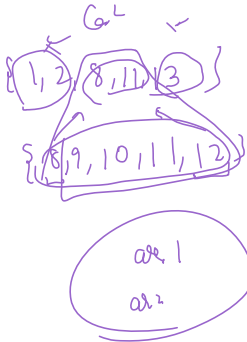
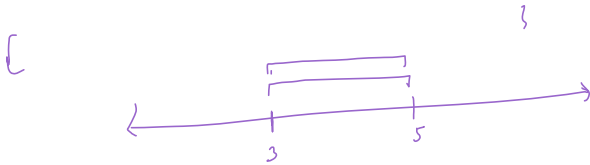


$\rightarrow \{1, 2, (3), 4, (5)\}$   
 $\rightarrow \{(3), 4, 5, 6, 7\}$   
 $\{1, 2, 3, 3, 4, 5, 6, 7\} \mid 5$



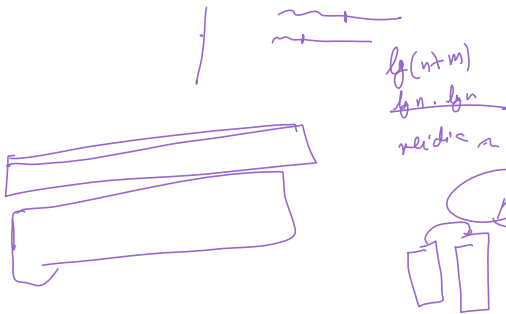
arr 2:

$0 \ 1$   
 $\{8, 1\}$   
 $\{9, \}$

$$\frac{m+n}{2} + \frac{m+n}{2}$$



$\Rightarrow \begin{matrix} F & F & T & T \\ 0 & 2 & 5 & 7 \\ 0 & 1 & 3 & 4 \\ \{8, 11, 15, 17\} \end{matrix}$   
 $\begin{matrix} F & T & T \\ 1 & 3 & 6 \\ \{9, 12, 13, 15\} \end{matrix}$   
 $f(n, n) \Rightarrow f(n) + f(n)$   
 $f(n, m)$



K sorted array

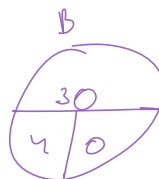
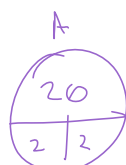
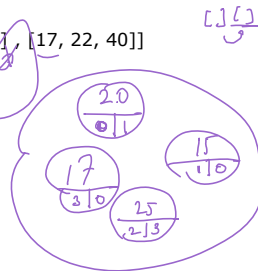
Q.  $[[10, 20, 30], [15, 35], [5, 7, 12, 25], [17, 22, 40]]$

M-1) n log n

M-2) n log k

$\Rightarrow \begin{matrix} 1 & 0 & 2 & 0 \\ 3 & & & \end{matrix}$

M-3) n log k  $\rightarrow \{5, 7, 10, 12, 25\}$



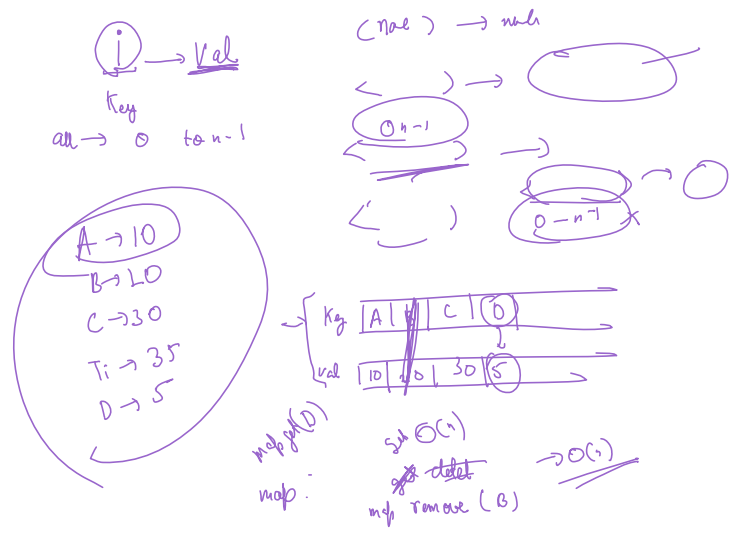
A. compare To (B)

