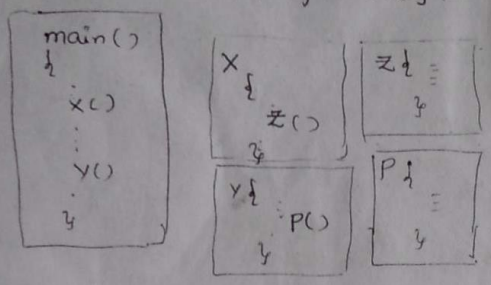
GATE NOTES BY DEVA SIR

| Contact: +91-9911691155

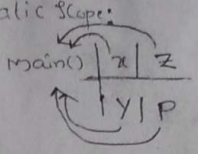
Programming & Data Structure:

SIR'S

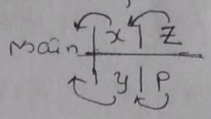
* Scoping [Static & Dynamic]:



*Static Scope:



*Dynamic Scope:



* Palametes passing:

Call by Value: "Actuals copied to formal", but not formal to actual

Call by reference: "Actuals & formals are uses same addressspace".

Call by Value Result: "Actual copied to formal & Formal copied to Actuals" (Restore)

Call by Result: "Actuals not copied to formal", but Formal copied to Actual".

Call by Constant: Formal values are never changed. 45

Call by Text: [Replaces formals by text appeared in function of main.] Actuals appeared as formula (Expressions), replaces with (formal) variable names in formals function [called function]

Call

If Actual appeared directly, then uses same name in the formals function.

Call by Name: Same as Call by text.

Call by need: Memory allocates only if formals are used in function

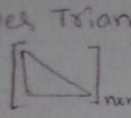
* "Call by reference" & "Call by name" palametes passing are gives different result when passing address of an array as palametes.

* "Call by Reference" & "Call by value Result" palametes passing gives different results when the presence of loops.

* Call by name: 1) If the actual palametes is Scalar variable, then same as call by reference

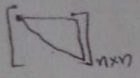
2) If the actual palametes is constant expression, then same as Call by value

Lower Triangular : $n + \frac{n^2-n}{2} = \frac{n(n+1)}{2}$



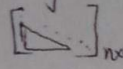
- Row major : $(j-1) + \frac{i(i-1)}{2}$
- Column major : $(i-j) + \left[(j-1)n - \frac{(j-1)(j-2)}{2}\right]$

Upper Δ^{le} : $n + \frac{n^2-n}{2} = \frac{n(n+1)}{2}$



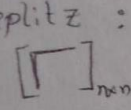
- Row major : $(j-i) + \left[(i-1)n - \frac{(i-1)(i-2)}{2}\right]$
- Column major : $\left[(i-1) + \frac{j(j-1)}{2}\right]$

Strictly lower Δ^{le} : $\frac{n^2-n}{2}$

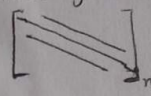


- Row major : $(j-1) + \frac{(i-1)(i-2)}{2}$
- Column major : $(i-1) + \frac{(j-1)(j-2)}{2}$

Toeplitz : $2n-1$



Tridiagonal : $3n-2$



Row : $2i+j-3$
Column : $i+2j-3$

C-Matrix : $3n-2$

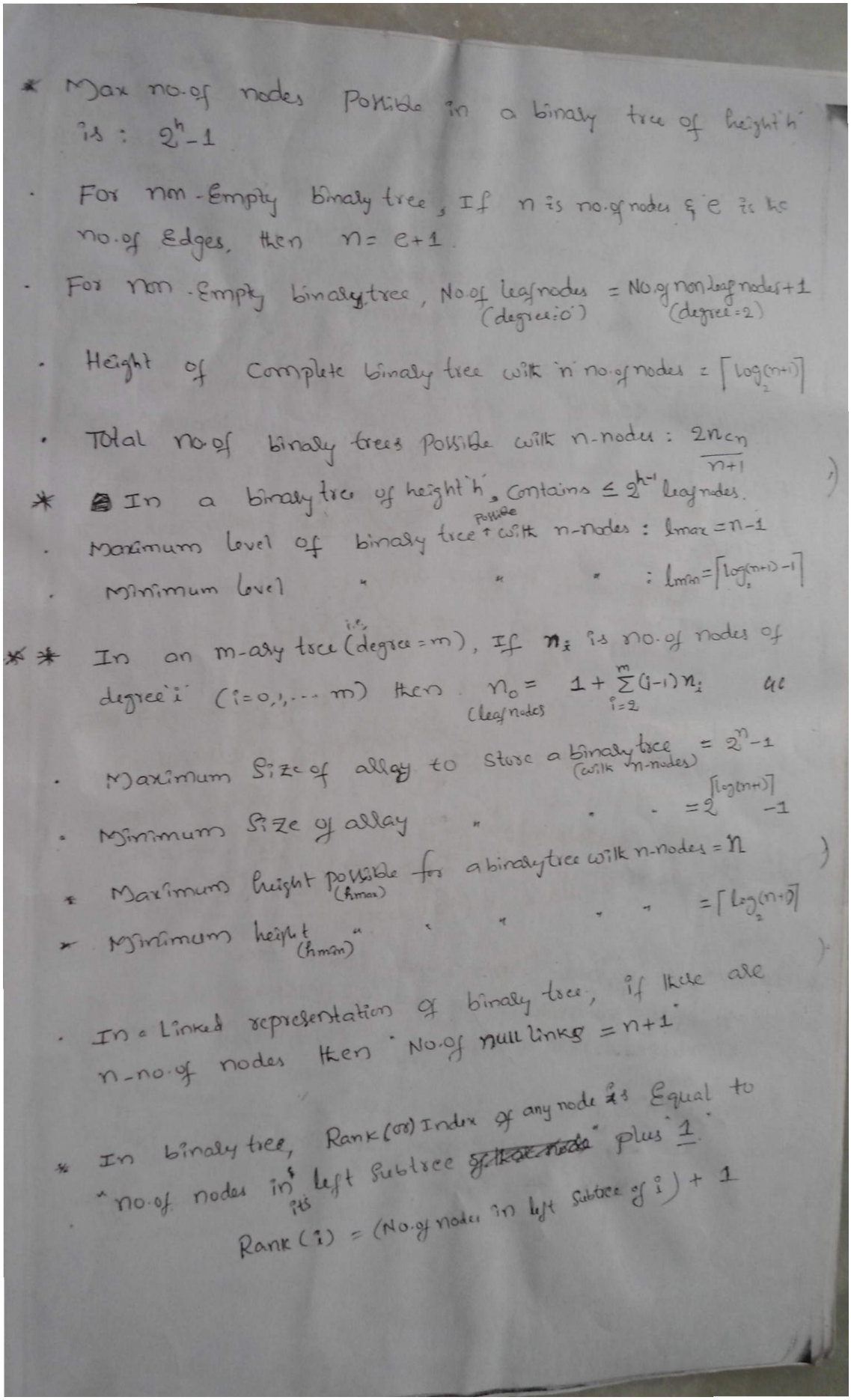
X-matrix : $2n-1$

Stack : (LIFO)
Push ⇒ Increment & Store.
Pop ⇒ Load & Decrement

- Applications of stack : Subroutines, Evaluation of Expressions (Postfix), Converting Arithmetic Expressions from infix to polish [Prefix/Postfix], Searching trees & graphs, Creating Expression trees, balancing of Symbols, Recursive functions, Backtracking algorithms, DFS trees.
- Queue : Front ⇒ Deletion of an Element
  Rear ⇒ Insertion of an Element.
- Applications of Queues : Multiprogramming, Resource Sharing, BFS, CPU Scheduling, Radix Sort, Real time system & storing operands in Evaluation of prefix Expressions.

* Converting Expressions :
  1) Prefix ⇒ Postfix : Prefix $\xrightarrow{\text{[Right to left]}}$ Expression Tree ⇒ Postfix
     Ex:- *+a*bc ←
  2) Postfix ⇒ Prefix : Postfix $\xrightarrow{\text{[left to Right]}}$ Expression Tree ⇒ Prefix.
     Ex:- bca*a+ →

** A tree can never be Empty, but binary tree may be Empty.



* Max no. of nodes possible in a binary tree of height 'h' is : $2^h - 1$

- For non-Empty binary tree, If n is no. of nodes & e is the no. of Edges, then $n = e + 1$.
- For non-Empty binary tree, No. of leaf nodes (degree: 0) = No. of non leaf nodes (degree = 2) + 1
- Height of Complete binary tree with 'n' no. of nodes = $\lceil \log_2(n+1) \rceil$
- Total no. of binary trees possible with n-nodes : $\frac{2nCn}{n+1}$

* In a binary tree of height 'h', contains $\leq 2^{h-1}$ leaf nodes.

- Maximum level of possible binary tree with n-nodes : $l_{max} = n - 1$
- Minimum level " " " : $l_{min} = \lceil \log_2(n+1) - 1 \rceil$

** In an m-ary tree (i.e. degree = m), If n_i is no. of nodes of degree 'i' ($i = 0, 1, \ldots m$) then n_0 (leaf nodes) $= 1 + \sum_{i=2}^{m} (i-1) n_i$ ae

- Maximum Size of array to store a binary tree (with n-nodes) $= 2^n - 1$
- Minimum Size of array " " " $= 2^{\lceil \log(n+1) \rceil} - 1$
- Maximum height possible (hmax) for a binary tree with n-nodes = n
- Minimum height (hmin) " " " " $= \lceil \log_2(n+1) \rceil$

- In a Linked representation of binary tree, if there are n-no. of nodes then No. of null links $= n + 1$

* In binary tree, Rank (or) Index of any node is Equal to "no. of nodes in its left Subtree plus 1".

Rank (i) = (No. of nodes in left Subtree of i) + 1

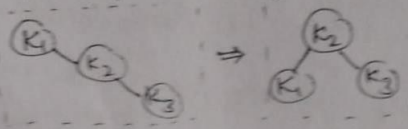
* Binary Sort : Inorder traversal of binary search tree called "binary sort". [Sorted order of data in ascending order]

(BST)

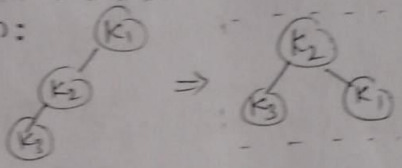
- In binary search tree : * Recursion of left child from the root gives the minimum element &
  * Recursion of right child from the root gives the maximum element.
- complexity for B.S.T [Search, Insert & Delete] : Avg case = $O(\log_2 n)$ | To visit root : $\Theta(1)$
  worst case = $O(n)$

* AVL tree [height balanced binary search tree] [balance factor (bf) = 0, or 1, or -1]

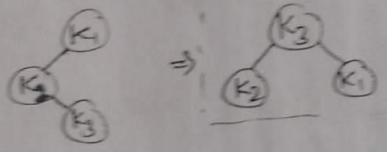
1) Right Rotation : [R-R] (Single)



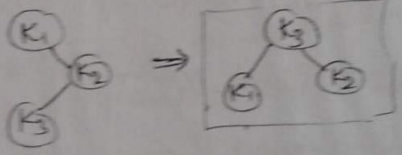
2) Left - Rotation : (Left-Left Rotation) (Single)



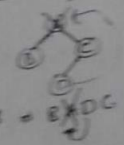
3) Left-Right Rotation : (Double Rotation) (LR)



4) Right-Left (RL) Rotation : [Double Rotation]



* "Deletion of a node" in Binary Search tree :
  1) Deletes that node; if it has no children
  2) Deletes that node & Replaces with child; if it has only one child.
  3) Deletes that node & Replaces with "inorder traversal of that node"; if it has more than one child.



- Heaptree : Complete binary tree with heap property (Min heap / max heap)
- Heap Sort : By deleting the root node from heap tree & placing the deleted node in o/p, until all the nodes are deleted. It gives sorted in ascending (Min heap) or descending (max heap) order.
- Priority Queue can be implemented using "Circular array", "Linked list" & "Heap tree".
- For any node 'i' (i=1, 2, ... n), ~~the left child of the node~~ In heap array, the Left child is at 2i location and Right child is at (2i+1)th location.
- In heap array, Parent of any node 'i' (i = 1 to n) is at : $\lfloor i/2 \rfloor$
- Balance factor (bf) of AVL tree (height balanced BST) is:

$$|bf| = |h_L - h_R| \leq 1$$

- Full binary tree : At each level, max no. of nodes Contained in a binary tree.
- Complete binary tree : At each level, max no. of nodes Contained, & Except at last level (left to right) possibly in a binary tree
- Strict binary tree : At each node, the no. of childs may Contained Exactly either 0 or 2 children.
- A Complete n-ary tree is one in which Every node has 0 or n sons. If 'x' is the no. of internal nodes of a Complete n-ary tree, the no. of leaves in it is : $x(n-1)+1$
- Let T(n) be the no. of different binary search trees on n-distinct Elements, Then $T(n) = \sum_{k=1}^{n} T(k-1)\,T(n-k+1)$

Algorithms:

- If $f(n) \leq K \cdot g(n)$ then $f(n)$ is $O(g(n))$ $(=)$
- If $f(n) \geq K \cdot g(n)$ then $f(n) = \Omega(g(n))$
- If $K_1 \cdot g(n) \leq f(n) \leq K_2 \cdot g(n)$ then $f(n) = \Theta(g(n))$
- If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ then $f(n) = o(g(n))$ [i.e., $f(n) < g(n)$]
- If $\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$ then $f(n) = \omega(g(n))$ [i.e., $f(n) > g(n)$]

* Greedy Approach:

It works in stage wise by considering one i/p at a time. [based on criteria]

1) Subset paradigm: Order determined with selection procedure - by "optimization" technique.
2) Ordering Paradigm: Order determined by "decision" that - already made but not selection.

* Knapsack problem: [$O(n) + O(n \log n)$] (without sorting, with sorting)

- Maximize $\sum_{1 \leq i \leq n} P_i x_i$, Subject to $\sum_{1 \leq i \leq n} w_i x_i \leq m$ & $0 \leq x_i \leq 1$

1) Maximum profit, 2) Minimum weight, 3) Maximum profit per unit weight (P/w)

* Job Sequencing with deadlines: [$\Theta(n^2)$]

It gives maximum profit, by taking the sum of profits (maximum) of jobs (J), which are completed in its deadline. [$\sum_{i \in J} P_i$]

* Minimum Cost Spanning trees:

- Prim's Algorithm: [$O(n^2)$]
  - with heap: $O(e \log n)$
  - with fibonacci heap: $O(e + \log n)$
- Kruskal's Algorithm [$O(e \log e)$]
  - with heap: $O(e \log n)$
- Optimal Randomized Algorithm (minimum cost spanning tree) [$O(V+E)$]
- Vertex Splitting. [$\Theta(n)$] ....

