AOA

DATE : / /

PAGE NO. :

Algorithm :⇒ It is a set of instructions designed to perform specific task.

(i) Input (ii) Output (iii) Definiteness (iv) finiteness (v) Effectiveness

Measurement :- (i) Is it correct?

(ii) Is it readable?

(iii) Performance Analysis

(a) Machine dependent :-

(A) H/w (B) OS (C) Compiler (d) Processor

(b) Machine independent :-

Complexity: The complexity means how complex an algorithm in which can be reflected as relative amount of time or space they required.

(i) Space Complexity :-

$$S_p = C + S_p(\text{Instance})$$

(ii) Time Complexity :-

$$T_p = C + T(i)$$

Where C = Compile time , T(i) = Run time

Eg- Algo. swap(a, b) → 0
{ temp = a; → 1
a = b; → 1
b = temp; → 1
} → 0

f(n) = 0+1+1+1+0 = 3

$T_p = 3$

Space : a → 1 , temp → 1
b → 1
counts = 3

$S_p = O(1)$

Eg - Sum(A, N) → 0
{ S = 0 → 1
for (i=0; i<n; i++) → n+1
{ S = S + A[i] } → n
return S; } → 1

$T_p = 1 + n + 1 + n + 1 = 2n + 3$

Space: S → 1
n → 1
i → 1
A → n
n+3

O(n)

Eg- Add(A, B, n)

```
{   for (i=0; i<n; i++)          → n+1
  { for (j=0; j<n; j++)          → n(n+1)
    {
      C[i,j] = A[i,j] + B[i,j]   → n*n
    } } }
```

$f(n) = n+1+n^2+n+n^2$, Space: $A \rightarrow n*n$, $B \rightarrow n*n$, $C \rightarrow n*n$, $i \rightarrow 1$, $j \rightarrow 1$, $n \rightarrow 1$

$f(n) = 2n^2+2n+1$

Time: $f(n) = O(n^2)$

$S(n) = 3n^2+3$

$O(n^2)$

Eg- mul(A, B, n)

```
{ for (i=0; i<n; i++)                    → n+1
  { for (j=0; j<n; j++)                  → n(n+1) = n²+n
    {       C[i,j] = 0;                  → n*n = n²
      for (K=0; K<n; K++)                → n*n(n+1) = n³+n²
      {
        C[i,j] = C[i,j] + A[i,K] * B(K,j]  → n*n(n+1)
      } } }
```

Time: $f(n) = 2n^3+3n^2+2n+1 \Rightarrow O(n^3)$

Space: $\left.\begin{matrix} A \rightarrow n^2 \\ B \rightarrow n^2 \\ C \rightarrow n^2 \end{matrix}\right] \rightarrow 3$ $\quad S(n) = 3n^2+4$

$\left.\begin{matrix} n \rightarrow 1 \\ i \rightarrow 1 \\ j \rightarrow 1 \\ K \rightarrow 1 \end{matrix}\right] \rightarrow 4$ $\quad S(n) = O(n^2)$

$Sp = C + S(i) = 4+3$

$1 < \log n < \sqrt{n} < n < n\log n < n^2 < n^3 < --- < 2^n < 3^n < --- < n^n$

Lower   Average   Upper

Asymptotic Notation:- The notation describe different rate of growth relation b/w defining function & the defined set of function.

(i) Big oh (O) [Worst case]:- It defines upper bounds of algorithm runtime.

$$f(n) \leq c * g(n)$$

(ii) Big Omega (Ω) [Best case]:- It defines lower bounds of algorithm runtime.

$$f(n) \geq c * g(n)$$

(iii) Theta (θ) [average case]:- It is used when function is bounded both from above & below by the function g(n).

$$f(n) = \theta(g(n))$$

$$c_1 * g(n) \leq f(n) \leq c_2 * g(n)$$

Q. (i) $f(n) = 10n^2 + 4n + 5$

$10n^2 + 4n + 5 \leq 10n^2 + 4n^2 + 5n^2$

$f(n) \leq 19n^2$

$c = 19, \; n \geq 1, \; O(n^2)$

$10n^2 + 4n + 5 \geq 9n^2$

$c = 9, \; n \geq 1, \; \Omega(n^2)$

(ii) $f(n) = 5n^3 + 2n^2 + 7$

$5n^3 + 2n^2 + 7 \leq 5n^3 + 2n^3 + 7n^3$

$f(n) \leq 14n^3$

$c = 14, \; n \geq 1, \; O(n^3)$

$5n^3 + 2n^2 + 7 \geq 4n^3$

$c = 4, \; n \geq 1, \; \Omega(n^3)$

(c) $f(n) = 10^n + 5n^3 + 3n^2 + 5$

$10^n + 5n^3 + 3n^2 + 5 \leq 2(10^n)$

$c = 2, \quad n \geq 2, \quad O(10^n)$

$10^n + 5n^3 + 3n^2 + 5 \geq 9^n$

$c = 9, \quad n \geq 1, \quad \Omega(n)$

(d) $f(n) = 6 * 2^n + n^2 + 5 \leq 7 * 2^n$

$c = 7, \quad O(2^n)$

Q. $f(n) = n^2 \log n + n$

$n^2 \log n + n \leq 10 n^2 \log n$

$O(n^2 \log n)$

$n^2 \log n + n \geq n^2 \log n \quad \Omega(n^2 \log n)$

$n^2 \log n \leq n^2 \log n + n \leq 10 n^2 \log n$

$\theta(n^2 \log n)$

Q. $f(n) = n! = n(n-1)(n-2) \ldots 3.2.1$

$1.1.1.1 \leq 1.2.3 \ldots n \leq n.n.n \ldots n$

$1 \leq n! \leq n^n$

$O(n^n) \quad \Omega(1)$

Q. $f(n) = \log n!$

$\log(1.1.1 \ldots 1) \leq \log(1.2.3 \ldots n) \leq \log(n.n.n \ldots n)$

$1 \leq \log n! \leq n \log n$

$O(n \log n), \quad \Omega(1)$

DATE : / /

PAGE NO. :

<u>Recurrences:-</u> A function or an algorithm which calls itself.

$T(n) = aT(n/2) + f(n)$, $n > 1$

4 method solved recurrence

(i) Substitution method

(ii) Iteration method

(iii) Recursion tree method

(iv) Master Method

Master Method:- It works only for following type of recurrences or for recurrences that can be transformed to following type.

$T(n) = a\,T(n/b) + f(n)$ $a \geq 1,\ b > 1$

where divide the n problem into a subproblem of size $\frac{n}{b}$.

↳ constant

$T(n/b) =$ Time of subproblem's solution

$f(n) =$ Merging Process

<u>Master Method:-</u>

Case-1:- If $f(n)$ is $O(n^{\log_b a - \epsilon})$ then $T(n)$ is $O(n^{\log_b a})$.

Case-2: If $f(n)$ is $O(n^{\log_b a} \log^k n)$ then $T(n)$ is $O(n^{\log_b a} \log^{k+1} n)$. $k \geq 0$

Case-3: If $f(n)$ is $\Omega(n^{\log_b a + \epsilon})$ then $T(n)$ is $\Theta(f(n))$.

