G, H, I
		B	E	D, E	
		D	C, G	E, C, G.	
		E	C,	C, G.	
		C	B, G	G.	
		G	H, I	H, I.	
		H	I, E	I	
		I	B		Same as Level order.

BFS Spanning Tree

Tree Edges =  $(A, B), (A, D), (B, E), (D, C), (D, G), (G, H), (G, I)$ .Cross Edges =  $(E, C), (B, C), (C, G), (H, I), (B, I), (H, E)$

(8) (a) &amp; (b)

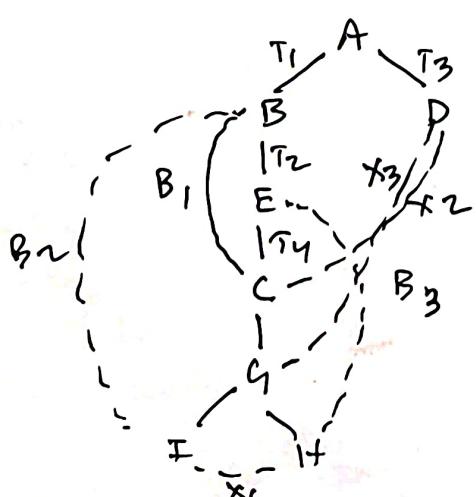
DFS  
Traversal

Tree  
 $(A, B), (A, D), (B, E), (E, C)$   
 $(C, G), (G, I), (G, H)$

Back  
 $(C, B), (I, B), (H, E)$

Cross  
 $(H, I), (D, C), (D, G)$

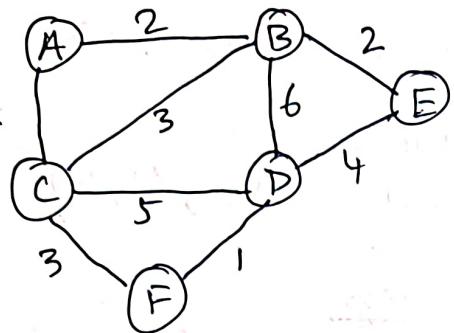
visit	Neighbour	Stack
A	B, D	A
B	E	A, B
E	C	A, B, <del>E</del> , E
C	B, G	A, B, E, C
G	I, H	A, B, E, C, G
I	B	A, B, E, C
G	I, H	A, B, E, G, G
H	I, E	A, B, E, C
C	B, G	A, B, E
E	C	A, B
B	I	A
A	B, D	A
D	C, G	
A		



DFS path

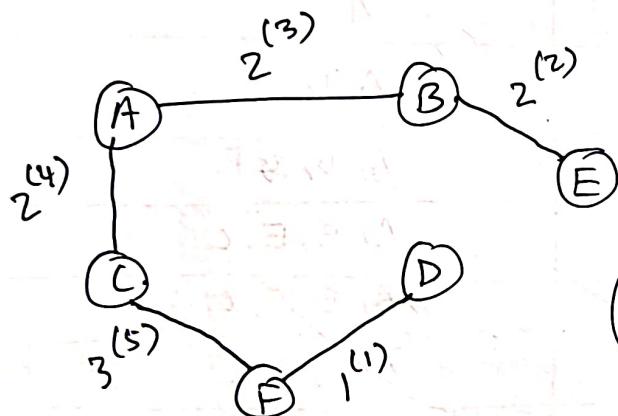
= A, B, E, C, G, I, H, D

pre-order Traversal  
path.

(9) Original GraphKruskal

Focus 1 = min cost

Focus 2 = should be connected



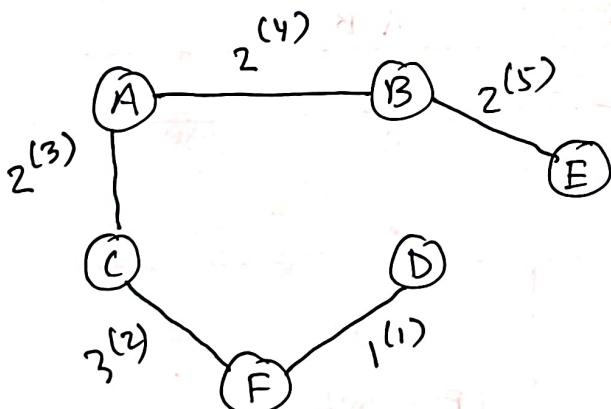
$$\text{min cost} = 2 \times 3 + 3 + 1 = 10$$

