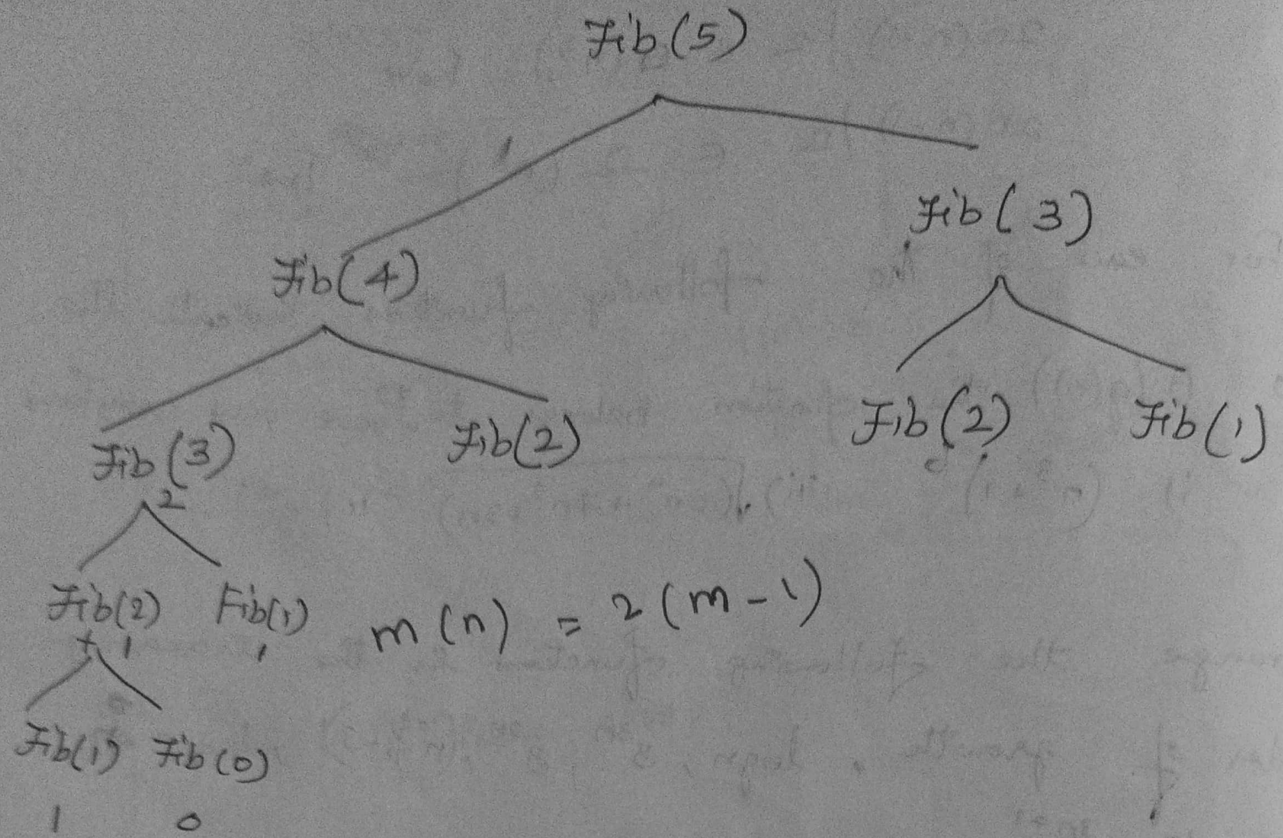


$m(n) = 2(m-1)$

MULTIPLICATION OF LARGE INTEGERS

$c_2 = a_1 * b_1$ - Product of their ~~first~~ first digit

$c_0 = a_0 * b_0$ - Product of their second digit

$c_1 = (a_1 + a_0) * (b_1 + b_0) - (c_2 + c_0)$ - Product of the sum of the a's digits and sum of the b's digits minus the sum of c_2 and c_0

$c = c_2 * 10^n + c_1 \times 10^{n/2} + c_0$

n - no. of digits in the given no,

Eg: 1) $\underset{a_1 a_0}{23} \times \underset{b_1 b_0}{14}$

$c_2 = 2 \times 1 = 2$

$c_0 = 3 \times 4 = 12$

$c_2 = 2 \times 1 = 2$

$c_0 = 3 \times 4 = 12$

$c_1 = (a_1 + a_0) * (b_1 + b_0) - (c_2 + c_0)$

$= (2+3) * (1+4) - (2+12)$

$= (5 * 5) - 14$

$c_1 = 11$

$c = 2 * 10^2 + 11 \times 10^1 + 12$

$= 200 + 110 + 12$

$c = 322$

ii) 9×78

09×78
$a_1 a_0 \quad b_1 b_0$

$c_2 = 0 \times 7 = 0$

$c_0 = 9 \times 8 = 72$

$c_1 = (0+9) * (7+8) - (c_0 + c_2)$

$= (9 * 15) - (72)$

$= 135 - 72$

$c_1 = 63$

$\boxed{n=2}$

$c = c_2 \times 10^n + c_1 \times 10^{n/2} + c_0$

$= 0 \times 10^2 + 63 \times 10^1 + 72$

$= 630 + 72$

$\boxed{c = 702}$

135
-72
63

630
+72
702

iii) $2345 * 678$

$a_0 = 45 \quad b_0 = 78$
$a_1 = 23 \quad b_1 = 06$

$45 * 78$

$c_2 = 4 \times 7 = 28$

$c_0 = 5 \times 8 = 40$

$c_1 = 9 * 15 - (68)$

$= 135 - 68$

$c_1 = 67$

23×06

$c_2 = 0 \times 2 = 0$

$c_0 = 3 \times 6 = 18$

$c_1 = 5 * 6 - (18)$

$= 30 - 18$

$c_1 = 12$

135
68
67

Scanned with OKEN Scanner

$C_1 = 28 \times 10^2 + 67 \times 10 + 40$

$= 28 \times 100 + 670 + 40$

$= 2800 + \cancel{670} 710$

$C_1 = 3510$

$C_0 = 0 \times 10^2 + 12 \times 10 + 18$

$C_0 = 120 + 18$

$C_0 = 138$

$C = (a*b) = (a_1 10^{n/2} + a_0) * (b_1 (10)^{n/2} + b_0)$

$= (a_1 * b_1) 10^{n/2} + (a_1 * b_0 + a_0 * b_1) 10^{n/2} + (a_0 * b_0)$

$(a_1 + a_0) * (b_1 + b_0) - (c_2 + c_0)$

$= c_2 10^n + c_1 10^{n/2} + c_0$

Eg: $2345 * 678$

$a_0 = 45 \quad b_0 = 78$

$a_1 = 23 \quad b_1 = 06$

$c_2 = a_1 * b_1 = 23 * 06 = 138$

$c_0 = a_0 * b_0 = 45 * 78 = 3510$

$c_1 = (a_0 + a_1) * (b_0 + b_1) - (c_2 + c_0)$

$= 68 * 84 - (138 + 3510)$

$c_1 = 2064$

$C = c_2 * 10^n + c_1 * 10^{n/2} + c_0$

$= c_2 * 10^4 + c_1 \times 10^2 + c_0$

$= 138 \times 10^4 + 2064 \times 10^2 + 3510$

$= 1380000 + 206400 + 3510$

$C = 1589910$

206400
3510
209910

1380000
209910
1589910

① 28
71
35

1 2 4
138 × ...
① 000
138
690
414
48438

Eg: 1234×4321

$a_1 = 12 \quad b_1 = 43$

$a_0 = 34 \quad b_0 = 21$

$c_2 = a_1 \times b_1 = 12 \times 43 = 516$

$c_0 = a_0 \times b_0 = 34 \times 21 = 714$

$c_1 = (a_0 + a_1) * (b_0 + b_1) - (c_2 + c_0)$

$= (46) * (64) - (1230)$

$= 2944 - 1230$

$c_1 = 1714$

$C = 516 \times 10^4 + 1714 \times 10^2 + 714$

$= 5160000 + 171400 + 714$

$C = 5332114$

43×12: 86, 43, 516

34×21: 34, 68, 714

$516 + 714 = 1230$

46×64: 184, 276, 2944

$2944 - 1230 = 1714$

$5160000 + 171400 + 714 = 5332114$

Basic Operation: Multiplication

$M(n) = 3M(n/2)$, Here in C multiplication takes place 3 times

Initial condition:

$M(1) = 1$

Divide & Conquer Approach

~~$n = n/2$~~

Time Complexity

$M(n) = 3(M(n/2))$

$M(n) = 3\left(3M\left(\frac{n/2}{2}\right)\right) = 3^2 M(n/2^2)$

$M(n) = 3(3M(n/4)) = 3^3 M(n/2^3)$

$M(n) = 3(3(3M(n/8)))$

$= 3^k M(n/2^k)$

$\cancel{= 3^k M(n/2^k)}$

$M(n) = 3M(n/2)$

By smoothness rule

$n = 2^k$

$= 3M\left(\frac{2^k}{2}\right)$

$M(2^k) = 3M(2^{k-1}) \rightarrow ①$ Backward substitution method

Sub $k = k-1$

$\cancel{M(2^k) = 3}$

$M(2^{k-1}) = 3M(2^{k-2})$

Sub $k = (k-2)$ $M(2^{k-2}) = 3M(2^{k-3})$

$\cancel{= 3[3 M(2^k}$

$M(2^k) = 3M(2^{k-1})$

$= \cancel{3M(2^{k-2})}$ $3(3M(2^{k-2}))$

$= 3^2 (M(2^{k-2}))$

$= 3^3 M(2^{k-3})$

$M(2^k) = 3^i M(2^{k-i})$

Sub $i = k$

$M(2^k) = 3^k M(2^{k-k})$

$= 3^k M(1)$

$M(2^k) = 3^k (1)$

$M(2^k) = \cancel{81}\ 3^k$ $\in O(3^k)$

$= 3^{\log_2 n}$

$M(2^k) = n^{\log_2 3}$ $\in O(n^{\log_2^3})$

$\underbrace{21}_{a_1}\underbrace{01}_{a_0} \times \underbrace{11}_{b_1}\underbrace{30}_{b_0}$