A. diff = B diff

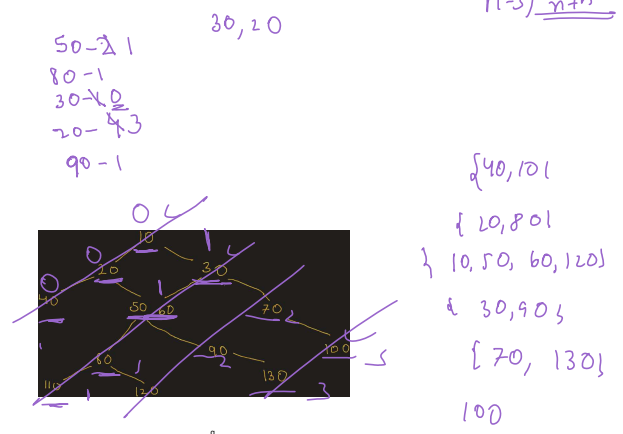
$\{1, 5, 6, 4, 3, 2, 10, 3\}$

$\{ \underline{1}, \underline{5}, \underline{6}, \underline{4}, \underline{3}, \underline{2}, \underline{10}, \underline{3} \}$   
 $1 \rightarrow 1$   
 $1, 2, 3, \dots \rightarrow 5, 6, 4 \rightarrow 7$   
 $1, 2, 3 \rightarrow 4$   
 $1, 2, 3, 4 \rightarrow 5, 6, 10$   
 $\downarrow \{ 1, 5, 6 \}$   
 $1, 4 \rightarrow 5, 6 \rightarrow 4$   
 $1, 3, 4 \rightarrow 5, 6 \rightarrow 4$

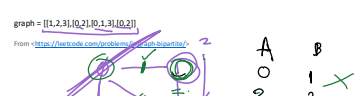
Hash-Map



$\text{int[]} \text{arr1} = \{ 30, 20, 40, 50, 70, 30, 20, 20, 50, 50 \};$   
 $\text{int[]} \text{arr2} = \{ 50, 80, 30, 20, 20, 20, 90, 50, 20 \};$   
 $n-1 \rightarrow n^2$   
 $n-2 \rightarrow n^3$   
 $n-3 \rightarrow n+n$

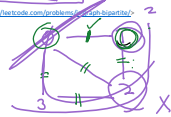


$\text{Map} \langle L, \text{ArrayList} \rangle$   
 $0 \rightarrow 10, 50, 60$   
 $1 \rightarrow 20$   
 $-1 \rightarrow 30$   
 $2 \rightarrow 40$   
 $-2 \rightarrow 70$

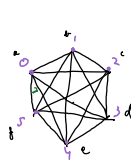


graph = [[1,2,3],[0,2],[0,1,3],[0,2]]

From <https://leetcode.com/problems/minimum-cost-to-connect-all-points/>



A	B
0	1
2	3



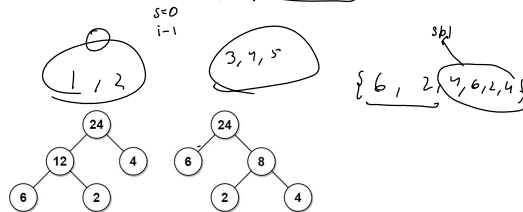
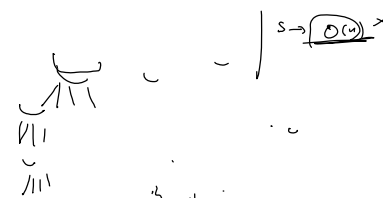
{0, 1, 2, 3, 4, 5}

$\rightarrow A + \{0, 2, 3, 4, 5\}$   
 $\Delta + \{1, 2, 3\} + \{3, 4, 5, 0\}$

0 5

1)

$sp1 \rightarrow \Delta + (4, e)$   
 $\rightarrow \Delta + (0, 2) (2, e)$   
 $\rightarrow \Delta + (0, 3) (3, e)$

