

UNIT-2

"Divide & Conquer"

- 3/12/18
- General Method (or) Abstraction Method of Divide and Conquer
 - Defective chess board
 - Binary search
 - Quick Sort
 - Merge Sort
 - Finding Maximum and Minimum
 - Performance Measurement
 - Randomized Sorting Algorithm, e.g. quick sort

Divide and conquer (General Method):-

- * If the given problem is a small problem then return a solution
- * If the given problem is a large then apply divide and conquer method.
- * If the given problem is large then it can be divided into possible number of sub problems.
- * For each sub problem we have to find out solutions
- * Finally all the solutions are combined

Explain Divide & Conquer Method Algorithm:

DAC(P)

↓

If (small (P))

↓

$S(P)$;

Else

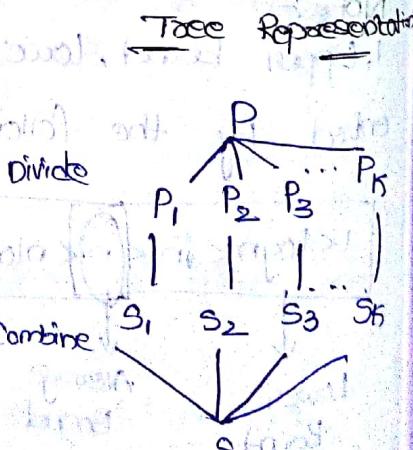
↓

divide P into $P_1, P_2, P_3, \dots, P_k$

Apply $DA(P_1), DA(P_2), DA(P_3), \dots, DA(P_k)$

Combine $DA(P_1), DA(P_2), DA(P_3), \dots, DA(P_k)$

↓



Note: 2

* All the divide and conquer algorithms are recursive hence we can find out the time complexity using Recurrence Relation

Binary Search :-

* In the binary search method the elements are sorted order that is ascending order.

* If we want search for the element say 'x'.

* Then we first divided the list at middle so that two sublists are created.

* If x greater than middle then right sub list consider and x is searched in the right sub list otherwise x is searched in the left sub list.

e.g. if the given list of elements are 3, 6, 8, 12, 14, 17, 25, 29, 31, 36, 42, 47, 53, 55, 62. and key element is 42. Then find a key element 42 is available or not and also find time complexity

* Consider the given list of elements are

3	6	8	12	14	17	25	29	31	36	42	47	53	55	62
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

let us assume l=1

high, h=n=15

then find mid value mid (m) = $\lfloor (l+h)/2 \rfloor$

if always low less than are equal to high then find

$$\text{mid } (m) = \frac{(l+h)}{2}$$

If (key = mid) then $42 = 29$ hence it is false.

If (key < mid)

$42 < 29$ false

If (key > mid)

$42 > 29$ true

then $low = mid + 1$

$$l = 8 + 1$$

$$l = 9$$

$$\text{now } l=9, h=15, m = \frac{9+15}{2} = \frac{24}{2} = 12.$$

if (key == mid) then

$$42 == 47 \text{ false}$$

if (key < mid) then

$$42 < 47 \text{ True.}$$

$$high = mid - 1$$

$$h = 12 - 1$$

$$h = 11$$

$$\text{now } h=11, l=9, m = \frac{11+9}{2} = \frac{20}{2} = 10$$

if (key == mid)

$$42 == 36 \text{ false}$$

if (key < mid) then

$$42 < 36 \text{ false}$$

if (key > mid) then

$$42 > 36 \text{ True}$$

$$low = mid + 1$$

$$l = 10 + 1$$

$$l = 11$$

$$\text{now } l=11, h=11, m = \frac{11+11}{2} = 11$$

if (key == mid) then

$$42 == 42 \text{ True}$$

the element is found.

Now key is 47.

The time complexity of binary search is

	Bestcase	Average Case	Worst case
successful	$O(1)$	$O(\log n)$	$O(\log n)$
unsuccessful	$O(\log n)$	$O(\log n)$	$O(\log n)$

Recursive Algorithm for Binary Search:- (Iterative)

```
int BinarySearch(A, n, Key)
{
    l=1, h=n
    while (l <= h)
    {
        mid = (l+h)/2;
        if (key == A[mid])
            return mid;
        if (key < A[mid])
            h=mid-1;
        else
            l=mid+1;
    }
    return 0;
}
```

Recursive algorithm:-

```
Algorithm RecursiveBinarySearch(l, h, Key)
{
    if (l == h)
    {
        if (A[l] == key)
            return l;
        else
            return 0;
    }
    else
    {
        mid = (l+h)/2;
        if (key == A[mid])
            return mid;
        if (key < A[mid])
            return RecursiveBinarySearch(l, mid-1, key);
        else
            return RecursiveBinarySearch(mid+1, h, key);
    }
}
```

Recurrence Relation:

$$T(n) = \begin{cases} 1 & \text{if } n=1 \\ T(\frac{n}{2}) + 1 & \text{if } n>1 \end{cases}$$

prove that the time complexity for average and worst case is $O(\log n)$ and unsuccessful search in three cases is also $O(\log n)$.

$$\text{Consider } T(n) = T(\frac{n}{2}) + 1 \rightarrow ①$$

Now substitute n is replaced by $\frac{n}{2}$ in eqn ①

then we get

$$T(n) = (T(\frac{n}{2^2}) + 1) + 1$$

$$= T(\frac{n}{2^2}) + 2 \rightarrow ②$$

$$= T(\frac{n}{2^3}) + 3 \rightarrow ③$$

$$= T(\frac{n}{2^4}) + 4 \rightarrow ④$$

Assume $\frac{n}{2^K} = 1$ then $n = 2^K$ then $K = \log n$

$$T(n) = T(\frac{n}{2^K}) + K$$

$$= T(1) + \log n$$

$$= 1 + \log n$$

Hence.

Time complexity is $O(\log n)$.

e.g. 11, 22, 30, 33, 40, 44, 55, 60, 66, 77, 80, 88, 99, $K=40, l=1$

l	m	h
11	22	30
1	2	3
30	33	40
3	4	5
40	44	55
5	6	7
55	60	66
7	8	9
60	66	77
8	9	10
66	77	80
9	10	11
77	80	88
10	11	12
80	88	99
11	12	13

① let us assume $l=1$ Key $= K=40$

$$\text{high } h=n=13$$

then find mid value $\text{mid}(m) = \left\lfloor \frac{l+h}{2} \right\rfloor$

if always low less than one equal to high then find mid

$$\text{mid}(m) = \frac{l+h}{2}$$

if $(\text{key} == \text{mid})$ then $40 == 55$ hence it is false

if (key < mid)

40 < 55 true

$$h = \text{mid} - 1;$$

$$= 7 - 1;$$

$$h = 6$$

$$\text{now } h = 6, l = 1, m = \frac{6+1}{2} = \frac{7}{2} = 3$$

if (key == mid) then

40 == 30 false

if (key < mid) then

40 < 30 false

if (key > mid) then

40 > 30 true

$$l = \text{mid} + 1$$

$$= 3 + 1$$

$$= 4$$

$$\text{now } l = 4, h = 6, m = \frac{4+6}{2} = \frac{10}{2} = 5$$

if (key == mid) then

40 == 40 true

the element is found.