21×11 $\qquad\qquad$ 01×30

$2\times1 \quad 1\times1$ $\qquad\qquad$ $0\times3 \quad 1\times0$

$x_2 = 2$ $\qquad\qquad$ $y_2 = 0$

$x_0 = 1$ $\qquad\qquad$ $y_0 = 0$

$x_1 = (2+1)*(1+1) - (2+1)$ $\qquad$ $y_1 = (0+1)*(3+0) - (0+0)$

$= 3*2 - (3)$ $\qquad\qquad$ $= 1*3$

$= 6-3$ $\qquad\qquad$ $y_1 = 3$

$x_1 = 3$

$y = 0\times10^2 + 3\times10 + 0$

$X = 2\times10^2 + 3\times10 + 1$ $\qquad$ $y = 30$

$X = 231$

$\therefore \boxed{C_2 = 231}$ $\qquad\qquad$ $\therefore \boxed{C_0 = 30}$

~~$C_1 = 231\times10$~~

$C_1 = (a_1+a_0)*(b_1+b_0) - (c_2+c_0)$

$= (21+01)*(11+30) - (231+30)$

$= 22*41 - (261)$ $\qquad$ $= 902 - 261$

$2\times4 \quad 2\times1$ $\qquad\qquad$ $\therefore \boxed{C_1 = 641}$

$$\begin{array}{r} 902 \\ -\,261 \\ \hline 641 \end{array}$$

$X_2 = 8$

$X_0 = 2$

$X_1 = (2+2)*(4+1) - (8+2)$

$= 4*(5) - (10)$

$= 20-10$

$X_1 = 10$

$X = 8\times10^2 + 10\times10 + 2$

$C = 231\times10^4 + 641\times10^2 + 30$

$= 2310000 + 64100 + 30$

$\boxed{C = 2374130}$

$$\begin{array}{r} 2310000 \\ 64100 \\ \hline 2374100 \end{array}$$

Strassen's Matrix Multiplication:

$$\begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} * \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix}$$

$$= \begin{bmatrix} m_1+m_4-m_5+m_7 & m_3+m_5 \\ m_2+m_4 & m_1+m_3-m_2+m_6 \end{bmatrix}$$

$$m_1 = (a_{00}+a_{11}) \times (b_{00}+b_{11})$$

$$m_2 = (a_{10}+a_{11}) * b_{00}$$

$$m_3 = a_{00} \times (b_{01}-b_{11})$$

$$m_4 = a_{11} \times (b_{10}-b_{00})$$

$$m_5 = (a_{00}+a_{01}) \times b_{11}$$

$$m_6 = (a_{10}-a_{00}) \times (b_{00}+b_{01})$$

$$m_7 = (a_{01}-a_{11}) * (b_{10}+b_{11})$$

$$a = \begin{bmatrix} 1 & 2 \\ 0 & 3 \end{bmatrix} \qquad b = \begin{bmatrix} -2 & 1 \\ 1 & -3 \end{bmatrix}$$

$$m_1 = (1+3) * (-2-3) = 4 * (-5) = -20$$

$$m_2 = (0+3) \times -2 = 3 \times -2 = -6$$

$$m_3 = 1 \times (1+3) = \cancel{2}\ 4$$

$$m_4 = 3 \times (1+2) = 9$$

$$m_5 = (1+2) \times (-3) = 3 \times -3 = -9$$

$$m_6 = (0-1) \times (-2+1) = -1 \times -1 = 1$$

$$m_7 = (-1) \times (-2) = 2$$

$$\begin{bmatrix} -20+9+9+2 & 4-9 \\ -6+9 & -20+4+6+1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -5 \\ 3 & -9 \end{bmatrix}$$

$$a = \begin{array}{c} \begin{array}{cc} a_{00} & a_{01} \end{array} \\ \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right] \\ \begin{array}{cc} a_{10} & a_{11} \end{array} \end{array} \qquad b = \begin{array}{c} \begin{array}{cc} b_{00} & b_{01} \end{array} \\ \left[\begin{array}{cc|cc} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ \hline 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{array}\right] \\ \begin{array}{cc} b_{10} & b_{11} \end{array} \end{array}$$

$$m_1 = (a_{00} + a_{11}) \times (b_{00} + b_{11})$$

$$= \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \times \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$$

$$\cancel{x_1 = (2+2) \times (4+4)}$$

$$\cancel{\Rightarrow x_1 = 4 \times 8 = 32}$$

$$\cancel{x_2 = (2+2)}$$

$$m_1 = \begin{bmatrix} 16 & 16 \\ 16 & 16 \end{bmatrix}$$

$$m2 = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} * \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

$$= \begin{bmatrix} 8 & 8 \\ 8 & 8 \end{bmatrix}$$

$$m3 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$M_4 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$M_5 = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} * \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

$$M_5 = \begin{bmatrix} 8 & 8 \\ 8 & 8 \end{bmatrix}$$

$$M_6 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$M_7 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 \\ 8 & 8 & 8 & 8 \end{bmatrix}$$

Time Complexity: Basic Operation: Multiplication

$M(n) = 7\,M(n/2)$ $\qquad M(1) = 1$

Let $n = 2^k$

$$M(2^k) = 7\,M(2^k/2)$$
$$= 7\,M(2^{k-1})$$
$$= 7^2\,M(2^{k-2})$$
$$= 7^i\,M(2^{k-i})$$

$i = k$

$M(2^k) = 7^k M(2^{k-k})$

$= 7^k M(1) = 7^k$

$M(2^k) = 7^k$

$M(2^k) = 7^{\log_2 n}$

$= n^{\log_2 7}$

$= n^{2.8}$

$\in \cancel{n^3}$

$\cancel{\in O(n^3)}$

$n = 2^k$

$\log_2 n = \log_2 2^k$

$\log_2 n = k \log_2 2$

$\boxed{k = \log_2 n}$

$\in O(n^3)$

Algorithm BinarySearch (A[0...n-1], K, low, high)

if low ≥ high
    return -1

else

    mid = (low + high)/2

    if K = A[mid]

        return mid

    else if K > A[mid]

        return Binary search (A, K, mid+1, high)

    else

        return Binary search (A, K, low, mid-1)

Scanned with OKEN Scanner

1) 1315 x 1212

2)
$$\begin{array}{cc} a_{00} & a_{01} \end{array}$$
$$\left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right] \quad \left[\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ \hline 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \end{array}\right]$$
$$\begin{array}{cc} a_{10} & a_{11} \end{array} \qquad \begin{array}{cc} b_{00} & b_{01} \\ b_{10} & b_{11} \end{array}$$

Ans

$\underset{a_1\ a_0}{\underline{13}\,\underline{15}} \times \underset{b_1\ b_0}{\underline{12}\,\underline{12}}$

$\underset{a_1\ b_1}{13 \times 12}$ $\quad$ $\underset{a_1\ b_1}{15 \times 12}$

$1 \times 1 \quad 3 \times 2 \qquad 1 \times 1 \quad 5 \times 2$

$x_2 = a_1 * b_1 = 1$ $\qquad$ $y_2 = 1 \times 1 = 1$

$x_0 = 3 \times 2 = 6$ $\qquad$ $y_0 = 5 \times 2 = 10$

$x_1 = (1+3)*(1+2) - (1+6)$ $\qquad$ $y_1 = (1+5)*(1+2) - (1+10)$

$= 4 * 3 - 7$ $\qquad$ $= 6 * 3 - 11$

$= 12 - 7$ $\qquad$ $= 18 - 11$

$x_1 = 5$ $\qquad$ $y_1 = 7$

$x = 1 \times 10^2 + 5 \times 10 + 6$ $\qquad$ $y = 1 \times 10^2 + 7 \times 10 + 10$

$= 100 + 50 + 6$ $\qquad$ $y = 100 + 70 + 10$

$x = 156$ $\qquad$ $y = 180$

$C_2 = 156$ $\qquad$ $C_0 = 180$

$C_1 = (13+15) * (12+12) - (156+180)$

$= 18 * 24 - (336)$

$1 \times 2 \quad 8 \times 4$

$x_2 = 2 \quad x_0 = 32$

$x_1 = (1+8) * (2+4) - (2+32)$

$x_1 = 9 * 6 - 34 = 54 - 34 = 20$

$x = 2 \times 10^2 + 20 \times 10 + 32$

$= 200 + 200 + 32$

$x = 432$

$C_1 = 432 - 336 = 96$

3 1212
487
336
96

3
18 x 24
(1) 72
36
(1) 432
180
156
336

$= 156\times10^4 + 96\times10^2 + 180$

$= 1560000 + 9600 + 180$

$C = 1569780$

$$\begin{array}{r} 1560000 \\ 9600 \\ 180 \\ \hline 1569780 \end{array}$$

2) →

$m1 = \begin{bmatrix} 0 & 2 \\ 2 & 2 \end{bmatrix} \times \begin{bmatrix} 4 & 6 \\ 6 & 4 \end{bmatrix}$

$m1 = \begin{bmatrix} 20 & 20 \\ 20 & 20 \end{bmatrix}$

$M2 = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \times \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$

$m2 = \begin{bmatrix} 10 & 10 \\ 10 & 10 \end{bmatrix}$

$m3 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$m4 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$M5 = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \times \begin{bmatrix} 3 & 4 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 10 & 10 \\ 10 & 10 \end{bmatrix}$

$M6 = \begin{bmatrix} 0 & 0 \\ 6 & 0 \end{bmatrix}$

$M7 = \begin{bmatrix} 0 & 6 \\ 6 & 0 \end{bmatrix}$

$M = \begin{bmatrix} 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 20 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \end{bmatrix}$

Sequential or linear Search → Unordered array

Binary Search → Ordered Array

0	1	2	3	4	5
12	13	14	15	18	21

$K = 13$

⇒ low = 0

high = 5

$$mid = \frac{0+5}{2} = \lfloor 2.5 \rfloor = 2$$

$13 < 14$

⇒ ∴ low = 0

high = 1

$$mid = \frac{0+1}{2} = \frac{1}{2} = \lfloor 0.5 \rfloor = 0$$

⇒ $13 > 12$

∴ low = 1

high = 1

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

$13 = 13$

K

∴ Elements found at index 1

Time Complexity:

Basic Operation: Comparison

⇒ Best Case → Time Complexity

If the search element is present in middle position of the array $C_{Best}(n) = O(1)$

→ Worst Case ⇒ Time Complexity

$C_{worst}(n) = C_{worst}(n/2) + 1$ $\quad O(\log n)$

$= \cancel{C_{worst}(n/4) + 2}$

$= \cancel{C_{worst}(n/2^2) + 2}$

$= \cancel{C_{worst}(n/2^3) + 3}$

$= \cancel{C_{worst}(n/2^k)}$

$C(1) = 1$

Let $n = 2^k$

$= C_{worst}(2^{k-1}) + 1$

$= C(2^{k-2}) + 2$

$= C(2^{k-3}) + 3$

$\frac{2^k}{2} = 2^{(k-1)}$

$C_{worst}(2^k) = C(2^{k-i}) + i$

$\log n = \log 2^k$

$\log n = k \log_2 2$

$\log n = k$

$i = k$

$C_{worst}(2^k) = C(2^{k-k}) + k$

$= C(1) + k$

$= 1 + k$

$= 1 + \log n$

$= \log n \in O(\log n)$

Algorithm Mergesort $(A[0 \ldots n-1])$

if $n>1$

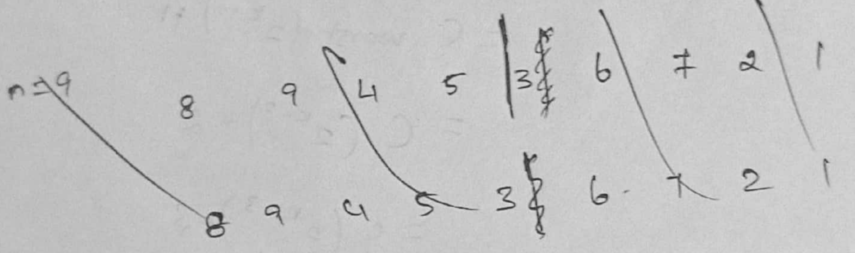
copy $A[0 \ldots \lfloor n/2 \rfloor - 1]$ to $B[0 \ldots \lfloor n/2 \rfloor - 1]$