Eg- $T(n) = 4\,T(n/2) + n$ solve using Master Method

Sol^n- $a = 4$, $b = 2$, $f(n) = n$

Case 1 apply when $f(n) < n^{\log_b a}$

case 2 apply when $f(n) = n^{\log_b a}$

case 3 apply when $f(n) > n^{\log_b a}$

$n^{\log_2 4} = n^{\log_2 2^2} = n^{2\log_2 2} = n^2$

So $f(n) < n^2$

As the rule, On applying case-1

If $f(n)$ is $O(n^{\log_b a - e})$ then $T(n)$ is $O(n^{\log_b a})$

If $f(n)$ is $O(n^{2-1})$ then $T(n)$ is $O(n^2)$

If $f(n)$ is $O(n)$ then $T(n)$ is $O(n^2)$

$F(n) = O(n)$, $T(n) = O(n^2)$

Q. $T(n) = 3T(n/2) + n^2$

Ans - $a = 3$, $b = 2$, $f(n) = n^2$

$n^{\log_b a} = n^{\log_2 3} = n^{\log_{10} 3 / \log_{10} 2} = n^{0.477/0.3010}$

$f(n) > n^{\log_2 3} \Rightarrow n^2 > n^{\log_2 3}$

We are applying case 3

If $f(n)$ is $\Omega(n^{\log_2 3 + 1})$ then $T(n) = \theta(f(n))$

$\boxed{T(n) = \theta(n^2)}$ Ans

Q. $T(n) = 4T(n/2) + n^2$

Ans - $a = 4$, $b = 2$, $f(n) = n^2$

$n^{\log_b a} = n^{\log_2 4} = n^2$

$f(n) = n^2$ which denotes case-2

If $f(n)$ is $O(n^{\log_2 4} \log^0 n)$ then $T(n) = O(n^{\log_b a} \log n)$

$\boxed{T(n) = O(n^2 \log n)}$ on $k = 0$

Q. $T(n) = T(n/2) + 2^n$

Ans - $a = 1$, $b = 2$, $f(n) = 2^n$

$n^{\log_2 1} = n^0 = 1$

$f(n) > 1$, Now We are applying case-3

If $f(n)$ is $\Omega(n^{\log_2 1+e})$ then $T(n)$ is $\theta(f(n))$

$\boxed{T(n) = \theta(2^n)}$

Q. $T(n) = 2^n\, T(n/2) + n^n$

Ans- $a = 2^n$, $b = 2$, $f(n) = n^n$

$n^{\log_b a} = n^{\log_2 2^n} = n^n$

Does not apply coz a is constant

~~$f(n) = n^n$ so On applying case-2~~

~~If $f(n)$ is $\theta(n^{\log_2 2^n} \log n)$ then $T(n)$ is $\theta(n^{\log_2 2^n} \log n)$~~

~~$T(n) = n^n$~~

Q. $T(n) = 2T(n/2) + n\log n$

Ans- $a = 2$, $b = 2$, $f(n) = n\log n$

$n^{\log_b a} = n^{\log_2 2} = n$

$f(n) > n$

Case-3 If $f(n) = \Omega(n^{\log_2 2+e})$ then $T(n) = \theta(n\log n)$

Q. $T(n) = 2T(n/2) + n\log^{-1} n$

Ans- $a = 2$, $b = 2$, $k = -1$, $f(n) = n\log^{-1} n$

DNA, non-polynomial b/w $f(n)$ & $n^{\log_b a}$

Q. $T(n) = 2T(n/4) + n^{0.51}$

Ans- $a = 2$, $b = 4$, $f(n) = n^{0.51}$

$n^{\log_4 2} = n^{\log 2/\log 4} = n^{0.50}$

$f(n) > n^{0.50}$

Case-3 $f(n) = \Omega(n^{0.50+1})$ then $\boxed{T(n) = \theta(n^{0.51})}$

Q. $T(n) = \sqrt{2}\,T(n/2) + \log n$

$n^{\log_2 \sqrt{2}} = n^{0.50} = \sqrt{n}$

$\log n < \sqrt{n} \Rightarrow f(n) < \sqrt{n}$

case-1 $f(n) = O(\sqrt{n})$ then $T(n) = O(n^{\log_2 \sqrt{2}}) = O(\sqrt{n})$

DATE : / /

PAGE NO. :

Q. $T(n) = 3T(n/2) + n$

$n^{\log_2 3} = n^{\log 3/\log 2} = n^{0.47/0.30}$, $f(n) = n$

$f(n) < n^{1.6}$

Case-1 $f(n) = O(n^{1.6})$ then $T(n) = O(n^{\log 3})$

Q. $T(n) = 4T(n/2) + n/\log n$

$n^{\log_2 4} = n^2$

$\frac{n}{\log n} < n^2$

case-1

$f(n) = O(n^{2})$ then $\boxed{T(n) = O(n^2)}$

Q.(i) $T(n) = 8T(n/2) + n^2$

Ans- $a = 8$, $b = 2$

$n^{\log_2 8} = n^4$

$f(n) < n^4$

Case 1 $f(n^{\log_2 8 - e}) = f(n^3)$ so $\boxed{T = O(n^4)}$

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

(iii) $T(n) = T(n/2) + c$

(iv) $T(n) = T(n/2) + N$

(v) $T(n) = 2T(n/2) + 1$

Ans- (ii) $T(n) = T(n-1) + n$

$a = 1$, $b = 1$, $f(n) = n$

$n^{\log_1 1} = n^0 = 1$ $f(n) > 1$

$O(n^2)$

~~b should be greater than 1 i.e. b>1~~

~~It Does not apply coz b is not constant~~

(iii) $a = 1$, $b = 2$, $f(n) = c$

$n^{\log_2 1} = n^0 = 1$

$f(n) = c > 1$

Case-3 $f(n) = \Omega(n)$ then $\boxed{T(n) = \theta(c)}$

DATE : / /
PAGE NO. :

(IV) $a=2, b=2, f(n)=1$
$n^{\log_2 2} = n$
$f(n) < n$
Case-1 $f(n) = O(1)$ then $\boxed{T(n) = O(n)}$

(iv) $a=1, b=2, f(n) = N \Rightarrow n^{\log_2 1} = n^0 = 1$
$f(n) \geq 1$ so case-3
$O(\log n)$
$f(n) = O(\log n)$ then $\boxed{T(n) = O(\log n)}$

Algorithm design :-

(i) Divide & Conquer Method :→

Problem → Binary search, Quick sort, merge sort, Strassen method

General algorithm:
Step-1 if trivial case
Step-2 solved
Step-3 else
Step-4 divide into sub problem
Step-5 Solved sub problem recursively
Step-6 Combine sol^n to sub problem
Step-7 endif

HOW TO TEACH PRADEEP SIR ?
Step 1 desiging method
Step 2 definition of problem
Step-3 Algo. of problem
Step-4 Numerial of problem
Step-5 Analysis of problem

8 Binary Search :-

Pro. Defi. :- Given any sorted array of n integer, search for a number (called key) in the array. Return the position of occurrence if found Report failure if not found.