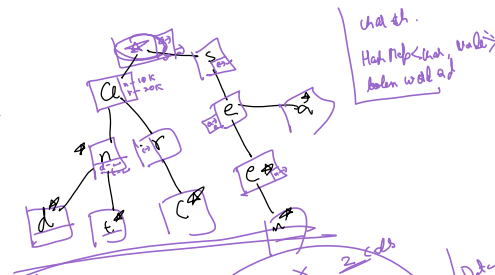


```
public int mctFromLeavesToRoot(int[] arr) {
    return solve(arr, 0, arr.length-1);
}

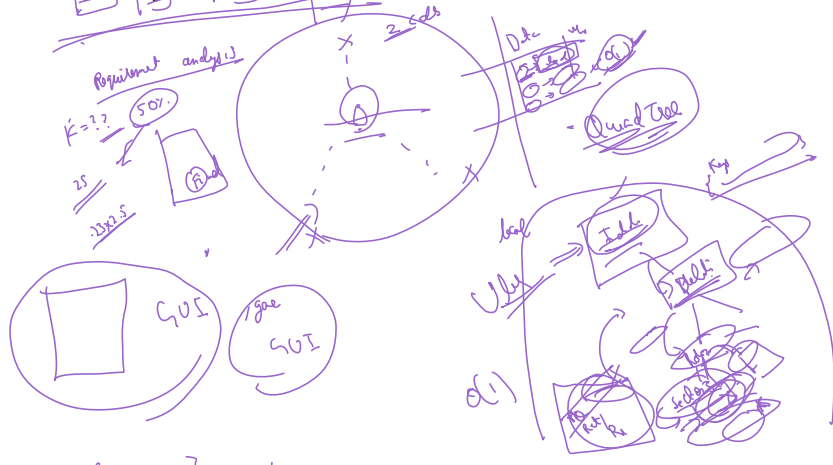
public int solve(int[] arr, int s, int e) {
    if(s==e){
        return 0;
    }
    int ans = Integer.MAX_VALUE;
    for(int i=s+1; i<=e; i++){
        // int sp = min(arr[s,i-1]*max(arr,i,e)+solve(arr,s,i-1)+solve(arr,i,e);
        ans = Math.min(ans, sp);
    }
    return ans;
}

public int mctFromLeavesToRoot(int[] arr, int s, int e) {
    int ans = arr[s];
    for(int i=s+1; i<=e; i++){
        ans = Math.max(ans, arr[i]);
    }
    return ans;
}
```

```
trie.addWord("ant");
trie.addWord("arc");
trie.addWord("and");
trie.addWord("see");
trie.addWord("sea");
trie.addWord("seen");
trie.addWord("an");
```

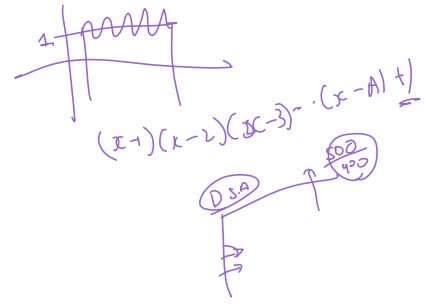


check  
 High Node < Low Node  
 below will add



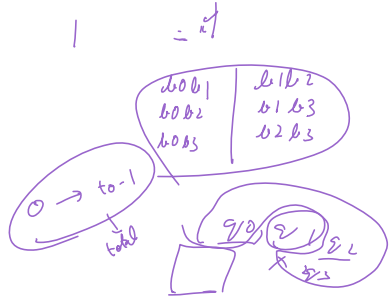
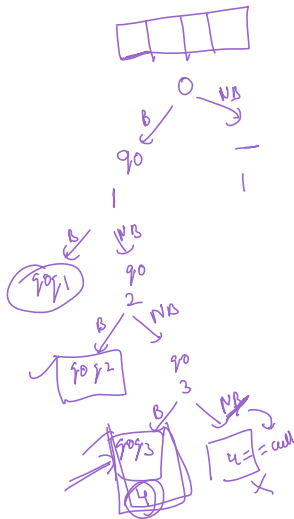
$f(1) = 1$   
 $f(2) = 1$   
 $f(3) = 1$

$f(1)=1$   
 $f(2)=1$   
 $f(3)=1$   
 $\vdots$   
 $f(A)=1$



3.

20

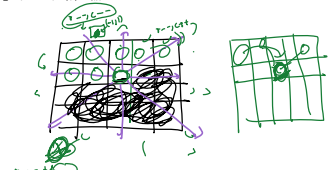
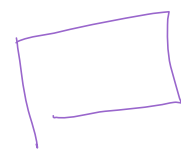


```

public static void combSeats(int total_box, int curr, int Qtp, String path) {
    if (Qtp == 0) { // ve BC
        System.out.println(path);
        return;
    }
    if (curr == total_box) { // -ve BC
        return;
    }
    combSeats(total_box, curr + 1, Qtp - 1, path + "Q" + curr); // include
    combSeats(total_box, curr + 1, Qtp, path); // exclude
}

```

pop 1  
 pop 2  
 pop 3



	0	1	2	3	4	5	6	7	8	9
0	5	3	8	0	7	1	6	4	9	2
1	6	0	4	1	9	5	0	1	7	8
2	8	9	8				6			
3	4	1	0	8	6		3	1	2	3
4	7									6
5	6						2	8		
6	2	0	4	1	9	2	1	4	5	
7							8		7	9

Ans {1, 2, 3, 4} Input  
 comb 1, 1, 1, 1  
 2, 2  
 2, 1, 1  
 3, 1  
 4