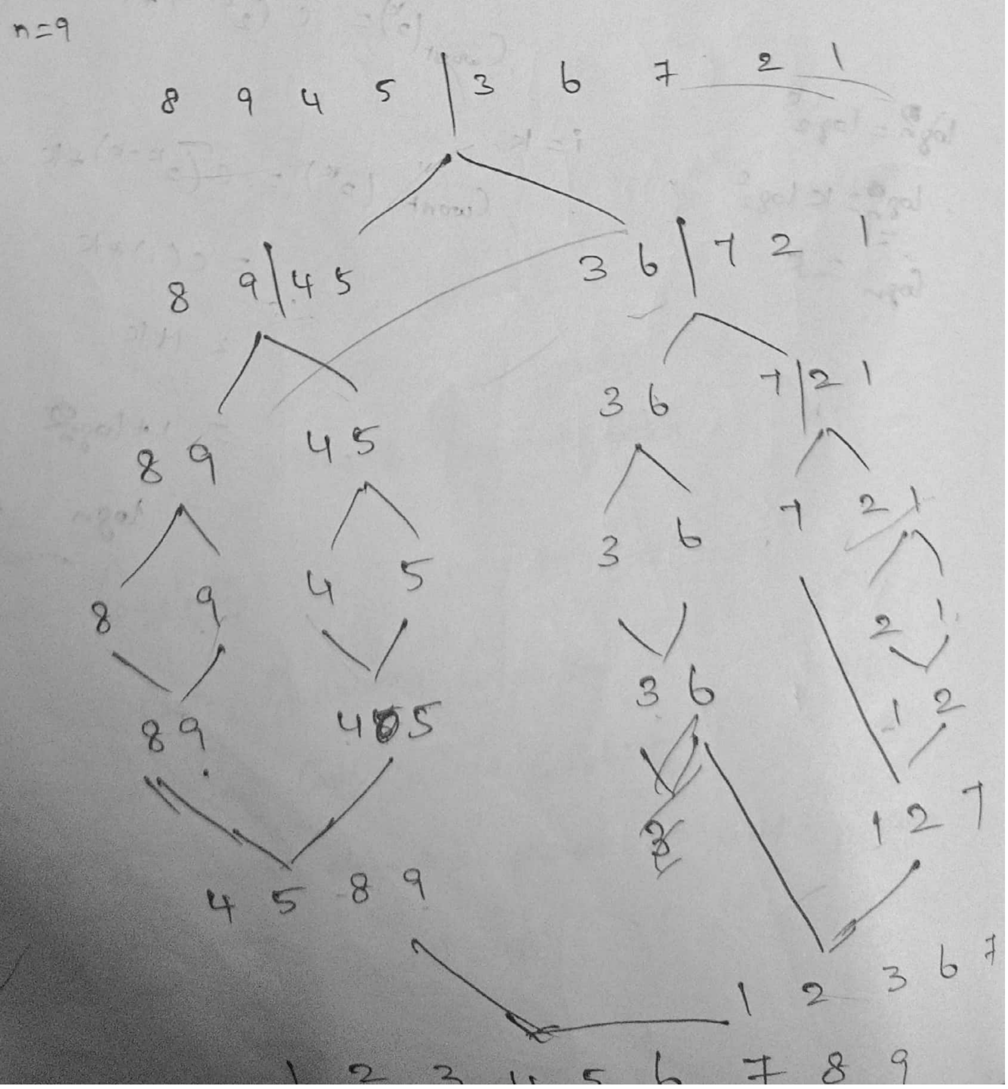
copy $A[\lfloor n/2 \rfloor \ldots n-1]$ to $C[0 \ldots \lceil n/2 \rceil - 1]$

Mergesort$(B[0 \ldots \lfloor n/2 \rfloor - 1])$

Mergesort$(C[0 \ldots \lfloor n/2 \rfloor - 1])$

Merge(B, C, A)





n=8

9 8 5 4 3 6 2 1 $\frac{n}{2}-1$

n>1

8>1 ✓

A[0...3] to B[0...3] 9 8 5 4

A[4...7] to C[0...3] 3 6 2 1

n=4 ✓ Mergesort (B[0...3])
4>1

B[0...3] → A[0...3]

A[0..1] to B[0...1] 9 8

A[2..3] to C[0...1] 5 4

n=2 Mergesort (B[0..1])
2>1 ✓

A[0] to B[0] 9

A[1] to C[0] 8

n=1 Merge (B,C,A) → 8 9
1>1 ✗

Merge sort (C[2..3])

A[0] to B[0] 5

A[1] to C[0] 4

Merge (B,C,A) → 4 5

Merge (B,C,A) → 4 5 8 9

Mergesort (C[0..3])

n=4
4>1 ✓

A[0..1] to B[0...1] 3 6

A[2..3] to C[0..1] 2 1

n=2 Mergesort (C[0..1])
2>1

A[0] to B[0] 3

A[1] to C[0] 6

Merge (B,C,A) → 3 6

Merge sort (C[2..3])

A[0] to B[0] 2

A[1] to C[0] 1

Merge (B,C,A) → 1 2

Merge (B,C,A) → 1 2 3 6

Merge (B,C,A) → 1 2 3 4 5 6 8 9

Basic Operation : Comparison , Initial condition : $C(1)=0$

Time Complexity :

$C(n) = C(n/2) + C(n/2) + n-1$ → no. of comparison in array

$= 2C(n/2) + n$

$= 2\left[2C(n/2^2) + n\right] + n$

$= 2\left[2\left(2C(n/2^3) + n\right) + n\right] + n$

$= 2^3 C(n/2^3) + 3n$

$= 2C$

$n = 2^k$

$C(2^k) = 2C(2^{k-1}) + 2^k - 1$

$= 2\left[2C(2^{k-2}) + 2^k - 1\right] + 2^k$

$= 2^2\left[C(2^{k-2})\right] + 2^{2k} - 2$

$= 2^2\left[C(2^{k-2})\right] + 2^{2k} - 2$

$= 2^2\left[2C(2^{k-3}) + 2^k - 1\right] + 2^{2k} - 2$

$= 2^3 C(2^{k-3}) + 2^{3k} - 3$

$= 2^i C(2^{k-i}) + 2^{ik} - i$

Sub $i = k$

$= 2^k C(2^{k-k}) + 2^{k^2} - k$

$= 2^k(0) + 2^{k^2} - k$

$= 2^{k^2} - k$

$n\log n - n + 1$

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

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

$\log n = k$

$C(2^k) = 2\,C(2^{k-1}) + 2^k - 1$

$= 2\left[2\,C(2^{k-2}) + 2^{k-1} - 1\right] + 2^k - 1$

$= 2^2 C(2^{k-2}) + 2^k - 2 + 2^k - 1$

$= 2^2 C(2^{k-2}) + 2.2^k - 3 = 2^2 C(2^{k-2}) + 2.2^k - 2^2 + 1$

$= 2^2\left[2C(2^{k-3}) + 2^{k-2} - 1\right] + 2.2^k - 2^2 + 1$

$= 2^3 C(2^{k-3}) + 2^k - 2^2 + 2.2^k - 2^2 + 1$

$= 2^3 C(2^{k-3}) + 3.2^k - 2^3 + 1$

General form

$= 2^i C(2^{k-i}) + i.2^k - 2^i + 1$

Sub $i = k$

$= 2^k C(2^{k-k}) + k.2^k - 2^k + 1$

$= 2^k C(1) + k.2^k - 2^k + 1$

$= 0 + k2^k - 2^k + 1$

$= 2^k(k-1) + 1$

$= 2^{\log_2^n}(\log_2^n - 1) + 1$

$= n^{\log_2^2}(\log_2^n - 1) + 1$

$= n^1 \log_2^n - n + 1$

$\in \Theta(n \log_2^n)$

Sort the given numbers using quick sort algorithm

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

1)

At first, 1st element is pivot

Pivot = 5

i → incr until

A[i] < A[p]

j → decr until

A[j] > A[p]

If i < j

swap A[i] & A[j]

If i > j

swap A[p] & A[j]

1.

5	3	1	9	8	2	4	7
↑	↑		↑			↑	↑
p	~~i~~		i			j	~~j~~

2.

5	3	1	4	8	2	9	7
			↑	↑	↑	↑	
			~~i~~	i	j	~~j~~	

3.

5	3	1	4	2	8	9	7
				↑↑	↑↑		
				j ~~i~~	i ~~j~~		

~~5 3 1 4 5~~

4.

2	3	1	4	|5|	8	9	7

5. 2 3 1 4 5 8 9 7

$\uparrow$ p, $\uparrow$ i, $\uparrow$ j; $\uparrow$ p, $\uparrow$ i, $\uparrow$ j

6. 2 3 1 4 5 8 9 7

$\uparrow$ p, $\uparrow$ i, $\uparrow$ j; $\uparrow$ p, $\uparrow$ i, $\uparrow$ j

7. 2 1 3 4 5 8 7 9

$\uparrow$ j, $\uparrow$ i; $\uparrow$ p, $\uparrow$ j, $\uparrow$ i

8. 1 | 2 | 3 4 5 7 | 8 | 9

Best Case :

$$C_{Best}(n) = C_{Best}(n/2) + C_{Best}(n/2) + n$$

$$C_{Best}(n) = 2C_{Best}(n/2) + n$$

$$C_{Best}(1) = 0$$

$n = 2^k$

$$C_{Best}(2^k) = 2C_{Best}(2^{k-1}) + 2^k = 2$$

$$= 2^2 C_{Best}(2^{k-2}) + 2 \cdot 2^k$$

$$= 2^3 C_{Best}(2^{k-3}) + 3 \cdot 2^k$$

$$= 2^i C_{Best}(2^{k-i}) + i \cdot 2^k$$

$$= 2^k C(0) + k \cdot 2^k$$

$$= k 2^k$$

$$= (\log_2 n)\, n = n \log_2 n$$

$$\in O(n \log_2 n)$$

Worst Case

$$C_{worst}(n) = (n+1) + n + (n-1) + \dots + 3$$

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

Avg Case

$$C_{avg}(n) = \frac{1}{n} \sum_{s=0}^{n-1} \left[(n+1) + C_{avg}(s) + C_{avg}(n-1-s) \right]$$

for n

$$C_{avg}(0) = 0 \quad C_{avg}(1) = 0$$

$$= \frac{1}{n} \sum_{s=0}^{n-1} (n+1) + \frac{1}{n} \sum_{s=0}^{n-1} 2\, C(s)$$

$$C(n) = \frac{n+1}{n} \sum_{s=0}^{n-1} 1 + \frac{2}{n} \sum_{s=0}^{n-1} 2\, C(s)$$

$$n\, C(n) = n(n+1) + 2 \sum_{s=0}^{n-1} C(s) \rightarrow 1$$

$$(n-1)\, C(n-1) = (n-1)(n-1+1) + 2 \sum_{s=0}^{n-2} C(s) \rightarrow 2$$

$$2 - 1 \Rightarrow (n-1)\, C(n-1) - n\, C(n) = (n-1)n + 2 \sum_{s=0}^{n-2} C(s) - n(n+1) - 2 \sum_{s=0}^{n-1} C(s)$$

$$n\, C(n) = (n-1)\, C(n-1) - n^2 + n + n^2 + n + 2 \left[\sum_{s=0}^{n-1} C(s) - \sum_{s=0}^{n-2} C(s) \right]$$

$$= (n+1)\, C(n-1) + 2n$$

$\div\ n(n+1)$

$$\frac{C(n)}{n+1} = \frac{C(n-1)}{n} + \frac{2}{n+1} \rightarrow 3$$

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

$n = n-2$ in 3

$$\frac{C(n-2)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n-1}$$

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

$$\frac{C(n)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n+1} + \frac{2}{n-1} + \frac{2}{n} \Rightarrow \frac{C(n)}{n+1} = \frac{C(n-i)}{n-(i-1)} + \frac{2}{n-(i-2)} + \frac{2}{n-(i-3)} \dots$$