Algo:- BINARY SEARCH(key, A, LB, UB)

1. if (LB > UB)
2. return "search fail"
3. m = (LB+UB)/2
4. If (key = A[m]) then
5. return "search successfully"
6. else if (key < A[m])
7. BINARY SEARCH (key , A, LB, m-1)
8. else BINARY SEARCH(key, A, m+1, UB)

Problem 4, 7, 8, 11, 13, 15, 21, 23, 28, 30

key = 21 , N = 10 , Lb = 1, Ub = 10

Lb > Ub no

M = (Lb+Ub)/2 = (1+10)/2 = 5

A [5] = 13

Compare A[m] and key 21 > 13 it means we should search Binary (key, A, m+1, Ub)

Binary(21, A, 6, 10)

Lb = 6 , Ub = 10

Lb > Ub no

m = (6+10)/2 = 8

A[8] = 23

Compare A[8] and key 21 < 23 it means

Binary (key, A, lb, m-1)

Binary (21, A, 6, 7)

Lb > Ub no

m = (6+7)/2 = 6

A[6] = 15

Compare A[6] & key 21 > 15 it means

Binary (key, A, m+1, ub) = Binary (21, A, 7, 7)

m = (7+7)/2 = 7 A[7] = 21

DATE : / /

PAGE NO. :

Compare A[7] & Key :- 21 = 21 then return "search successful"

Analysis :-

Best case : $O(1)$ if key = A[m]

Average case: $T(n) = T(n/2) + K$

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

Worst case: $O(\log n)$

* Quick Sort :-

Given array of N integer to apply Quick and sort the element increasing or decreasing order. It is done by Divide & Conquer method.

Algo:- QUICK SORT (A, P, R)

1. If P < R then
2. Q = partition (A, p, r)
3. QUICK SORT (A, P, q-1)
4. QUICK SORT (A, q+1, r)

partition (A, p, r)

Step-1 pivot = A[p]

Step-2 up = r

Step-3 down = p + 1

Step-4 while(down < up)

Step-5 while (A[down] <= pivot)

down = down + 1

Step-6 while (A[up] ~~>~~= pivot)

up = up - 1

Step-7 if (down < up)

swap(A[down], A[up])

Step-8 end while

Step-9 swap(A[up], pivot) return (up)

DATE : / /
PAGE NO. :

Eg- 23, 14, 8, 2, 11, 28, 31, 5, 19, 25 Solve by Quick sort

p = 1 r = 10

Step-1 if p < r then

q = partition (A, 1, 10)

pivot = A[1] = 23

up = 10

down = 2 # The down pointer travelling till it find element smaller than pivot.

while(2 < 10)

while(14 <= 23)

The up pointer travels till it find element greater than pivot.

23, 14, 8, 2, 11, [28]down, 31, 5, 19[up], 25

if (down < up) ⇒ if(6 < 9)

swap (28, 19)

→ 23, 14, 8, 2, 11, 19, 31, 5, 28, 25

23, 14, 8, 2, 11, 19 31[down], 5[up], 28, 25

→ 23, 14, 8, 2, 11, 19, 5, 31, 28, 25

23, 14, 8, 2, 11, 19, 5[up], 31[down], 28, 25

if(8 < 7)

then swap(A[up], pivot)

→ $\underbrace{5, 14, 8, 2, 11, 19,}_{A}$ 23, $\underbrace{31, 28, 25}_{B}$

5, 14, 8, 2, 11, 19

5, 14[down], 8, 2[up], 11, 19

5, 2[up], 8[down], 14, 11, 19

Swap (A[up], pivot)

→ 2, 5, $\underbrace{8, 14, 11, 19}$

8[up], 14[down], 11, 19

8, 14, 11[up], 19[down] ⇒ 8, 11, 14, 19

DATE : / /
PAGE NO. :

~~After applying Quick sort on A elements are~~

2, 5, 8, 11, 14, 19

B ⇒ 31, 28, 25   31, 28[down], 25[up]

31, 25, 28   31, ~~25[down] 28[up]~~ ?

swap (A[up], pivot)   down   we could not find down so it will out from loop

~~28, 25, 31~~   25, 28, 31

~~28~~, 25[up], 31[down]

swap (A[up], pivot)   UP could not find

25, 28, 31

Best & Average : T(n) pivot produce each time then every iteration step sub array • $T(n) = T(n/2) + n$

Eg ⇒ ~~99, 82, 94, 15, 43, 56, 88, 80, 69, 89, 46, 70, 56, 64, 18, 61~~

$T(n) = O(n \log n)$ It depends on pivot element.

Worst :- $O(n^2)$

Merge Sort :- It divides input array in 2 halves, calls itself for 2 halves & then merges the 2 sorted halves.

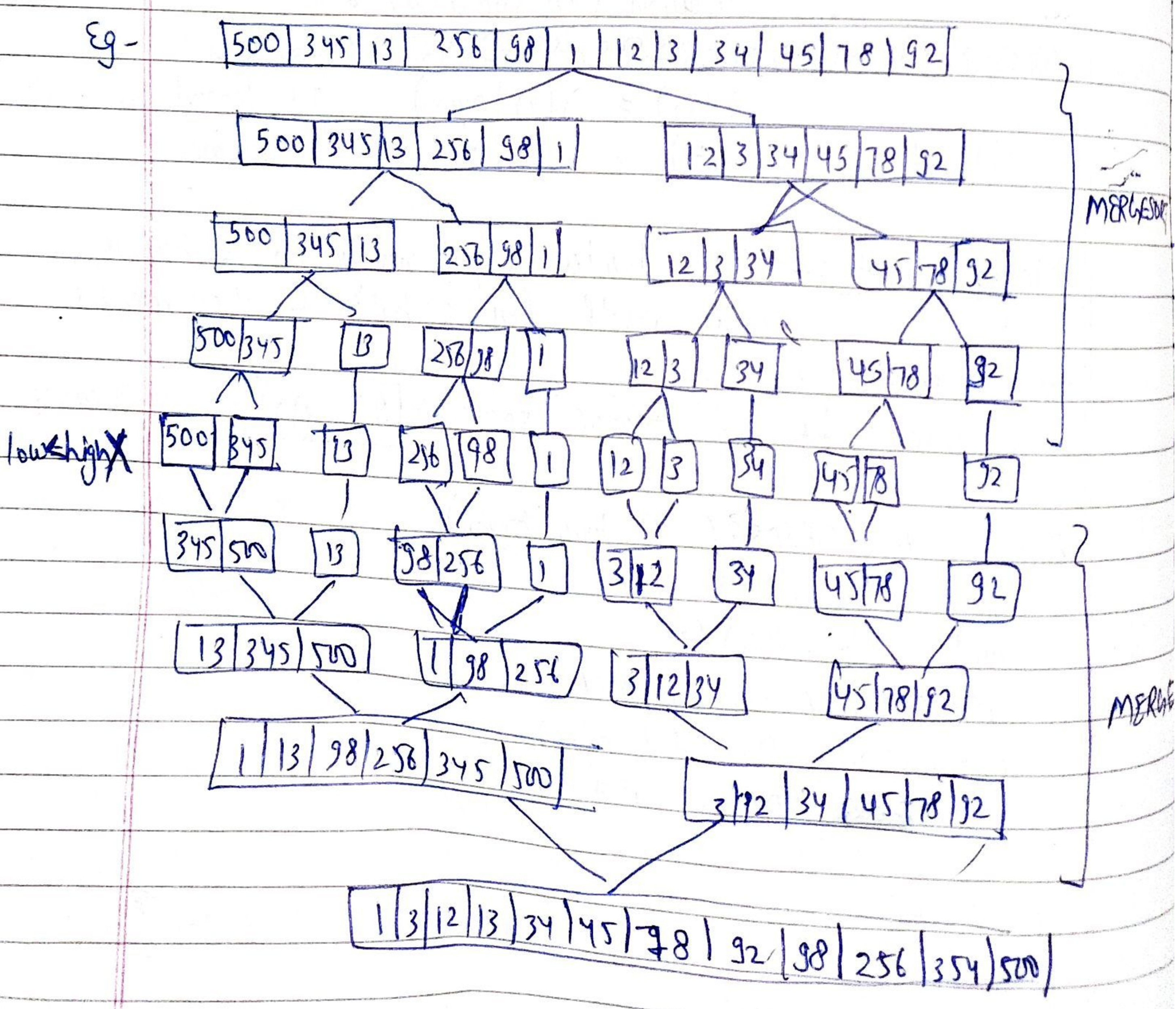
Time complexity of Merge sort is $\boxed{\theta(n \log n)}$ in all 3 cases.