* Optimal Storage on tapes: [O(n log n)]
  - No. of permutations for ordering n-programs = n!
  - (Minimize) Mean Retrieval Time (MRT) = $\sum_{j=1}^{n} \left(\sum_{k=1}^{j} l_{ik} \right)$

[where 'i' is the i^{th} permutation order, which contains the lengths of program in "increasing order"]

Ex:- $(l_1, l_2, l_3) = (5, 10, 3) \Rightarrow$ Increasing order = 3, 5, 10

MRT = 3 + (3+5) + (3+5+10) = 29

Optimal order is: (3, 5, 10)

  - Large files with Smaller records ⇒ Insertion Sort uses.
  - Small files with Larger records ⇒ Selection Sort uses.

* Optimal Merge Pattern [Huffman Coding] [O(n log n)].
  - The no. of bits per message = $\sum$ (frequency of External node 'i' (q_i)) * (No. of bits required for q_i)
  - weighted External path length = $\sum q_i d_i$ ↳ distance from root to External node 'i'.

[eg: Equivalent decimal]

* Coins Change problem
* Single Source Shortest path [Dijkstra's Algorithm] [O(e log n)]
  (with +ve weights) [with heap - O(e + n log n)]

~~* Coins Change problem~~

* Union & find operations for disjoint sets:
  - Time Complexity for union : O(n)
  - Time Complexity for find(i) : O(i) ↳ element
  - To find all the n elements at different 'i' levels the time Complexity become : $[O(\sum_{i=1}^{n} i) = O(n^2)]$

Dynamic Programming:

It is the Sequence of decisions by "Principle of optimality."

- Multistage graph: [O(n+e)]

$Cost(i,j) = \min_{\text{from } j^{th} \text{ node}} \{ c(j,l) + Cost(i+1,l) \}$ (i^{th} stage, $l \in V_{i+1}$)

$V_0, V_1, V_2 \ldots V_{n-1}, V_n$ (1, n) $\Rightarrow$ $V_1 = d(1, V_0) = d(1,1)$, $V_2 = d(2, V_1) = d(2, d(1,1))$, $V_3 = d(3, V_2)$

- All pairs Shortest path: (Floyd's-Warshall's) [$O(n^3)$] [With heap: $O(n^2 \log n)$]

$A^0(i,j) = C(i,j)$

$A^K(i,j) = \min \{ A^{K-1}(i,j), A^{K-1}(i,K) + A^{K-1}(K,j) \}$

* Single Source Shortest path (Bellman-ford) with +ve & -ve weights [$O(ne)$ with Adjacency list; $O(n^3)$ with Adjacency matrix]

$dist^1[u] = C[1,u]$

$dist^K[u] = \min \{ dist^{K-1}[u], \min_i \{ dist^{K-1}[i] + C(i,u) \} \}$

- Optimal Binary Search tree: [O(nlogn)] (Lexically ordered tree)

$j-i=1$, $j-i=2$, $j-i=3$ ...

$cost(i,j) = \min_{i<K\le j} \{ cost(i,K-1) + cost(K,j) + w(i,j) \}$

$w(i,j) = P(j) + q(j) + w(i,j-1)$

$w(i,i) = q(i)$ & $cost(i,i) = 0$

$Cost = \sum_{i}^{n} P_i * level\ a_i + \sum_{i}^{n+1} q_i (level\ E_i - 1)$ (Successful) (unsuccessful)

Shortcut: Select max frequency node as a root and similarly for left & right subtrees until all completes

String Editing [O(mn)]

0/1 knapsack problem [$O(n^2)$]

$S^0 = \{(0,0)\}$; $S_1^K = S^{K-1} + (P_K, \omega_K)$ ⇒ Addition

$S^K = S_1^K \cup S^{K-1}$ ⇒ Merging

$S^{i-1} \Rightarrow (P_j, \omega_j)$, $S_1^i \Rightarrow (P_K, \omega_K)$ } If $(P_j \le P_K)$ & $(\omega_j \ge \omega_K)$ then Discard (P_j, ω_j)

- Travelling Salesperson problem: $O(n^2 2^n)$, Exponential time (Milliam)

  |S|=0
  |S|=1
  |S|=2
  :

  $g(i,\phi) = c_{i1}$ & $g(i,S) = \min_{j \in S} \{ c_{ij} + g(j, S-\{j\}) \}$

- Reliability design : [O(nlogn)]

* Divide & Conquer :

Splitting the inputs into K distinct subsets ; $1 \le K \le n$

- $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$; $n > 1$

  $= 1$; $n = 1$

- Binary Search: $\left[\underset{\text{(time)}}{T(n)} = T(n/2) + C \text{ & } \underset{\text{(space)}}{S(n)} = n + \log n\right]$ [O(logn)]

- Minimum & Maximum items: $[T(n) = 2 \cdot T(n/2) + 2$ & $S(n) = C + n + \log n]$

  [O(n)] $= \frac{3n}{2} - 2$

- Merge Sort : (Stable) $\left[T(n) = 2 \cdot T(n/2) + cn \text{ & } S(n) = 2n + \log n + C\right]$ [O(nlogn)]

- Quick Sort (unstable) $\left[T(n) = 2 \cdot T(n/2) + n \text{ & } S(n) = 2n \right.$

  $= T(n-k) + T(k) + cn \quad \left. = n + \log n\right]$ $\begin{bmatrix} O(n \log n) - \text{avg} \\ O(n^2) - \text{worst} \end{bmatrix}$

- Strassen's Matrix Multiplication : $[T(n) = 7\,T(n/2) + an^2]$ $[O(n^{2.81}) = O(n^{\log_2 7})]$

  (7 - multiplications & 18 - additions required)

- Convex Hull : [O(nlogn)]

- Towers of hanoi

* Knapsack problem $[O(2^{n/2})]$

- Problem of multiply large integers.

→ $T(n) = a \cdot T(n/b) + f(n)$; $n \ge d$ $\quad a \ge 1, a > 0, b > 1, c > 0, \epsilon > 0, \delta > 1$

$= c$; $n < d$

1) If $n^{\log_b a} > f(n) \Rightarrow \boxed{T(n) = O(n^{\log_b a})}$ & $f(n) = O(n^{\log_b a - \epsilon})$

Else 2) If $n^{\log_b a} \log^k n > f(n) \Rightarrow \boxed{T(n) = O(n^{\log_b a} \log^{k+1} n)}$ & $f(n) = \Theta(n^{\log_b a} \log^k n)$

Else 3) $a \cdot f(n/b) \le \delta \cdot f(n) \Rightarrow \boxed{T(n) = O(f(n))}$ & $f(n) = \Omega(n^{\log_b a + \epsilon})$

* DFS/DFT : Preorder traversal of ordered tree [O(n+e)]
  - Stack & Backtracking technique
  - Computes the no of paths b/w two vertices [O(n³) - cost] (Adj matrix)
  - Computing a cycle & TO find articulation points
  - Biconnected Components & Strong Components
  - Eulers paths & No. of Connected Components
  - Whether the graph is Connected or not.
* BFS/BFT : Level by level [level order] traversal of ordered tree. [O(n+e)]
  - Queue & greedy technique.
  - Used for Topological Sort & Dijkstra's Algorithm
  - Prim's Algorithm.
  - Whether the graph is Connected or not.
  - No. of Connected Components
  - * Transitive Closure of a graph
  - Computing a cycle.

* Prim's Algorithm : [O(n²)] & [O(elogn) with heap]
  - It starts with vertex [min cost vertex]
  - Intermediate result "tree"
  - Greedy technique.

* Kruskal's Algorithm: [O(eloge)] & [O(elogn) with heap]
  - It starts with Edge [minimum cost Edge]
  - Intermediate result may either tree or forest
  - Greedy technique.

* Collision Resolution Techniques:
  1) Closed Hashing [Linear Probing] [primary clustering]
     $i = H(K) = (K \bmod h) + 1$ & Searches the locations $i, i+1, i+2, \ldots$ until free location is found (no collision occurs).
  2) Random probing: $i = (i+m) \bmod h + 1$ repeats until the Empty location found.
  3) Double Hashing: $i = H_1(K) = (K \bmod h) + 1$ & $m = H_2(K) = [K \bmod (h-4)] + 1$
     $i = [(i+m) \bmod h] + 1$ repeats until Empty location found
  4) Quadratic probing: $i = (i + x^2) \bmod h$ for $x = 1, 2, 3, \ldots$
     Probing sequence $i+1, i+2^2, i+3^2, \ldots$
     [Atmost $(h+1)/2$ positions probed]
  5) Open Hashing (Chaining) (Separate chaining).

* Ha

- Selection Sort (Exchange Sort) (Inspace) : $\frac{n(n-1)}{2}$: $O(n^2)$ $O(n^2)$
- Insertion Sort : " " "
- Bubble Sort (Inspace) : " " "
- Quick Sort (Extra $\log_2 n$ Entries) : $1.4\, n\log n$ $O(n\log n)$ $O(n^2)$
- Merge Sort (Stable) (Extra n-Entries) : $n\log n$ $O(n\log n)$ $O(n\log n)$
- Heap Sort (Inspace) (Elegant Sort) : $3n\log n$ $O(n\log n)$ $O(n\log n)$

* Shell Sort (Diminishing Increment) : $O(n)$ $O(n^{1.2})$

- Binary Sort : $O(\log n)$ $O(n\log n)$

* . Radix Sort [Bucket] (Extra space for link) : $O(m+n)$ $O(m+n)$
- Address Calculation Sort (Extra space for link) : $O(n)$ $O(n^2)$
- Comparision Based Sorting : $O(n\log n)$ $O(n^2)$
- Non-Comparision Based Sorting (Radix, Shell, Count, Binary, etc) : $O(n)$ $\leq O(n^2)$

- Linear Search : $O(n)$ $O(n)$
- Binary Search : $O(\log n)$ $O(\log n)$

* Hashing : 1) Division Method : $H(K) = K \bmod h$; if indices starts from 0 $[(K \bmod h)+1$: if starts from 1$]$
2) Mid Square Method : Selecting appropriate number of digits from the middle of square of key. (Even positions from right most digit). Key 1234
$[K=1234 \Rightarrow K^2 = 1522756 \Rightarrow H(K) = 525]$
3) Folding Method : (Chopping)
(a) Pure folding $[H(K) = K_1 + K_2 + K_3 + \dots + K_n]$ (Ignore carry)
$[K = 15227\,56 \Rightarrow H(K) = 01+52+27+56 = 136 = 36]$
(b) Fold Shifting : Even number parts are reversed before addition.
$[K = 15\,22\,75\,66 \Rightarrow H(K) = 01+25+27+65 = 190 = 90$ (ignore carry)$]$
(c) Fold Boundary : First & last parts are reversed before addition.
$[H(K) = 10+52+23+65 = 54$ (ignore carry)$]$
4) Digit Analysis Method : Extracting the digits from key
$K = 6\,7\,2\,2\,5\,4\,1 \Rightarrow H(K) = 427$ [Even position digits extracted and reversed it]

Compiler Design:

- **Ambiguous grammar:** For a given i/p string if there exists more than one parse tree or more than one LMD or more than one RMD, then the grammar is called "Ambiguous".
- **Left Recursive:** $A \to A\alpha / \beta \Rightarrow A \to \beta A'$ & $A' \to \alpha A' / \epsilon$ — Elimination
- **Left factoring:** $A \to \alpha\beta_1 / \alpha\beta_2 \Rightarrow A \to \alpha A'$ & $A' \to \beta_1 / \beta_2$
- Problem with Left recursive grammar is that if such a grammar is used by the parser, that Left most derivation (LMD) in deriving a string, then there is a danger of going into an "infinite loop".
- Left factoring avoids back-tracking in parser, but not ambiguity.
- Solution for Eliminating ambiguity.

* TOP-Down Parsing: [LL(1) Parser] (Left most derivation, Left to right scanning)

* LL(1) grammar is not Ambiguous. & It must be left-factored, and also It should be free from Left Recursion".

- First( ) & Follow( ) Sets are used to Construct LL(1) Parsing table.
- For Each rule $A \to \alpha$ in Grammar:
  1. For Each terminal 'a' Contained in First(α), add $A \to \alpha$ to [A, a] Entry in Parsing table
  2. If First(A) Contains 'ϵ', For Each terminal 'b' in follow(A), Add $A \to \epsilon$ to [A, b] Entry in Parsing table.
- TO check (whether) given grammar is LL(1): or not;
  1) If $A \to \alpha_1 / \alpha_2 \Rightarrow \{First(\alpha_1) \cap First(\alpha_2)\} = \phi$
  2) If $A \to \alpha / \epsilon \Rightarrow \{First(\alpha) \cap Follow(A)\} = \phi$

  } is LL(1).
- TOP down parser follows Left most derivation.

* TOP Down parsers
- Brute force (Backtracking).
- Recursive descendent parsing [Procedures for Each production]
- * Predictive parsing:
  - Recursive predictive parsing (Procedures for First & follow of each variable)
  - * Non-Recursive predictive parsing. [LL(1)]

* Bottom Up Parsers : [Shift-Reduce Parsers]

* OPParsers : [OPP] : Operator Precedence (OPP) LR-Parsers

- Operator Precedence Parser uses "operator precedence grammar", in which "no ε-Rules & does not contain two adjacent non-terminals on Right Side of any production" with Precedence-rules.
[Operator grammar with Precedence rules called Operator Precedence-grammar]

- Operator precedence Parsing Algorithm:

Let 'a' is top of stack symbol & 'b' is Symbol pointed by i/p Pointer.

1) If a=b=$, Succesful completion of parsing
2) If a<b or a≐b then push 'b' onto stack and advance i/p Pointer to next symbol.
3) If a>b then then pop 'a' & Reduction takes place for 'a'

* Operator precedence grammar may be either ambiguous or unambiguous grammar., But LR Parsers are contained (uses) only unambiguous grammar

* LR Parsers : [LR(0), SLR(1) (Simple LR(1)), LALR(1) (Lookahead LR(1)) & CLR(1) (Canonical) (Genesis)]
↳ Left to right Scanning → Right most derivation in reverse

- LR parsing Algorithm:

Let S is state on top of stack & 'a' is Symbol pointed by input pointer. Initially Push 0 [I_0 is initial state of DFA] on top of stack & advance i/p pointer to next Symbol.

1) If action [S, a] = S_i, Push 'a' & then 'i' onto stack and advance i/p pointer to the next Symbol.
2) If action [S, a] = γ_i, If γ_i is : A→β, Pop off 2×|β| Symbols from stack and replace by A. If S' is state below A in stack, then push goto [S', A] on to stack.
3) If action [S, a] = accept, Successful completion
4) If action [S, a] = blank, Error.

* Bottom up parser follows Right most derivation in reverse.
* Closure ( ) & goto ( ) functions are used to create canonical collection of LR items.