$\Rightarrow i = n-1$

$$= 0 + \frac{2}{3} + \dots + \frac{2}{n} + \frac{2}{n+1} = 2 \ln(n+1)$$

$$C(n) = 1.38\, n \log_2 n$$

Closest Pair Problem Using Brute Force Approach:

Algorithm BruteForce Closest Pair (P)

~~d ← ∞~~ $d \leftarrow \infty$

~~for i ←~~ for $i \leftarrow 1$ to $n-1$ do

for $j \leftarrow i+1$ to n do

$d \leftarrow \min(d, \text{sqrt}((x_i - x_j)^2 + (y_i - y_j)^2))$

return d

A set of points $(2,3)$, $(12,30)$, $(40,50)$, $(5,1)$, $(12,10)$, $(3,4)$

1 2 3 4 5 6

$i=1, j=2$

$x_1, y_1 - (2,3)$

$x_2, y_2 - (12,30)$

$= \sqrt{(2-12)^2 + (3-30)^2}$

$= \sqrt{(-10)^2 + (-27)^2}$

$= \sqrt{10^2 + 27^2}$

$= \sqrt{829}$

$= 28.7923601$

$i=1, j=3$

$d \leftarrow 28.79$

$x_1, y_1 - (2,3)$

$x_2, y_2 - (40,50)$

$= \sqrt{(2-40)^2 + (3-50)^2}$

$= \sqrt{(-38)^2 + (-47)^2}$

$= \sqrt{3653}$

$= 60.44$

$i=1, j=4$

$d \leftarrow 28.79$

$x_1, y_1 - (2,3)$

$x_2, y_2 - (5,1)$

$= \sqrt{(2-5)^2 + (3-1)^2}$

$= \sqrt{(-3)^2 + 2^2} = \sqrt{9+4}$

$= \sqrt{13} = 3.61$

$d \leftarrow 3.61$

$i=1 \quad j=5$

$x_1, y_1 = (2,3)$ $\quad = \sqrt{(-10)^2 + (-7)^2}$

$x_2, y_2 = (12,10)$ $\quad = \sqrt{100+49}$

$= \sqrt{149}$

$d \leftarrow 3.61$ $\quad = 12.206$

$i=1 \quad j=6$

$x_1, y_1 = (2,3)$ $\quad = \sqrt{1^2+1^2}$

$x_2, y_2 = (3,4)$ $\quad = \sqrt{2} = 1.414$

$d \leftarrow 1.414$

$i=2 \quad j=3$

$x_1, y_1 = (12,30)$ $\quad = \sqrt{(28)^2 + (20)^2}$

$x_2, y_2 = (40,50)$ $\quad = \sqrt{1184}$

$= 34.4$

$d \leftarrow 1.414$

$i=2 \quad j=4$

$x_1, y_1 = (12,30)$ $\quad = \sqrt{7^2 + 29^2}$

$x_2, y_2 = (5,1)$ $\quad = \sqrt{49+841}$

$= \sqrt{890}$

$d \leftarrow 1.414$ $\quad = 29.83$

$i=2 \quad j=5$

$x_1, y_1 = (12,30)$ $\quad = \sqrt{20^2} = 20$

$x_2, y_2 = (12,10)$

$d \leftarrow 1.414$

$i=2 \quad j=6$

$x_1, y_1 = (12, 30)$

$x_2, y_2 = (3, 4) \qquad = \sqrt{9^2 + 26^2}$

$d \leq 1.414 \qquad = \sqrt{757} = 27.51$

$i=3 \quad j=4$

$x_1, y_1 = (40, 50) \qquad = \sqrt{35^2 + 49^2}$

$x_2, y_2 = (5, 1) \qquad = \sqrt{3626}$

$d \leq 1.414 \qquad = 60.22$

$i=3, \ j=5$

$x_1 \qquad = \sqrt{28^2 + 40^2}$

$= \sqrt{2384} \qquad 48.83$

$d \leq 1.414 \qquad = \cancel{354.18}$

$i=3 \quad j=6$

$= \sqrt{37^2 + 46^2}$

$= \sqrt{3485} = 59.04$

$i=4 \quad j=5$

$= \sqrt{7^2 + 9^2} = \sqrt{49+81}$

$= \sqrt{130} = 11.40$

$i=4 \quad j=6$

$= \sqrt{2^2 + 3^2} = \sqrt{4+9} = \sqrt{13}$

$= 3.61$

$i=5 \quad j=6$

$$= \sqrt{9^2+6^2}$$

$$= \sqrt{81+36}$$

$$= \sqrt{117} = 10.82$$

$d \in 1.414$

Time Complexity:

Basic Operation: Comparison

$$C(n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 1$$

$$= \sum_{i=1}^{n-1} (n-i-\cancel{1}+\cancel{1})$$

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

$$= (n-1)+(n-2)+(n-3)+\dots$$

$$\dots + (\cancel{n}-\cancel{n}+1)$$

$\cancel{n} \; \cancel{1}+2$

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

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

$$\in \Theta(n^2)$$

Quick Sort

Average Case

Recurrence Relation: $C_{avg}(n) = \frac{1}{n} \sum_{s=0}^{n-1} \left[(n+1) + C_{avg}(s) + C_{avg}(n-1-s) \right]$ for $n>1$

Initial Cond $C_{avg}(0)$; $C_{avg}(1) = 0$

Analysis:

Since $C(s) = C(n-1-s)$

$$C(n) = \frac{1}{n} \sum_{s=0}^{n-1} \left[(n+1) + 2C(s) \right]$$

$$= \frac{1}{n} \sum_{s=0}^{n-1} (n+1) + \frac{1}{n} \sum_{s=0}^{n-1} 2C(s)$$

$$= \frac{n+1}{n} \sum_{s=0}^{n-1} + \frac{2}{n} \sum_{s=0}^{n-1} C(s)$$

$$C(n) = (n+1) + \frac{2}{n} \sum_{s=0}^{n-1} C(s)$$

$$nC(n) = n(n+1) + 2 \sum_{s=0}^{n-1} C(s) \rightarrow (1)$$

Sub $n = n-1$ in (1)

$$(n-1)C(n-1) = (n-1)(n-1+1) + 2 \sum_{s=0}^{n-2} C(s) \rightarrow (2)$$

(2) (1)

$$(n-1)C(n-1) - nC(n) = (n-1)n + 2\sum_{s=0}^{n-2} C(s) - n(n+1) - 2\sum_{s=0}^{n-1} C(s)$$

$$nC(n) = (n-1)C(n-1) - n^2 + n + n^2 + n + 2\left[\sum_{s=0}^{n-1} C(s) - \sum_{s=0}^{n-2} C(s)\right]$$

$$nC(n) = (n-1)C(n-1) + 2n + 2\left[C(n-1)\right]$$

$$nC(n) = (n+1)C(n-1) + 2n$$

$\div$ by $n(n+1)$

$$\frac{C(n)}{n+1} = \frac{C(n-1)}{n} + \frac{2}{n+1} \rightarrow (3)$$

Sub $n = n-1$ in (3)

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

Sub $n = n-2$ in (3)

$$\frac{C(n-2)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n-1}$$

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

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

$$\frac{C(n)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n+1} + \frac{2}{n-1} + \frac{2}{n}$$

General

$$\frac{C(n)}{n+1} = \frac{C(n-i)}{n-(i-1)} + \frac{2}{n-(i-2)} + \frac{2}{n-(i-3)} + \frac{2}{n-(i-4)}$$

Put $i = n-1$

$$\frac{C(n)}{n+1} = \frac{C(1)}{n-(n-1-\cancel{1})} + \frac{2}{n-(n-1-2)} + \dots + \frac{2}{n} + \frac{2}{n+1}$$

$$= 0 + \frac{2}{3} + \dots + \frac{2}{n} + \frac{2}{n+1}$$

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

$$= 2\sum_{i=3}^{n+1} 1/i$$

$$= 2\left[\sum_{i=3}^{n+1} 1/i - 1/1 - 1/2\right]$$

$$= 2\left[\sum_{i=3}^{n+1} 1/i - 3/2\right]$$

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

Sub $n = n-1$ in (3)

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

Sub $n = n-2$ in (3)

$$\frac{C(n-2)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n-1}$$

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

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

$$\frac{C(n)}{n+1} = \frac{C(n-3)}{n-2} + \frac{2}{n+1} + \frac{2}{n-1} + \frac{2}{n}$$

General

$$\frac{C(n)}{n+1} = \frac{C(n-i)}{n-(i-1)} + \frac{2}{n-(i-2)} + \frac{2}{n-(i-3)} + \frac{2}{n-(i-4)}$$

Put $i = n-1$

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

$$= 0 + \frac{2}{3} + \dots + \frac{2}{n} + \frac{2}{n+1}$$

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

$$= 2\sum_{i=3}^{n+1} 1/i$$

$$= 2\left[\sum_{i=3}^{n+1} 1/i - 1/1 - 1/2\right]$$

$$= 2\left[\sum_{i=3}^{n+1} 1/i - 3/2\right]$$

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

$c(n) \approx (n+1)\, 2\ln(n+1)$

$\approx 2n\ln(n)$

$C(n) = 1.38 n \log_2^n$

FAKE COIN PROBLEM

Basic Operation : Weighing the coins

Time Complexity

Initial cond

$W(1) = 0$

$W(n) = W(n/2) + 1$

Sub $n = 2^k$

$= W(2^{k-1}) + 1$

$= W(2^{k-2}) + 2$

$= W(2^{k-3}) + 3$

$= W(2^{k-i}) + i$

Sub $i = k$

$= W(2^0) + k$

$= W(1) + k$

$= 0 + k = k = \log n$

$\in \Theta(\log n)$

Algorithm Insertion sort $(A[0 \dots n-1])$

for $i \leftarrow 1$ to $n-1$ do

$v \leftarrow A[i]$

$j \leftarrow i-1$

while $j \geq 0$ and $A[j] > v$ do

$A[j+1] \leftarrow A[j]$

$j \leftarrow j-1$

$A[j+1] \leftarrow v$

	87 |	49	23	64	17	21
$v=49$	49	87 |	23	64	17	21
$v=23$	23	49	87 |	64	17	21
$v=64$	23	49	64	87 |	17	21
$v=17$	17	23	49	64	87 |	21
$v=21$	17	21	23	49	64	87