Algo: MERGESORT(A, low, high)
1. If (low < high) then
2. Mid = (low+high) / 2
3. MERGESORT(A, low, mid);
4. MERGESORT(A, mid+1, high);
5. MERGE (A, low, mid, high);

MERGE(a , low, mid, high);
1. L = low   2. H = high   3. J = mid+1
4. K = low (array B)

```
While( low <= mid AND j<=high) do
  If ( a[low]< a[j] ) then
      B[k]= a[low];
      K = k+1;
      Low = Low+1;
  Else
    B[K] = a[j];
    K = k+1;
    J = j+1;
  While ( low <= mid ) do
    B[k] = a[low];
    k = k+1;
    Low = low+1
  while (j <= high) do
    B[K] = a[j];
    K = K+1;
    J = j+1;
  For i = 1 to n do
    A[i] = B[i].
```

Eg-



DATE : / /

PAGE NO. :

Strassen's Matrix Multiplication :-

Problem :- Given two matrix A & B of size $n * n$ each, obtain their product matrix C.

Strassen's Method ⇒ This method reduce the number of multiplication to 7 & we have 18 addition or subtraction.

We will calculate value of C_{ij} using 7 multiplication for 7 $(n/2 * n/2)$ matrix and 10 additions or subtractions as follows $\begin{vmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{vmatrix}$ $\begin{vmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{vmatrix}$

7*
10+-

$P = (A_{11} + A_{22})(B_{11} + B_{22})$, $Q = (A_{21} + A_{22}) * B_{11}$, $R = A_{11} * (B_{12} - B_{22})$

$S = A_{22} * (B_{21} - B_{11})$, $T = (A_{11} + A_{12}) * B_{22}$, $U = (A_{21} - A_{11}) * (B_{11} + B_{12})$

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

8+-

$C_{11} = P + S - T + V$, $C_{12} = R + T$, $C_{21} = Q + S$, $C_{22} = P + R - Q + U$

(using 8 additions or subtractions)

Eg- $A = \begin{bmatrix} -5 & 2 \\ 0 & 4 \end{bmatrix}$ $B = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix}$ $C = ?$

$P = (-5+4)(1+0) = -1$, $Q = (0+4) * (1) = 4$, $R = (-5) * (1-0) = -5$

$S = 4 * (3-1) = 8$, $T = (-5+2) * 0 = 0$, $U = (0+5) * (1+1) = 10$

$V = (2-4) * (3+0) = -6$

$C_{11} = -1 + 8 - 0 - 6 = 1$, $C_{12} = -5 + 0 = -5$, $C_{21} = 4 + 8 = 12$, $C_{22} = -1 - 5 - 4 + 10 = 0$

$C = \begin{bmatrix} 1 & -5 \\ 12 & 0 \end{bmatrix}$ Ay

SSM Analysis ⇒ Basic complexity = $O(n^3)$

$$T(n) = 7T(n/2) + 18n^2$$

Using Iteration Method

$$n = n/2$$

$$T(n) = [7\, T(n/4)] + 18(n/2)^2$$

$$T(n) = 7(7T(n/4) + \frac{18n^2}{4}) + \frac{18n^2}{4}$$

DATE : / /
PAGE NO.:

$= 7^{\log_2 n} + c.n^{\log_2 7}$

$T(n) = O(n^{2.81})$

(ii) Greedy Method ⇒ {We have 2 type of solⁿ

(i) Feasible solⁿ
(ii) Optimal Solⁿ}

Greedy method is a method of selecting a subset of input as the solution which is optimal

General Algo :-

1. Choose an element e belonging to dataset D.
2. Check whether e can be included into the solⁿ set S, if yes then include
3. Continue from step 1 until S is filled or D is empty.

Problem (i) Knapsack problem :-

- Fraction KP → Solⁿ by Greedy
- 0/1 KP → Solⁿ by Dynamic

Definition :- Given a knapsack of capacity m unit and n items Each having weight W_i and value v_i. The problem is to fill the knapsack with item such that total profit value is maximum.

```
GREEDY KNAPSACK (w, v, n, m)
1.  for (i=1 to n)
2.      x[i] = 0;
3.      s[i] = v[i] / w[i];
4.      end for
5.  profit = 0;
6.  Sort s[i] in decreasing order and according
    sort v and w.
7.  For (i=1 to n)
8.  if (w[i] ≤ m)
9.      x[i] = 1;
10.     m = m - w[i];
11. else exit loop
12. if i ≤ n
13. x[i] = m / w[i];
14. for i=1 to n
15. Profit = Profit + (V[i] * x[i])
16. return (x, Profit)
```


Q. Find a solution by Greedy approach of the following instance of Knapsack problem

$W = \{4, 10, 5, 6, 8, 3\}$

$V = \{20, 15, 30, 18, 16, 21\}$

$M = 20$

Sol^n:-

$$S[i] = \frac{v[i]}{W[i]} = \left\{ \frac{20}{4}, \frac{15}{10}, \frac{30}{5}, \frac{18}{6}, \frac{16}{8}, \frac{21}{3} \right\}$$

$s[i] = \{5, 1.5, 6, 3, 2, 7\}$

Item	vi/wi	wi	Vi	x[i]	W[m]
I_6	7	3	21	1	17
I_3	6	5	30	1	12
I_1	5	4	20	1	8

I_4	3	6	18	1	2
I_5	2	8	16	0.25	0
I_2	1.5	10	15	—	—

For $i=1$, check $w[i] \le M$
$3 \le 20$ yes then $x[i]=1$,
$m = m - w[i] = 20-3 = 17$

For $i=2$, check $w[i] \le M$
$5 \le 17$ yes then $x[i]=1$ &
$m = 17-5 = 12$

For $i=3$, check $w[i] \le M$
$4 \le 12$ yes then $x[i] = 1$ & $m = 12-4 = 8$

For $i=4$, check $w[i] \le M$
$6 \le 8$ yes then $x[i]=1$ & $m = 8-6 = 2$

For $i=5$, check $w[i] \le M$
$8 \le 2$ No then
if $i \le n$ yes
$x[i] = M / w[i] = 2/8 = 0.25$
$M = 0$

$x = \{1, 1, 1, 1, 0.25, 0\}$

Profit = Profit + $(21 \times 1 + 30 \times 1 + 20 \times 1 + 18 \times 1 + 16 \times 0.25 + 15 \times 0)$

