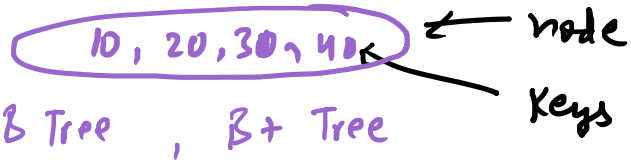
Lecture 11-12

Wednesday, October 27, 2021 8:00 PM

External Searching :- To locate or retrieve a record which is stored in a secondary memory like disk.

Multiway Search Tree :- A M-way search tree contains multiple keys in a node, which leads to reduction of the overall height of the tree.

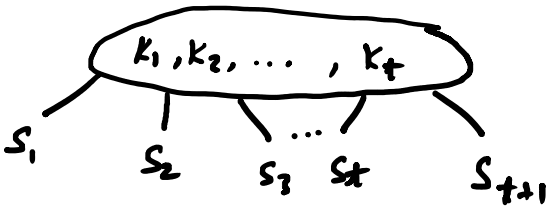


eg:- B Tree , B+ Tree

B Tree :- Balanced Tree

A B Tree of order 'm' is a multiway search tree with the following properties :-

1) The root can have atleast 1 key & atmost m-1 keys
2) All nodes (except the root) can have atleast $\lceil m/2 \rceil$ keys and atmost m-1 keys.
3) All leaves are at the same depth.
4) If a node has 't' keys, then it will have 't+1' children
5) keys of a node are sorted in ascending order.



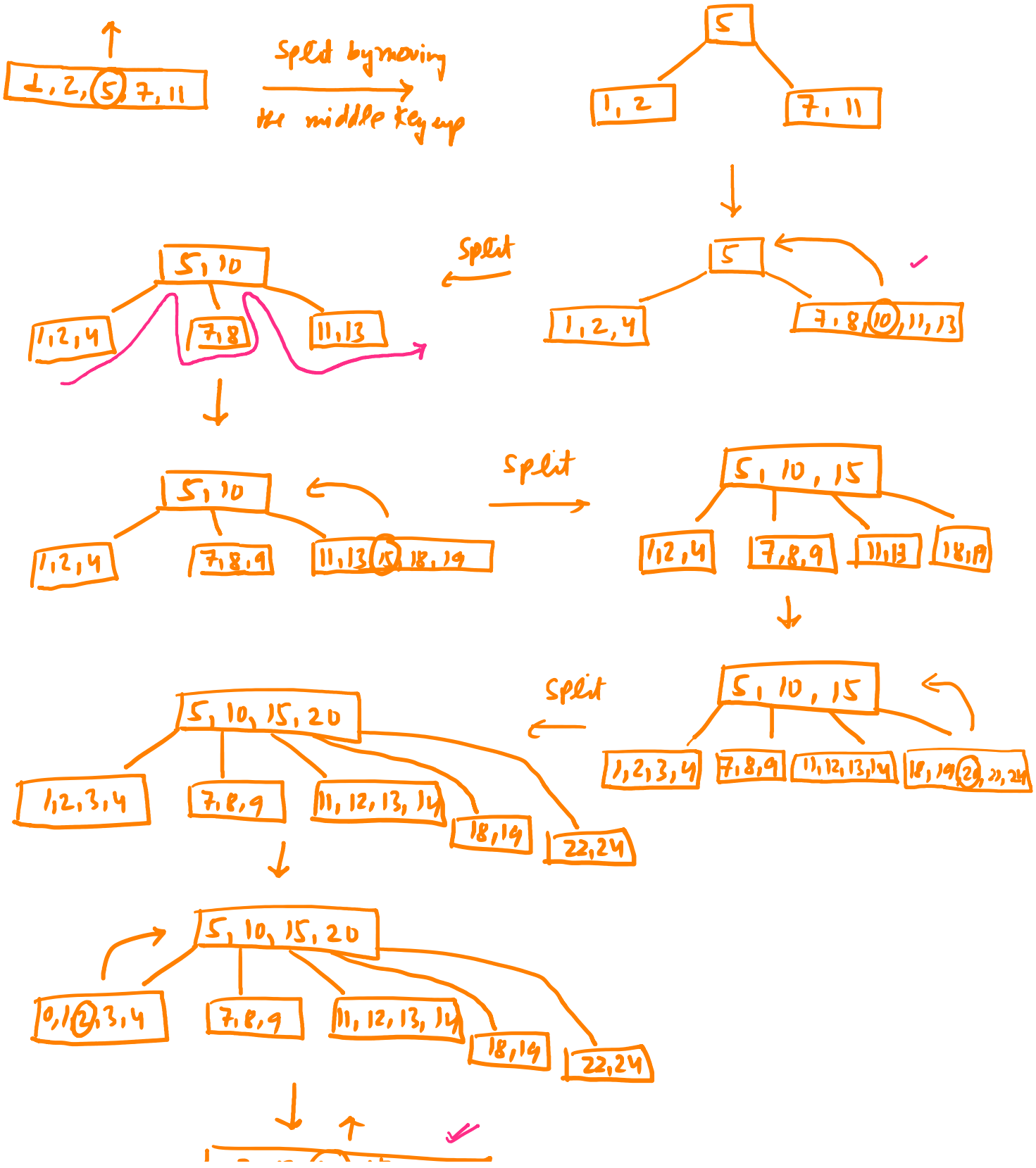
$K_1 < K_2 < \ldots < K_t$

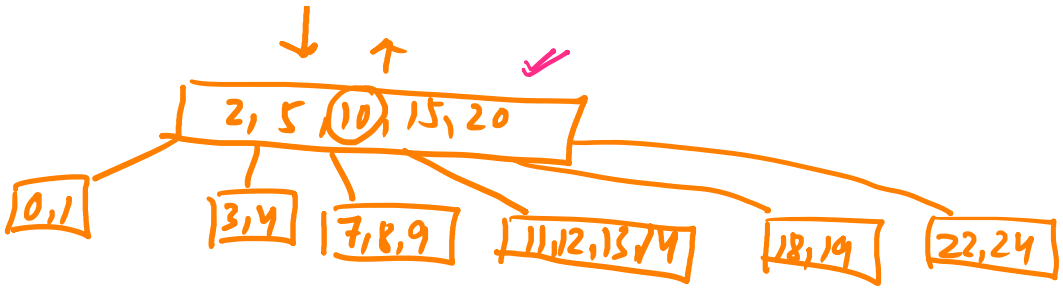
6) All the keys in the subtrees pointed by $S_1, S_2, \ldots, S_{t+1}$,