79 | 44 68 90 29 34 37

v=44 44 79 | 68 90 29 34 37

v=68 44 68 79 | 90 29 34 37

v=90 44 68 79 90 | 29 34 37

v=29 29 44 68 79 90 | 34 37

v=34 29 34 44 68 79 90 | 37

v=37 29 34 37 44 68 79 90

1 | 2 3 4 5 6 7

v=2 1 2 | 3 4 5 6 7

v=3 1 2 3 | 4 5 6 7

v=4 1 2 3 4 | 5 6 7

v=5 1 2 3 4 5 | 6 7

v=6 1 2 3 4 5 6 | 7

v=7 1 2 3 4 5 6 7

9 8 4 7

	7 |	6	5	4	3	2	1
v=6	6	7 |	5	4	3	2	1
v=5	5	6	7 |	4	3	2	1
v=4	4	5	6	7 |	3	2	1
v=3	3	4	5	6	7 |	2	1
v=2	2	3	4	5	6	7 |	1
v=1	1	2	3	4	5	6	7

Basic Operation: Comparison

Worst Case Time Complexity:

If the elements are present in the array in the decreasing order. For every i, the loop j iterates fully.

$$= \sum_{i=1}^{n-1} \sum_{j=0}^{i-1} 1$$

$$= \sum_{i=1}^{n-1} i-\cancel{1}-0+\cancel{1}$$

$$= \sum_{i=1}^{n-1} i = 1+2+\dots+(n-1)$$

$$= \frac{(n-1)n}{2} = \frac{n^2-n}{2} \in O(n^2)$$

Best case Time Complexity :

If the elements present in the array are in increasing order, for every i value, j loop iterates only once

$$= \left(\sum_{i=1}^{n-1} 1 \right) \times 1$$

$$= n-1-\cancel{1}+\cancel{1}$$

$$= n-1 \in O(n)$$

Average Case Time Complexity :

For item pos

Position Probability

$$= \frac{1}{i+1} \sum_{j=0}^{i-1} (j+1)$$

$$= \frac{1}{i+1} (1+2+\dots+i)$$

$$= \frac{1}{\cancel{i+1}} \frac{i\cancel{(i+1)}}{2}$$

$$= i/2$$

For n numbers

$$= \sum_{i=1}^{n-1} i/2$$

$$= 1/2 \left(1+2+\dots+(n-1)\right)$$

$$= \frac{1}{2}\left(\frac{(n-1)n}{2}\right) = \frac{1}{4}n^2 - \frac{1}{4}n$$

$$\in O(n^2/4)$$

$P =$

4^+

5

→ For random order array insertion sort makes on average half as many comparisons of as on decreasing arrays.

→ The task is to determine how many key comparisons are done average, to insert one new element into the array.

→ Assume key are distinct element at in position i, it has, i+1 possible location to be inserted.

Algorithm Design Technique
ADT

TRANSFORM AND CONQUER:

→ In the transformation stage, the problem's instance is modified to be, for one reason or another, more amenable to a solution.

→ ~~First~~ In the conquering stage it is solved.

→ There are 3 variations of this idea

i) Transformation to a simple or more convenient instance of the same problem, we call it instant simplification. - Eg. ~~Pre~~ Pre sorting

ii) Transformation to a different representation of the same instance, we call it representation change. Eg: AVL Tree

iii) Transformation to an instance of a different problem for which an algorithm is already available we call it problem reduction.

INSTANT SIMPLIFICATION

Brute force

Presorting — Uniqueness element in array — $O(n^2)$

Presorting — Mode value — $O(n^2)$

Presorting — Searching — $O(n)$, $O(\log n)$

i) For a sorted array, only one loop needed for unique element

12 14 18 19 21 26

$i=0$ $A[0] == A[1] \Rightarrow 12 == 14 \leftarrow$ No

$i=1$ $A[1] == A[2] \Rightarrow 14 == 14 \leftarrow$ Yes

Algorithm PresortElementUniqueness ($A[0 \ldots n-1]$):

for $i \leftarrow 0$ to $n-2$ do

  if $A[i] = A[i+1]$

    return false

return true

Time Complexity:

$$T(n) = T_{sort} + T_{unique}$$

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

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

Sort - Mergesort $n\log n$

Algorithm PresortMode($A[0 \ldots n-1]$)

Sort the array A

$i \leftarrow 0$

modefrequency $\leftarrow 0$

while $i \le n-1$ do

  runlength $\leftarrow 1$; runvalue $\leftarrow A[i]$

  while i+runlength $\le n-1$ and $A[i$+runlength$]$ = runvalue if runlength > modefrequency

modefrequency ← runlength; modevalue ← run~~length~~ value

i ← i + runlength

return modevalue

Mode Computation:

Brute force Approach

5 3 2 1 ~~1~~ 5 7 8 25 18

5	3	2	1	7	8
~~1~~	1	1	1	1	~~1~~
~~2~~					2
3					

Worstcase Time

→ If all elements are unique, then an element in i^{th} position makes $i-1$ comparison

Basic ~~Comparison~~ Operation : Comparison

$$C(n) = \sum_{i=1}^{n} (i-1)$$

$$= 1 + 2 + \dots + n-1$$

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

1 2 3 5 5 5 7 8 8

$i = 0$

runvalue = 1 modefrequency ← 1

runlength = 1 ; $i = i + \text{runlength}$

$= 0 + 1 = 1$

$i = 1$ runvalue = 2

runlength = 1 modefrequency ← 1

$i = 1 + 1 = 2$

$i = 2$ runvalue = 3 modefrequency → 1

runlength = 1 $i = i + 1 = 3$

$i = 3$ runvalue = 5 $i = i + 3 = 3 + 3 = 6$

runlength = $\cancel{1}$ $\cancel{2}$ 3 modefrequency → 3

modevalue → 5

$i = 6$ runvalue = 7 $i = i + 1 = 7$

runlength = 1

$i = 7$ runvalue = 8 $i = i + 2 = 9$

runlength = $\cancel{1}$ 2

$9 \leq n - 1$ X

∴ Modevalue = 5

Time Complexity

$C(n) = C_{sort} + C_{mode}$

$= n\log n + \sum_{i=0}^{n-1} 1$

$= n\log n + n - \cancel{1} + \cancel{1}$

$\approx n\log n$

$= O(n\log n)$

For searching normal method is better compared to transform & Conquer method

COMPUTING AN MEDIAN IN SOLUTION PROBLEM:

$n=9$ $\quad K = \lceil 9/2 \rceil = 5$, $i > j$ - Swap.

4	1	10	9	7	12	8	2	15
p	i	i					j	j

4	1	2	9	7	12	8	10	15
p		i j	i j	j	j	j	j	

2 1 4 (9 7 12 8 10 15)

S

S - first element in which all left side elements are small

$k=5$

$K < S$ ⇐ left half of array

$K > S$ ⇐ right " " "

$K == S$

$S > 3$

9	7	12	8	10	15
p	i	i	j	j	j

9	7	8	12	10	15
p		i	j		

9 7 8 12 10 15

i j i

4 5 6 7 8 9

8 7 9 12 10 15

S

~~6>5~~

5<6

8 7

p j i

4 5

7 8

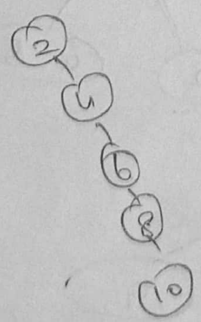
S

k = 5

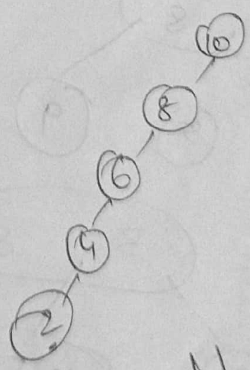
k^{th} smallest element is 8

<u>BST</u>

1) 2 4 6 8 10



2) 10 8 6 4 2

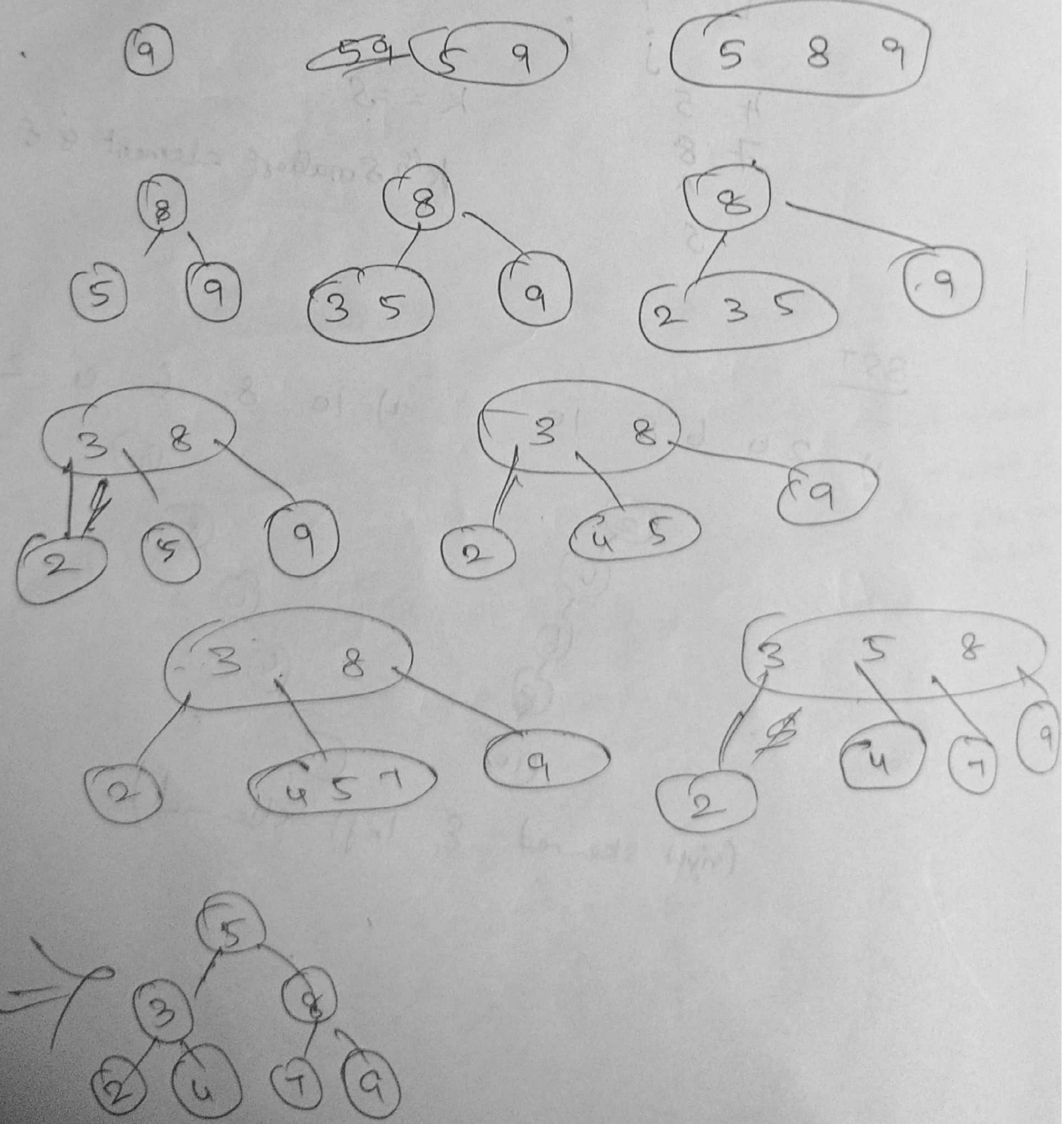


(right skewed & left skewed)