Profit = $0 + 93 = 93$ Ans

Fraction Knapsack with using Greedy Method

Analysis:-

Step 1-4 :- $O(n)$

Complexity $\Rightarrow O(n)$

DATE : / /

PAGE NO. :

Q. $n=7$, $M=15$ $v=\{10, 5, 15, 7, 6, 18, 3\}$ $w=\{2,3,5,7,1,4,1\}$

Ans $S[i] = \frac{V[i]}{w[i]} = \left\{ \frac{10}{2}, \frac{5}{3}, \frac{15}{5}, \frac{7}{7}, \frac{6}{1}, \frac{18}{4}, \frac{3}{1} \right\}$

$S[i] = \{ 5, 1.6, 3, 1, 6, 4.5, 3 \}$

Item	v[i]/w[i]	w_i	V_i	x[i]	W[m]
I_5	6	1	6	1	14
I_1	5	2	10	1	12
I_6	4.5	4	18	1	8
I_3	3	5	15	1	3
I_7	3	1	3	1	2
I_2	1.6	3	5	0.67	0
I_4	1	7	7	–	–

For i=1, check $w[i] \le M \Rightarrow 1 \le 15$ yes then $x[i]=1$ $m=15-1=14$

For i=2, $2 \le 14$ yes then $x[i]=1$, $m=12$

For i=3, $4 \le 12$ yes then $x[i]=1$, $m=8$

For i=4, $5 \le 8$ yes then $x[i]=1$, $m=3$

For i=5, $1 \le 3$ yes then $x[i]=1$, $m=2$

For i=6, $3 \le 2$ No then

if $i \le n$ yes $x[i] = m/w[i] = 2/3 = 0.67$

$m = 0$

$x = \{ 1, 1, 1, 1, 1, 0.67 \}$

Profit = Profit + ($6\times1 + 10\times1 + 18\times1 + 15\times1 + 3\times1 + 5\times0.67 + 7\times0$)

Profit = 0 + $6+10+18+15+3+3.35$

$\boxed{\text{Profit} = 55.35}$ Ans

JOB SEQUENCING :- Given a set of n jobs to execute each on of which takes a unit time to process. At any time Instant we can do only one job. It is required to sequence the job execution such that profit is maximum.

JOB_GREEDY (d, p)

1. for t = 1 to max deadline repeat step-2 to step-3
2. Prepare a set of jobs with d[i] = t
3. Select job with maximum profit
4. Exit.

Eg- Schedule following jobs so as to have max. profit

	Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7	Job 8
Deadline	2	1	3	2	4	1	3	3
Profit	10	15	8	20	9	12	16	11

Sol^n Using the algorithm

→ At t=1, d=1 the jobs with deadline are J2 & J6
Compare their profit p2 > p6, select J2
J = { J2 }

→ At t=2 for d=t=2
J1 & J4 have this deadline then compare their profit p1 < p4, select J4
J = { J2, J4 }

→ At t=3 for d=t=3
J3, J7 & J8 have this deadline
Maximum profit p7 hence select J7
J = { J2, J4, J7 }

→ At t=4 for d=t=4
Only J5 has this deadline so select J5
J = { J2, J4, J7, J5 }

Total profit $p2 + p4 + p7 + p5 = 15 + 20 + 9 + 16$
$= 60$

Optimal Merge Pattern :-

Given n file of variable length m1, m2 --- mn. We have to find an order in which these file be merge two at a time. So that total operations are minimum.

Algo -

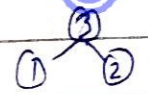
1. Sort file in increasing order of length.
2. Merge first 2 file, replaces them with resulting file in the list.
3. Repeat from step-① till list has only one file
4. Exit

Q. 8, 2, 9, 1, 12, 10, 18, 15, 14, 17

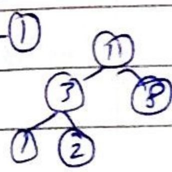
Ans -

Step-① Sort in increasing order of length :-
1, 2, 8, 9, 10, 12, 14, 15, 17, 18

Step-② Merge first two file
3, 8, 9, 10, 12, 14, 15, 17, 18



Repeat step-①

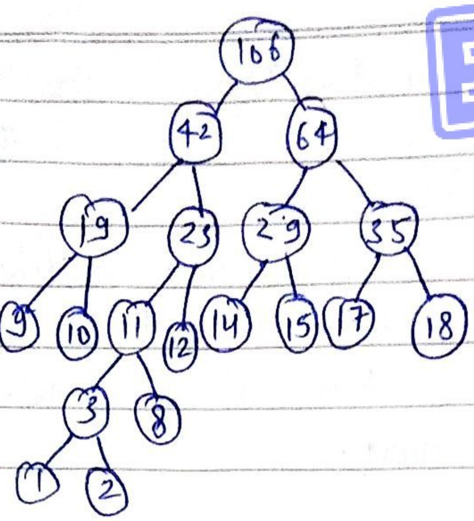


Step-③ 3, 8, 9, 10, 12, 14, 15, 17, 18
11, 9, 10, 12, 14, 15, 17, 18
→ 9, 10, 11, 12, 14, 15, 17, 18
19, 11, 12, 14, 15, 17, 18
→ 11, 12, 14, 15, 17, 18, 19
23, 14, 15, 17, 18, 19
→ 14, 15, 17, 18, 19, 23
29, 17, 18, 19, 23
→ 17, 18, 19, 23, 29
35, 19, 23, 29
→ 19, 23, 29, 35

42, 29, 35
→ 29, 35, 42
64, 42
→ 42, 64
106 Ans

DATE : / /

PAGE NO. :



Minimum Spanning tree :— A spanning tree is sub-graph of G is a tree and contains all the vertices of G. If graph is weighted every spanning tree will have a cost attached to it.

A tree of a graph with minimum cost is MST.

Application ⇒ (i) Telephone Network
(ii) Finding Airline routes

Greedy approach
(i) Kruskal's algo.
(ii) Prim's algo.

Problem definition :- Given a weighted connected graph G, find a spanning tree T such that sum of weights of all edge in T is minimum.

(i) Kruskal's algo :—

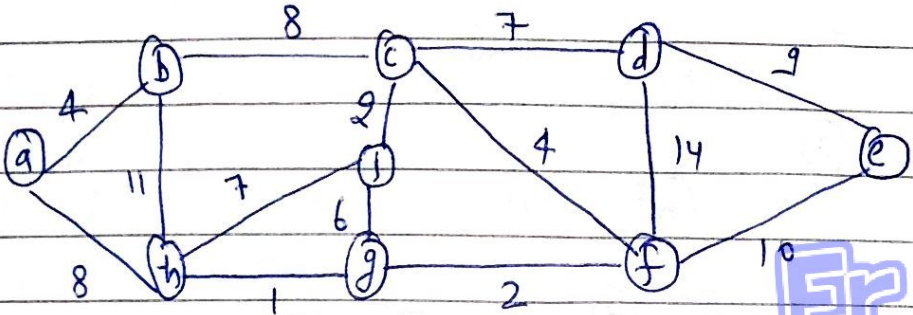
The Greedy choice is to pick the smallest weight edge that doesn't cause a cycle in MST constricted so far.

Steps of K'A :⇒
(i) Sort all edges in non-decreasing order.
(ii) Pick smallest edge. If cycle is not formed include edge

(iii) Repeat step-② until there is (V-1) edges in spanning tree.