$S_1 < K_1 < S_2 < K_2 < S_3 \ldots < S_t < K_t < S_{t+1}$

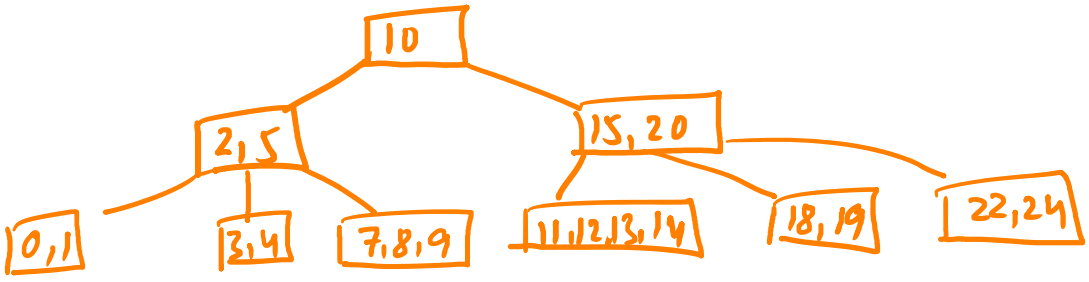
Insert the following keys in a B Tree of order 5.

1, 7, 5, 2, 11, 4, 8, 13, 10, 15, 19, 9, 18, 24, 3, 12, 14, 20, 21, 6





split



Deletion :-

1) For a key in a non-leaf node → swap the key with its inorder successor & then delete the inorder successor.

2) For a key in a leaf node

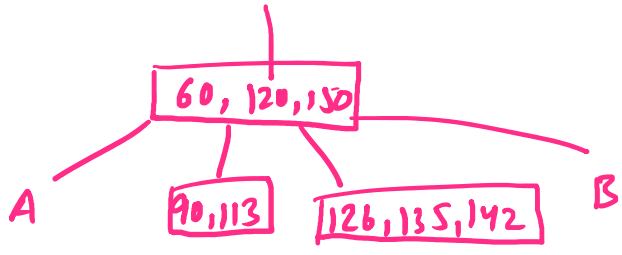
(a) If after deletion of the key property 2 holds then no issues

(b) But if property 2 does not hold then there is a issue.
keys in that leaf node < $\lceil m/2 \rceil$, in case of more than 1 nodes or a non-root node