② now key K = 62

$$l = 1, h = 13, k = 62$$

consider if ($l \leq h$)

$$\text{mid} = \frac{l+h}{2} = \frac{1+13}{2} = \frac{14}{2} = 7$$

if (key == mid) then

62 == 55 false

if (key < mid) then

62 < 55 false

if (key > mid)

62 > 55 true

$$l = 7 + 1 = 8$$

$$\text{now } l = 8, h = 13, m = \frac{8+13}{2} = \frac{21}{2} = 10$$

if ($l \leq h$)

$$\text{mid} = \left(\frac{8+13}{2} \right) = 10$$

if (key == mid)
62 == 77 false

if (key < mid)
62 < 77 true
high = mid - 1
= 10 - 1
h = 9

Finding Maximum and Minimum element

* Identify low index and High index.

* calculate Mid Value based on low and high indexes
that is mid $m = \lfloor \frac{l+h}{2} \rfloor$

* Based on the mid element the given list can be divided into two sub lists.

1) Left sub list

2) Right sub list

* consider left sub list, identify low, high indexes and calculate mid element.

* until the left sub list contain one (or) two elements in the list.

* Consider Right sub list, identify low, high indexes and calculate mid element. until you get the list contains (or) 2 elements.

* Compare left & right sub lists identify maximum element and minimum element.

Q1) Identify Maximum and minimum element from the following given list 50, 40, -5, -9, 45, 90, 65, 25, 75.

50	40	-5	-9	45	90	65	25	75
0	1	2	3	4	5	6	7	8

$$l=0 \quad h=8$$

$$mid = \left\lfloor \frac{0+8}{2} \right\rfloor = 4$$

50	40	-5	-9	45
0	1	2	3	4

left sub list
l=0, h=4

$$mid = \left\lfloor \frac{0+4}{2} \right\rfloor = 2$$

50	40	-5
0	1	2

$$l=0, h=2$$

$$mid = \left\lfloor \frac{0+2}{2} \right\rfloor = 1$$

50	40
0	1

$$\max = 50$$

$$\min = 40$$

$$\max = -5$$

$$\min = -5$$

$$\max = 50$$

$$\min = -5$$

$$\begin{matrix} -9 & 45 \end{matrix}$$

$$\begin{matrix} 3 & 4 \end{matrix}$$

$$\max = 45$$

$$\min = -9$$

90	65	25	75
5	6	7	8

Right sub list

$$l=5, h=8$$

$$mid = \left\lfloor \frac{5+8}{2} \right\rfloor = 6$$

90	65
5	6

$$\max = 90$$

$$\min = 65$$

25	75
7	8

$$\max = 75$$

$$\min = 25$$

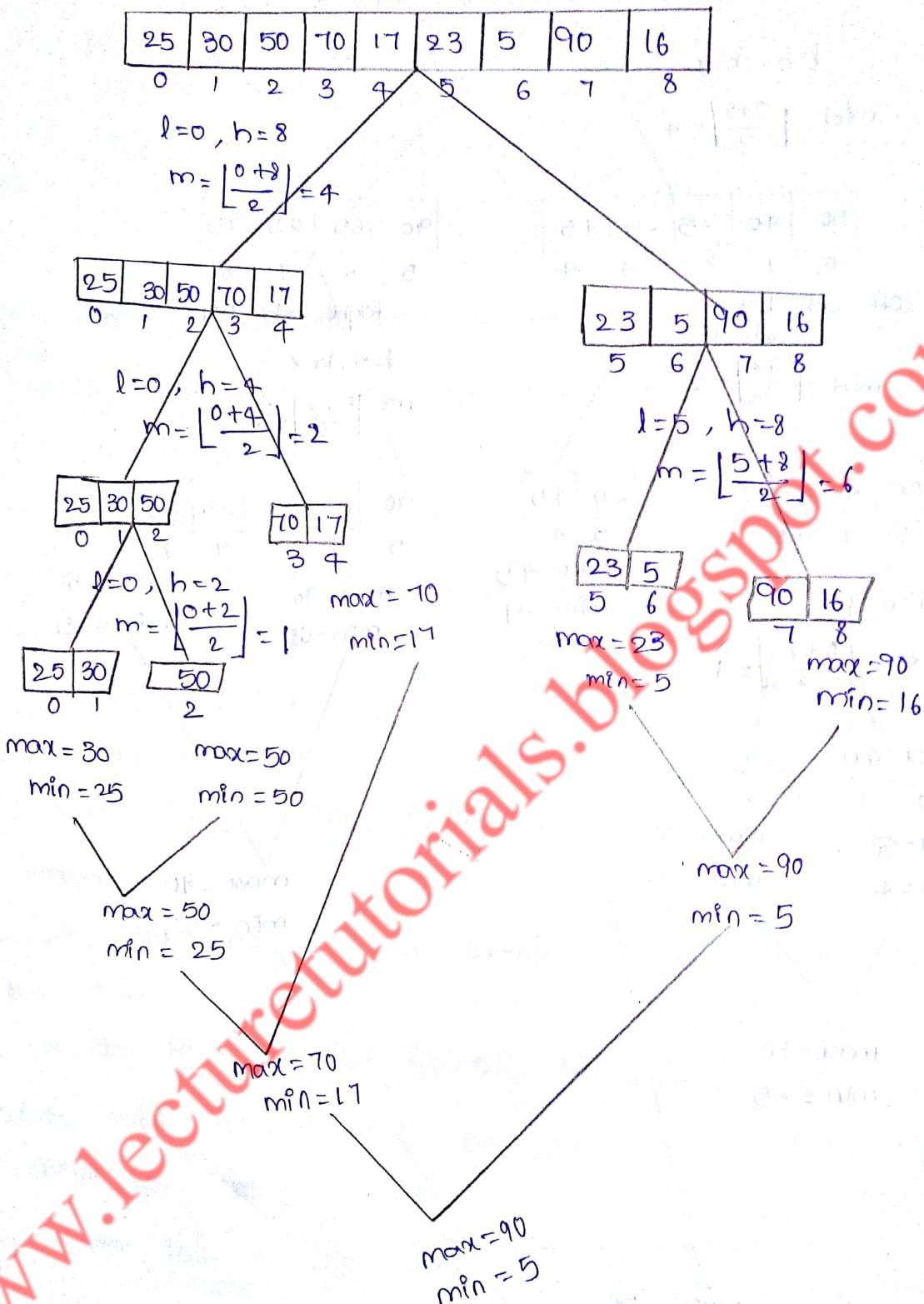
$$\max = 90$$

$$\min = 25$$

Hence the maximum number is 90

minimum number is -9

Ex:2 25, 30, 50, 70, 17, 23, 5, 90, 16.