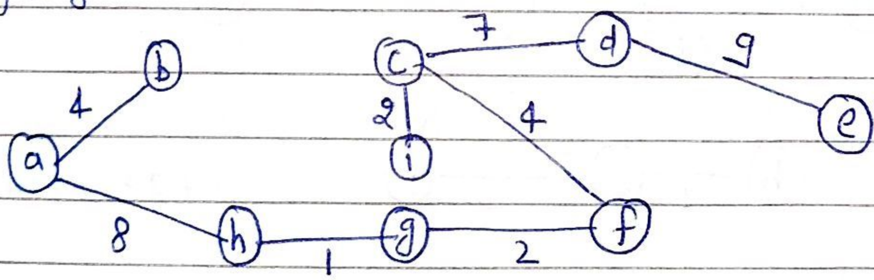
$O(V \log E)$

Eg-



(h,g) = 1
(g,f) = (i,c) = 2
(a,b) = (c,f) = 4
(g,i) = 6
(h,i) = (c,d) = 7
(a,h) = 8

(d,e) = 9
(f,e) = 10
(b,h) = 11
(d,f) = 14



(ii) PRIM's Algo :- The idea is using key values is to pick minimum weight edge from cut.

Algo ⇒

(i) Create a set mstset that keep track of vertices already included in MST.

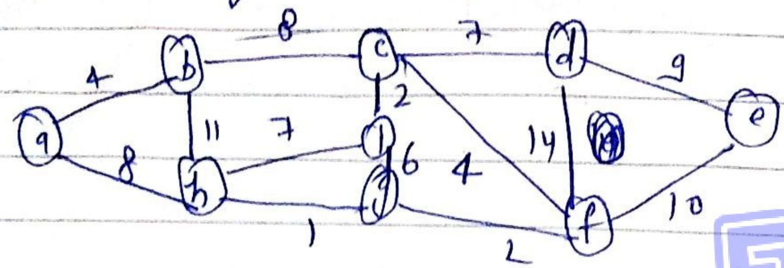
(ii) Assign key value to all vertices in input graph.

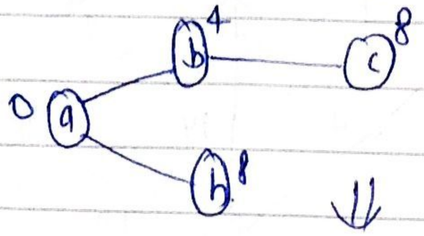
(iii) Which mstset doesn't include all vertices

(a) Pick vertex u which is not in mstset

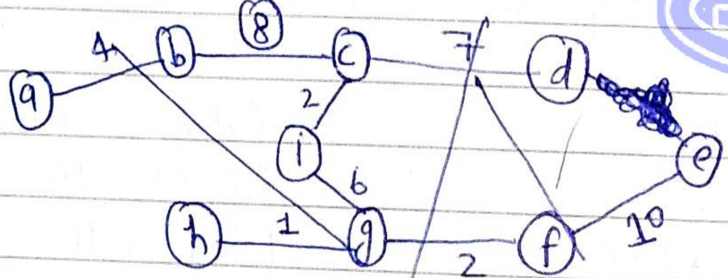
DATE : / /
PAGE NO. :

(b) Include u to mstset
(c) Update key value of all adjacent vertices of u.

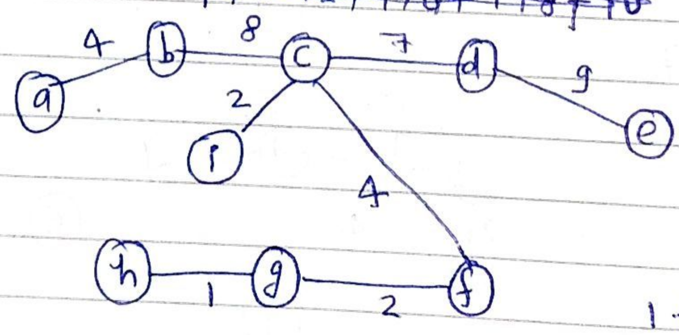
Q. 

Ans 

⇓

⇒ 

~~1+2+2+4+6+7+8+10~~

⇒ 

1+2+2+4+4+7+8+9 = 37 Ans

(iii) Dynamic Programming:— Solved all possible smaller problem and combines them. During this combining process only a subset of smaller problem is used but this set keeps changing for a better solution. This strategy is called Dynamic

General Algo:

[A] Determine the optimal substructure of the problem
[B] Define the value of the objective function recursively

that is how the value of objective function will be calculate for the problem and how will they be combined.

(c) Compute the value of objective function for the problem and store the result.

(d) Compute the final value called optimal sol^n.

Matrix Chain Multiplication (MCM) :—

Definition: Given a Chain of matrix (A_1 A_2 A_3 - - . A_n) to be multiplied, we need to find a parentthesization of sequence such that no. of operation perform and minimize

Algorithm :—

M_C_ORDER(P)

1. $n \leftarrow length[P] - 1$
2. for $i \leftarrow 1$ to n set $m[i,j] = 0$
3. For $l = 2$ to n
4. For $i = 1$ to $n - l + 1$ set $j = i + l - 1$

   $m[i,j] = \infty$
5. for $K = 1$ to $j - 1$ set $q = m[i, K] + m[k+1, j] + P_{i-1} P_j P_K$

   if $q < m[i, j]$ then $m[i,j] = q$

   $S[i,j] = K$
6. Return m & s and Exit

Q. {5, 10, 3, 12, 5, 50, 6} MCM

Sol^n $P_0 = 5$, $P_1 = 10$, $P_2 = 3$, $P_3 = 12$, $P_4 = 5$, $P_5 = 50$, $P_6 = 6$