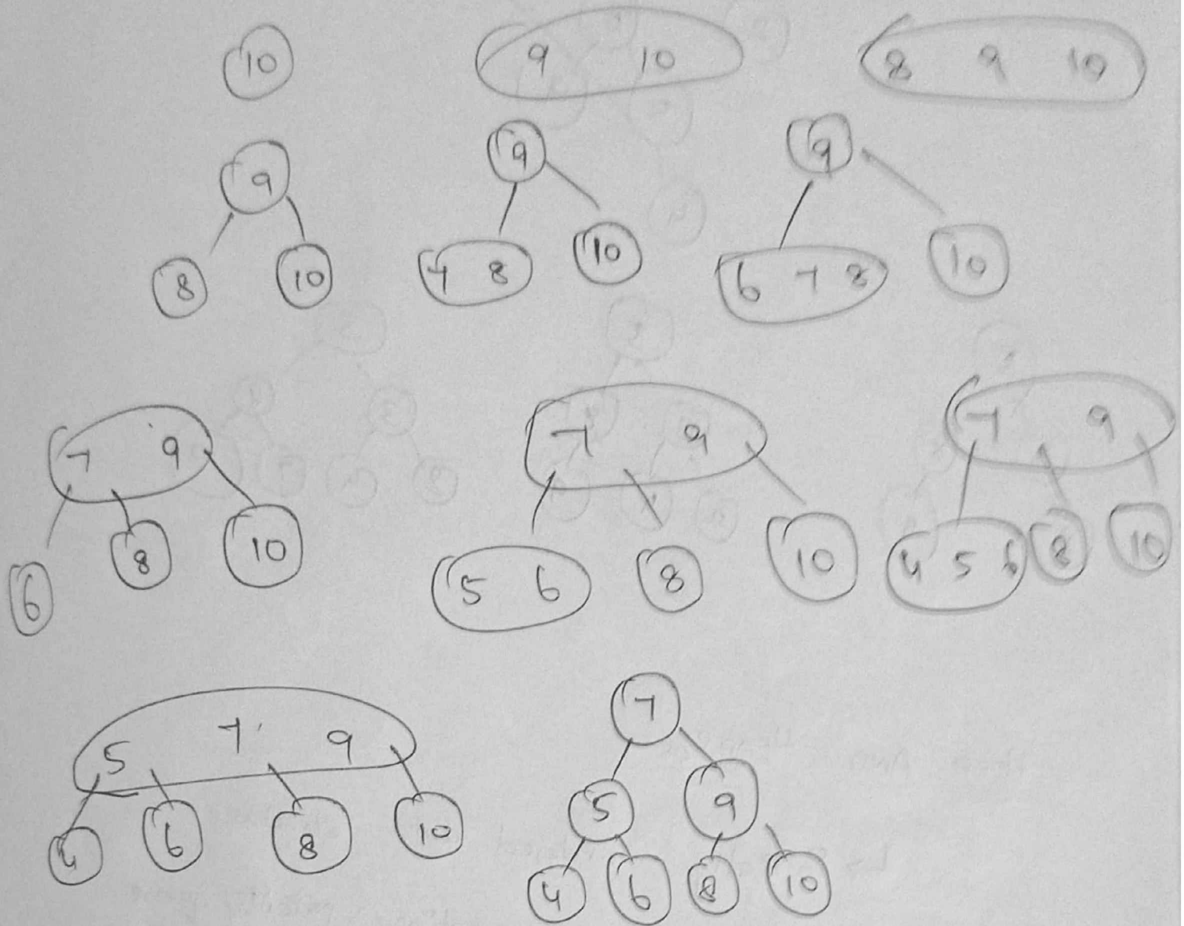
2-3 tree

- 2 nodes — single key — k
  - left child < k
  - right child > k
- 3 nodes
  - 2keys → k1, k2
  - k1 < k2
    - left child < k1
    - middle child k1 to k2
    - right child > k2

9 5 8 3 2 4 7

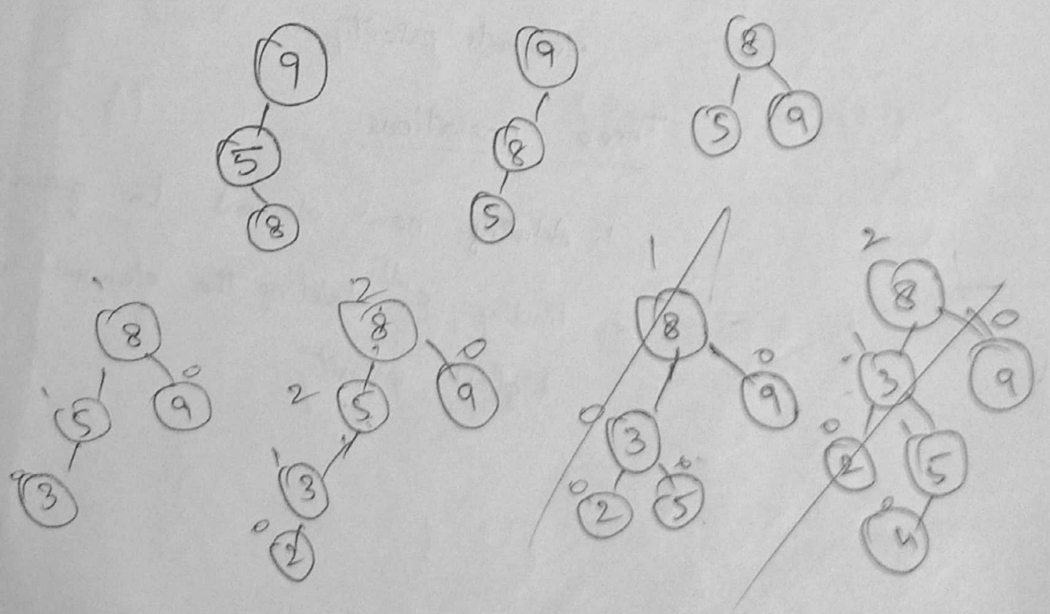


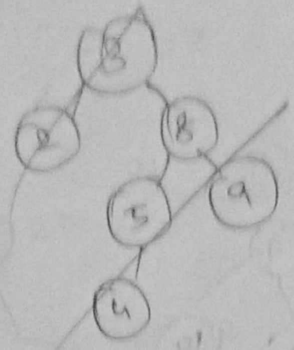
10 9 8 7 6 5 4

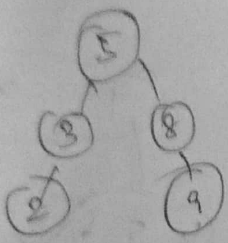


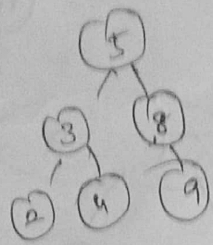
AVL Tree:

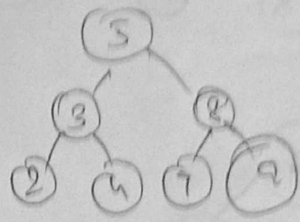
9 5 8 3 2 4 7











HEAP AND HEAPSORT:

↳ Partially ordered data structure

↳ Useful for implementing priority queue

Priority Queue:

↳ Orderable characteristic

||

elements priority

Three operations:

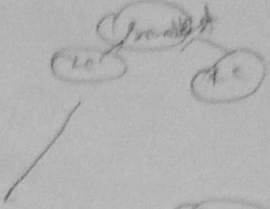
1. Adding new element to queue
2. Finding & 3) deleting the element with highest priority

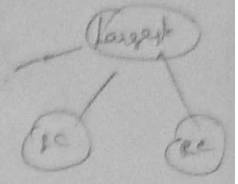
Heap :

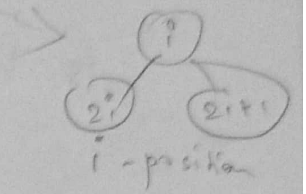
Two properties

↳ Shape — min heap, max heap

↳ Structural

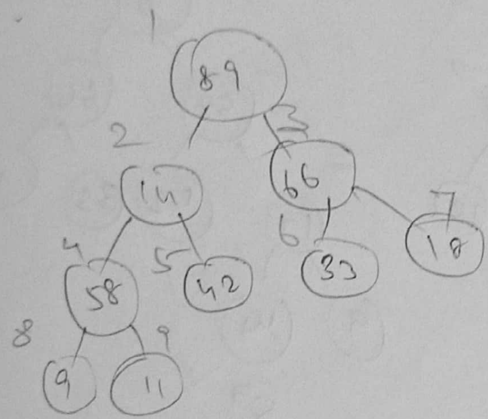






1 2 3 4 5 6 7 8 9

89, 14, 66, 58, 42, 33, 18, 9, 11



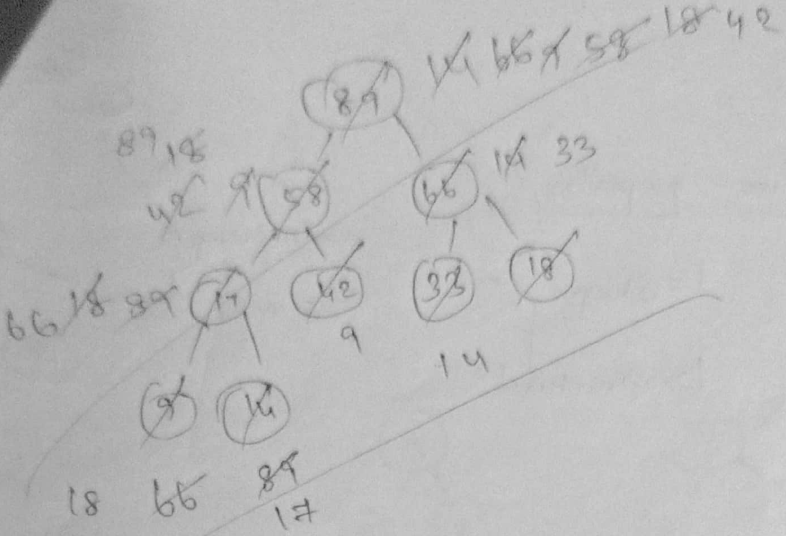
Here n = 9

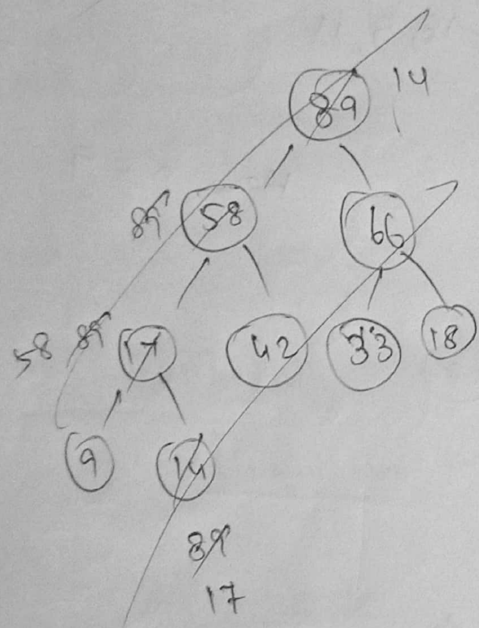
Bottom Up Approach

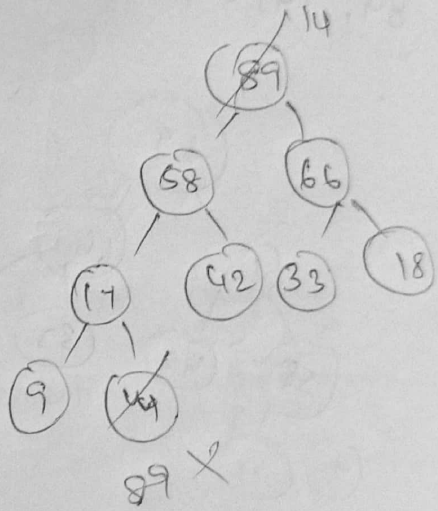
Starts from $\lfloor \frac{n}{2} \rfloor$ to 1