Algorithm MaxMin(i, j, max, min)

{

if ($i = j$) then

max := min := $a[i]$; //list contain only one element.

else if ($i = j - 1$)

{

if ($a[i] < a[j]$)

{

max := $a[j]$;

min := $a[i]$;

}

else

{

max := $a[i]$; //list contains 2 elements.

min := $a[j]$;

}

}

else mid = $\lfloor \frac{i+j}{2} \rfloor$; max min(i, mid, max, min) //max min(mid+1, j, max1, min1)

if (max < max1) then //more than one element.

max := max1;

if (min > min1) then

min := min1;

}

}

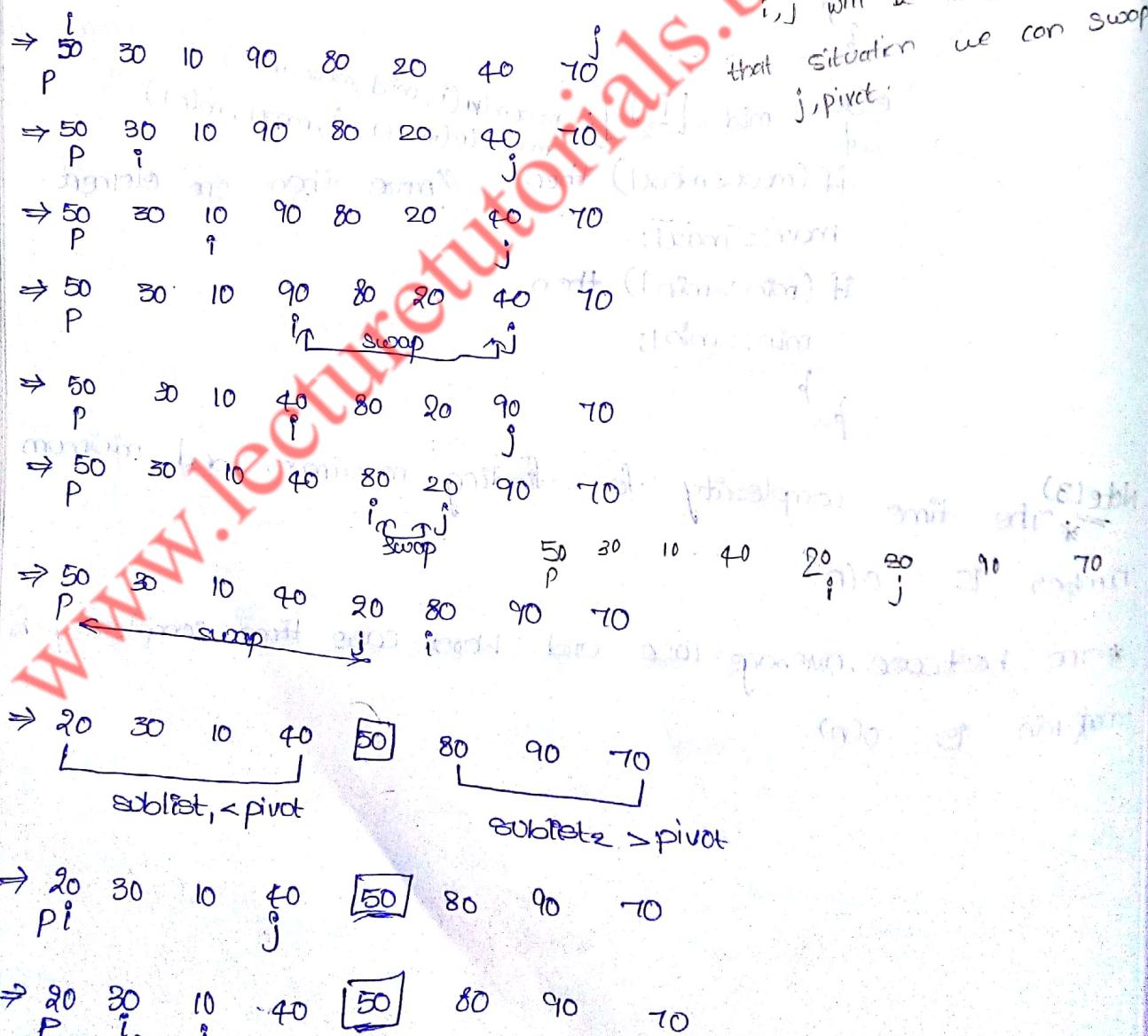
Note(3) * The time complexity for finding maximum and minimum number is $O(n)$.

* The best case, average case and worst case time complexity for max & min is $O(1)$.

Quick Sort

- * In quick sort method we are going to identify i (low), j (high) and pivot.
- * Always $\text{pivot} = a[i]$; a is array list of elements}
- * i has to be incremented when $a[i] \leq \text{pivot}$.
- * j has to be decremented when $a[j] \geq \text{pivot}$.
- * Otherwise no change to be made if $i & j$ has stop then swap $a[i]$ & $a[j]$.
- * Then again continue the process.
- * If $i & j$ crosses each other then swap $a[i]$, pivot.
- * Finally, we can generate the list is sorted order i.e. ascending order.

Eg: 50, 30, 10, 90, 80, 20, 40, 70.



$\Rightarrow 20 \ 10 \ 30 \ 40 \ 50 \ 80 \ 90 \ 70$
P i j

$\Rightarrow 20 \ 10 \ 30 \ 40 \ 50 \ 80 \ 90 \ 70$
P swap j

$\Rightarrow 10 \ 20 \ 30 \ 40 \ 50 \ 80 \ 90 \ 70$
P i j

$\Rightarrow 10 \ 20 \ 30 \ 40 \ 50 \ 80 \ 90 \ 70$
P i j swap

$\Rightarrow 10 \ 20 \ 30 \ 40 \ 50 \ 80 \ 70 \ 90$
P i j

$\Rightarrow 10 \ 20 \ 30 \ 40 \ 50 \ 80 \ 70 \ 90$
P i j swap

$\Rightarrow 10 \ 20 \ 30 \ 40 \ 50 \ 70 \ 80 \ 90$

ex: 62, 71, 72, 80, 82, 60, 52, 51, 42.

$\Rightarrow 62 \ 71 \ 72 \ 80 \ 82 \ 60 \ 52 \ 51 \ 42$
P i j

$\Rightarrow 62 \ 71 \ 72 \ 80 \ 82 \ 60 \ 52 \ 51 \ 42$
P i j swap

$\Rightarrow 62 \ 42 \ 71 \ 72 \ 80 \ 82 \ 60 \ 52 \ 51 \ 71$
P i j swap

$\Rightarrow 62 \ 42 \ 51 \ 80 \ 82 \ 60 \ 52 \ 72 \ 71$
P i j swap

$\Rightarrow 62 \ 42 \ 51 \ 80 \ 82 \ 60 \ 52 \ 72 \ 71$
P i j swap

$\Rightarrow 62 \ 42 \ 51 \ 52 \ 80 \ 60 \ 82 \ 72 \ 71$
P i j swap

$\Rightarrow 60 \ 42 \ 51 \ 52 \ 62 \ 82 \ 80 \ 72 \ 71$
P i j swap

$\Rightarrow 60 \ 42 \ 51 \ 52 \ 62 \ 82 \ 80 \ 72 \ 71$
P i j swap

$\Rightarrow 60 \ 42 \ 51 \ 52 \ 62 \ 82 \ 80 \ 72 \ 71$
P i j swap

$\Rightarrow 60 \ 42 \ 51 \ 52 \ 62 \ 82 \ 80 \ 72 \ 71$
P i j swap

$\Rightarrow 52 \ 42 \ 51 \ 60 \ 62 \ 82 \ 80 \ 72 \ 71$

$\Rightarrow 52 \ 42 \ 51 \ 60 \ 62 \ 82 \ 80 \ 72 \ 71$

$\Rightarrow 52 \quad 42 \quad 51 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

P $\xrightarrow{\text{swap}} i:j$

$\Rightarrow 51 \quad 42 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

P $\xrightarrow{i:j}$

$\Rightarrow 51 \quad 42 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

P $\xrightarrow{\text{swap}} i:j$

$\Rightarrow 42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{P:i:j}$

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

P

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{j:j}$

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{P:i:j}$

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{P:i:j}$

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{P:i:j}$

$42 \quad 51 \quad 52 \quad 60 \quad [62] \quad 82 \quad 80 \quad 72 \quad 71$

$\xrightarrow{P:i:j}$

Ex:- 65, 70, 75, 80, 85, 60, 55, 50, 45.