$n = length[P] - 1 = 6$

$i = j = m[i,j] = 0$

$m_{11} = m_{22} = m_{33} = m_{44} = m_{55} = m_{66} = 0$

$m[i,j] = \infty$

$m_{ij} = i \leq k < j \{ m[i,k] + m[k+1,j] + P_{i-1} P_j P_k \}$

→ $i=1,\ j=2$

$m_{12} = 1 \leq 1 < 2 \{ m[1,1] + m[2,2] + P_0 P_2 P_1 \}$

$= 0 + 0 + 5 \times 3 \times 10 = 150$

$\cancel{m_{23} = 2 \leq 1 \leq 3 \{ m[2,1}$

→ $i=2,\ j=3$

$m_{23} = 2 \leq 2 \leq 3 \{ m[2,2] + m[3,3] + P_1 P_3 P_2 \}$

$= 0 + 0 + 10 \times 12 \times 3 = 360$

→ $i=3,\ j=4$

$m_{34} = 3 \leq 3 \leq 4 \{ m[3,3] + m[4,4] + P_2 P_4 P_3 \}$

$= 0 + 0 + 3 \times 5 \times 12 = 180$

→ $i=4,\ j=5$

$m_{45} = 4 \leq 4 < 5 \{ m[4,4] + m[5,5] + P_3 P_5 P_4 \}$

$= 0 + 0 + 12 \times 50 \times 5 = 3000$

→ $i=5,\ j=6$

$m_{56} = 5 \leq 5 < 6 \{ m[5,5] + m[6,6] + P_4 P_6 P_5 \}$

$= 0 + 0 + 5 \times 6 \times 50 = 1500$

→ $i=1,\ j=3$

$m_{13} = 1 \leq 1 < 3 \{ m[1,1] + m[2,3] + P_0 P_3 P_1 \}$

$= 0 + 360 + 5 \times 12 \times 10 = 960$

→ $i=2,\ j=4$

$m_{24} = 2 \leq 2 < 4 \{ m[2,2] + m[3,4] + P_1 P_4 P_2 \}$

$= 0 + 180 + 10 \times 5 \times 3 = 330$

→ $i=3,\ j=5$

$m_{35} = 3 \le 3 < 5 \{ m[3,3] + m[4,5] + P_2 P_5 P_3 \}$

$= 0 + 3000 + 3 \times 50 \times 12 = 4800$

→ $i=4,\ j=6$

$m_{46} = 4 \le 4 < 6 \{ m[4,4] + m[5,6] + P_3 P_6 P_4 \}$

$= 0 + 1500 + 12 \times 6 \times 5 = 1860$

→ $i=1,\ j=4$

$m_{14} = 1 \le 1 < 4 \{ m[1,1] + m[2,4] + P_0 P_4 P_1 \}$

$= 0 + 330 + 5 \times 5 \times 10 = 580$

→ $i=2,\ j=5$

$m_{25} = 2 \le 2 < 5 \{ m[2,2] + m[3,5] + P_1 P_5 P_2 \}$

$= 0 + 4800 + 10 \times 50 \times 3 = 6300$

→ $i=3,\ j=6$

$m_{36} = 3 \le 3 < 6 \{ m[3,3] + m[4,6] + P_2 P_6 P_3 \}$

$= 1860 + 3 \times 6 \times 12 = 2076$

→ $i=1,\ j=5$

$m_{15} = 1 \le 1 < 5 \{ m[1,1] + m[2,5] + P_0 P_5 P_1 \}$

$0 + 6300 + 5 \times 50 \times 10 = 8800$

→ $i=2,\ j=6$

$m_{26} = 2 \le 2 < 6 \{ m[2,2] + m[3,6] + P_1 P_6 P_2 \}$

$= 0 + 2076 + 10 \times 6 \times 3 = 2256$

→ $i=1,\ j=6$

$m_{16} = 1 \le 1 < 6 = \{ m[1,1] + m[2,6] + P_0 P_6 P_1 \}$

$= 0 + 2256 + 5 \times 6 \times 10$

$= 2256 + 300$

$\boxed{m_{16} = 2556}$

Longest Common Subsequence :— The problem is to find a common substring of string X and Y of maximum length.

LCS problem deals with string with the comparision

DATE : / /

PAGE NO. :

of string.

Working :-

~~Algorithm~~ Application :-

(i) It is used in DNA matching.

LCS problem can be solved using dynamic programming

1. Structure of optimal solution
2. A Recursive problem

$$c[i,j] = \begin{cases} 0 & , \text{if } i=0 \text{ or } j=0 \\ c[i-1, j-1]+1 \; \nwarrow & , \text{if } i,j>0 \; \& \; x_i = y_j \\ \max(c[i,j-1] \uparrow, c[i-1,j] \leftarrow) & , x_i \neq y_j \end{cases}$$

3. Computing the cost of optimal solution (length of LCS)
4. Optimal Solution.

Algo :-

```
m = length(x)
n = length(y)
for i=1 to m do
    c[i,0] = 0
for j=1 to n
    c[0,j] = 0
for i = 1 to m do
    for j=1 to n do
        if xi = yj
            c[i,j] = c[i-1,j-1] + 1
            B[i,j] = 'D'   '↖'
else if c[i-1,j] ≥ c[i,j-1]
            c[i,j] = c[i-1, j] + 1
            B[i,j] = 'U'   '←'
else    c[i,j] = c[i,j-1]
        B[i,j] = 'L'   '↑'
return c & B
```

Ex- $X = \langle MLNOM \rangle$
$Y = \langle MNOM \rangle$ Find out the LCS

Sol^n. length [X] = 5
length [Y] = 4
$m = 5, \; n = 4$

	$c[i,j]$ y_j	1	2	3	4
x_i	0	M	N	O	M
0	0	0	0	0	0
1 M	0	1 ↖	1 ←	1 ←	1 ↖
2 L	0	1 ↑	1 ←	1 ←	1 ←
3 N	0	1 ↑	2 ↖	2 ←	2 ←
4 O	0	1 ↑	2 ↑	3 ↖	3 ←
5 M	0	1 ↖	2 ↑	3 ↑	4 ↖

$c[1,2] = \max\{c[0,2], c[1,1]\}$
$\max\{0, 1\}$

Ans - MNOM

If one of the sequence is of 0 length
put $c[i,j] = 0$
if $x_i = y_j$ $c[i,j] = c[i-1, j-1] + 1$, $B[i,j] =$ '↖'
elseif $c[i-1,j] \geq c[i,j-1]$
$c[i,j] = c[i-1,j] + 1$, $B[i,j] =$ '←'
else $c[i,j] = c[i,j-1]$, $B[i,j] =$ '↑'

0/1 knapsack problem: - In 0/1 knapsack problem, the list of items are indivisible, that means either we take the item or discard it. Or we can say that item cannot be broken in small pieces, they can only be either rejected or accepted from the list.

Algo:- 0-1-knapsack (v, w, n, W)
1. for w = 0 to W do
2. c[0, w] = 0
3. for i = 1 to n do

```
c[i, 0] = 0
for w = 1 to W do
if wi ≤ W then
  if   vi + c[i-1 , W-wi] then
     c[i,w] = vi + c[i-1 , W-wi]
     else    c[i,W] = c[i-1, W]
else
    c[i,w] = c[i-1, W]
```

Eg- $W=6$, $n=3$, $w_i = \{2,3,3\}$, $v_i = \{1,2,4\}$

Soln-

Table c will contain n rows and W columns

i \ W	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1
2	0	0	1	2	2	3	3
3	0	0	1	4	4	5	6

1. $c[0,W] \rightarrow 0$ for $w=0$ to W do
2. for $i=1$, check for each value of W

if $w_i \le W$ X

→ $w_1 > W$ $\quad 2 > 1$

$c[1,1] = c[i-1, W] = c[0,1] = 0$

→ $W=2$, $w_1 = 2$

$w_i = W$ then

$v_i + c[i-1, W-w_i] > c[i-1, W]$

$1 + c[0,0] > c[0,2]$

$1 > 0$ then

$c[1,2] = 1$

→ $W=3$, $w_1 = 2$

$w_i \le W$ then

$V_i + c[i-1, W-w_i] > c[i-1, W]$

$1 + c[0,1] > c[0,3]$

$1 > 0$ then

$c[1,3] = 1$

→ $W = 4$, $w_1 = 2$

$w_i \leq W$ then

$1 + c[0,2] > c[0,4]$

$1 > 0$ then $c[1,4] = 1$

→ $W = 5$, $w_1 = 2$

$w_i \leq W$ then

$1 + c[0,3] > c[0,5]$

$1 > 0$ then $c[1,5] = 1$

→ $W = 6$, $w_1 = 2$

$w_i \leq W$ then

$1 + c[0,4] > c[0,6]$

$1 > 0$ then $c[1,6] = 1$

for $i = 2$, check for each value of W

→ $W = 1$, $w_2 = 3$

$w_i > W$

$c[2,1] = c[1,1] = 0$

→ $W = 2$, $w_2 = 3$

$w_i > W$

$c[2,2] = c[1,2] = 1$

→ $W = 3$, $w_2 = 3$

$w_i = W$ then

$V_i + c[i-1, W-w_i] > c[i-1, W]$

$2 + c[1,0] > c[1,3]$

$2 > 1$ then $c[2,3] = 2$

→ $W = 4$, $w_2 = 3$, $w_i \leq W$ then

$2 + c[1,1] > c[1,4]$

$2 > 1$ then $c[2,4] = 2$

$W=5,\ w_2=3$ $\quad w_i \le w$ then $\quad 2 + c[1,2] > c[1,5] \Rightarrow 3 > 1$