Solutions to 2 -(b) :-

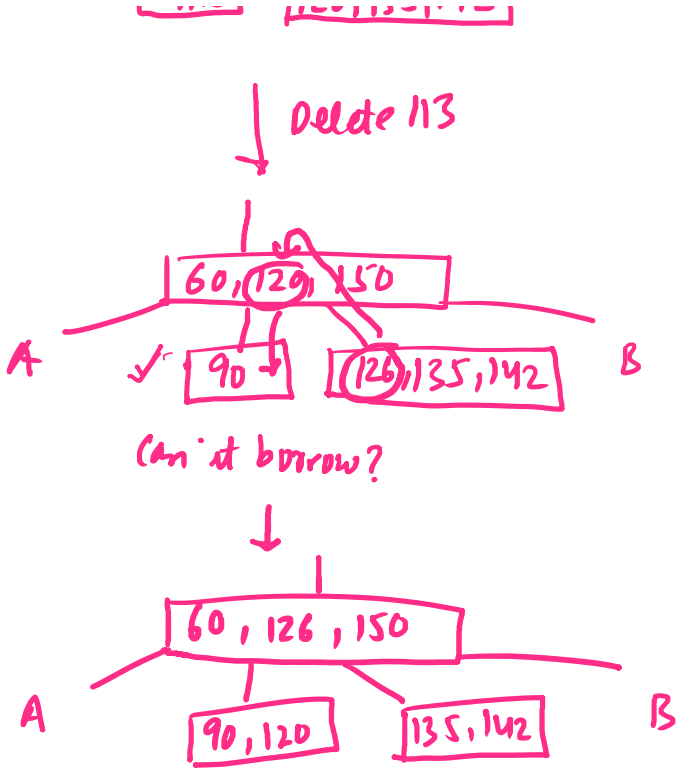
Assume $m = 5$

1) Borrowing



2 options :-

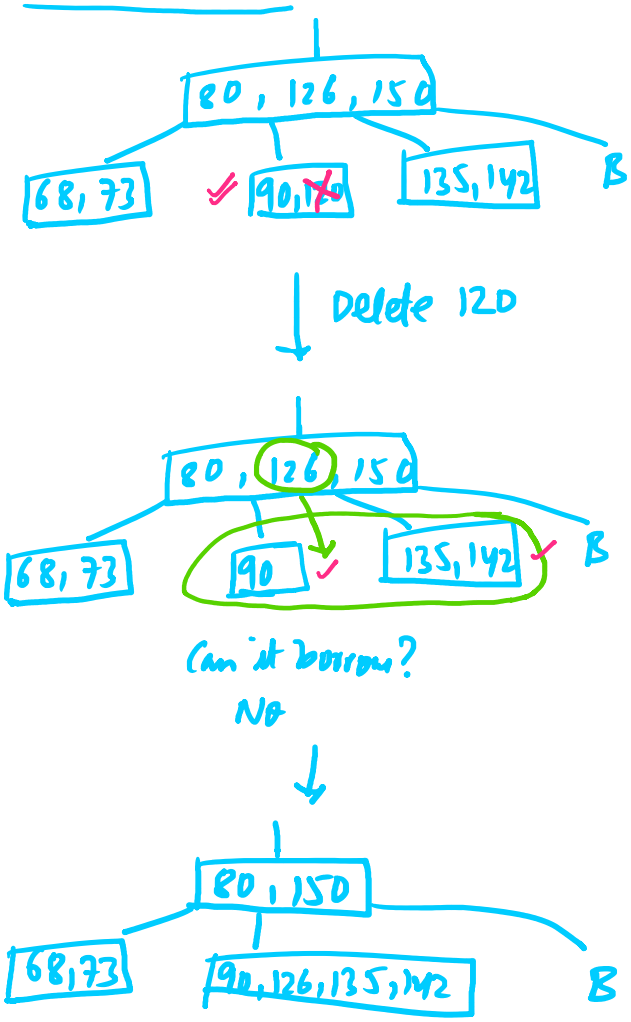
1) First ke... right



2 options:-

1) First key of right sibling node via parent
2) Last key of left sibling node via parent

2) Consolidation :-

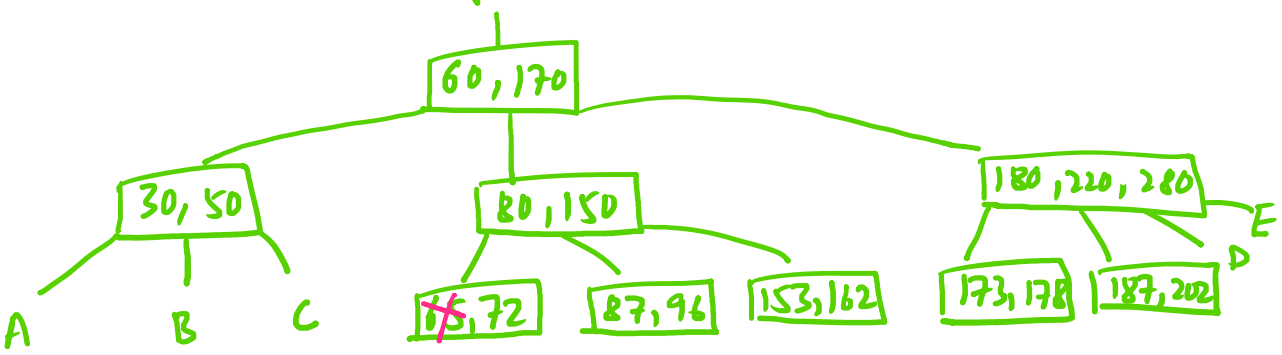


2 options:-

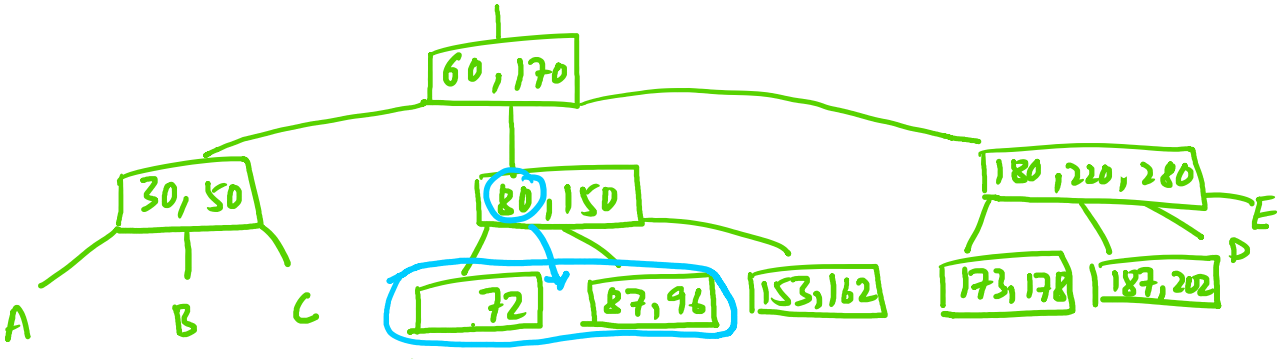
1) Combine with the right sibling node by moving the middle key down.
2) Combine with the left sibling node by moving the middle key down

B → C → B → C → ...
i parent(i)