* LR(0) parsers:

- Shift Reduced Conflicts (SR Conflict): $A \to \alpha . x\beta$ (Shift), $B \to \gamma .$ (Reduce) } are in same state.
- Reduce-Reduce Conflict (RR Conflict): $A \to \alpha .$ (Reduced), $B \to \gamma .$ (Reduced) } are in same state.

- LR(0) Parsing table Construction:

1) If goto $(I_i, a) = I_j$ then set action $[i, a] = S_j$ (Shift Entry)
   (a: terminal)
2) If goto $(I_i, A) = I_j$, then set action $[i, A] = j$ (State Entry)
   (A: non-terminal)
3) If I_i contains $A \to \alpha .$ (Reduced production) (Complete) then the action $[i,$ All Entries$] = R_p$, where 'p' is production numbers. ($P: A \to \alpha$)
   (R_p: Reduced Entry)
4) If $S' \to S.$ is in I_i, then set action $[i, \$]$ to accept.

- The state, it Contains either Shift-Reduced or Reduce-Reduce Conflicts - then that state is called Inadequate-state.

* SLR(1) Parsers:

- Shift Reduced (SR) Conflicts: $A \to \alpha . x\beta$, $B \to \gamma .$ If follow(B) contains 'x'
- Reduce-Reduce (RR) Conflicts: $A \to \alpha .$, $B \to \gamma .$ If follow(A) $\cap$ follow(B) $\neq \phi$

- SLR(1) parsing table Construction:

"SLR(1) parsing table Construction same as LR(0) parsing, Except reduced Entries:"

If I_i contains $A \to \alpha .$ (Reduced production), then find follow(A), & for each (all) a in follow of A, set action $[i, a] = R_p$ (Production number P: A→α)

* CLR(1) [LR(1)] parsing table Construction:

"Except reduced Entries, Remaining Entries are same as SLR(1)"

If I_i contains $A \to a .,\ \$/a/b$ then Reduced Entry (R_p) $A \to a$ in row 'i' under the terminals given by lookahead ($\$, a, b$) in LR(1) items. [i.e. R_p Entry Entered into $[i, \$]$, $[i, a]$ & $[i, b]$]

* Shi
* Redu

* LALR(1) Passes:

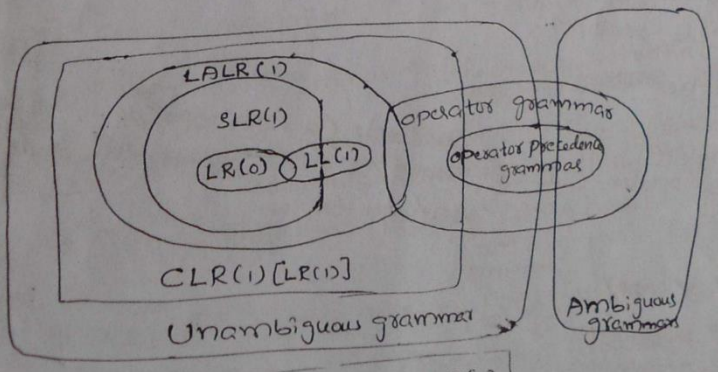
"It is constructed from CLR(1), If two states has same productions & that contains same core but different lookaheads, with those two states of CLR(1) combined into single state in LALR(1)." [& Hence there is no chance of Shift-reduced conflicts, but there may be chance to get reduced-reduced conflicts in LALR(1)]

* CLR(1) & LALR(1) Passes:

Shift-Reduced Conflicts: $A \to \alpha . x \beta , a$; $B \to \gamma . , x$

Reduce-Reduce Conflicts: $A \to \alpha . , a$; $B \to \gamma . , a$

- OPP < LL(1) < LR(0) < SLR(1) ≤ LALR(1) ≤ LR(1) [CLR(1)] [Power of Parsers]
- The no. of states (SLR(1)) = No. of states (LALR(1)) ≤ No. of states (CLR(1))



LR(0) ⊂ SLR(1) ⊂ LALR(1) ⊂ CLR(1)
LL(1) ⊂ LALR(1) ⊂ CLR(1)

* Shift actions: All Edged labelled with terminals in transition diagram (or) No. of Shift Entries in the Parsing table [LR(0), SLR(1), CLR(1) & LALR(1)]

* Reduced actions:
LR(0): No. of productions except Padding.
SLR(1): Total no. of Elements in the follows of Each & Every completed-production items
CLR(1): Total no. of lookaheads of all completed items.
LALR(1): "

* Syntax Directed Translation (SDT): [Translation rules placed within right side part]
  - Syntax Directed Definition (SDD): Translation rules placed at the end of production.
  - Synthesized Attribute: [Attribute value determined from children]
    - Follows Bottom up Approach, $A \to BC \ \{ A.i = f(B.i \text{ or } C.i) \}$
    - Follows Postfix Notation.
  - Inherited Attribute: [Attribute value determined from parent or sibling]
    - Follows Prefix Notation. $A \to BC \ \{ B.i = f(A.i \text{ or } C.i) \}$
    - Follows Top-down approach.

→ A Parse tree, shows attribute value at each node called "Annotated / Decorated parse tree".

- S-Attributed definition:
  An SDT which uses only synthesized attributes is called "S-attributed definition".

- L-Attributed definition:
  An SDT in which attributes are restricted to inherit either from parent or left siblings is called "L-attributed definition". [S-attribute definition is L-attributed]

$A \to BCD \ \{ C.i = f(A.i \text{ or } B.i) \}$ or
$\{ D.i = f(A.i \text{ or } B.i \text{ or } C.i) \}$ or
$\{ B.i = f(A.i) \}$ or
$\{ A.i = f(B.i \text{ or } C.i \text{ or } D.i) \}$ → S-attributed

- Every S-attributed definition is L-attributed, need not converse.

** Syntax directed translation (SDT) follows Depth first search notation.

* Intermediate Code generation:
  1) Syntax tree
  2) DAG
  3) Postfix Notation
  4) Three Address Code: → Quadruple (4-fields: op, op1, op2, Result)
     → Triples (3 fields: operator, op1, op2)
     → Indirect triples

* Runtime Environments:
  Static: Definition of a procedure,
  Declaration of a name, &
  Scope of declaration
  Dynamic: Activation of a procedure
  Binding of a name
  Lifetime of a binding

- Storage: Static → Target code & static data objects
  Heap → Dynamic data objects
  Stack → Automatic data objects

- Activation Record:
  - It Contains (in stack): Temporary data, Local data, Machine status, optional access link, Optional Control link, Actual parameters & Returned Value.

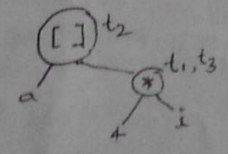
* Code optimization:
  - Machine dependent: optimizations applied on target code.
  - Machine Independent: optimizations applied on 3-Address Code.
- After applying optimization, there is no guarantee that generated code is optimal.
- Basic Block: Sequence of Consecutive Statements, in which Control Enters at beginning and leaves at End without halt or possibility of branching Except at the end.

* Machine Independent optimizations:
  1) Loop optimization: a) Code motion/Loop invariant Elimination/frequency-reduction.
     b) Loop Unrolling [duplication of body]
     c) Loop Jamming/Loop fusion/Loop Combining.
  2) Folding / Constant folding: [3+5 ⇒ 8]
  3) Constant propogation: [pi/180 ⇒ 3.1415/180]
  4) Copy propogation: [x=y & z=1.0+x ⇒ x=y & z=1.0+y]
  5) Dead Code Elimination: [Removes instructions without changing the behaviour (using DAG)]
  6) Common Sub Expression Elimination (Redundancy Elimination): [x=(a+b)-(a+b)/4 ⇒ [c=a+b, x=c-c/4] [using DAG]]
  7) Strength Reduction: (Replacing Costlier Operation by Cheaper one) [x**2 = x*x (Exponential)]
  8) Algebraic Simplification: [Eliminates x=x*1; & y=y+0]

* DAG [Directed Acyclic graph]: Minimizes temp usage & Eliminates Common Sub Expressions.

$t_1 = 4*i$
$t_2 = a[t_1]$
$t_3 = 4*i$

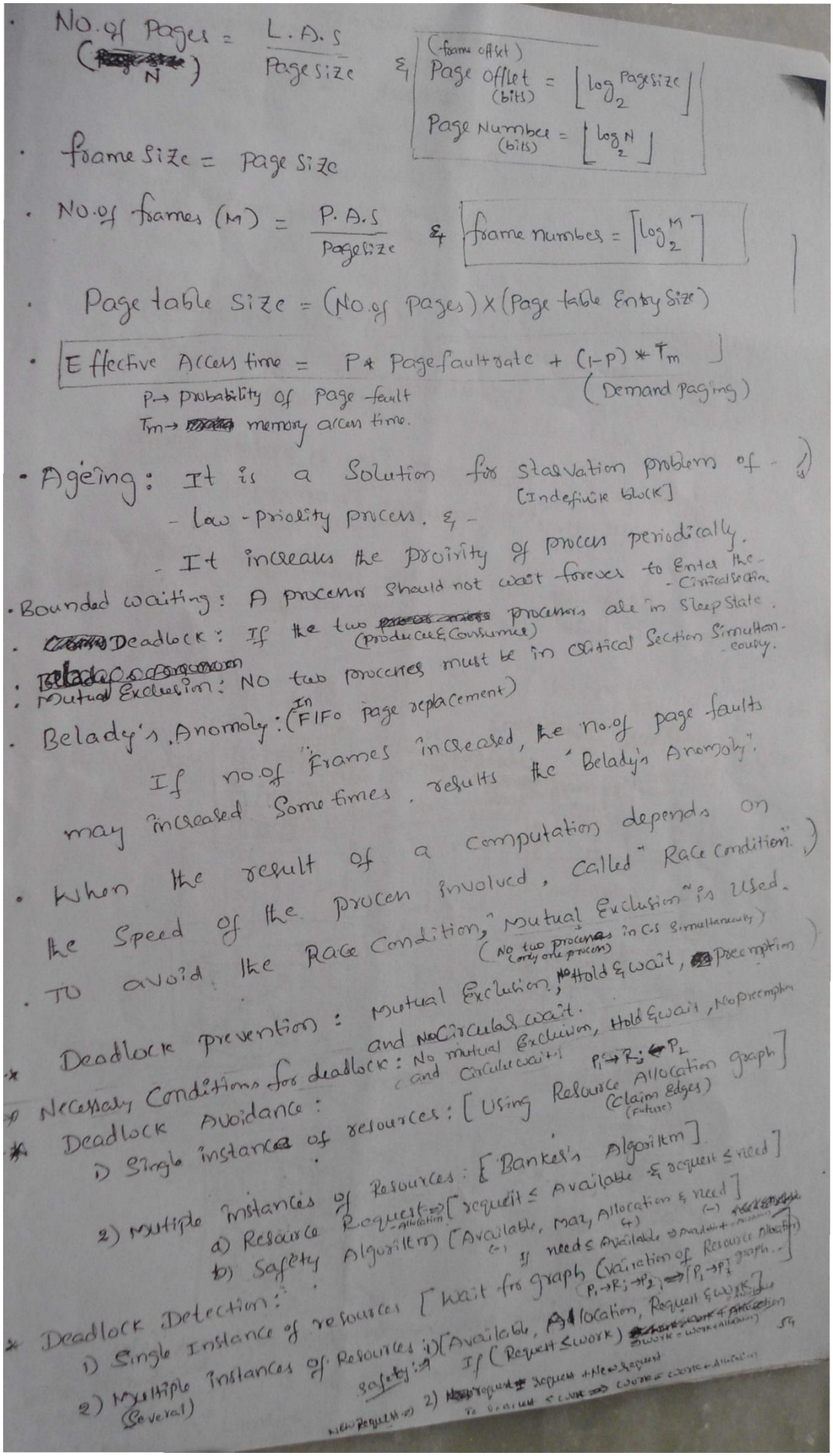
⇒



Operating System:

- Arrival time = Submission time [A.T]
- CPU Burst time = Service time [B.T].
- Completion time = Finished time [C.T]
- Turn Around Time [T.A.T] = C.T − A.T
- Waiting time [W.T] = T.A.T − B.T
- Response time = First Response time − Arrival Time.
- Highest Response Ratio Next (Non-preemptive) selects the highest "response ratio" (w+s/s) process. [Response Ratio = (Waiting time + Service time) / Service time]
- First Fit: Allocation of first available free space.
- Best Fit: " " best available free space.
- Worst Fit: " " largest available free space.
- Next Fit: Same as first fit, but it starts from last allocated process.
- Logical address space (LAS) > Size of physical address space (read) [P.A.S].
- Logical Address space < Virtual Address Space (VAS) (TSIK)
- Overlays: Do not require any special support from operating system. [Keeping instructions & data in memory, that are needed at any time.] [only those]
- Dynamic Loading: A Routine is not loaded until called & All routines are kept in "relocatable load format". It does not require any special support from OS, but storing libraries taken support from O.S.
- Dynamic Linking: "Rather than Loading being postponed until Execution time, Linking is postponed". [It requires support from O.S]
- Static Loading: All modules of programs are stored in memory before calling.
- [r ≥ p(n−1) + 1] r → no. of available resources (for deadlock free) n → max no of resources needed by each process p → No. of processes
- Convoy Effect: At time T_0, All processes are in ready queue & at time T_1, All processes are blocked [in block/device queue]. Thus gives Poor Utilization of resources. (in FCFS CPU scheduling)

External fragmentation can be avoided by "compaction".



- No. of Pages = L.A.S / Page size (N)
- frame size = Page size
- No. of frames (M) = P.A.S / Page size
- Page table size = (No. of pages) × (Page table Entry size)
- Effective Access time = P * Page-fault rate + (1-P) * Tm
  P → probability of page fault
  Tm → memory access time.
- & (frame offset) Page offset (bits) = $\lceil \log_2^{\text{Page size}} \rceil$
  Page Number (bits) = $\lceil \log_2^{N} \rceil$
- & frame number = $\lceil \log_2^{M} \rceil$
- (Demand Paging)
- Ageing: It is a Solution for starvation problem of low-priority process. [Indefinite block]
  It increases the priority of process periodically.
- Bounded waiting: A process should not wait forever to enter the Critical Section.
- Deadlock: If the two processes (Producer & Consumer) are in Sleep state.
- Mutual Exclusion: No two processes must be in critical Section Simultaneously.
- Belady's Anomaly: (In FIFO page replacement)
  If no. of frames increased, the no. of page faults may increased Sometimes, results the "Belady's Anomaly".
- When the result of a computation depends on the Speed of the process involved, called "Race Condition".
- To avoid the Race Condition, "Mutual Exclusion" is used. (No two processes / only one process in CS simultaneously)
- Deadlock prevention: Mutual Exclusion, Hold & wait, Preemption and No Circular wait.
- Necessary Conditions for deadlock: No mutual Exclusion, Hold & wait, No Preemption and Circular wait.
- Deadlock Avoidance:
  1) Single instance of resources: [Using Resource Allocation graph] $P_i \to R_j \leftarrow P_i$ (Claim Edges) (Future)
  2) Multiple instances of Resources: [Banker's Algorithm]
     a) Resource Request → [request ≤ Available & request ≤ need]
     b) Safety Algorithm [Available, max, Allocation & need]
- Deadlock Detection:
  1) Single Instance of resources [wait for graph (Variation of Resource allocation graph)]
  2) Multiple (Several) Instances of Resources: [Available, Allocation, Request & Work]
     safety: If (Request ≤ work)

* Recovery from deadlock: 1) Resource Termination:
  a) Abort all deadlock process.
  b) Abort one process at a time until the deadlock cycle is eliminated.
  2) Resource preemption:
  a) Selecting a victim
  b) Rollback
  c) Starvation (Fixing upper limit of processes)

- Valid/Invalid bit:
  Valid → Associated page is legal and in memory
  Invalid → Either the page is not valid or the page is valid but currently on the disk.