$65 \quad 70 \quad 75 \quad 80 \quad 85 \quad 60 \quad 55 \quad 50 \quad 45$

P

$a[i] \leq \text{pivot}$

$a[j] \geq \text{pivot}$

$65 \quad 70 \quad 75 \quad 80 \quad 85 \quad 60 \quad 55 \quad 50 \quad 45$

P $i \leftarrow \text{swap} j$

$65 \quad 45 \quad 75 \quad 80 \quad 85 \quad 60 \quad 55 \quad 50 \quad 70$

P

$65 \quad 45 \quad 75 \quad 80 \quad 85 \quad 60 \quad 55 \quad 50 \quad 70$

P $i \leftarrow \text{swap} j$

$65 \quad 45 \quad 75 \quad 80 \quad 85 \quad 60 \quad 55 \quad 75 \quad 70$

P

$65 \quad 45 \quad 50 \quad 80 \quad 85 \quad 60 \quad 55 \quad 75 \quad 70$

P $i \leftarrow \text{swap} j$

$65 \quad 45 \quad 50 \quad 55 \quad 85 \quad 60 \quad 80 \quad 75 \quad 70$

P $i \leftrightarrow j$

$65 \quad 45 \quad 50 \quad 55 \quad 60 \quad 85 \quad 80 \quad 75 \quad 70$

$65 \quad 45 \quad 50 \quad 55 \quad 60 \quad 85 \quad 80 \quad 75 \quad 70$

P

60 45 50 55, **65** 85 80 75 70

Sublist1 < pivot Sublist2 > pivot

$\frac{60}{P}$	45	50	55	$\boxed{65}$	85	80	75	70
P^60	45	50	55	$\boxed{65}$	85	80	75	70

60 45 50 55 [65] p 85 80 75 70
P i j i j

60 45 50 55 **65** 85 80 75 70
P ↙ scoop ↘ i P ↗ scoop ↘ i

$P^{55} \quad 45 \quad 50 \quad 60 \quad 65 \quad 70 \quad P_i \quad 80 \quad 75 \quad 85 \quad i$

p 55 45 50 60 65 70 80 75 85
~~(sway)~~ ji

50 45 55 60 65 70 80 75 85
P i j P swapji

45 50 55 60 65 80 70 75 85

45 50 55 60 65 80 70 75 85

45 50 55 60 65 75 70 80 85

45 50 55 60 65 75 70 80 85

45 50 55 60 65 70 75 80 85

P. 809 \rightarrow $(P_1 \wedge P_2 \wedge P_3) \vee (P_1 \wedge P_2 \wedge P_4) \vee (P_1 \wedge P_3 \wedge P_4) \vee (P_2 \wedge P_3 \wedge P_4)$

45 50 55 60 65 70 75 80 85

10/12/18
Quick Sort Algorithm:

```
quicksort(l,h)
{
    if (l < h)
    {
        j = partition(l,h);
        quicksort (l, i);
        quicksort (j+1, h);
    }
}

partition (l,h)
{
    pivot = A[l];
    i = l; j = h;
    while (i < j),
    do
    {
        i++;
    } while (A[i] ≤ pivot);
    do
    {
        j--;
    } while (A[j] ≥ pivot);
    if (i < j)
        swap (A[i], A[j]);
    swap (A[l], A[j]);
    return j;
}
```

* Time complexity for QuickSort

Note:-

Best Case Time complexity is $O(n \log n)$

Average Case Time complexity is $O(n \log n)$

Worst Case Time complexity is $O(n^2)$

* Time complexity for Best and Average Case.

To construct left sublist i.e. $T(\frac{n}{2})$

To construct Right sublist i.e. $T(\frac{n}{2})$

Time to build partition i.e. n

Hence total time $T(n) = T(\frac{n}{2}) + T(\frac{n}{2}) + n$

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

now the recurrence relation is

$$T(n) = \begin{cases} 1 & \text{if } n=1 \\ 2T\left(\frac{n}{2}\right) + n & \text{if } n>1 \end{cases}$$

$$\text{Consider } T(n) = 2T\left(\frac{n}{2}\right) + n \quad \text{--- (1)}$$

$$T(n) = 2 \left[2T\left(\frac{n}{4}\right) + \frac{n}{2} \right] + n$$

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

$$= 2^2 T\left(\frac{n}{2^2}\right) + 2n \quad \text{--- (2)}$$

$$= 2^3 T\left(\frac{n}{2^3}\right) + 3n \quad \text{--- (3)}$$

$$\left. \begin{array}{l} \text{since } T(n) = 2T\left(\frac{n}{2}\right) + n \\ T\left(\frac{n}{2}\right) = 2 \cdot T\left(\frac{n}{2^2}\right) + \frac{n}{2} \end{array} \right\}$$

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

$$= 2^K T\left(\frac{n}{2^K}\right) + Kn \quad \text{--- (4)}$$

Assume

$$\frac{n}{2^K} = 1$$

$$n = 2^K \Rightarrow K = \log_2 n$$

$$T(n) = n \cdot T(1) + \log_2 n \cdot n$$

$$T(n) = n \cdot T(1) + n \log_2 n$$

$$= n + n \log_2 n$$

$$= O(n \log_2 n)$$

Time complexity for worst case:

The recurrence relation $T(n) = \begin{cases} 1 & \text{if } n=0 \\ T(n-1)+n & \text{if } n>0 \end{cases}$

BS

max freq

$$T(n) = T(n-1) + n \quad \textcircled{1}$$

$$T(n) = T(n-2) + (n-1) + n \quad \textcircled{2}$$

$$T(n-3) + (n-2) + (n-1) + n \quad \textcircled{3}$$

$$T(n-4) + (n-3) + (n-2) + (n-1) + n \quad \textcircled{4}$$

let assume $n-K=0$

$$= T(n-K) + (n-(K-1)) + (n-(K-2)) + \dots + (n-1) + n$$

$$= T(n-K) + (n-K+1) + (n-K+2) + \dots + (n-1) + n$$

$$= T(K-n) + (n-n+1) + (n-n+2) + \dots + (n-1) + n$$

$$= T(0) + 1 + 2 + \dots + (n-1) + n$$

$$= T(0) + \frac{n(n+1)}{2}$$

$$= \frac{1+n(n+1)}{2}$$

$$= O(n^2)$$

Note:-

Select pivot as middle element

* Select Random element as pivot

MERGE SORT:-

In merge sort based on the middle element the list can be divided into left sublist and Right sublist.