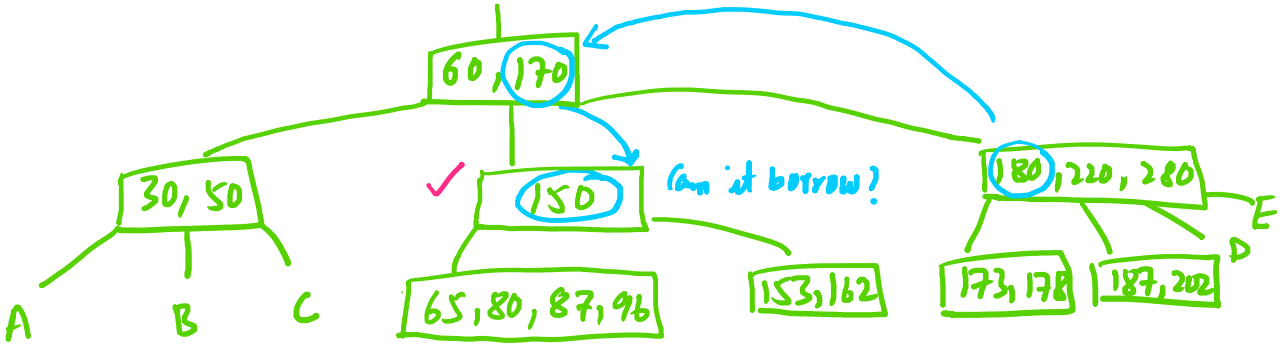
3) Consolidation & Borrowing



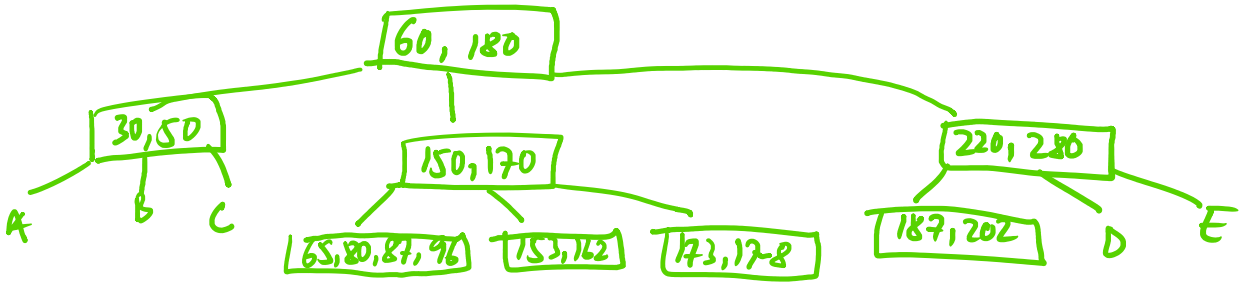
↓ Delete 65



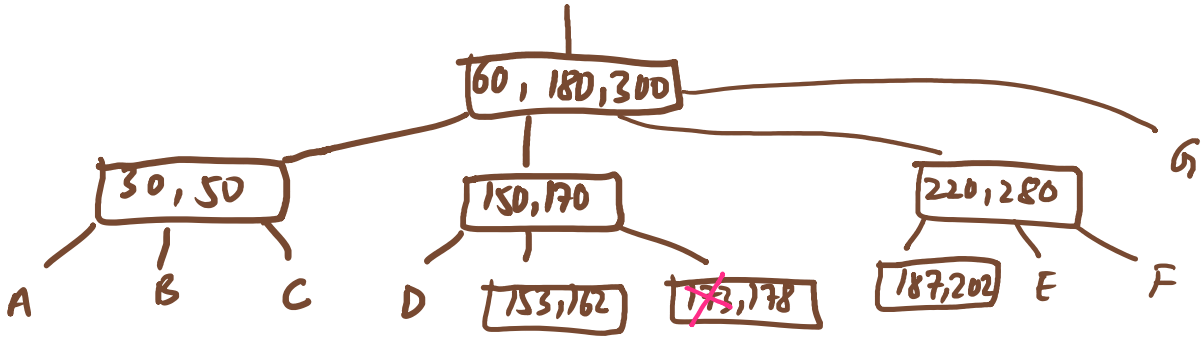
Can it borrow?
No
Consolidation



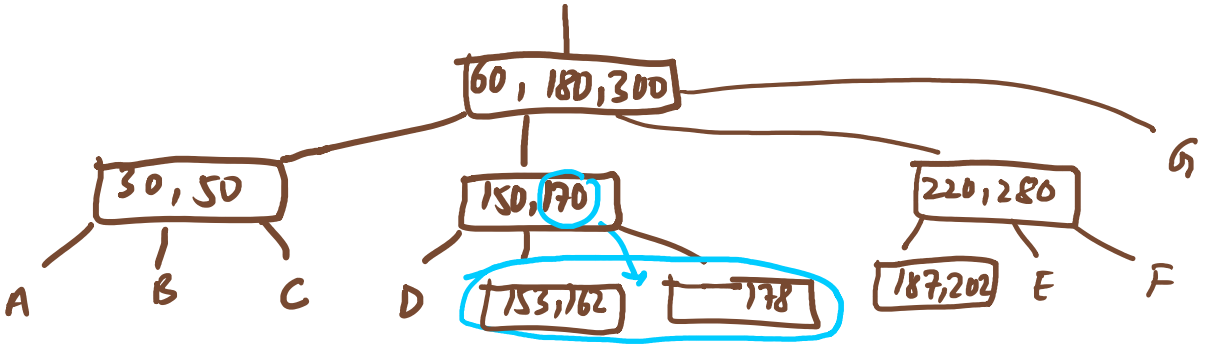
↓



4) Double Consolidation :-

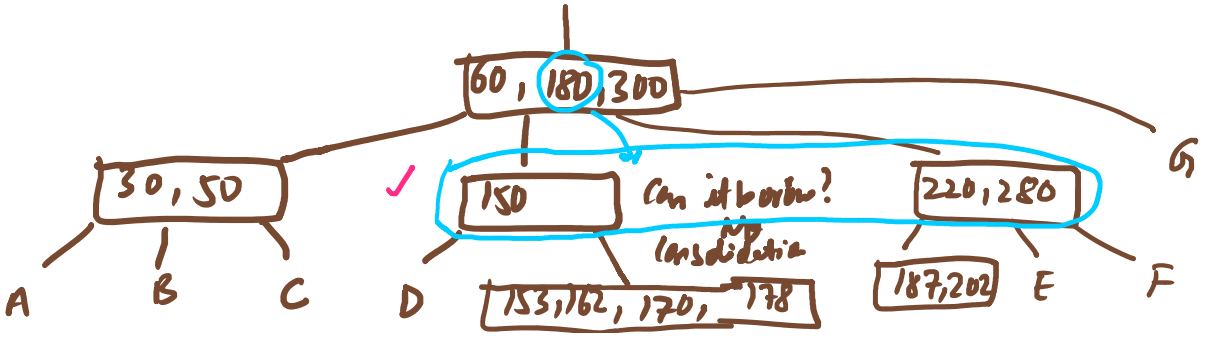


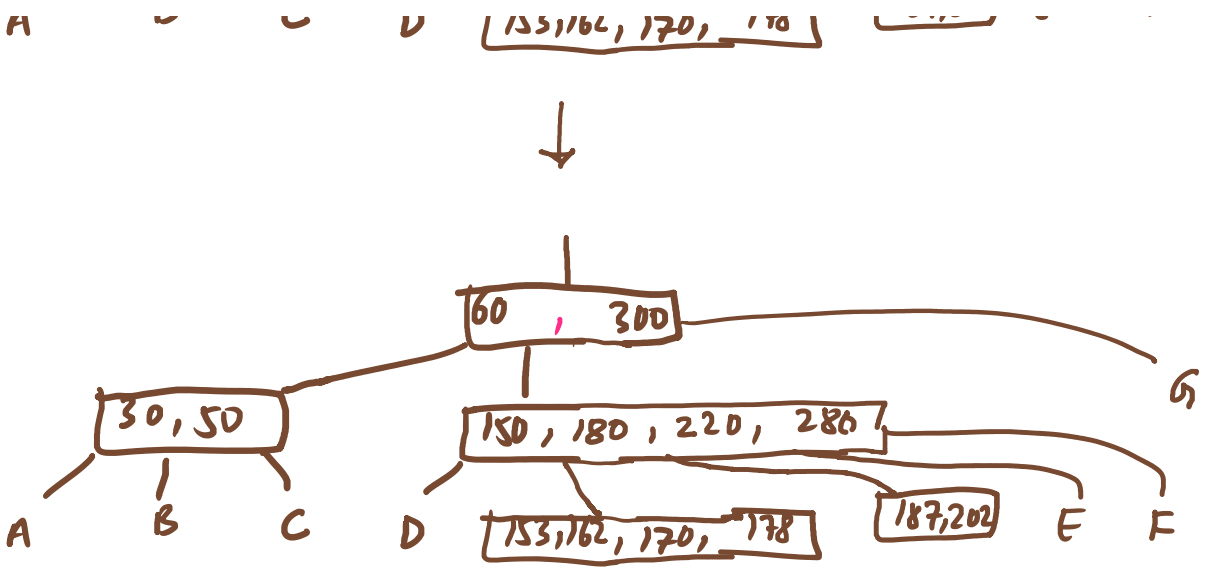
↓ Delete 173



Can it borrow?
No
Consolidation

↓



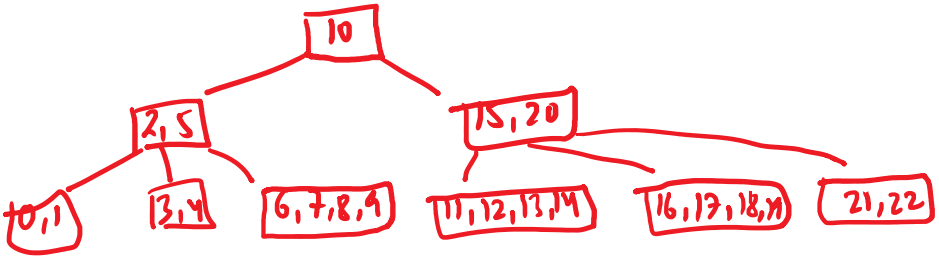


B + Tree :-

Drawback of a B tree :-

1) (Key, Value) pair is present in all the nodes (internal & external)

Index Data



This will increase the no. of levels thereby increasing the height.

2) Searching of multiple keys is time consuming
eg:- 1 , 2 what if the leaves are connected.

B + Tree :- Two types of nodes

1) Internal Nodes – Only keys are stored → index information
2) External Node – Actual data or record will be stored.

All the external nodes are connected via linked list.