- Thrashing: High page fault rate [i.e, A process spending more time for paging than executing].

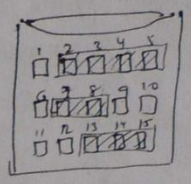
- Dirty bit: To indicate that the page has been modified (Dirtybit=1) (Modify) Otherwise, the page is not modified if Dirtybit=0.

* File Allocation Methods:

1) Contigeous Allocation:

Directory:

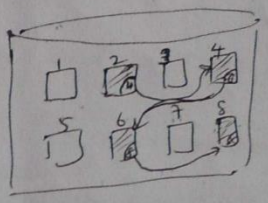
Filename	Starting Block	Count (No. of allocated blocks)
x	2	4
y	7	2
z	13	3



2) Linked Allocation:

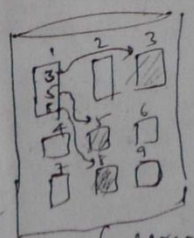
Directory:

Filename	Starting block address	Ending block
x	2	8



3) Indexed Allocation:

Filename	Index block Address
x	1



Disadvantage: Index block may not sufficient for largest files

* Free Space Management:

1) Bitmap / Bit vector: 'Free space list' is used to indicate (Keep) track of free disk space, It records all the free disk blocks. In Bitmap, each block is represented by 1-bit, either 0 or 1; If block is free, the bit is 1; If block is allocated, the bit is 1.

Ex:- Free disk blocks are: 2, 3, 4, 5, 8, ...

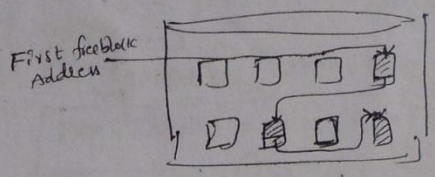
Then Free space bit map: 0011110010...
(1 2 3 4 5 6 7 8)

Calculation of First free block number:

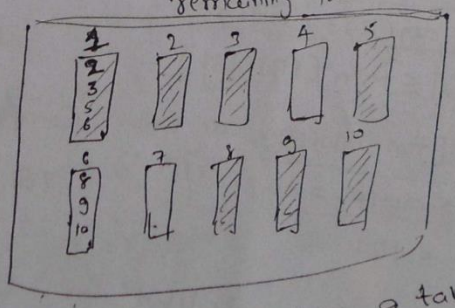
= (No. of bits per word) x (no. of 0-value words) + offset of first 1-bit.

2) Linked list:

Address of first free block is placed in cache & remaining free block addresses are pointed by previous of that free block.

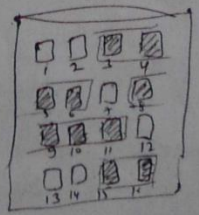


* 3) Grouping: First free block contains, next 'n' free blocks address in it, and in nth free block remaining free blocks addresses from (n+1)th free block.



4) Counting: It maintains a table, in which, Contiguous free blocks are counted and gives the first address of free block from HDD.

First free block	count
3	4
8	4
11	2



* Disk Scheduling Algorithms:

1) FCFS, 2) SSTF, 3) SCAN (Elevator)

4) C-SCAN, 5) LOOK-C

Databases:

- DDL commands: CREATE, ALTER, DROP, TRUNCATE, GRANT, REVOKE & COMMENT.
- DCL Commands: COMMIT, SAVEPOINT, ROLLBACK & SET TRASACTION.
- DML Commands: SELECT, INSERT, UPDATE, DELETE, CALL, LOCKTABLE and EXPLAIN PLAN.

* Selection operation [σ] (Restriction): $deg(\sigma_c(R)) = deg(R)$ [No. of attributes]

$0 \le |\sigma_c(R)| \le |R|$ [No. of tuples]

- $\sigma_{c_1}(\sigma_{c_2}(R)) = \sigma_{c_2}(\sigma_{c_1}(R)) = \sigma_{c_1 \wedge c_2}(R)$

* Projection operation [Π]: $0 \le deg(\Pi_A(R)) \le deg(R)$

$|\Pi_{A_1}(R)| = |R|$

- Possible no. of projections over Relation with n-attributes $= 2^n$

* Rename [ρ]: $deg(\rho_S(R)) = degree(R)$ & $|(\rho_S(R))| = |R|$

* Union operation [U]: [Two relations are union compatible (same degree & domain)]

* $deg(R \cup S) = deg(R) = deg(S)$

$\max(|R|, |S|) \le |(R \cup S)| \le (|R| + |S|)$

* Intersection operation [$\cap$]: [Two relations are union compatible]

$deg(R \cap S) = deg(R) = deg(S)$

$0 \le |(R \cap S)| \le \min(|R|, |S|)$

* Set Difference [−]: [Two Relations are union compatible]

$deg(R - S) = deg(R) = deg(S)$

$0 \le |(R - S)| \le |R|$

* Union compatible of two relations are used for: $\cup, \cap, \&\ -$

* Cartesian product [cross product or cross join] [X]

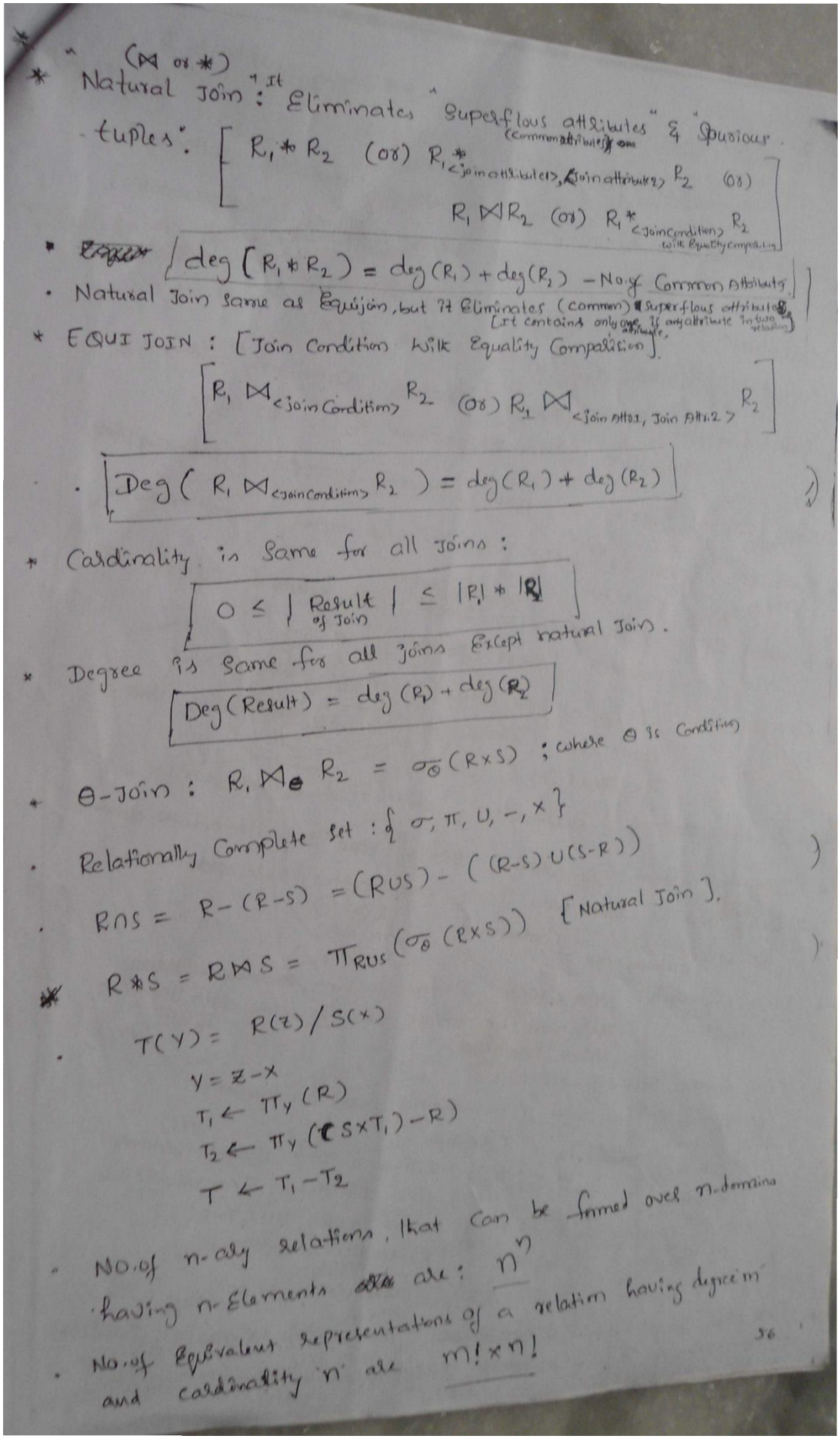
$Deg(R \times S) = deg(R) + deg(S)$

* $|(R \times S)| = |R| * |S|$

* No. of Super keys formed over $R(A_1, A_2 \ldots A_n)$ having:

① A_i as key $= 2^{n-1}$ super keys, ② $A_i A_j$ as key $= 2^{n-2}$ super keys

③ A_i & A_j as key $= 2^{n-1} + 2^{n-1} - 2^{n-2} = 3/4\, 2^n$ ④ $A_i A_j$ & $A_K A_L$ as keys $= 2^{n-2} + 2^{n-2} - 2^{n-4}$



* Natural Join ($\bowtie$ or $*$): It Eliminates "Superflous attributes" & "Spurious tuples".
  [$R_1 * R_2$ (or) $R_1 *_{<\text{join attribute1}>, <\text{join attribute2}>} R_2$ (or) $R_1 \bowtie R_2$ (or) $R_1 *_{<\text{Join Condition}>} R_2$]
  (Common attributes) (with Equality comparing)
  - $\deg(R_1 * R_2) = \deg(R_1) + \deg(R_2) - \text{No. of Common Attributes}$
  - Natural Join same as Equijoin, but it Eliminates (common) Superflous attributes [It contains only one if any attribute in two relations]
* EQUI JOIN : [Join Condition with Equality Comparision].
  [$R_1 \bowtie_{<\text{Join Condition}>} R_2$ (or) $R_1 \bowtie_{<\text{Join Attr1, Join Attr2}>} R_2$]
  - $\text{Deg}(R_1 \bowtie_{<\text{Join Condition}>} R_2) = \deg(R_1) + \deg(R_2)$
* Cardinality is same for all joins :
  $0 \le |\text{Result of Join}| \le |R_1| * |R_2|$
* Degree is same for all joins Except natural Join).
  $\text{Deg}(\text{Result}) = \deg(R_1) + \deg(R_2)$
* θ-Join : $R_1 \bowtie_\theta R_2 = \sigma_\theta(R \times S)$; where θ is Condition
- Relationally Complete set : $\{\sigma, \pi, \cup, -, \times\}$
- $R \cap S = R - (R - S) = (R \cup S) - ((R - S) \cup (S - R))$
* $R * S = R \bowtie S = \pi_{R \cup S}(\sigma_\theta(R \times S))$ [Natural Join].
- $T(Y) = R(Z) / S(X)$
  $Y = Z - X$
  $T_1 \leftarrow \pi_Y(R)$
  $T_2 \leftarrow \pi_Y((S \times T_1) - R)$
  $T \leftarrow T_1 - T_2$
- No. of n-ary relations, that can be formed over n-domains having n-Elements are: n^n
- No. of Equivalent representations of a relation having degree n and cardinality n are $n! \times n!$

* Recoverable : If T_2 reads a value(x) previously written by T_1, then C_1 appears befor C_2 / a_2

* Cascading rollback : If T_2 reads a value(x) previously written by T_1, then a_1 appears before C_2 [It is not recoverable].

* Cascadeless : If T_2 reads a value (x) previously written by T_1, then C_1/a_1 appears before $R_2(x)$.

* Strict Schedule : If T_2 reads a value(x) previously written by T_1 then C_1/a_1 appears before $R_2(x)$ or $W_2(x)$

* Basic 2PL : Locking is growing phase & unlocking is shrinking phase.

* Strict 2PL : All the write locks are released after Commit / Abort operation.

* Rigorous 2PL : All the locks [shared read & exclusive write] are released after Commit / Abort operation.

* Conservative 2PL : Locks all the items it access before the transaction begins Execution by predeclaring its read set & write set.

* Conflict operations : Read-write, write-read & write-write.