Again consider left sublist findout mid value again it can be divide into left sublist₁ & right sublist₁, until we get 1 element in the list.

Then combine all the elements ^(apply merge sort)

In the same way right sublist also divide the list until we get 1 element in the other list.

Finally combined left & Right sort list.

20	50	30	10	40	60
----	----	----	----	----	----

Consider the list, Now find out middle element $m = \frac{1+6}{2}$

$$\frac{1+6}{2} = 3.5 = 3$$

The list 20, 50 & 30 $\boxed{20 \ 50 \ 30}$

$$\text{find out mid value } m = \left\lfloor \frac{1+3}{2} \right\rfloor = \frac{1+3}{2} = \frac{4}{2} = 2$$

Now the mid element is 2

Hence $\boxed{20 \ 50}$ $\boxed{30}$

$$\text{Now again find out mid element } m = \left\lfloor \frac{1+2}{2} \right\rfloor = \frac{1+2}{2} = 1$$

The final elements are $\boxed{20}$ $\boxed{50}$ $\boxed{30}$

Now combine first two elements, hence the list is

$\boxed{20 \ 50}$.

Now combine index 1, 2 and 3 $\boxed{20 \ 30 \ 50}$.

The list $\boxed{10 \ 40 \ 60}$

4

5

6

find out middle value $m = \left\lfloor \frac{1+6}{2} \right\rfloor = \left\lfloor \frac{4+6}{2} \right\rfloor = \frac{10}{2} = 5$

Now the mid element is 5

Hence $\boxed{10 \ 40}$ $\boxed{60}$

$$\text{Now again find out mid element } m = \left\lfloor \frac{1+5}{2} \right\rfloor = \left\lfloor \frac{4+5}{2} \right\rfloor = \frac{9}{2} = 4$$

The final elements are $\boxed{10}$ $\boxed{40}$ $\boxed{60}$

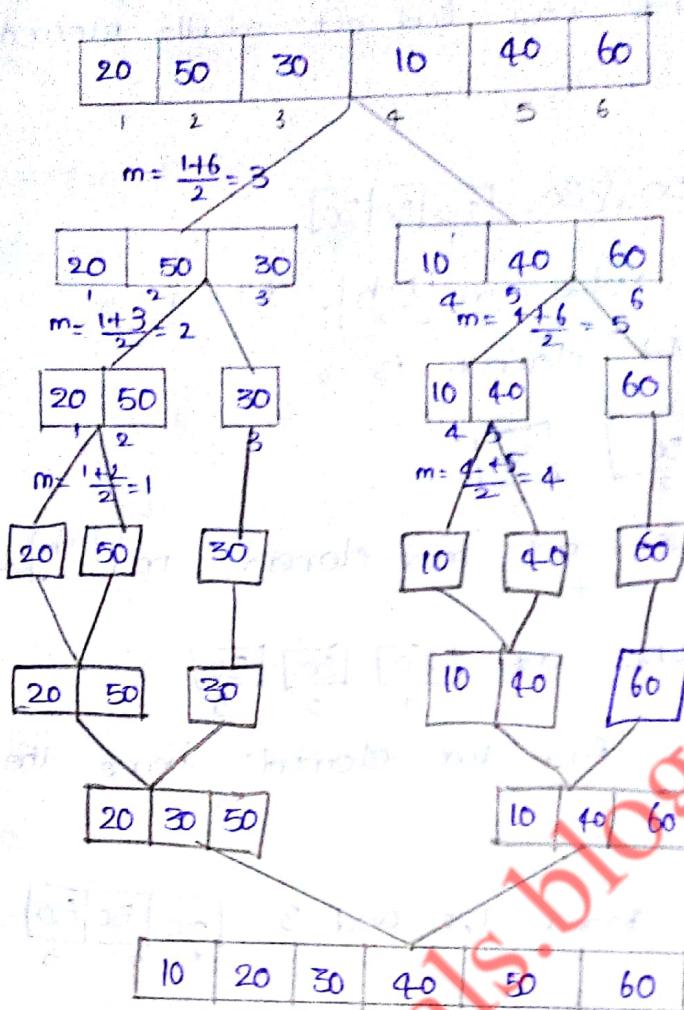
Now combine first two elements hence the list is $\boxed{10 \ 40}$

Now combine index 4, 5 and 6 $\boxed{10 \ 40 \ 60}$

Now combine the indexes [1, 3] and [4-6]

$\boxed{20 \ 30 \ 50}$ $\boxed{10 \ 40 \ 60}$

Hence the final sorted list is $\boxed{10 \ 20 \ 30 \ 40 \ 50 \ 60}$



Ex: 65, 70, 60, 75, 85, 80, 55, 50, 45, 57

Consider the list

65	70	60	75	85	80	55	50	45	57
1	2	3	4	5	6	7	8	9	10

Find out middle element $m = \frac{1+h}{2} = \frac{1+10}{2} = \frac{11}{2} = 5$

The list

65	70	60	75	85
1	2	3	4	5

Find out mid value $m = \frac{1+h}{2} = \frac{1+5}{2} = 3$

Now the mid. element is 3

Hence

65	70	60	75	85
1	2	3	4	5

Now again find out mid element $m = \frac{1+3}{2} = 2$.

65	70	60	75	85
1	2	3	4	5

Now again find out mid element $m = \frac{1+2}{2} = 1$

65	70	60	75	85
1	2	3	4	5

Now again find out mid value $m = \frac{t+5}{2} = \frac{9}{2} = 4$
the mid element is 4

65	70	60	75	80
1	2	3	4	5

Now combine first two elements hence the list is

65	70
1	2

Now combine indexes 1, 2 & 3 [60 | 65 | 70]

Now combine indexes 1, 2, 3 & 4 [60 | 65 | 70 | 75]

Now combine indexes 1, 2, 3, 4 & 5 [60 | 65 | 70 | 75 | 80]

The list

80	55	50	45	57
6	7	8	9	10

Find out middle value $m = \frac{1+t}{2} = \frac{6+10}{2} = \frac{16}{2} = 8$

Now the middle element is 8, hence [80 | 55 | 50] [45 | 57]

Now again find out mid element $m = \frac{6+8}{2} = \frac{14}{2} = 7$

Now the middle element is 7, hence [80 | 55] [50] [45 | 57]

again find out middle element $m = \frac{6+7}{2} = \frac{13}{2} = 6$

Now the mid value is 6, [80 | 55] [50] [45 | 57]

Again find out middle value $m = \frac{9+10}{2} = \frac{19}{2} = 9$

Now the mid value is 9, [80 | 55] [50] [45] [57]

Now combine first two elements hence the list is

80	55	80
6	7	

Now combine indexes 6, 7 & 8

50	55	80
6	7	8

Now combine indexes 6, 7, 8 & 9 [45 | 50 | 55 | 80]

Now combine indexes 6, 7, 8, 9 & 10 [45 | 50 | 55 | 57 | 80]

Now combine indexes [-5] & [6-10]