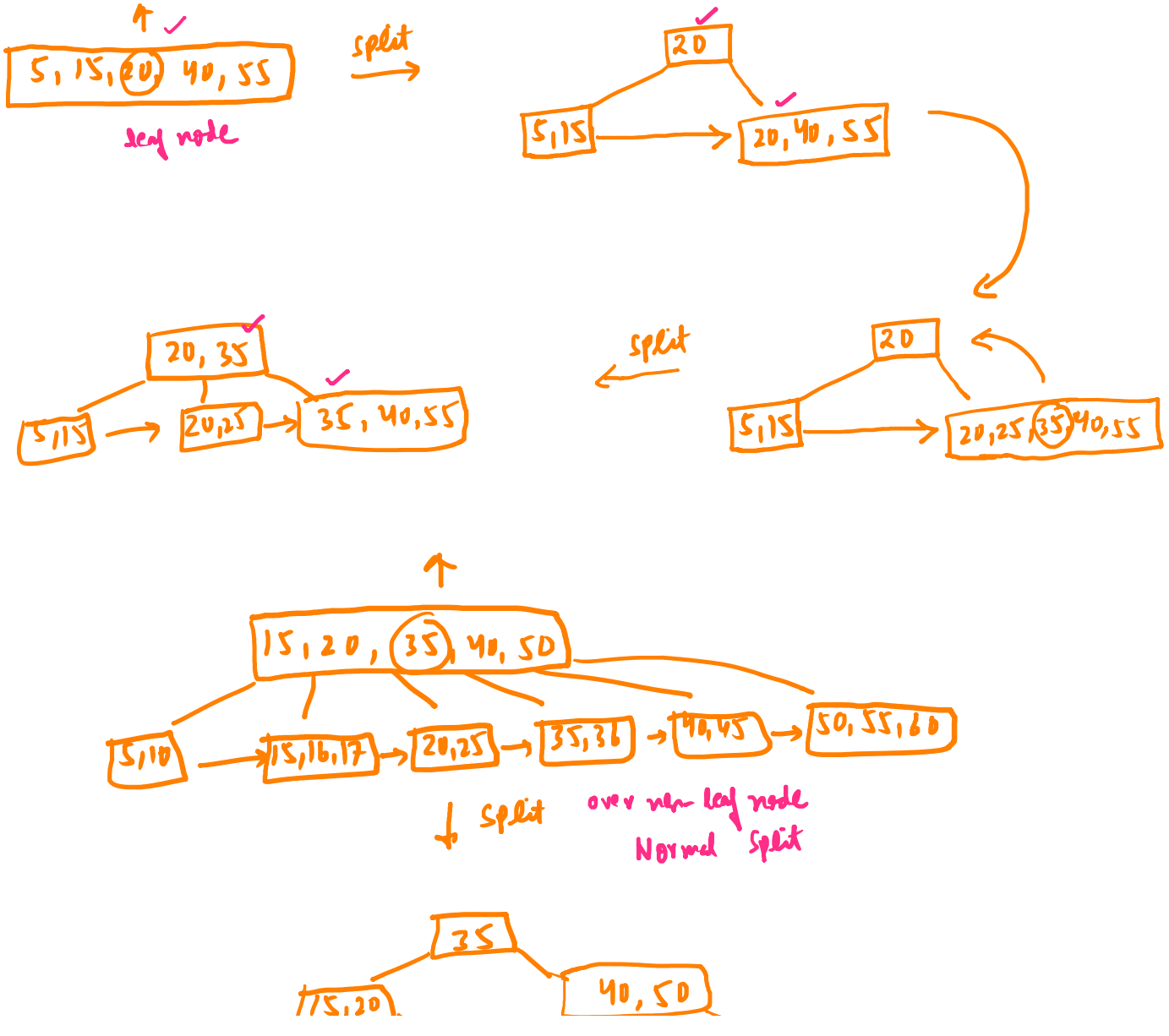
B+ Tree Insertion :-

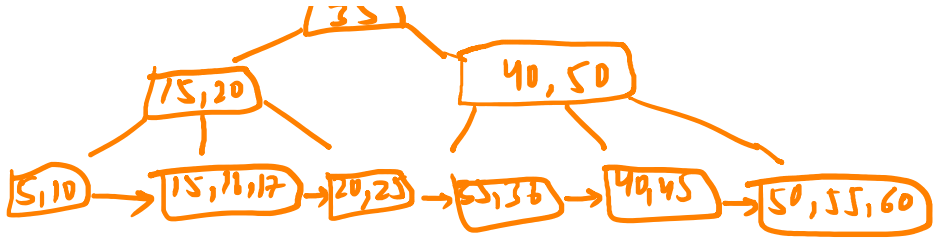
→ B+ Tree of order 'm' will follow all the properties of a B Tree of order 'm'.

→ Whenever a split is performed over a leaf then retain a copy of that middle element in the right child leaf node

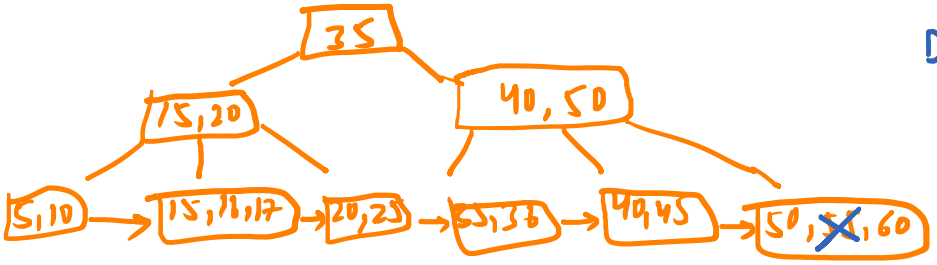
Insert the following keys in a B+ Tree of order 5.