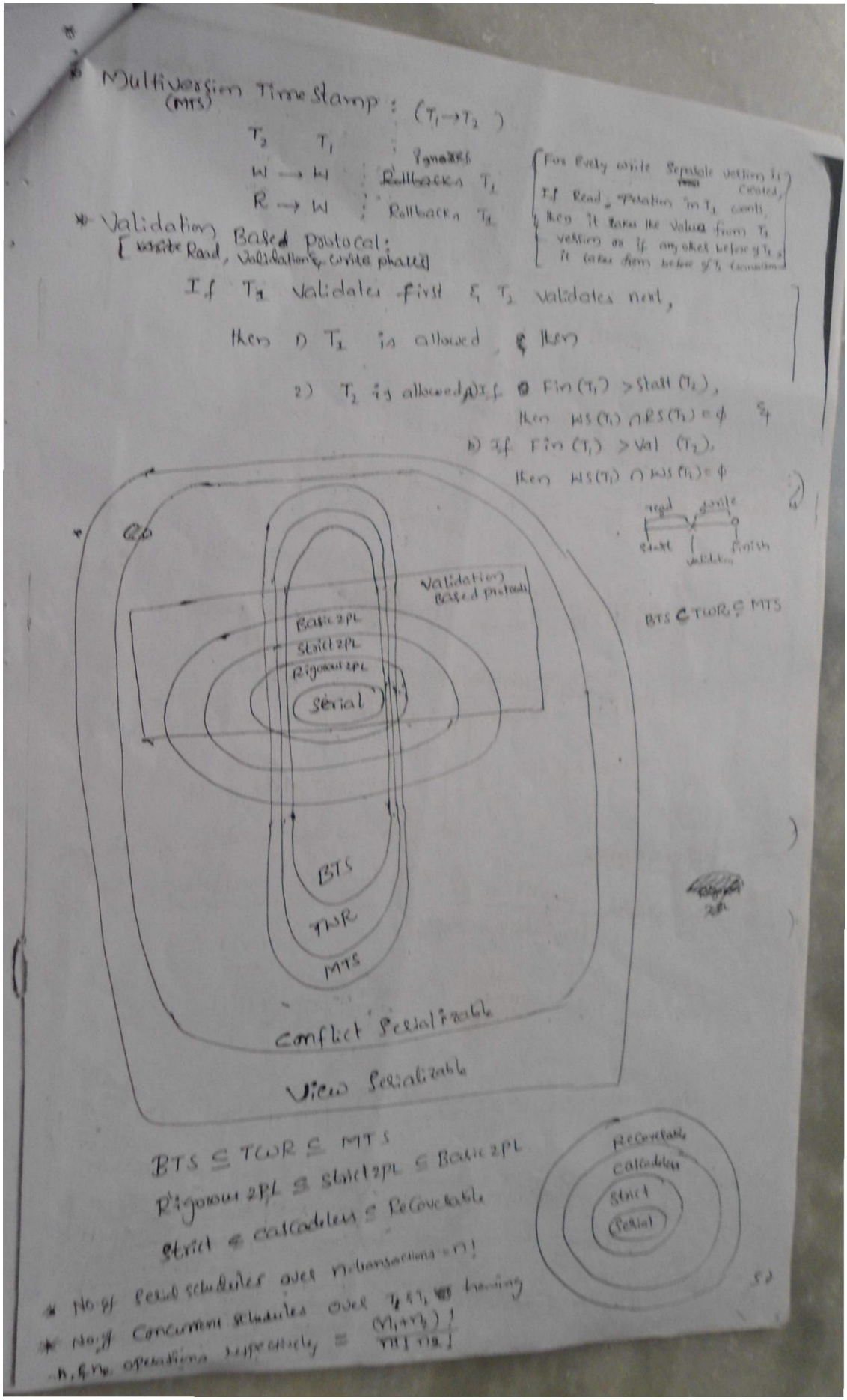
* Consistent Schedule : In which, Before the read or write or lock operations it locks over the same data item and after those operation(s), the unlock must be contained. (it releases lock)

* Legal Schedule : In which, Before locking, there must be unlock over the same data item in other transaction.

* Time stamp Based Protocal :
If $T_1 \rightarrow T_2$ is order of transactions based on timestamp.

T_2	T_1 (request)	
W	W	; Rollbacks T_1 in Basic timestamp (BTS), But Ignores in Thomas write Rule (TWR).
R	W	; Rollbacks in BTS & TWR
W	R	; Rollbacks in BTS & TWR
R	R	; Allows in both.

* Basic 2PL, Strict 2PL & Rigorous 2PL are always Conflict Serializable Schedules [& deadlock may occur].


Multiversion Timestamp : ($T_1 \to T_2$)
(MTS)
T_2 T_1
W → W : Rollback T_1
R → W : Rollback T_2
For every write Separate version is Created,
If Read operation in T_1 comes then it takes the value from T_2 version as if any other before T_1 it takes from before of T_1 execution
Validation Based Protocol:
[Read, Validation & write phase]
If T_1 validates first & T_2 validates next,
then 1) T_1 is allowed & then
2) T_2 is allowed if a) Fin(T_1) > Start(T_2), then $WS(T_1) \cap RS(T_2) = \phi$ &
b) If Fin(T_1) > Val(T_2), then $WS(T_1) \cap WS(T_2) = \phi$
read
write
Start
Finish
Validation
Validation Based protocol
Basic 2PL
Strict 2PL
Rigorous 2PL
Serial
BTS
TOR
MTS
Conflict Serializable
View Serializable
BTS ⊆ TOR ⊆ MTS
Rigorous 2PL ⊆ Strict 2PL ⊆ Basic 2PL
Strict ⊆ cascadeless ⊆ Recoverable
No. of Serial schedules over n transactions = n!
No. of Concurrent schedules over T_1 & T_2 having n_1 & n_2 operations respectively = $\frac{(n_1+n_2)!}{n_1! \, n_2!}$
Recoverable
cascadeless
Strict
Serial

* Indexing : 1) Single level Index
  - Primary Index (sparse)
  - Clustered Index (non-dense)
  - Secondary Index (Dense)
  2) Multilevel Index
  - Indexed Sequential access method
  - B-Tree Index
  - B^+-Tree Index.

- Primary Index [ordered with Key field]
- Clustered Index [ordered with non-key field]
- Secondary Index [ordered with Either key or non-Key field].

* Sequential file :

$$\text{No. of records per block} = \text{Blocking factor} = \left\lfloor \frac{\text{Block size}}{\text{Record size}} \right\rfloor$$

$$\text{No. of}^{\text{record}}\text{ blocks} = \left\lceil \frac{\text{total no. of records}}{\text{Blocking factor}} \right\rceil$$

No. of block accesses (Avg) by linear search:

$$= \left\lceil \frac{\text{No. of}^{\text{record}}\text{ blocks} + 1}{2} \right\rceil$$

No. of block accesses by Binary Search:

$$= \left\lceil \log_2 {}^{\text{No. of}^{\text{record}}\text{ blocks}} \right\rceil$$

* Indexed File :

1) Primary Index :

- Index blocking factor $= \left\lfloor \frac{\text{Block size}}{\text{Key field size} + \text{Block Pointer size}} \right\rfloor$
- First (single) level Index blocks : $= \left\lceil \frac{\text{No. of}^{\text{block}}\text{ of}^{\text{record}}\text{ key}}{\text{Index blocking factor}} \right\rceil$
- No. of block accesses $= \left\lceil \log_2 (\text{First level Index blocks}) \right\rceil + 1$

[redacted]

3) Clustered Index:

- Index blocking factor (bf) = $\left\lfloor \frac{\text{Block size}}{\text{Key field size} + \text{Block pointer size}} \right\rfloor$
- Single (First) level Index blocks = $\left\lceil \frac{\text{No. of distinct values over non-key field (Clustered Index)}}{\text{Index Blocking factor}} \right\rceil$
- No. of block accesses = $\left\lceil \log_2 (\text{Single level index blocks}) \right\rceil + 1$

4) Secondary Index:

- Index blocking factor = $\left\lfloor \frac{\text{Block size}}{\text{Key field size} + \text{Block pointer size}} \right\rfloor$
- Single level Index blocks (first) = $\left\lceil \frac{\text{Total no. of Records}}{\text{Index blocking factor}} \right\rceil$

No. of block accesses = $\left\lceil \log_2 (\text{Single level Index blocks}) \right\rceil + 1$

→ Index blocking factor same for all indexes.

4) Multi level Index: (It is same for Primary, clustered & Secondary Indexes)

Level 1 = "First level Index blocks" computed by given Index (Primary, clustered or Secondary)

Level 2 = $\left\lceil \frac{\text{No. of blocks in Level 1}}{\text{Index Blocking factor}} \right\rceil$

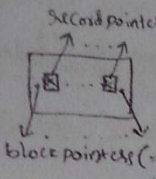
⋮

Level n = $\left\lceil \frac{\text{No. of blocks in Level (n-1)}}{\text{Index Blocking factor}} \right\rceil = 1$

- No. of levels = n
- No. of blocks = $\sum_{i=1}^{n} (\text{no. of blocks in level } i)$
- No. of block accesses = n + 1

* B-Tree : (Bayer's / Balanced search tree).

Degree / order "n" : 1) Root node : 2 to n children [Pointers]



2) Internal node : $\lceil \frac{n}{2} \rceil$ to n children [Pointers]

3) Leaf nodes are all at the same level.

$$\text{BlockSize} \geq P*(\text{Size of block Pointer}) + (P-1)*(\text{Size of Key field} + \text{Size of Record Pointer (data Pointer)})$$

P → order (degree) of B-Tree / max no. of block pointers.

h → height

- Minimum (Minimum) No. of nodes $= 1 + \left(\frac{2\left[\left(\frac{P}{2}\right)^{h} - 1\right]}{\left(\frac{P}{2}\right) - 1} \right)$

- Maximum No. of nodes $= \frac{P^{h+1} - 1}{P - 1}$

- Minimum possible height (h_{min}) $= \lceil \log_P l \rceil$ — l → no. of leaves (leaf nodes)

- Maximum possible height (h_{max}) $= \left\lfloor 1 + \log_{\left(\frac{P}{2}\right)} \left(\frac{l}{2}\right) \right\rfloor$

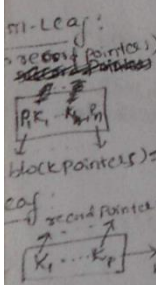
* B^+ Tree :

It is same as B-Tree, except

1) All the records are available at leaf (last) level & some of them are duplicated in one non-leaf level [above the leaf level]

2) It allows both random & sequential access, but in B-Tree only random access allowed.

3) All the leaf nodes are connected to next leaf node by block pointers. [Every leaf node has one block pointer]



* Order of Internal (non-leaf) node :

$$\lceil [P * \text{Size of block Pointer}] + [(P-1) * \text{Size of Key field}] \rceil \leq \text{Block size}$$

P → order of B^+-tree (non-leaf)

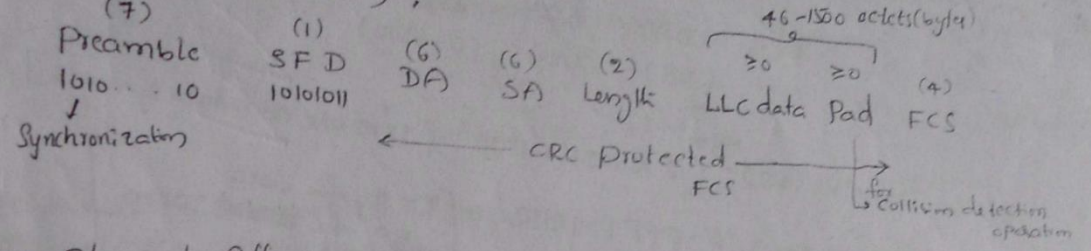
- Order of leaf node :

$$P_{leaf} * (\text{Size of Key field} + \text{Size of record pointer}) + \text{Size of block pointer} \leq \text{Block size}$$

Computer Networks:

SM's copy

* Ethernet (IEEE 802.3):



$$\text{Channel Efficiency (utilization)} = \frac{1}{1+2BLe/vf} = \frac{1}{1+5.44a} = \frac{P}{P+2T/A}$$

(P → Transmission time)

$2T$ → Duration of each slot ($=2 *$ Propagation delay)

A → Slot Probability ($=1/e$)

f → frame length

v → Speed of Signal propagation

e → Contention slots / frame

L → Cable Length ($=d$)

B → N/w bandwidth

$$a = \frac{\text{Propagation delay}}{\text{Transmission delay}} = \frac{d/v}{f/B}$$

a → Propagation delay / transmission time

P → mean time to transmit a frame ($P = f/B$) [Transmission time].

- Minimum length of frame is 72 bytes (or) 64 bytes
- Minimum Data is 46 bytes & Maximum data is 1500 bytes

* CSMA/CD for Ethernet:

$$\text{Utilization} = \frac{\left(\frac{1}{2a}\right)}{\left(\frac{1}{2a}\right)+\left(\frac{1-A}{A}\right)} = \frac{1}{1+3.44a} = \frac{1}{1+6.44a}$$

$$\text{Utilization} = \frac{T.P}{T.P+C.P} = \frac{\left(\frac{1}{2a}\right)}{\left(\frac{1}{2a}\right)+\left(\frac{1-A}{A}\right)}$$

$\frac{1}{2a}$ → Transmission (TP) = P · Interval (Period)

$\frac{1-A}{A}$ → Contention · Interval

$A = \left(1-\frac{1}{N}\right)^{N-1}$

N → No. of stations

- It offers Connection less Communication (Ethernet)
- No Acknowledgements in Ethernet.
- No Error Control & No flow Control.
- Maximum duration for Contention slot = 2 * Propagation delay.

$\therefore$ Contention period $= \frac{2 * \text{Propagation delay}}{A}$; $A = 1/e$

* Token Ring (IEE 802.5)

	SD	AC	FC	DA	SA	Data	checksum	ED	FS
bytes →	(1)	(1)	(1)	(2/6)	(2/6)	(0-4500)	(4)	(1)	(1)

Starting Delimiter (SD): J K 0 J K 0 0 0 [J & K are non-data symbols]

Access Control (AC): P P P T M R R R [T = 1 for data, = 0 for Token]

Ending Delimiter (ED): J K 1 J K 1 I E [I → Intermediate frame indication; E → Error Detection bit]

Frame Status (FS): A C r r A C r r [r → unused bit; A → Destination, C → Frame acceptance]

A	C	
0	0	→ Destination not present & Frame not accepted.
0	1	→ Not Possible
1	0	→ Destination Present & Frame not accepted
1	1	→ Destination Present & Frame accepted

- Max data is 4500 octets
- Min frame length is : 3 bytes

Token frame: | SD | AC | ED | (1) (1) (1)

* Ring latency: (Time taken for a bit to travel around Ring. (RL)

$$R.L. = \frac{d}{v} + \frac{Mb}{R} \quad \text{(seconds)}$$

$$= \frac{dR}{v} + Mb \quad \text{(bits)}$$

d → length of link (m)
v → velocity (m/s)
M → no. of station
R → Data rate of link (bps)

- Data transfer rate $= \frac{f}{t_1 + t_2 + \frac{f}{B}}$;

f → frame length
B → channel capacity or Bandwidth
t_2 → token observing time
t_1 → Ring latency time.

- Length of Link (bits) $= R \times \frac{d}{v}$

- Utilization $= \frac{T_1}{T_1 + T_2}$

$= \frac{1}{1 + \frac{a}{N}}$; if $a < 1$

$= \frac{1}{a(1 + \frac{1}{N})}$; if $a > 1$

T_1 → Avg time to transmit a data frame
T_2 → Avg time to pass a token
N → No. of stations

* Stop & wait ARQ:

$$\text{Utilization [Efficiency } (\eta)] = \frac{(t_{transmit})}{(t_{transmit} + t_{propagating})} = \frac{t_{transmit}}{t_{transmit} + \text{Round trip time (RTT)}}$$

$$= \frac{f/B}{f/B + RTT} = \frac{1}{1+2a} \quad \left[\because a = \frac{t_{propagating}}{t_{transmit}}\right]$$

- Positive ACK's are numbered in Stop&wait & $= \frac{1-P}{1+2a}$
- Negative ACK's are not numbered.
- Throughput = 1 Data/RTT = 1 Packet/RTT

$P = 10^{-3}$
P → probability of single frame Error.