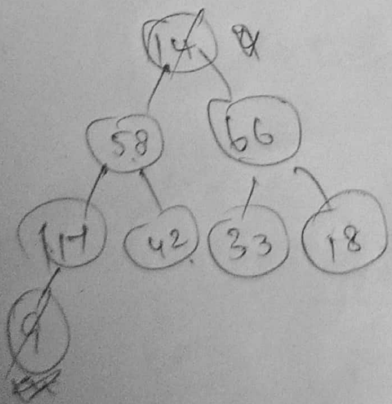
$$\lfloor n/2 \rfloor = \lfloor 9/2 \rfloor = 4$$

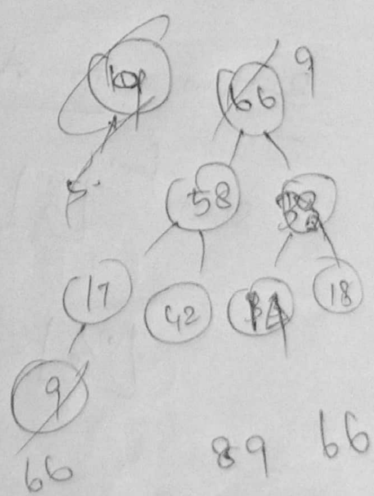
Reduce the size by 1

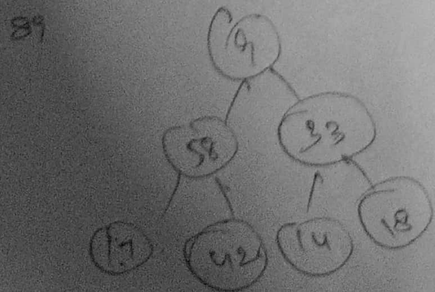

89
14 66 17 58 18 42
89 18
42 9
58
66
14 33
66 18 89
17
42
33
18
9
14
9
14
18 66 89
17

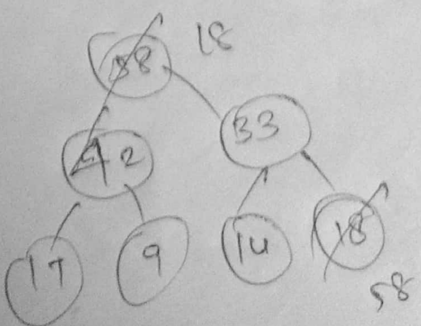

89
14
89
58
66
58 89
17
42
33
18
9
14
89
17

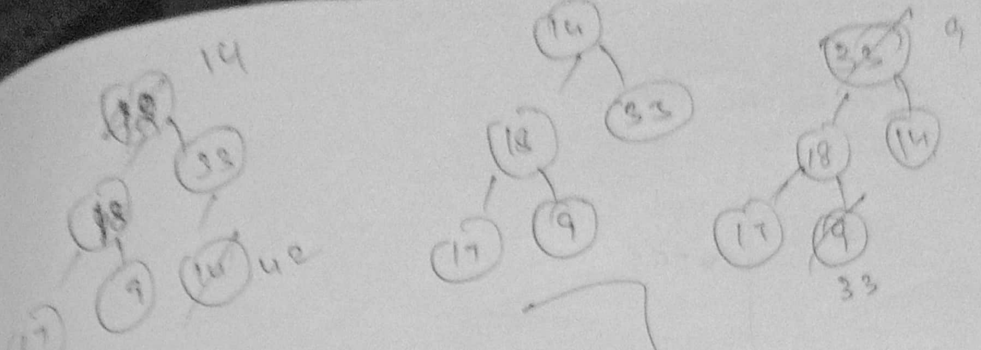

14
89
58
66
17
42
33
18
9
14
89


14
9
58
66
17
42
33
18
9


9
66
58
17
42
18
9
66
89 66

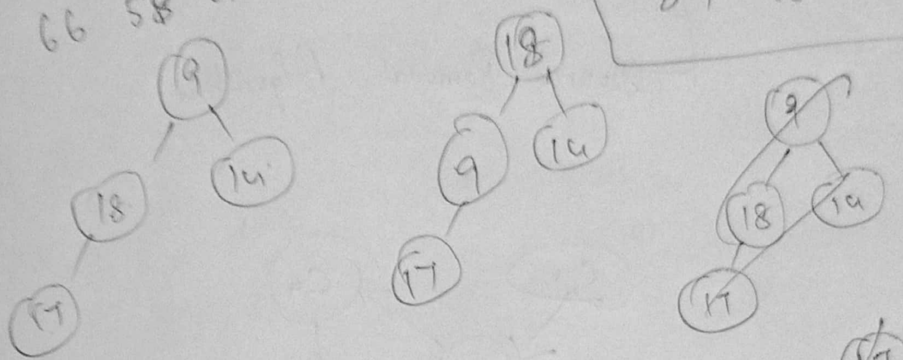

89
9
58
33
17
42
14
18

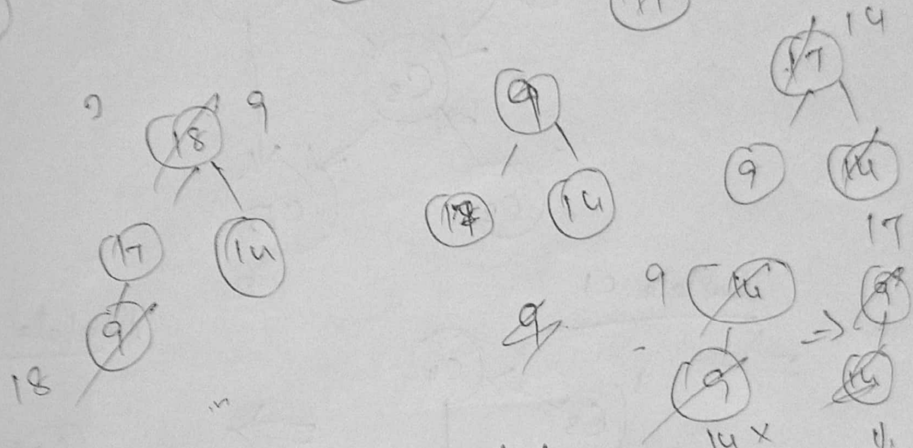

18
42
33
17
9
14
18
58



89 66 58 42

89 66 58 42 33



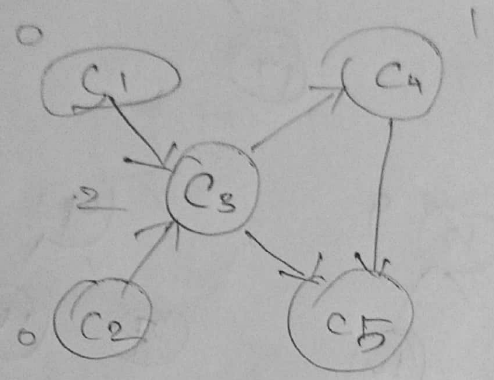


89 66 58 42 33 18 17 14 9

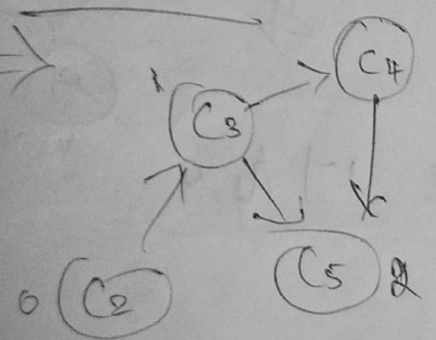
∴ Sorted array is 89 66 58 42 33 18 17 14 9

Topological Sort:

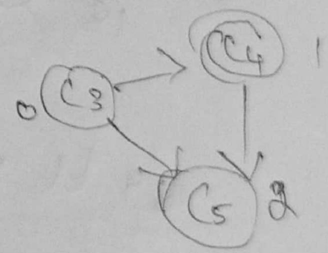
- DFS
- Source Removal Algorithm



Delete C1

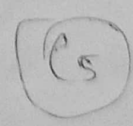


Delete C2



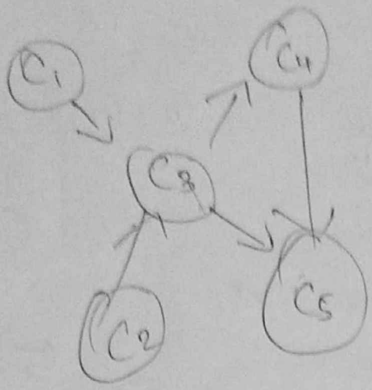
Delete C3

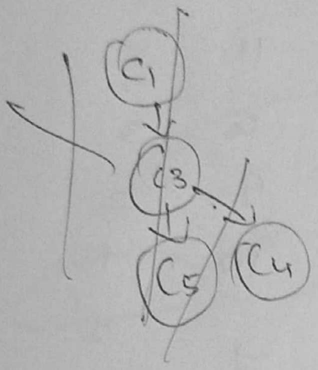
Delete C4

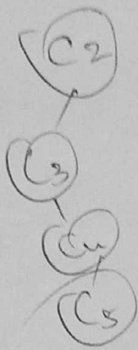


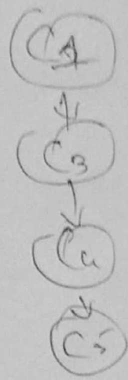
Order: C1 ⇒ C2 ⇒ C3 ⇒ C4 ⇒ C5

DFS



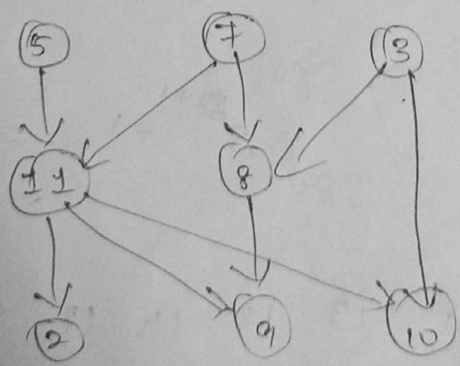


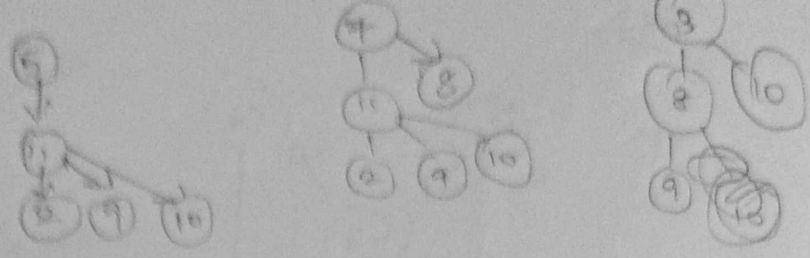




C_1	1	4
C_3	2	3
C_4	3	2
C_5	4	1

$C_1 \to C_3 \to C_4 \to C_5$





5 → 11 → 2 → 9

5	5
11	4
2	3
9	2
10	1
	2
~~7~~ 8	1
3	1

1) Construct a 2-3 tree with maximum of $3^{h+1}-1$ nodes.