60	65	70	75	85
1	2	3	4	5

45	50	55	57	80
6	7	8	9	10

Hence the final sorted list is

45	50	55	57	60	65	70	75	80	85
1	2	3	4	5	6	7	8	9	10

65	70	60	75	85	80	55	50	45	57
1	2	3	4	5	6	7	8	9	10

$$m = \frac{1+10}{2} = 5$$

65	70	60	75	85
1	2	3	4	5

$$m = \frac{1+5}{2} = 3$$

80	55	50	45	57
6	7	8	9	10

$$m = \frac{6+10}{2} = 8$$

65	70	60
1	2	3

$$m = \frac{1+3}{2} = 2$$

75	85
4	5

$$m = \frac{4+5}{2} = 4$$

80	55	50
6	7	8

$$m = \frac{6+8}{2} = 7$$

45	57
9	10

$$m = \frac{9+10}{2} = 9$$

65	70
1	2

$$m = \frac{1+2}{2} = 1$$

60
3

75	85
4	5

$$m = \frac{4+5}{2} = 4$$

80	55	50
6	7	8

$$m = \frac{6+7}{2} = 6$$

45	57
9	10

$$m = \frac{9+10}{2} = 9$$

65	70
1	2

60
3

75	85
4	5

60	65	70	75	85
----	----	----	----	----

45	50	55	57	80
----	----	----	----	----

45	50	55	57	60	65	70	75	80	85
----	----	----	----	----	----	----	----	----	----

310	285	179	652	351	423	861	254	450	520
1	2	3	4	5	6	7	8	9	10

$$m = \frac{1+10}{2} = 5$$

310	285	179	652	351	423	861	254	450	520
1	2	3	4	5	6	7	8	9	10

$$m = \frac{1+5}{2} = 3$$

$$m = \frac{6+10}{2} = \frac{16}{2} = 8$$

310	285	179	652	351	423	861	254	450	520
1	2	3	4	5	6	7	8	9	10

$$m = \frac{1+3}{2} = 2$$

$$m = \frac{4+5}{2} = 4$$

$$m = \frac{6+8}{2} = \frac{14}{2} = 7$$

$$m = \frac{9+10}{2} = 9$$

310	285	179	652	351	423	861	254	450	520
1	2	3	4	5	6	7	8	9	10

$$m = \frac{1+2}{2} = 1$$

$$m = \frac{6+7}{2} = 6$$

$$m = \frac{9+10}{2} = 9$$

310	285	179	652	351	423	861	254	450	520
1	2	3	4	5	6	7	8	9	10

285 310

179 285 310

179	285	310	351	652
-----	-----	-----	-----	-----

254	423	861	254	423	861	254	423	450	520	861
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

179	254	285	310	351	423	450	520	652	861
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Algorithm's

Merge Sort algorithm

Algorithm MergeSort (low, high)

{
if (low < high) // if there are more than one element
{

mid = $\lfloor \frac{(low+high)}{2} \rfloor$; //divide p into sub problems

//find where to split the set.

MergeSort (low, mid); //solve the sub problems

MergeSort (mid+1, high); //combine the solutions

} Merge (low, mid, high);

}

Merge

Merging two sorted sub arrays using auxiliary storage

Algorithm Merge (low, mid, high)

{

h := low;

i := low;

j := mid+1;

while (h <= mid) & & (j <= high)

{

if (a[h] <= a[j]) then

{

b[i] = a[h];

h := h+1;

}

else

{

b[i] := a[j];

j := j+1;

}

i := i+1;

}

for $k := j$ to $high$ do

{

$b[i] := a[k];$

$i := i + 1;$

}

else

{

for $k := j$ to mid do

{

$b[i] := a[k];$

$i := i + 1;$

}

for $k := low$ to $high$ do

$a[k] := b[k];$

}

Time Complexity:

Note:- In merge sort the time complexity is to be

Best case $O(n \log n)$

Average Case $O(n \log n)$

Worst Case $O(n \log n)$

Recurrence Relation:-

$$T(n) = \begin{cases} 1 & \text{if } n=1 \\ 2T\left(\frac{n}{2}\right) + n & \text{if } n>1 \end{cases}$$

$$T(n) = 2T\left(\frac{n}{2}\right) + n \quad \text{--- (1)}$$

$$T(n) = 2^1 \left[2T\left(\frac{n}{2^2}\right) + \frac{n}{2} \right] + n$$

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

$$= 2^2 + \left(\frac{n}{2^3}\right) + 2n \quad \text{--- (2)}$$

$$= 2^3 + \left(\frac{n}{2^3}\right) + 3n \quad \text{--- (3)}$$

:

$$= 2^k + \left(\frac{n}{2^k}\right) + kn \quad \text{--- (4)}$$

Assume $\frac{n}{2} = 1 \Rightarrow n=2^k$
 $K = \log_2 n$

$$\begin{aligned}T(n) &= n \cdot T\left(\frac{n}{2}\right) + \log_2 n \\&= n \cdot T(1) + n \log_2 n \\&= n + n \log_2 n \\&= O(n \log n)\end{aligned}$$

Strassen's Matrix Multiplication:-

* In general matrix multiplication is the no. of columns in the 1st matrix = the no. of rows in the 2nd matrix.

* Let us consider $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ $B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$

* The resultant matrix $C = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$ where

$$c_{11} = a_{11} * b_{11} + a_{12} * b_{21}$$

$$c_{12} = a_{11} * b_{12} + a_{12} * b_{22}$$

$$c_{21} = a_{21} * b_{11} + a_{22} * b_{21}$$

$$c_{22} = a_{21} * b_{12} + a_{22} * b_{22}$$

* Here we have to compute 8 multiplications hence the time complexity of matrix multiplication is $O(n^3)$.

* In case of 4x4 matrix we are going to compute large no. of multiplications needed.

* Let us assume $A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$ and

$$B = \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix}$$

* Now the above matrix can be divided into 4 equal 2×2 matrices

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix}$$

* Now finally $C = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$

* The recurrence relation for general (matrix) multiplication is

$$T(n) = \begin{cases} 1 & \text{if } n \leq 2 \\ 8T\left(\frac{n}{2}\right) + n^2 & \text{if } n > 2 \end{cases}$$

* Now the strassen's reduce the time complexity is

$$T(n) = \begin{cases} 1 & \text{if } n \leq 2 \\ 7T\left(\frac{n}{2}\right) + n^2 & \text{if } n > 2 \end{cases}$$

* Instead of using 8 multiplications
multiplications use only 7 multiplications.