* Go-Back N ARQ: [Accepts in order of packets]

$$\text{Efficiency} = (2^k - 1) * \frac{t_{transmit}}{t_{transmit} + R.T.T}$$

$$= \frac{1-P}{1-P+P^{N+1}}$$

P → Packet loss probability
W → Sender window size
k → No. of bits for sequence numbers of frames

- $W_S + W_R \le$ A.S.N (Available Sequence Number)
- Sender window size = N (SWS) & Receiver window size = 1 (RWS)
- Uses Cumulative / Piggybacking acknowledgements.
- Go-back N is called as "Conservative protocol".

$t_{transmit} = f/B$
$R.T.T = 2 \times t_{propg}$
$a = \frac{t_{propagation}}{t_{transmit}}$
$\text{Throughput} = \frac{\text{one packet}}{R.T.T}$

* Selective Repeat: [Selective Reject]

- It accepts out of order of packets.

$$\text{Efficiency} = (2^{k-1}) * \frac{t_{transmit}}{t_{transmit} + R.T.T}$$

- It uses Piggybacking / Cumulative / Independent Acknowledgements
- $W_S = W_R = \frac{N}{2}$; N → Max Available seq number (2^k)

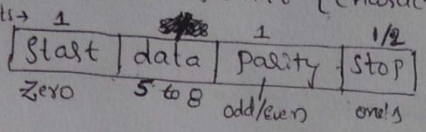
* RTT (Round Trip time): It is minimum ACK knowledgement waiting time. (RTT = 2 * Propogation delay)

* Timeout: It is the maximum Acknowledgement waiting time.
[Timeout = 2 * RTT]

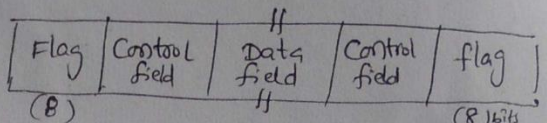
- Stop & wait ARQ is an Example of Closed Loop Control protocol [Based on o/p, the input is adjusted, & it is Reliable & Connection oriented]
- Throughput = $\frac{1 \text{ Packet}}{R.T.T}$ & with Piggyback (Throughput) = $\frac{2 * \text{Packet}}{RTT}$
- Repeater: physical layer, Bit, Nearest LANs of same type, Fully H/w [Simplified]
- Bridges: Data link layer (Physical & data), frame, Nearest different LANs, Much H/w (Less S/w [Frame connection])
- Routers: Network layer (Logical address), packet, Far away LAN's (LAN-WAN), Less H/w & Much S/w. [Routing]
- Gateways: All layers, message, Different WANS (WAN-WAN), fully S/w [Protocol Conversion]

5

* Asynchronous frame format [Characters format]:



* Synchronous frame format:



* Routing Algorithms:

1) Flooding: Every incoming packet is sent out on every outgoing line except the one it arrived on.

2) Broad cast Routing: a) Flooding, b) multidimensional routing c) Sink tree (spanning tree) and d) Reverse path forwarding.

It sends the packets from a host to all other remaining hosts in the network simultaneously.

3) Multicast Routing:

It sends the packets from a host to a specific groups.

a) Shared tree, b) Source based tree, c) Least unit cost path tree & d) Reverse path multicasting. [uses prune message]

4) Distance Vector routing: (RIP, IGRP) [uses split-horizon technique].

In which, Routing only to neighbours & knowledge about whole network.

5) Link State Routing (OSPF) (Dijkstra's Algorithm).

It discovers the neighbours of (cost of reaching each of neighbours) each routes and learn their n/w addresses, sends the packet to all other routers.

6) Flow-based Routing:

$$T = \frac{1}{\mu C - \lambda}$$

T → mean delay time
μ → mean packet size (bits)
C → capacity (bps)
λ → mean flow (packets/sec)

7) Hierarchical Routing: (Intra region & Inter region routings):

Routing is based on regions, & It maintains the two tables namely Full table [for all routers in all regions] & hierarchical table [for all routers within the region & knows about other regions but not routers of other regions] for the Gateway routers.

8) Random Routing : Outgoing link is chosen randomly by assigning probability to each link. $\left[P_i = \frac{R_i}{\Sigma R_i}\right]$

9) Path-Vector (Policy) Routing : [BGP] :

It lists all of Autonomous Systems visited, inorder to reach the destination n/w by this route.

* Congestion Control :

1) Back pressure : Connection-Oriented

2) Choke packet : [ICMP Source quench packet] [Back to Source node]

3). Implicit Congestion Signalling : [End systems]
- Connection oriented & Connection-less.

4). Explicit Congestion Signalling [Forward / Backward]
- Connection oriented.

5) Policing / Reservation : [End Systems].

Leaky Bucket :
S (Speed) $= \frac{C}{M-P}$
C → Capacity

* Application Layer protocals :

FTP – 20 for data
21 for Control
} ⇒ TCP

Telnet – 23
SMTP – 25 ⇒ TCP
DNS – 53 ⇒ UDP
HTTP – 80 ⇒ TCP
POP3 – 110 ⇒ TCP
ICMP ⇒ UDP [i30 → query, 131 → Report, 132 → Termination]

* Physical Layer [Bit transmission] :

* Data Link Layer : [Framing]
- Physical Address (48-bit) / NIC / IEEE / Ethernet / Hardware / MAC address. [NIC card] stored at
- Access Control, Flow Control & Error Control
- Synchronization
- Node-Node data transfer. Ex :- HDLC, LLC ↳ Bit oriented

6

* Network Layer : [Packeting].
- Logical Address (32-bit) / IP / Internet / Subnet Address [Hard disk] stored at
- Routing & Congestion Control, Switching & Relaying.
- Source - Destination data transfer Ex :- CCITT X.25, X.75 Gateway

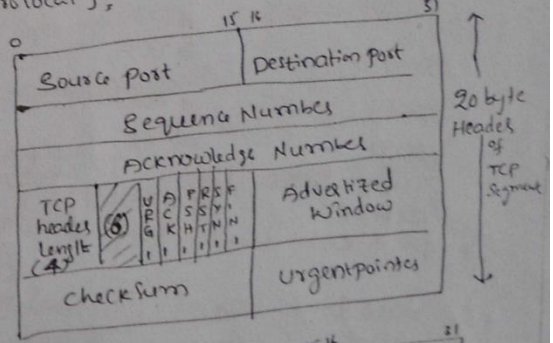
* Transport Layer: [Segmentation]
  - Port Address (16-bit) [Stored at Main memory]
  - Connection Oriented & Connection Less Service
  - Segmentation & Reassembly
  - End - End data transfer

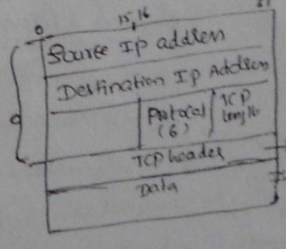
* Session Layer: [Sessions]
  - Synchronization, Dialog Control, TOKen management & Activity Management.
  - Mapping Logical names to physical address & vice-versa
  - Grouping & Recovery (checkpoints).

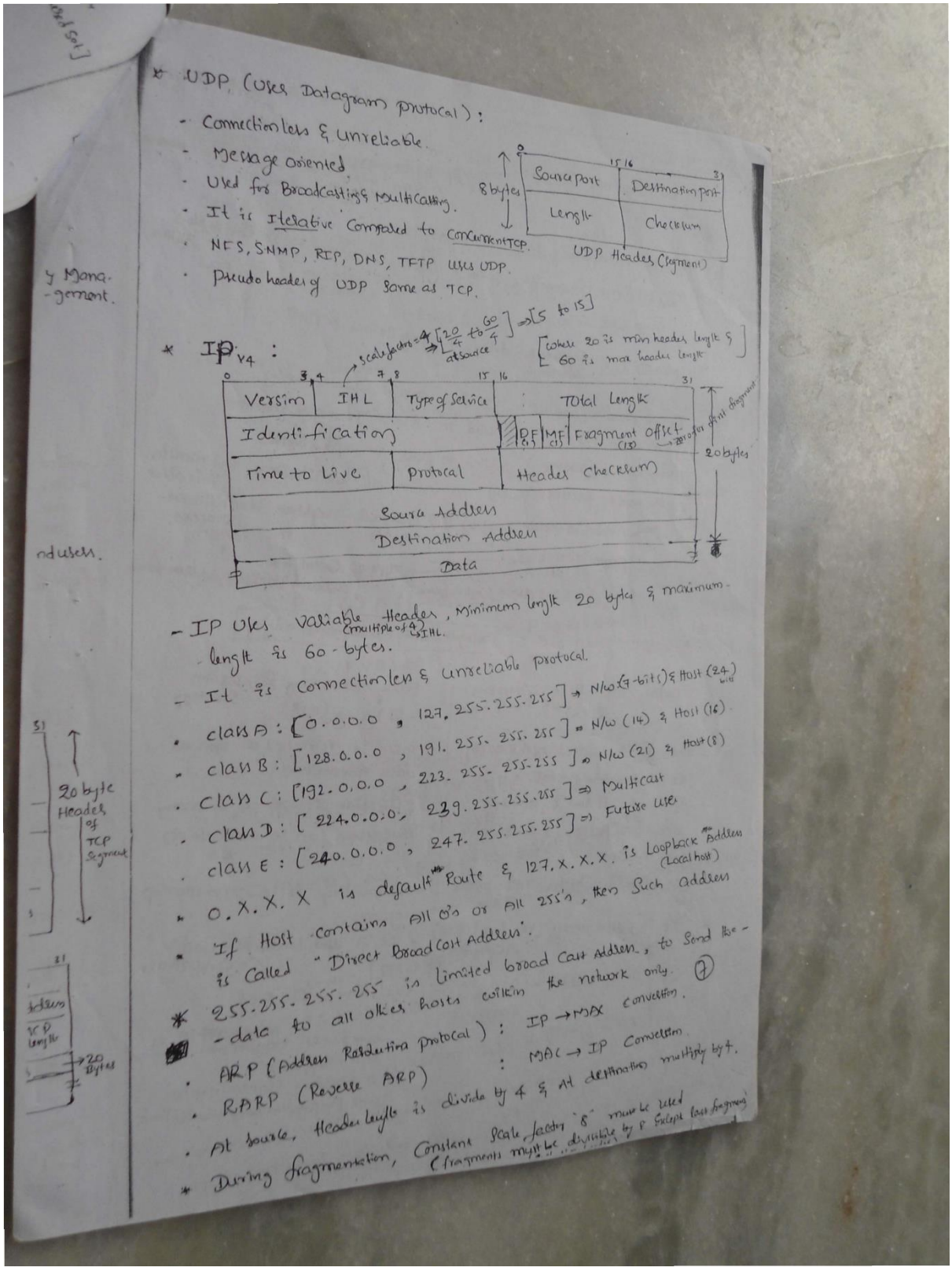
* Presentation Layer: [Formatting].
  - Encryption / Decryption (cipher text) (plain text) & Code translation
  - Compression / DeCompression.

* Application Layer:
  - Client-Server, Platform-independent, Synchronization b/w Endusers.
  - Network management.
  - FTP, Telnet, SMTP, DNS, HTTP, POP3 & ICMP.

* TCP [Transmission Control Protocal];
  - Connection oriented & Reliable
  - Byte oriend
  - Transport peer to peer protocal
  - Cumulative Acknowledge used
  - Uses Sliding window protocal (GBN & SR)
  - Not useful for Broad Calling & Multicalling
  - It is also called "Slow start protocal".
  - HTTP, HTTPS, FTP, Telnet & SMTP uses TCP.
  - Congestion window always Starts MSS=1 and increases Exponentially upto threshold value, then increases Linearly upto RWS (Receiver window size) and when time out occurs, congestion window reduced to 1 MSS & threshold value reduced to half of congestion window size.
  - Data Link layer uses Static RTT for time out & Transport Layer uses Dynamic RTT for time out calculation. (Round trip time)



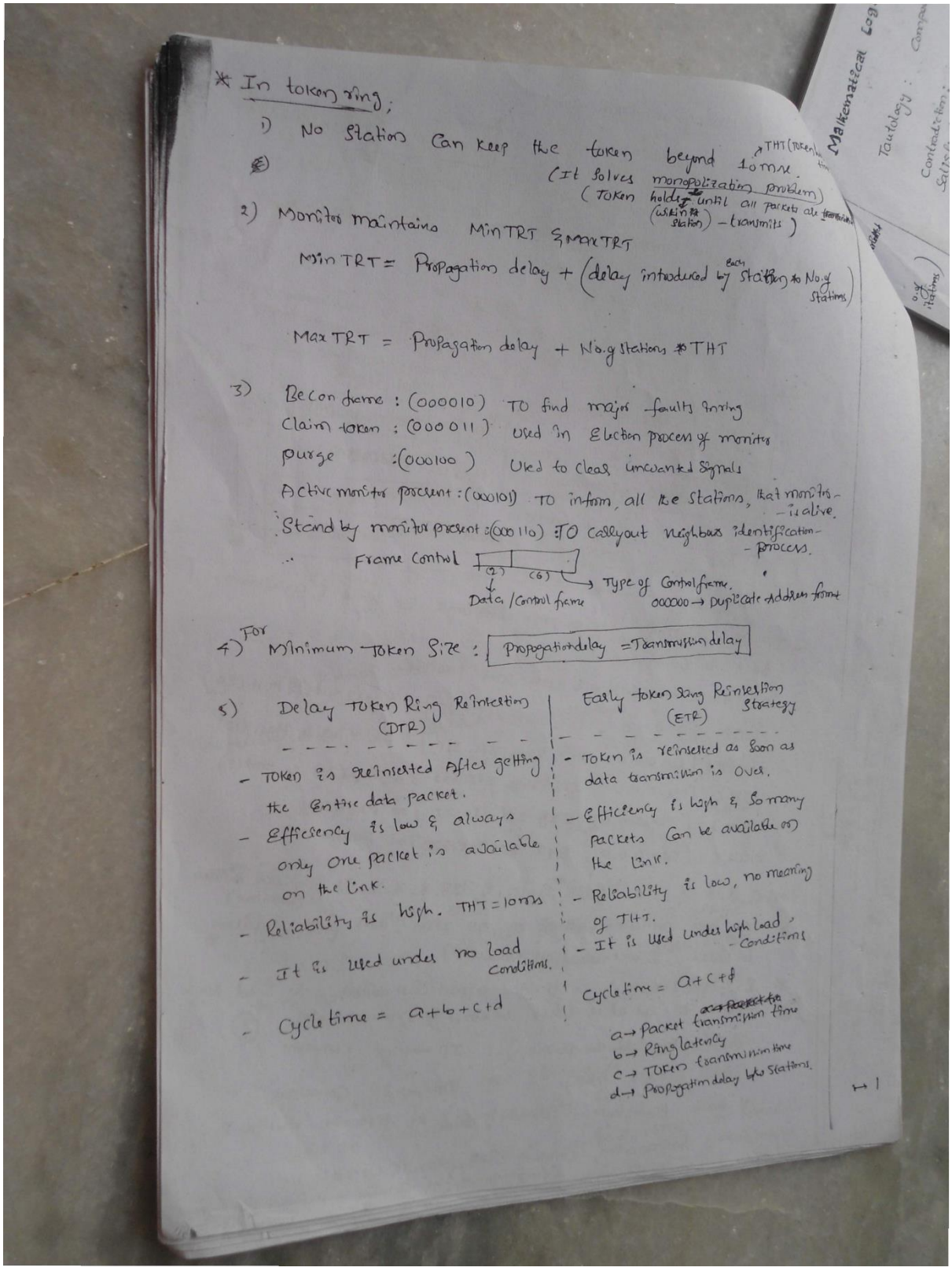




* UDP (User Datagram protocal):
  - Connectionless & Unreliable.
  - Message oriented.
  - Used for Broadcasting & Multicasting.
  - It is Iterative Compared to Concurrent TCP.
  - NFS, SNMP, RIP, DNS, TFTP uses UDP.
  - Pseudo header of UDP same as TCP.