2) Construct a full 2 tree with maximum of $2^{h+1}-1$ nodes.

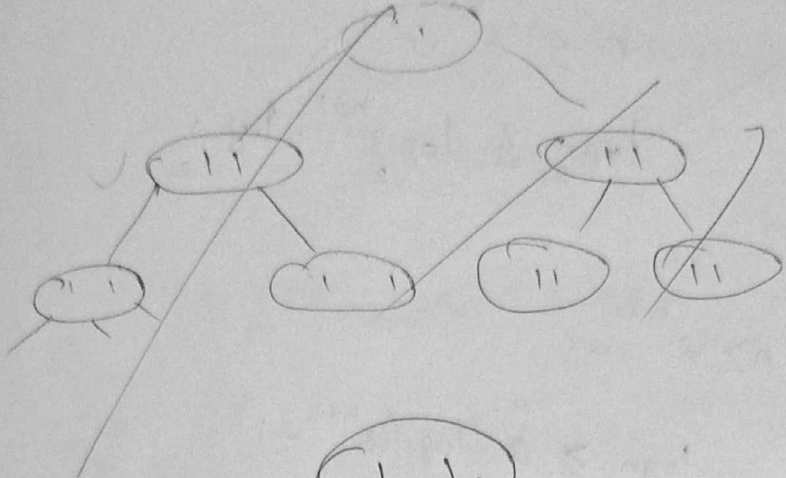
Soln

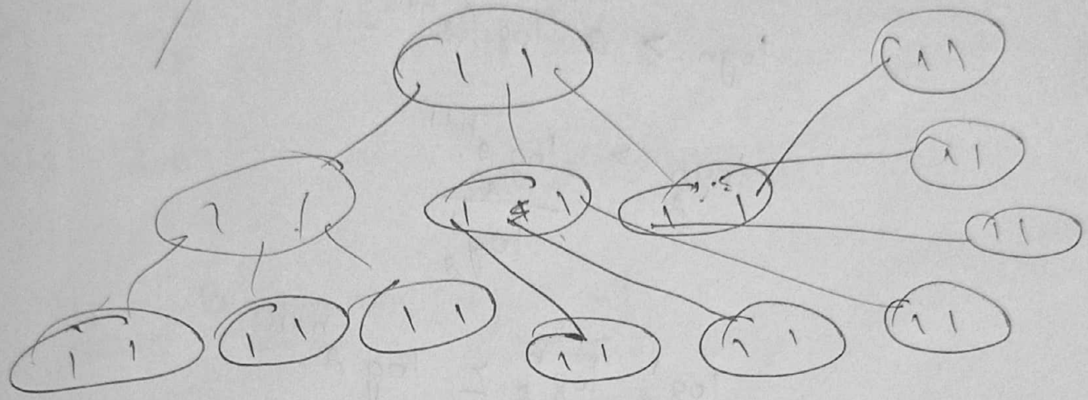
1) $h = 2$ $\quad 3^{2+1}-1 = 3^3-1 = 27-1 = 26$

19 15 18 13 12 14 17 9 5 8

3 2 4 7 29 25 28 23 22 24 27

39 35 38





→ Smallest possible no. of nodes in a 2-3 tree of full 2-tree – $n \geq 2^{h+1} - 1$

→ Largest possible no. of nodes in a 2-3 tree of full 3-tree – $n \leq 3^{h+1} - 1$

(i) Take log on both sides

$$\log_2 n \geq \log(2^{h+1} - 1)$$

$$\log_2 n \geq \log_2 2^{h+1} - \log_2 1$$

$$\log_2 n \geq \cancel{\log} (h+1) \log_2 2 - \log_2 1$$

$$(h+1) - \log_2 1 \leq \log_2 n \qquad \log a + \log b$$

$$\log ab$$

$$h+1 \leq \log_2 n + \log_2 1$$

$$h+1 \leq \log_2 (n+1)$$

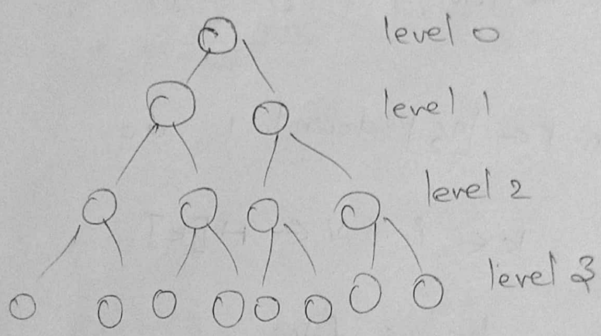
$$h = \log_2 n - 1$$

HEAPSORT ANALYSIS:

$C(n) =$ Heap Construction + Sorting

Heap Construction:

$$C_{worst}(n) = \sum_{i=0}^{h-1} \sum_{\text{level } i \text{ keys}} 2(h-i)$$



$2(n-\log_2(n+1))$

$h = 3$

$n/2 = \frac{15}{2} = 7$

7 to 1

$i = 2 \quad = 2(h-i)$

$= 2(3-2)$

$= 2(1) = 2$ comparisons

$$C_{worst}(n) = \sum_{i=0}^{h-1} \sum_{\text{level } i \text{ keys}} 2(h-i)$$

Here for ith level there are 2^i keys

$$\therefore \sum_{\text{level } i \text{ keys}} 2(h-i) = 2(h-i)2^i$$

$$C_{worst}(n) = \sum_{i=0}^{h-1} \sum_{\text{level } i \text{ keys}} 2(h-i)$$

$$= \sum_{i=0}^{h-1} 2(h-i)\, 2^i$$

$$= \sum_{i=0}^{h-1} 2(h)\, 2^i - \sum_{i=0}^{h-1} (2i)\, 2^i$$

$$= 2h \sum_{i=0}^{h-1} 2^i - \left(2 \sum_{i=0}^{h-1} i 2^i\right)$$

$$= \left[2h(2^h - 1)\right] - \left\{2\left[(h-2)2^h + 2\right]\right\}$$

$$= 2h2^h - 2.h - 2h2^h + 4 \times 2^h - 4$$

$$= -2h + 2^{h+2} - 4$$

$$= 2^{h+2} - 2h - 4$$

$$= 2 * 2^{h+1} - 2^h - 4$$

$$= 2(2^{h+1} - h - 2)$$

$$= 2(2^{h+1} - 1 - h - 1)$$

$$= 2(n - \log_2(n+1) - 1)$$

$$= 2[n - \log_2(n+1) - 1 - 1]$$

$$= 2[n - \log_2(n+1) - 2]$$

$$\in \Theta(n)$$

$2 + 3(2^2) + 3(2^3) + \dots + (h-1)$

Time Complexity for Heapsort

$C(n)$ = Time complexity for heap construction + Time complexity for maximum deletions

$C_1(n)$ = Time complexity for heap construction (or Heapify) $= \Theta(n)$

The number of key comparisons $C(n)$, needed for eliminating the root keys from the heaps of diminishing sizes from n to 2, we get the following inequality:

$$C_2(n) \leq 2\lfloor \log_2 (n-1) \rfloor + 2\lfloor \log_2 (n-2) \rfloor + \dots + 2\lfloor \log_2 (1) \rfloor \leq 2 \sum_{i=1}^{n-1} \log_2 i$$

$$\leq 2 \sum_{i=1}^{n-1} \log_2 (n-1)$$

$$\leq 2(n-1) \log_2 (n-1)$$

$$\leq 2n \log_2 n$$

$$\in O(n \log_2 n)$$

Time Complexity for Heapsort

$$C(n) = C_1(n) + C_2(n)$$

$$= \Theta(n) + \Theta(n \log_2 n)$$

$$= \Theta(n \log_2 n)$$

(2,3) (12,30) (40,50) (5,1) (12,10) (3,4) (4,2) (4,9)

Sort using x coordinates. If two x values are same then go to y coordinates

(2,3) (3,4) (4,2) (4,9) (5,1) (12,10) (12,30), (40,50)

Strip area
x coordinate of each point - x coordinate of mid point

mid = 4

(2,3) (3,4) (4,2) (4,9)

mid = 2

(2,3) (3,4)

$\sqrt{1+1} = \sqrt{2}$

$d_L = 1.414$

(4,2) (4,9)

$= \sqrt{0+7^2}$

$d_R = 7.0$

d = 1.414

Strip

2 - 4 = 2 < d ∈ No

3 - 4 = 1 < d ∈ Yes

4 - 4 = 0 < d ∈ Yes

4 - 4 = 0 < d ∈ Yes

dr = 7.0

dl = 1.4142

ds = 2.236

(3,4) (4,2) (4,9)

i) 2 - 4 > = 1.4142 ∈ Yes

$d = \sqrt{1+2^2} = \sqrt{5}$

= 2.236

ii) 9 - 4 > = 1.4142 ∈ Yes

$d = \sqrt{1+5^2} = \sqrt{26}$

= 5.09

iii) 9 - 2 > = 1.4142 ∈ Yes

(5,1) (12,10) (12,30) (40,50)

(5,1) (12,10)

$= \sqrt{7^2+9^2}$

$= \sqrt{130}$

= 11.4

(12,30) (40,50)

$= \sqrt{28^2 + 2}$

$= \sqrt{784 + 400}$

= 34.40

d = 11.4

5 - 12 < 11.4 ∈ Yes

12 - 12 < 11.4 ∈ Yes

12 - 12 < 11.4 ∈ Yes

40 - 12 < 11.4 ∈ No

(5,1) (12,10) (12,30)

i) = 10 - 1 > = 11.4 ∈ No

ii) 30 - 1 > 11.4 ∈ Yes

$d = \sqrt{7^2 + 29^2} = 29.8$

iii) 30 - 10 > 11.4 ∈ Yes

$d = \sqrt{0^2 + 10^2} = 10$

$d_L = 11.4$

$d_R = 34.4$

$d_S = 10$