* Hence the time complexity is $\underline{\underline{O(n^{2.81})}}$

* Strassen's introduce 7 different formulas are

$$\begin{aligned} P &= (A_{11} + A_{22}) \cdot (B_{11} + B_{22}) \\ Q &= B_{11} (A_{21} + A_{22}) \\ R &= A_{11} (B_{12} - B_{22}) \\ S &= A_{22} (B_{21} - B_{11}) \\ T &= B_{22} (A_{11} + A_{12}) \\ U &= (A_{21} - A_{11}) \cdot (B_{11} + B_{12}) \\ V &= (B_{21} + B_{22}) \cdot (A_{12} - A_{22}) \end{aligned}$$

$$\begin{array}{c} (A_{11} + A_{22}) \cdot (B_{11} + B_{22}) \\ \downarrow \quad \downarrow \\ A_{11} \quad A_{22} \quad B_{11} \quad B_{22} \end{array}$$

$$\begin{array}{c} (A_{21} + A_{22}) \cdot (B_{11} + B_{22}) \\ \downarrow \quad \downarrow \\ A_{21} \quad A_{22} \quad B_{11} \quad B_{22} \end{array}$$

$$\begin{array}{c} (B_{11} + B_{22}) \cdot (A_{12} - A_{22}) \\ \downarrow \quad \downarrow \\ B_{11} \quad B_{22} \quad A_{12} \quad A_{22} \end{array}$$

$$C_{11} = P + S - T + V$$

$$C_{12} = -R + T$$

$$C_{21} = Q + S$$

$$C_{22} = P + R - Q + U$$

ROT
add

Ex:

$$A = \begin{pmatrix} 1 & 3 \\ 7 & 5 \end{pmatrix}$$

$$B = \begin{pmatrix} 6 & 7 \\ 3 & 8 \end{pmatrix}$$

find C

$$\text{Here consider } A_{11} = 1, A_{12} = 3, A_{21} = 7, A_{22} = 5$$

$$B_{11} = 6, B_{12} = 7, B_{21} = 3, B_{22} = 8$$

$$P = (A_{11} + A_{22}) \cdot (B_{11} + B_{22})$$

$$= (1 + 5) \cdot (6 + 8) = (6) \cdot (14) = 84$$

$$Q = B_{11} \cdot (A_{21} + A_{22})$$

$$= 6(7 + 5)$$

$$= 6(12)$$

$$= 72$$

$$R = A_{11} \cdot (B_{12} - B_{22})$$

$$= 1(7 - 8) = -1$$

$$S = A_{22} \cdot (B_{21} - B_{11})$$

$$= 5(3 - 6) = 5(-3) = -15$$

$$T = B_{22} \cdot (A_{11} + A_{12})$$

$$= 8(1 + 3) = 8(4) = 32$$

$$U = (A_{21} - A_{11}) \cdot (B_{11} + B_{12})$$

$$= (7 - 1)(6 + 7) = (6)(13) = 78$$

$$V = (B_{21} + B_{22}) \cdot (A_{12} - A_{22})$$

$$= (3 + 8)(3 - 5) = (11)(-2) = -22$$

$$C_{11} = P + S - T + V$$

$$= 84 + (-15) - 32 + (-22)$$

$$= 84 - 15 - 32 - 22$$

$$= 15$$

$$C_{12} = R + T = -1 + 32 = 31$$

$$C_{21} = Q + S = 72 + (-15) = 57$$

$$C_{22} = P + R - Q + U$$

$$= 84 + (-1) - 72 + 78$$

$$= 89$$

Hence the resultant matrix $C = \begin{pmatrix} 15 & 31 \\ 57 & 89 \end{pmatrix}$

e.g. $A = \begin{pmatrix} 4 & 5 \\ 6 & 9 \end{pmatrix}$ $B = \begin{pmatrix} 12 & 20 \\ 3 & 14 \end{pmatrix}$ Then find $AXB = C$.

Here consider $A_{11} = 4$ $A_{12} = 5$ $A_{21} = 6$ $A_{22} = 9$

$$B_{11} = 12 \quad B_{12} = 20 \quad B_{21} = 3 \quad B_{22} = 14$$

$$P = (A_{11} + A_{22}) \cdot (B_{11} + B_{22})$$

$$= (4 + 9)(12 + 14) = (13)(26) = 338$$

$$Q = B_{11} \cdot (A_{21} + A_{22}) = 12(6 + 9) = 12(15) = 180$$

$$R = A_{11} \cdot (B_{12} - B_{22}) = 4(20 - 14) = 4(6) = 24$$

$$S = A_{22} (B_{21} - B_{11}) = 9(3 - 12) = 9(-9) = -81$$

$$T = B_{22} (A_{11} + A_{12}) = 14(4 + 5) = 14(9) = 126$$

$$U = (A_{21} - A_{11})(B_{11} + B_{12}) = (6 - 4)(12 + 20) = (2)(32) = \frac{64}{32}$$

$$V = (B_{21} + B_{22})(A_{12} - A_{22}) = (3 + 14)(5 - 9) = (17)(-4) = -68$$

$$C_{11} = P + S - T + V$$

$$338 - 275 = 63$$

$$= 338 - 81 - 126 - 68 = 219 - 95 = 124$$

$$C_{12} = R + T = 24 + 126 = 150$$

$$C_{21} = Q + S = 180 + 81 = 261 \quad 99$$

$$C_{22} = P + R - Q + U = 338 + 24 - 180 + 64 = 394 - 180 = 214 \quad 246$$

Hence the resultant matrix $C = \begin{pmatrix} 63 & 150 \\ 214 & 246 \end{pmatrix}$

Algorithm:

Algorithm MatrixMul (A, B, n)

```

if (n ≤ 2)
  {
    C = 4 formulas
  }
else
  {
    mid ... n/2
    MM (A11, B11, n/2) + MM (A12, B21, n/2);
    MM (A11, B12, n/2) + MM (A12, B22, n/2);
    MM (A21, B11, n/2) + MM (A22, B21, n/2);
    MM (A21, B12, n/2) + MM (A22, B22, n/2);
  }
}
```

Time complexity:

$$O(D^{2.81})$$

$$(P + \epsilon)S = (P + \epsilon)(P + \delta)S = (P^2 + (P + \epsilon)\delta)S = P^2S + (P + \epsilon)(P + \delta)S$$

$$P^2S + P\epsilon S + P\delta S + \epsilon\delta S$$

$$P^2S + P\epsilon S + P\delta S + \epsilon\delta S = P^2S + P(S + \epsilon) + P(\delta - \epsilon)S + \epsilon\delta S$$

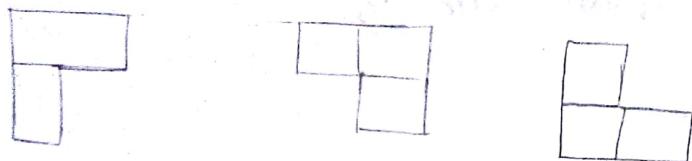
24/12/18.

Prepared by Saurav

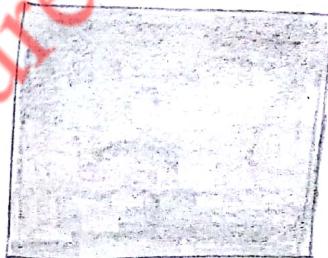
* Defective chess board Algorithm problem :-

→ Given chessboard of size $n \times n$ i.e. $n = 2^k$.

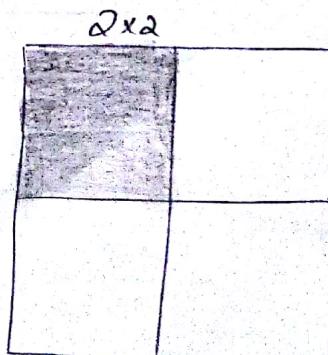
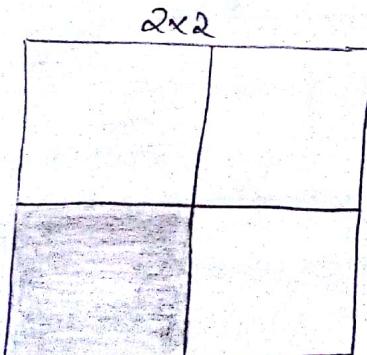
A chessboard contains exactly one defective square indicated by ^{(cor) add}Square, to fill chessboard with help of L-shaped tiles. These L-shaped tiles are called Triaminoes.