* IP$_{V4}$:

- IP uses Variable Header, Minimum length 20 bytes & maximum length is 60 bytes. (multiple of 4 in IHL)
- It is Connectionless & Unreliable protocal.
- class A: [0.0.0.0 , 127.255.255.255] ⇒ N/w (7-bit) & Host (24 bits)
- class B: [128.0.0.0 , 191.255.255.255] ⇒ N/w (14) & Host (16)
- class C: [192.0.0.0 , 223.255.255.255] ⇒ N/w (21) & Host (8)
- class D: [224.0.0.0 , 239.255.255.255] ⇒ Multicast
- class E: [240.0.0.0 , 247.255.255.255] ⇒ Future use
- 0.X.X.X is default Route & 127.X.X.X is Loopback Address (Local host)
- If Host contains All 0's or All 255's, then Such address is called "Direct Broadcast Address".
- 255.255.255.255 is limited broad cast address, to send the data to all other hosts within the network only. (7)
- ARP (Address Resolution Protocal): IP → MAC convertion.
- RARP (Reverse ARP): MAC → IP Convertion.
- At source, Header length is divide by 4 & at destination multiply by 4.
- During fragmentation, Constant scale factor "8" must be used (Fragments must be divisible by 8 except last fragment)



* In token ring;

1) No station can keep the token beyond THT (Token holding time) time.
   (It solves monopolization problem)
   (Token holding until all packets are transmitted within the station) - transmits)

2) Monitor maintains MinTRT & MaxTRT

   MinTRT = Propagation delay + (delay introduced by each station * No.of stations)

   MaxTRT = Propagation delay + No.of stations * THT

3) Beacon frame : (000010) TO find major faults occuring
   Claim token : (000011) Used in Election process of monitor
   purge : (000100) Used to clear unwanted signals
   Active monitor present : (000101) TO inform, all the stations, that monitor is alive.
   Stand by monitor present : (000110) TO callyout neighbors identification process.

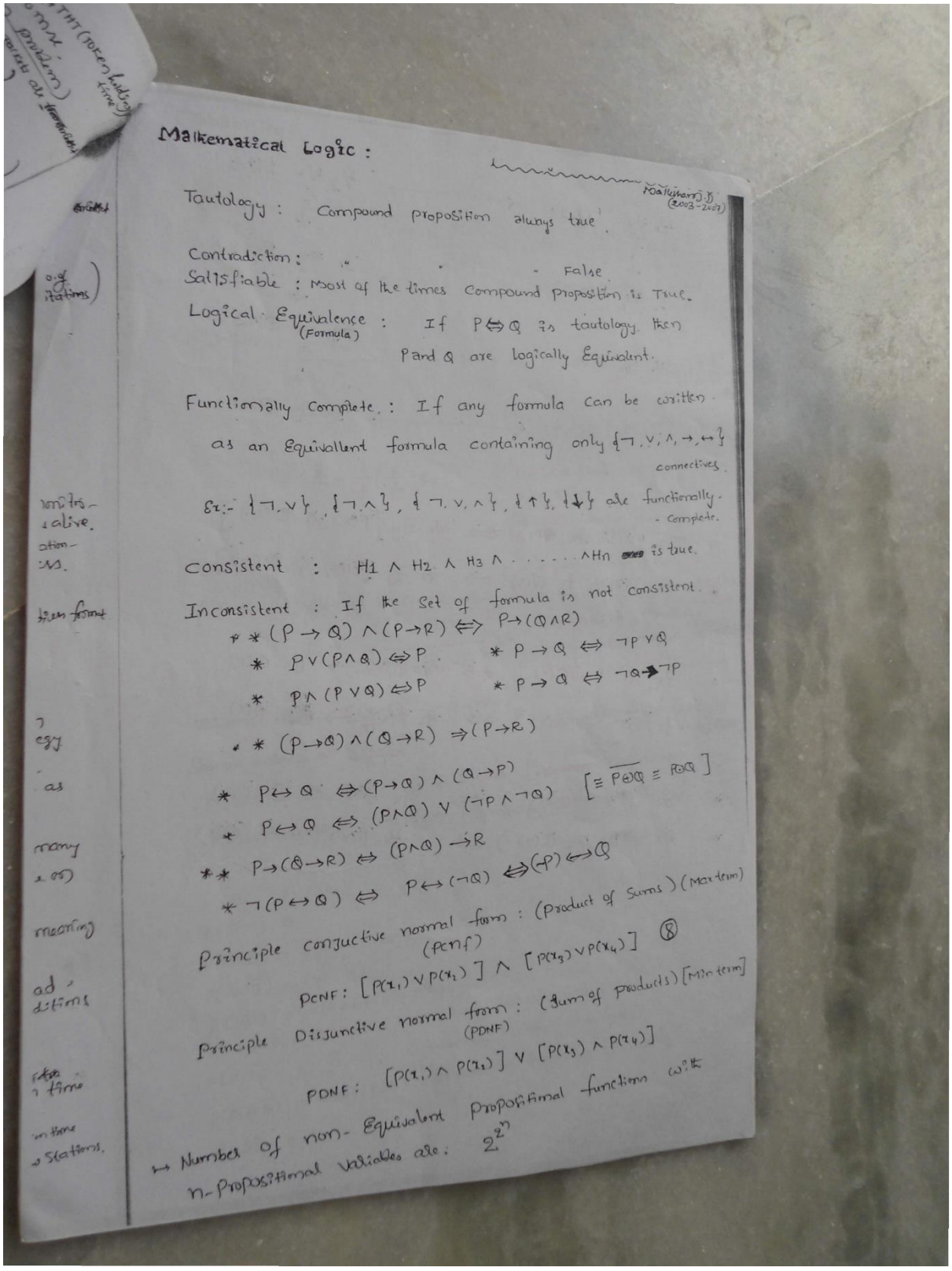
   Frame control: (2) (6) → Type of control frame. 000000 → Duplicate Address format
   (2) → Data / control frame

4) For Minimum Token Size : Propagation delay = Transmission delay

5)

Delay Token Ring Reinsertion (DTR)	Early token Ring Reinsertion Strategy (ETR)
- Token is reinserted after getting the entire data packet.	- Token is reinserted as soon as data transmission is over.
- Efficiency is low & always only one packet is available on the link.	- Efficiency is high & so many packets can be available on the link.
- Reliability is high. THT = 10ms	- Reliability is low, no meaning of THT.
- It is used under no load conditions.	- It is used under high load conditions
- Cycle time = a+b+c+d	Cycle time = a+c+d

a → Packet transmission time
b → Ring latency
c → Token transmission time
d → Propagation delay b/w stations.



Mathematical Logic :

Tautology : Compound proposition always true.

Contradiction : " " False

Satisfiable : Most of the times Compound proposition is True.

Logical Equivalence (Formula) : If $P \Leftrightarrow Q$ is tautology then P and Q are logically Equivalent.

Functionally Complete : If any formula can be written as an Equivalent formula containing only $\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$ connectives.

Ex:- $\{\neg, \vee\}$, $\{\neg, \wedge\}$, $\{\neg, \vee, \wedge\}$, $\{\uparrow\}$, $\{\downarrow\}$ are functionally complete.

Consistent : $H1 \wedge H2 \wedge H3 \wedge \ldots \wedge Hn$ is true.

Inconsistent : If the set of formula is not consistent.

* $(P \rightarrow Q) \wedge (P \rightarrow R) \Leftrightarrow P \rightarrow (Q \wedge R)$
* $P \vee (P \wedge Q) \Leftrightarrow P$
* $P \wedge (P \vee Q) \Leftrightarrow P$
* $P \rightarrow Q \Leftrightarrow \neg P \vee Q$
* $P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P$
* $(P \rightarrow Q) \wedge (Q \rightarrow R) \Rightarrow (P \rightarrow R)$
* $P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$
* $P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$ $[\equiv \overline{P \oplus Q} \equiv P \odot Q]$
* $P \rightarrow (Q \rightarrow R) \Leftrightarrow (P \wedge Q) \rightarrow R$
* $\neg (P \leftrightarrow Q) \Leftrightarrow P \leftrightarrow (\neg Q) \Leftrightarrow (\neg P) \leftrightarrow Q$

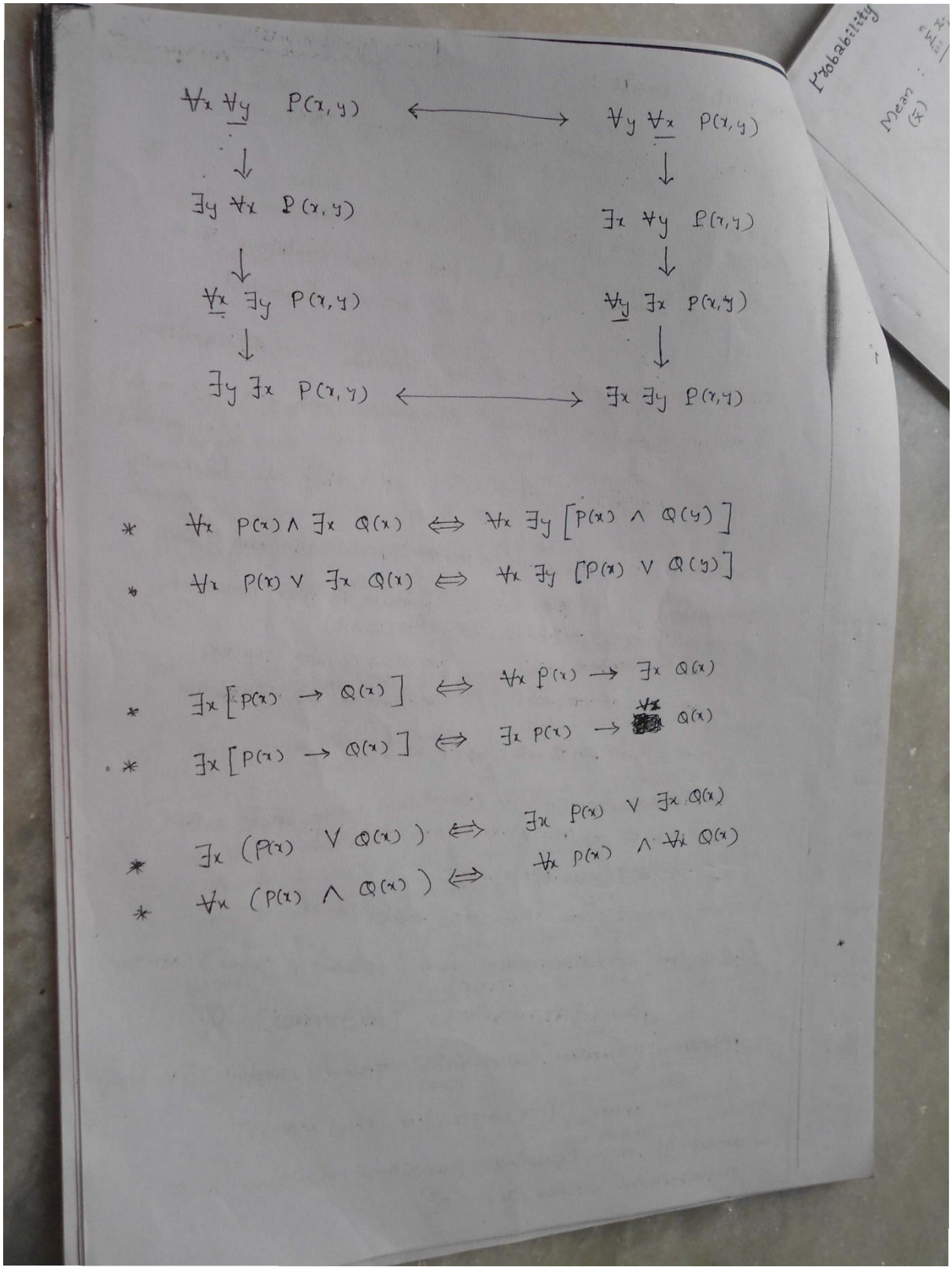
Principle conjuctive normal form (PCNF) : (Product of Sums) (Maxterm)

PCNF : $[P(x_1) \vee P(x_2)] \wedge [P(x_3) \vee P(x_4)]$ (8)

Principle Disjunctive normal form (PDNF) : (Sum of products) [Minterm]

PDNF : $[P(x_1) \wedge P(x_2)] \vee [P(x_3) \wedge P(x_4)]$

↦ Number of non-Equivalent propositional functions with n-propositional variables are : 2^{2^n}


$\forall x \forall y\ P(x,y) \longleftrightarrow \forall y \forall x\ P(x,y)$
$\exists y \forall x\ P(x,y)$
$\exists x \forall y\ P(x,y)$
$\forall x \exists y\ P(x,y)$
$\forall y \exists x\ P(x,y)$
$\exists y \exists x\ P(x,y) \longleftrightarrow \exists x \exists y\ P(x,y)$
$\forall x\ P(x) \wedge \exists x\ Q(x) \Leftrightarrow \forall x \exists y [P(x) \wedge Q(y)]$
$\forall x\ P(x) \vee \exists x\ Q(x) \Leftrightarrow \forall x \exists y [P(x) \vee Q(y)]$
$\exists x [P(x) \rightarrow Q(x)] \Leftrightarrow \forall x\ P(x) \rightarrow \exists x\ Q(x)$
$\exists x [P(x) \rightarrow Q(x)] \Leftrightarrow \exists x\ P(x) \rightarrow$ [redacted] $Q(x)$
$\exists x\ (P(x) \vee Q(x)) \Leftrightarrow \exists x\ P(x) \vee \exists x\ Q(x)$
$\forall x\ (P(x) \wedge Q(x)) \Leftrightarrow \forall x\ P(x) \wedge \forall x\ Q(x)$

Probability :

Mean ($\bar{x}$) : $\frac{\sum_{i=1}^{n} x_i}{n} = \frac{\sum_{i=1}^{n} f_i x_i}{\sum f_i} = A + \frac{\sum f_i d_i}{\sum f_i} = A + h \cdot \frac{\sum f_i u_i}{\sum f_i}$

$A \rightarrow$ mid value of class at maximum frequency only.