5, 40, 15, 20, 55, 35, 25, 10, 36, 50, 45, 60, 16, 17

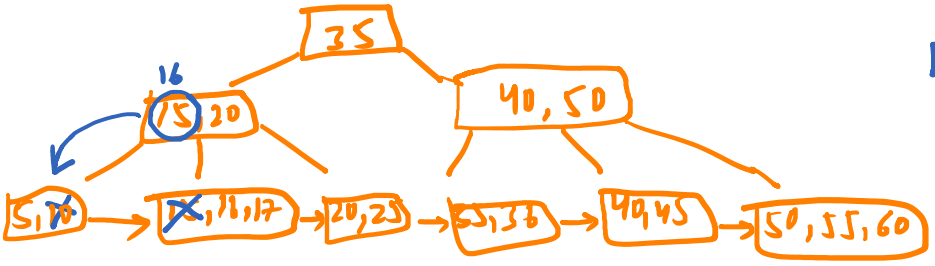




B+ Tree Deletion :- Logic is same as B Tree

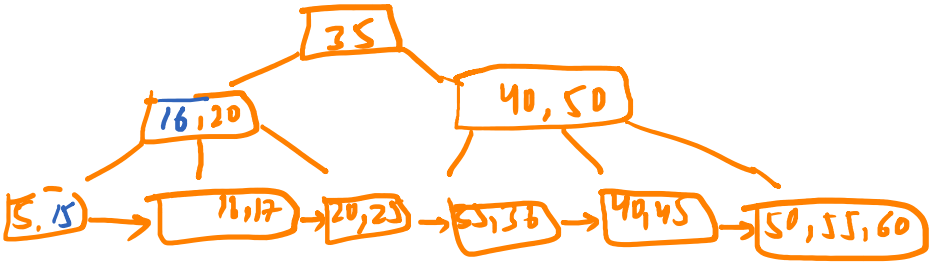


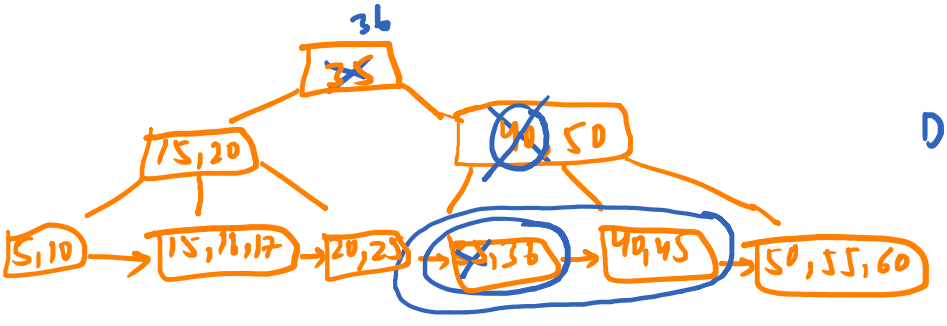
Delete 55



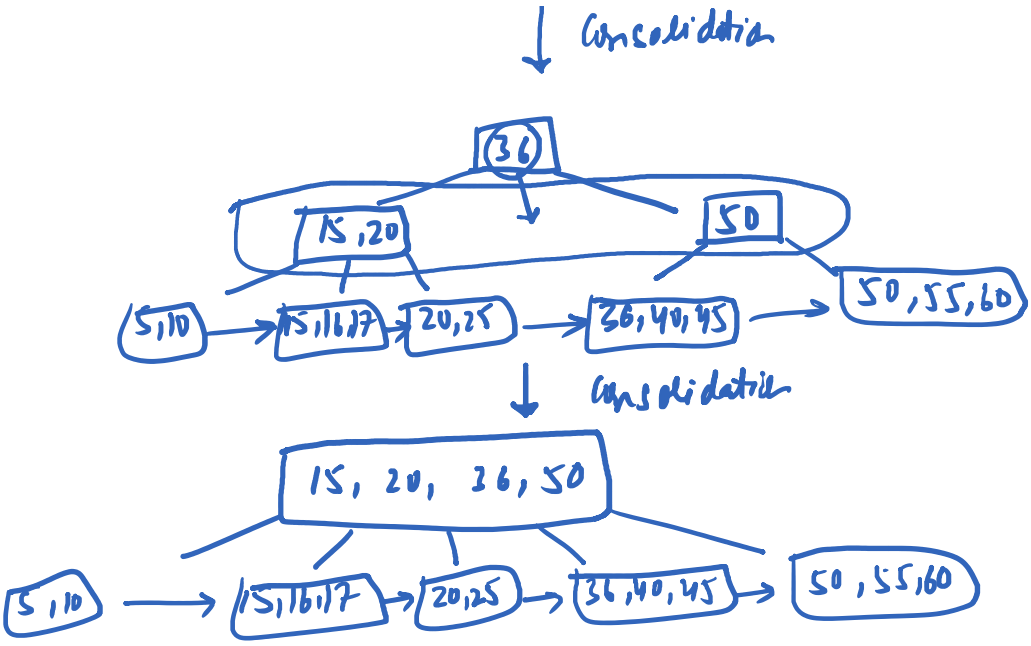
Delete 10

↓ Borrowing





Delete 35



Hashing :- Time Complexity to search a record

$\theta(1)$, $O(n)$

mapping the data to its location.

Hash function $h(x) = y$

(x ↑ data, y ↑ location)

DS used is a hash table

(collision : if x_1 & x_2 are 2 data values where $x_1 \neq x_2$

but $h(x_1) = h(x_2)$

Example :- Insert the following elements in a hash table of size 10

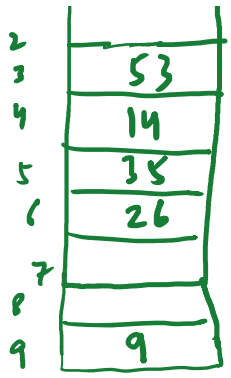
20, 35, 14, 26, 9, 53

0	20
1	
2	
3	53

Hash function = $x \% n$

x mod n

(x ↑, n ↑)



x mod m

↑ data ↑ size of hash table

$h(x) = x \,\%\, 10$

$h(20) = 20 \,\%\, 10 = 0$

$= 35 \,\%\, 10 = 5$

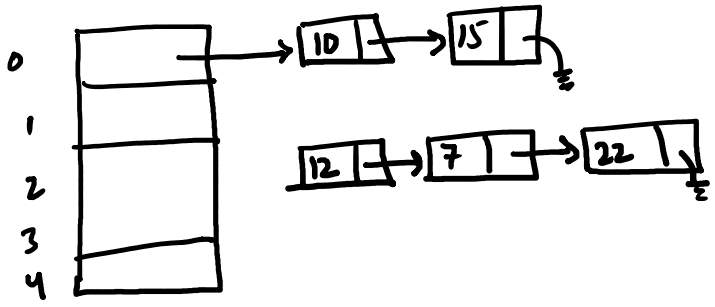
$h(26) = 26 \,\%\, 10 = 6$

$\Theta(1)$

Collision Resolution Techniques :-

1) Direct chaining :- Array of linked lists

Separate chaining



$h(x) = x \,\%\, 5$

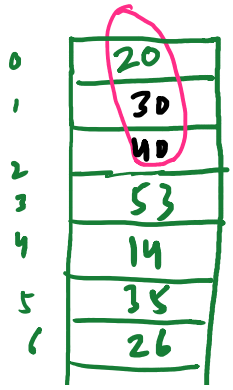
2) Open Addressing :- Array based implementation

Probing ┬ Linear Probing

└ Quadratic Probing

Linear Probing :- $h(x) = (x + i) \,\%\, n$ rehasing function

where $i = 1, 2, 3, \ldots$



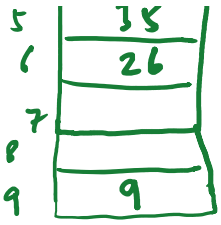
Insert 30

$h(30) = 30 \,\%\, 10 = 0$ collision

$re\text{-}h(30+1) = 31 \,\%\, 10 = 1$ variant

Insert HD

Primary Clustering



Primary Clustering

Insert 40

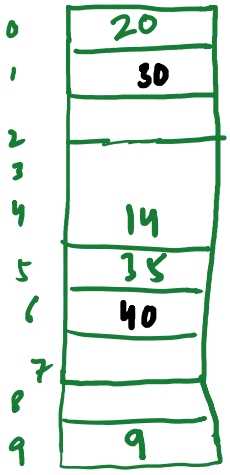
$h(40) = 0$ collision

re- $h(40+1) = 1$ collision

re- $h(40+2) = 2$ vacant

Quadratic Probing $re\text{-}h(x) = (x + i^2) \% n$

where $i = 1, 2, 3, \ldots$



Insert 40

$h(40) = 0$ collision

$h(40+1^2) = 1$ collision

$h(40+2^2) = 4$ collision

$h(40+3^2) = 9$ collision

$h(40+4^2) = 6$ vacant

Huffman Tree :- Used for data compression

Symbol	Code (Fixed length)
A	0 0 0
B	0 0 1
C	0 1 0
D	0 1 1
E	1 0 0

000 001 010

A B C

length of message = n symbols

size of message = 3n

3 bits per symbol

Can this be reduced?

Lossless compression

Huffman tree used variable length codes.

Frequently occuring symbol should be given shorter codes.