$c[2,5]=3$ $\quad$ similarly $\Rightarrow$ $c[2,6]=3$

for $i=3$ $\quad w_3=3,\ W=1$ $\quad c[3,1]=0$

$c[3,2]=1,\quad c[3,3]=4,\ c[3,4]=4$

$c[3,5]=5,\quad c[3,6]=6$

Eg- $\quad W(M)=7,\quad n=7$

$w_i = \{5,3,4,1,2\},\quad V_i = \{15,12,16,8,10\}$

i \ W	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	15	15	15
2	0	0	0	12	12	15	15	15
3	0	0	0	16	16	16	16	28
4	0	8	8	8	20	24	24	28
5	0	8	10	18	18	24	30	34

1. $c[0,W] \rightarrow 0$
2. for $i=1$, check for each value of W

$\rightarrow$ $W=1,\ w_1=5$

$w_i > W$ then $c[1,1] = c[0,1] = 0$

$\rightarrow$ $W=2,\ w_1=5$

$w_i > W$ then $c[1,2] = c[0,2] = 0$

$\rightarrow$ $W=3,\ w_1=5$

$w_i > W$ then $c[1,3] = c[0,3] = 0$

$\rightarrow$ $W=4,\ w_1=5$ $\quad c[1,4] = 0$

$\rightarrow$ $W=5,\ w_1=5$

$w_i = W$ then $v_i + c[i-1, W-w_i] = c[i, W]$

DATE:

PAGE NO.

$15 + c[0,0] > c[0,5]$

$15 > 0$ then

$c[1,5] = 15$

$W = 6$, $w_1 = 5$

$w_i \leq W$ then $15 + c[0,1] > c[0,6]$

$15 > 0$

$c[1,6] = 15$

$W = 7$, $w_1 = 5$

$w_i \leq W$ then $15 + c[0,2] > c[0,7]$

$15 > 0$

$c[1,7] = 15$

for $i = 2$

$W = 1$, $w_2 = 3$

$w_i > W$ then $c[2,1] = c[1,1] = 0$

$W = 2$, $w_2 = 3$

$w_i > W$ then $c[2,2] = c[1,2] = 0$

$W = 3$, $w_2 = 3$

$w_i = W$ then $V_i + c[i-1, W - w_i] > c[i-1, W]$

$12 + C[1,0] > c[1,3]$

$12 > 0$ then $c[2,3] = 12$

$W = 4$, $w_2 = 3$

$w_i \leq W$ then $12 + c[1,1] > c[1,4]$

$12 > 0$ then $c[2,4] = 12$

$W = 5$, $w_2 = 3$

$w_i \leq W$ then $12 + c[1,2] > c[1,5]$

$12 > 15$

$c[2,5] = c[1,5] = 15$

$W = 6$, $w_2 = 3$

$w_i \leq W$ then $12 + c[1,3] > c[1,6]$

$12 > 15$ $c[1,6] = 15$

$w = 7$, $w_2 = 3$ $c[1,7] = 15$

DATE: / /

PAGE NO.:

for $i=3$,

→ $W=1,\ w_3=4$

$w_i > W$ then $c[3,1] = c[2,1] = 0$

→ $W=2,\ w_3=4$

$w_i > W$ then $c[3,2]=0$

→ $W=3,\ w_3=4$

$w_i > W$ then $c[3,3]=12$

→ $W=4,\ w_3=4$

$w_i = W$ then $v_i + c[i-1, W-w_i] > c[i-1, W]$

$16 + c[2,0] > c[2,4]$

$16 > 12$

$c[3,4]=16$

→ $W=5,\ w_3=4$

$w_i \le W$ then $16 + c[2,1] > c[2,5]$

$16 > 15$

$c[3,5]=16$

→ $W=6,\ w_3=4$

$w_i \le W$ then $16 + c[2,2] > c[2,6]$

$16 > 15$

$c[3,6]=16$

→ $W=7,\ w_3=4$

$w_i \le W$ then $16 + c[2,3] > c[2,7]$

$16 + 12 > 15$

$28 > 15$ then $c[3,7]=28$

for $i=4$,

→ $W=1,\ w_4=1$, $w_i = W$ then

$v_i + c[i-1, W-w_i] > c[i-1, W]$

$8 + c[3,0] > c[3,1]$

$8 > 0$ then $c[4,1]=8$

→ $W=2,\ w_4=1$ $w_i \le W$ then

$8 > 0$ then $c[4,2]=8$

DATE : / /
PAGE NO. :

→ $W=3,\ w_4=1$, $w_i \le W$ then
$8 > 0$ then $c[4,3]=8$

→ $W=4,\ w_4=1$, $w_i \le W$ then
$8+c[3,3] > c[3,4]$
$8+12 > 16$ ~~then c[i,w] = c[i-1, w] = c[3,4]~~
$c[4,4] = 20$

→ $W=5,\ w_4=1$, $w_i \le W$ then
$8+c[3,4] > c[3,5]$
$24 > 16$ then $c[4,5]=24$

→ $W=6,\ w_4=1$, $w_i \le W$ then
$8+c[3,5] > c[3,6]$
$24 > 16$ then $c[4,6]=24$

→ $W=7,\ w_4=1$, $w_i \le W$ then
$8+16 > c[3,7]$
$24 > 28$ then $c[4,7]=c[3,7]=28$
$c[4,7]=28$

for $i=5$,

→ $W=1,\ w_5=2$
$w_i > W$ then $c[i-1,W] = c[4,1]=8$
$c[5,1]=8$

→ $W=2,\ w_5=2$
$w_i = W$ then $10+c[4,0] > c[4,2]$
$10 > 8$ then
$c[5,2]=10$

→ $W=3,\ w_5=2$
$w_i \le W$ then $10+c[4,1] > c[4,3]$
$18 > 8$ $c[5,3]=18$

→ $W=4,\ w_5=2$
$w_i \le W$ then $10+8 > 16$ $c[5,4]=18$

→ $W=5,\ w_5=2$
$w_i \le W$ then $10+8 > 24$ $c[5,5]=c[4,5]=24$

→ $W = 6, \; w_5 = 2$

$w_i \le W$ then $\quad 10 + c[4,4] > c[4,6]$

$10 + 20 > 24$

$c[5,6] = 30$

→ $W = 7, \; w_5 = 2$

$w_i \le W$ then $\quad 10 + c[4,5] > c[4,7]$

$10 + 24 > 28$

$34 > 28$

then $\quad c[5,7] = 34$

$\boxed{\text{Ans} - 34}$