Kruskal

Prim

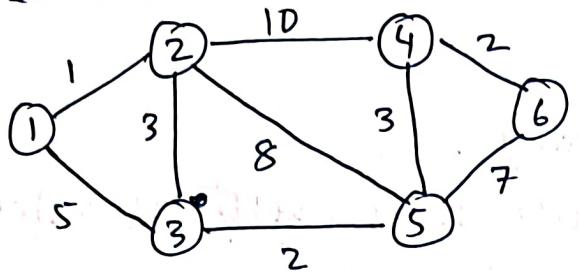
Focus 1 = connected

Focus 2 = min cost



$$\text{min cost}_{\text{prim}} = 10$$

### Dijkstra



### Problem

From source = 1

Find the min distances  
to all nodes.

constraint - Path should exist

objective - min distance from source

### Relaxation Logic

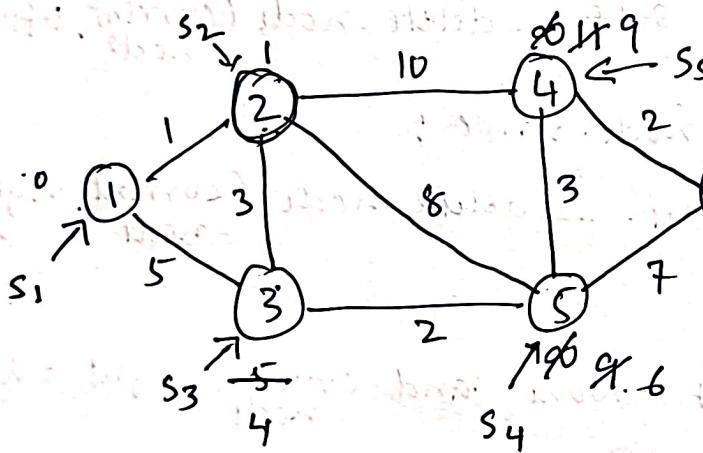
$$d[u] + c(u,v) < d[v]$$

$$d[v] = d[u] + c(u,v)$$

\* choose  
least value global min

always

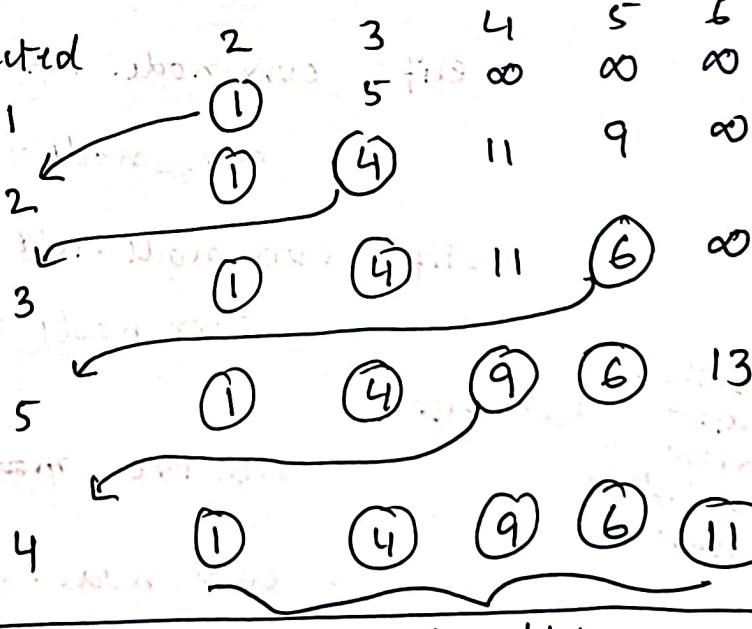
NOT selected  
node.



	1 - 2	1 - 3	1 - 4	1 - 5	1 - 6
	1	5	9	6	11
	2	4			
	3				
	4				

min distances

selected



min distances

## 11. Delete node in B.S.T

Idea

```
def delete-node(self, value):
```

```
    return self. -- delete-node(self.root, value)
```

```
def -- delete-node(self, current-node, value):
```

if current-node is None:

```
    return None
```

Search

elif value > current-node.value:

```
    current-node = self. -- delete-node(current-node.left, value)
```

elif value < current-node.value:

```
    current-node = self. -- delete-node(current-node.right, value)
```

Value found

else:

if current-node.left == None and curr-node.right == None

```
    return None
```

elif curr-node.left == None and curr-node.right != None

```
    curr-node = curr-node.right
```

elif curr-node.left != None and curr-node.right == None

```
    curr-node = curr-node.left
```

Both  
left +  
right  
child  
exist

else:

sub-tree-min = self.min-value(curr-node.right)

```
curr-node.value = sub-tree-min
```

```
curr-node.right = self. -- delete-node
```

```
(curr-node.right, sub-tree-min)
```

```
def min-value(self, curr-node):
```

while curr-node.left is not None:

```
    curr-node = curr-node.left
```

```
return curr-node.value
```