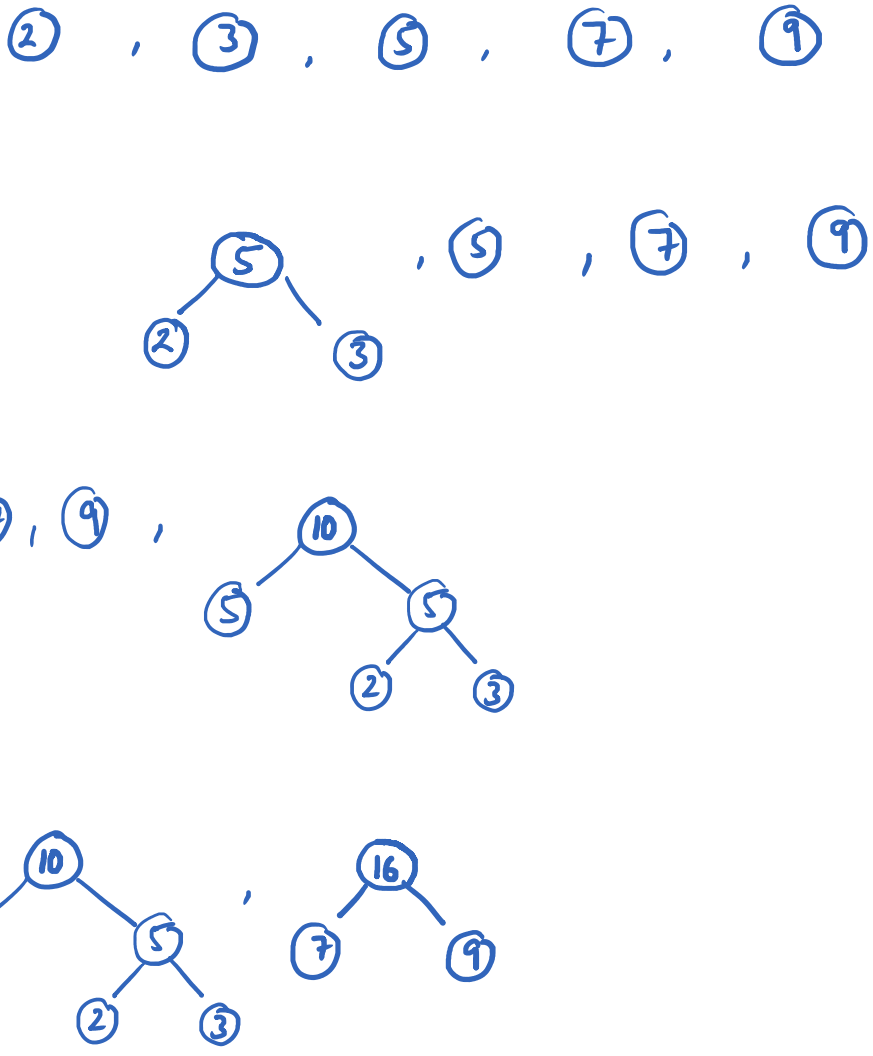
Symbol	Frequency
A	2
B	5

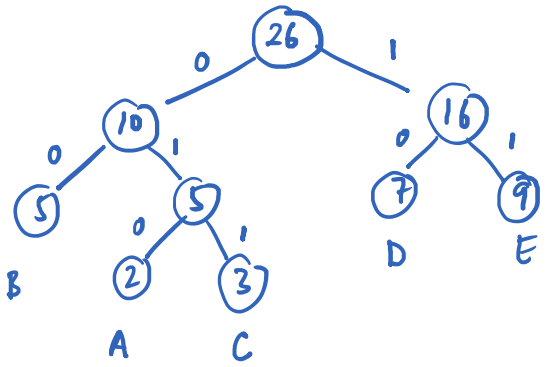
Huffman Tree Algorithm :-

1) All the symbols with their frequencies shall

A	2
B	5
C	3
D	7
E	9
	26

1) All the symbols with their frequencies shall become individual nodes (or single node tree)
2) Select 2 trees with minimum root value.
3) Combine these 2 trees by making one as Left Subtree & other as right subtree, and their overall root node shall have the value equivalent to the sum of individual roots.
4) Repeat step 2 to 3 until one tree remains & shall be the Huffman Tree.





Huffman Tree

Symbol	Frequency	Huffman code
A	2	0 10
B	5	00
C	3	0 11
D	7	10
E	9	11

Average bits/Symbol $= \frac{3+2+3+2+2}{5}$

$= 2.4$ bits/Symbol

Average bits/Symbol $= \frac{\sum_{i=1}^{n} f_i x_i}{\sum_{i=1}^{n} f_i} = \frac{2\times3 + 5\times2 + 3\times3 + 7\times2 + 9\times2}{26}$

$= \frac{6 + 10 + 9 + 14 + 18}{26}$

$= 2.19$ bits/Symbol

$-\log_2 P_i$

Average information content

Shannon's Entropy $= -\sum_{i=1}^{n} p_i \log_2 p_i$

$= -\left[\frac{2}{26}\log_2\frac{2}{26} + \frac{5}{26}\log_2\frac{5}{26} + \frac{3}{26}\log_2\frac{3}{26} + \frac{7}{26}\log_2\frac{7}{26} + \frac{9}{26}\log_2\frac{9}{26} \right]$

$= 2.5$ bits/Symbol

A A A A A

$E = -\sum_{i=1}^{n} p_i \log_2 p_i$

$E = -P_A \log_2 P_A \qquad P_A = 1$

min value of entropy $\boxed{E = 0} \Rightarrow$ no randomness

---

A B C D E

$E = -\sum_{i=1}^{n} p_i \log p_i$

$P_i = \frac{1}{5}$

A B C D E

$\sum_{i=1}$ $p_i \log p_i$

$= -5\left[\frac{1}{5}\log\frac{1}{5}\right]$ $\quad p_i = \frac{1}{5}$

$= -\log_2 \frac{1}{5} = \boxed{\log_2 5}$

max. value of entropy $= \log_2 n$

Highly random