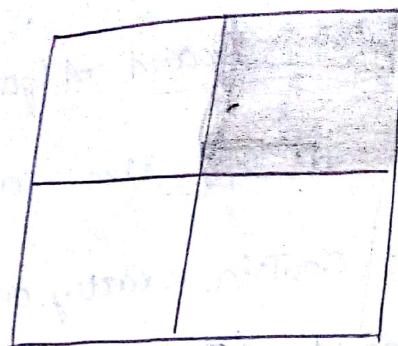
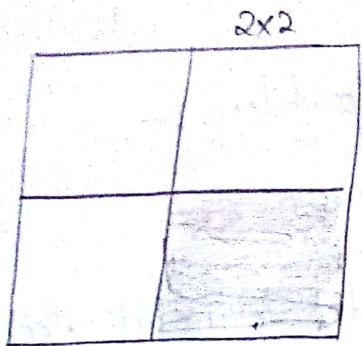


- In a chessboard total no. of squares = $n \times n$, where $n = 2^k \Rightarrow$ i.e. $2^k \times 2^k \Rightarrow 2^{2k}$.
- In a chessboard, no. of non-defective squares = $2^{2k} - 1$.
- This number is always divisible by 3.
- In 1x1 chessboard only one defective square.
- In 1x1 chessboard the no. of Non-defective chessboard is 0.



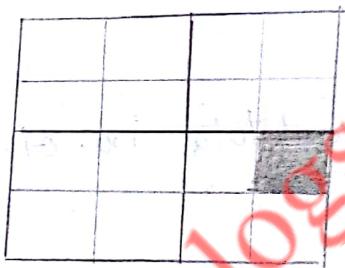
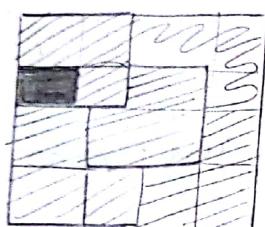
- In 2x2 chessboard, exactly one defective square & remaining three are comes under to Non-defective squares.





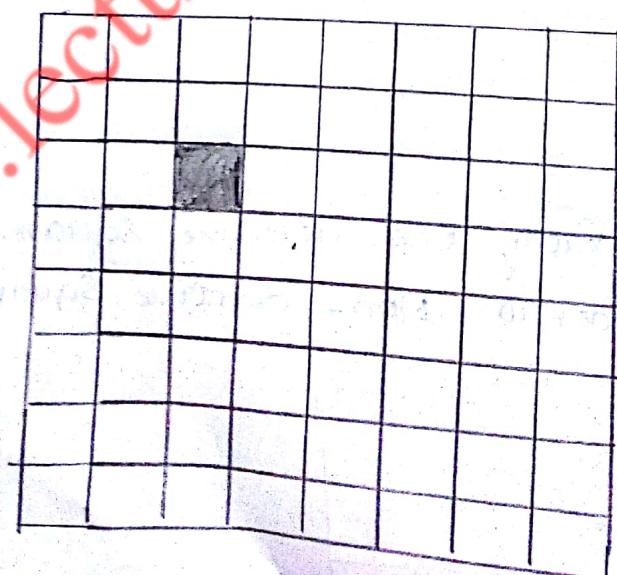
→ In 4×4 chessboard, exactly one defective square.

→ The no. of non-defective squares are 15.



→ size $k=1$, i.e 2×2 problem has a simple solution, if the problem is very large then it can be divided into smaller ones finally combine solutions to smaller problems.

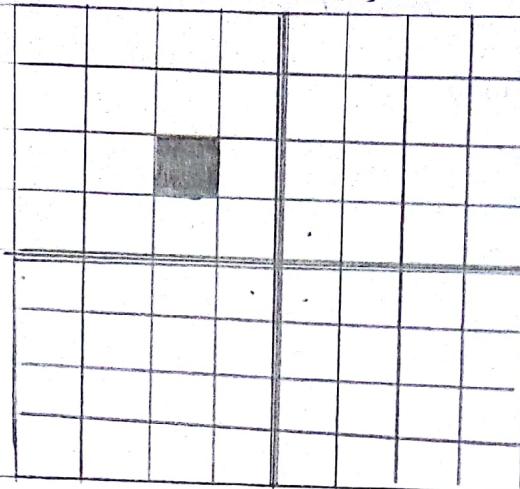
Ex:- Tiling 8×8 Defective chessboard.



8x8

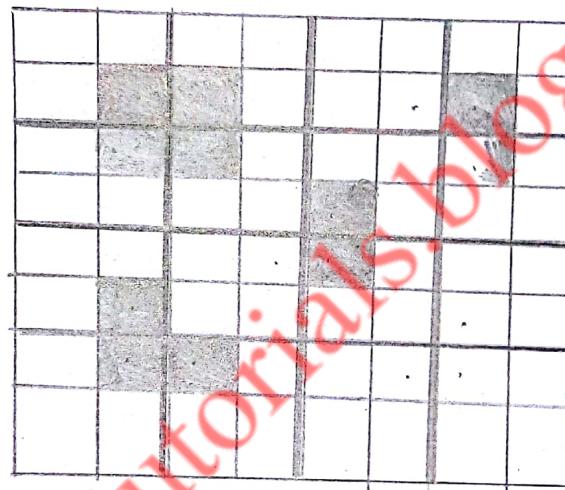
→ Now, the 8×8 chessboard can be divided into 4×4 chessboards.

→ Now, introduce artificial triaminos.



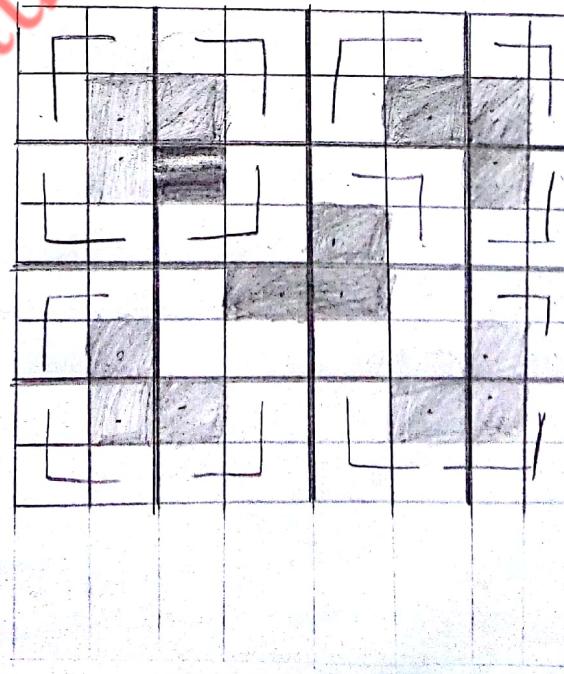
4×4

→ Now, again each 4×4 can be divided into possible no. of 2×2 chessboards.



→ Now add triaminos for each & every 2×2 chessboard.

→ Now combine all the solutions.



Note :- The Time Complexity of the defective chessboard
is $O(n^2)$ i.e., $\Theta(n^2)$

www.lecturetutorials.blogspot.com