$d_i = x_i - A$

$u_i = \frac{x_i - A}{h}$

Median : $\frac{n}{2} + \left(\frac{n}{2}+1\right)$; n is Even

: $\frac{n+1}{2}$; n is odd

$= L + \frac{h}{C_f}\left(\frac{N}{2} - C_p\right)$; $L \rightarrow$ Lower limit of median class

$N \rightarrow \sum f_i$

$C_f \rightarrow$ Cumulative frequency of ($N/2$ contained) - median class.

$C_p \rightarrow$ Cumulative frequency of - preceding class. - median

Mode : Value of x corresponding to max. frequency.

$= L + \frac{f_m - f_1}{(f_m - f_1) + (f_m - f_2)} \times h$; $L \rightarrow$ Lower limit of modal class.

$f_m \rightarrow$ max. frequency (modal class)

$f_1 \rightarrow$ preceding frequency of - modal class

$f_2 \rightarrow$ Following frequency of - modal class.

Variance $= \sigma^2 = \frac{\sum (x - \bar{x})^2}{n}$; $\bar{x} \rightarrow$ mean

Sample variance $= \frac{\sum (x - \bar{x})^2}{n-1}$;

Mode = 3 Median − 2 Mean [Asymmetric distribution]

Mean = Mode = Median [Symmetric distribution]

$P(\bar{A}) = 1 - P(A)$

$P(\phi) = 0$

$P(A - B) = P(A) - P(A \cap B)$

$P(A \cap \bar{B}) = P(A - B)$

$P(A \cup B) = P(A) + P(B) - P(A \cap B)$

$P(A \cup B) = P(A) + P(B)$; mutually Exclusive Events (A & B)

$P(A \cap B) = \phi$; mutually Exclusive Events (A & B)

$P(A \cap B) = P(A) \cdot P(B)$; Independent Events (A & B) [arbitrary Events]

$P(S) = 1$

$$P(A_1 \cap A_2 \cap \dots \cap A_n) \geq \sum_{i=1}^{n} P(A_i) - (n-1)$$

* $$P(A_1 \cup A_2 \cup \dots \cup A_n) \leq \sum_{i=1}^{n} P(A_i)$$

$$P(A/B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A/B) \cdot P(B) = P(A \cap B)$$

* $P(A \cap B) = P(A) \cdot P(B)$; A & B are independent Events

$P(A/B) = P(A)$; A & B are independent Events.

$P(E_1 \cap E_2 \cap \dots \cap E_n) = P(E_1) \cdot P(E_2) \dots P(E_n)$; mutually independent.

$$P(E_2/X) = \frac{P(X/E_2) \cdot P(E_2)}{P(X/E_1) \cdot P(E_1) + P(X/E_2) \cdot P(E_2) + P(X/E_3) \cdot P(E_3)}$$

$$P(E_i/X) = \frac{P(X/E_i)\, P(E_i)}{\sum_{j=1}^{n} P(X/E_j)\, P(E_j)}$$

* Bernoulli Trials (Jacob Bernoulli) : $P(r) = {}^{n}C_r\, p^r q^{n-r}$

Random Variable (Stochastic Variable): It assigns a real number to each possible outcome.

Let X be a discrete random variable, then

* 1) $F(x) = P(X \leq x)$ is called distribution function of X. $\Rightarrow \sum_{i=0}^{x} P(i)$
* 2) Mean or Expectation of $X = \mu = E(X) = \sum_{i=1}^{n} x_i P(x_i)$
* 3) Variance of $X = \sigma^2 = E(X^2) - [E(X)]^2 = \sum_{i=1}^{n} (x_i - \mu)^2 P(x_i)$
* 4) Standard deviation of $X = \sigma = \sqrt{\text{Variance}}$
* 5) $\sum_{i=1}^{n} P(x_i) = 1$

Random Variable :

1) Discrete Random Variable :

"Finite Set of values" or "Countably infinite"

2) Continuous Random Variable : (Non-discrete)

"Infinite no. of uncountable values"

Discrete distributions ; (Binomial & Poisson)

** Binomial Distributions : [Fixed no. of trials 'n' & Probability of success 'p']

$P(X=r) = P(r) = {}^{n}C_r \, p^r q^{n-r}$ (Binomial / Bernoulli distribution)

$r = 0, 1, \ldots n$

$q = 1-p$

$E(x) = \text{mean} = np$

$V(x) = \text{Variance } (\sigma^2) = npq$

* Poisson Distribution : $(n \to \infty, p \to 0)$

$P(X=x) = \frac{e^{-\lambda} \cdot \lambda^x}{x!}$; $x = 0, 1, 2, \ldots \infty$

$\lambda = \text{mean } (\mu)$

$= \text{Variance } (\sigma^2)$

$\boxed{\lambda = np}$; $q = 1$

Continuous Distribution : (Uniform, Exponential & Normal distributions)

$X \to$ Continuous Random variable.

Density functions:

* 1) $P(X \le a) = \int_{-\infty}^{a} f(x).dx$

* 2) $P(a \le X \le b) = \int_{a}^{b} f(x).dx$

* 3) $E(x) = \text{Mean} = \int_{-\infty}^{\infty} x \cdot f(x).dx$

* 4) Variance of $X = V(X) = \int_{-\infty}^{\infty} [x - E(x)]^2 \cdot f(x).dx$

* 5) $\int_{-\infty}^{\infty} f(x).dx = 1.$

(10)

* Uniform Distribution : (Rectangular Distribution)

- Density function : $f(x) = \frac{1}{b-a}$; $a \le x \le b$

  $= 0$; otherwise

- Cumulative function : $f(x) = P(X \le x)$

  $= \int_{-\infty}^{x} f(x).dx.$

$$P(x \le x) = \int_{-\infty}^{x} f(x).dx = \begin{cases} 0 & ; \text{if } x \le a \\ \frac{x-a}{b-a} & ; \text{if } a \le x \le b \\ 1 & ; \text{if } x > b \end{cases}$$

$$\text{Mean} = \frac{a+b}{2} \quad ; \quad \text{Variance } (\sigma^2) = \frac{(b-a)^2}{12}$$

* Exponential Distribution :

Density function : $f(x) = \lambda . e^{-\lambda x}$; $x > 0$

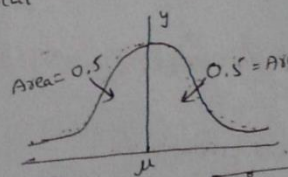
$= 0$; otherwise

$$\text{Mean} = \frac{1}{\lambda} \quad ; \quad \text{Variance } (\sigma^2) = \frac{1}{\lambda^2} \quad \left[\text{or. Mean} = S.D = \frac{1}{\lambda}\right]$$

* Normal Distribution :

Density function : $f(x) = \frac{1}{\sigma\sqrt{2\pi}} . e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$; $-\infty \le x \le \infty$, $\sigma > 0$, $-\infty < \mu < \infty$

- Normal Distribution is Symmetrical
- Mean $= \mu$; Variance $= \sigma^2$
- Mean = Median = Mode
- $f(x) \ge 0$ for all x
- $\int_{-\infty}^{\infty} f(x).dx = 1$ & Variate $(z) = \frac{x-\mu}{\sigma} = \frac{x-\text{Mean}}{\text{S.D}}$



$\int_{-\infty}^{0} P(z) = 0.5$

$\int_{0}^{\infty} P(z) = 0.5$

* $P(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$; $-\infty \le z \le \infty$ & $z = \frac{x-\mu}{\sigma}$

$z \rightarrow$ Standard normal variate

Sets :

Set is a collection of objects & objects are definite and distinguishable.

Symmetric Difference [$A \oplus B$ or $A \Delta B$]:

$$A \Delta B = (A-B) \cup (B-A) = (A \cup B) - (A \cap B)$$
$$= \{x / x \in A \text{ or } x \in B \text{ but not both}\}$$

Cardinality : Number of elements in the set ($|A|$)

Power set : Set of all subsets of any finite set. ($P(A)$)

* If $|A| = m$ then $|P(A)| = 2^m$ elements; & $|\phi| = 0$
  $\Rightarrow |P(\phi)| = 1$, i.e. $P(\phi) = \{\phi\}$ (↳ Null set) & $P(\{\phi\}) = \{\phi, \{\phi\}\}$

Comparable : If $A \subseteq B$ or $B \subseteq A$ then A & B are comparable.

* If $|A| = n$ then $|P(A \times A)| = 2^{n^2}$ elements.
* If $|A| = m$ & $|B| = n$ then no. of relations from A to B $= 2^{mn}$
  (or) $|P(A \times B)| = 2^{mn}$
* A, B, C & D are four sets then the no. of regions of Universal set 'U' divided into : 2^4 regions

$A - (B \cup C) = (A-B) \cap (A-C)$ & $A - (B \cap C) = (A-B) \cup (A-C)$

* Principle of Extension : Two sets A & B are Equal iff they have same members (elements).
* Principle of Abstraction : Given Set 'U' and property 'P', there is Set 'A' such that the elements of Set 'A' are Exactly those members (elements) of U which have property 'P'.
* Inclusion - Exclusion principle :

$$n(A_1 \cup A_2 \cup \dots \cup A_n) = \sum_{1 \le i \le n} n(A_i) - \sum_{1 \le i < j \le n} n(A_i \cap A_j) + \sum_{1 \le i < j < k \le n} n(A_i \cap A_j \cap A_k) - \dots + (-1)^{n-1} n(A_1 \cap A_2 \cap \dots \cap A_n)$$

(11)

* $A - B = A \setminus B = A$ but not B [Relative Complement of two sets]
* For any set A; $A \subseteq A$ & $\phi \subseteq A$ [A & ϕ are called Trivial subsets of A]
* $A \cup \sim A = U$; $\sim \phi = U$; $A \times (B \cap C) = (A \times B) \cap (A \times C)$

Relations :

If $n(A)=p$, $n(B)=q$ then total no.of relations $= 2^{pq}$ (A to B)

If $n(A)=p$, then No.of relations from A to A $= 2^{p^2}$

Inverse Relation $R^{-1} = \{(b,a) / (a,b) \in R\}$

Diagonal Relation $\Delta_A = \{(a,a) / a \in A\}$

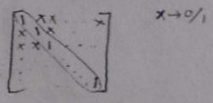
Composition of relations R & S (RS or RoS):

$R \subseteq A \times B$, $S \subseteq B \times C$ $\Big\}$ $RS = RoS = \{(a,c) / \exists b \in B$ for which $(a,b) \in R$ & $(b,c) \in S\}$

* $(RoS)oT = Ro(SoT)$

Reflexive Relation : If $(a,a) \in R$, $\forall a \in A$

i.e., If $\boxed{\Delta_A \subseteq R}$

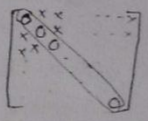


If $i=j$ then $M_{ij}=1$

Irreflexive Relation :

If $(a,a) \notin R$, $\forall a \in A$

If $i=j$ the $M_{ij}=0$

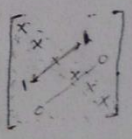


Symmetric Relation :

If $(a,b) \in R$ then $(b,a) \in R$, $\forall a,b \in R$

i.e., If $\boxed{R = R^{-1}}$

If $M_{ij}=1$ then $M_{ji}=1$



Antisymmetric Relation:

def: * If $(a,b) \in R$ & $(b,a) \in R$ then $a=b$, $\forall a,b \in R$

[Note: * If $(a,b) \in R$ then $(b,a) \notin R$, $\forall a,b \in R$ [Asymmetric Relation

i.e., If $\boxed{R \cap R^{-1} \subseteq \Delta_A}$ Antisymmetric.

* If $M_{ij}=1$ & $M_{ji}=1$ then $i=j$ [Antisymmetric]

Transitive Relation : If $R \circ R \subseteq R$

If $(a,b) \in R$ & $(b,c) \in R$ then $(a,c) \in R$

No. of Reflexive Relations $= 2^{n^2-n} = 2^{n(n-1)}$

" Irreflexive " $= 2^{n^2-n} = 2^{n(n-1)}$

" Symmetric " $= 2^{\frac{n^2-n}{2}} \cdot 2^n = 2^{n(n+1)/2}$

" ASymmetric " $= 3^{\frac{n^2-n}{2}} = 3^{n(n-1)/2}$

" Antisymmetric " $= 3^{\frac{n^2-n}{2}} \cdot 2^n = 3^{n(n-1)/2} \cdot 2^n$

" Reflexive & Symmetric " $= 2^{\frac{n^2-n}{2}} = 2^{n(n-1)/2}$

" Neither Reflexive Nor Irreflexive $= 2^{n^2} - 2 \cdot 2^{n(n-1)}$

* Comparability : Reflexive & Symmetric
* Quasiorder : Irreflexive & Transitive
* Partial order : Reflexive, Antisymmetric & transitive.
* Equivalence : Reflexive, Symmetric & Transitive. (2)

. ϕ is not reflexive [Empty Relation]

. $A \times A$ is Reflexive [universal relation].

* Partition: Partition of 'S' is a collection $P = \{A_i\}$ of non-Empty subsets of 'S',

1) Such that Each a in S belongs to exactly one of the A_i [cell].

2) The sets of P are mutually disjoint: If $A_i \neq A_j$ then $A_i \cap A_j = \phi$

Ex:- $S = \{1,2,3,4,5,6\} \Rightarrow P_1 = [\{1,2\}, \{3,4,6\}, \{5\}]$

* Each Element of 'S' belongs to Exactly one of the cell in 'P'.

* ϕ does not belongs to a partition.

. No. of partitions with Set $\{1,2,3\}$:

$P = \{1,2,3\}$, $P_1 = \{\{1\},\{2\},\{3\}\}$, $P_2 = \{\{1\},\{2,3\}\}$, $P_3 = \{\{1,2\},\{3\}\}$, $P_4 = \{\{1,3\},\{2\}\}$, $P_5 = \{\{1,2,3\}\}$ → 5 partitions

↦ Let P_1 & P_2 are two partitions of S then Intersection of each cell in P_1 with each cell in P_2 is called "Cross partition" is also a partition

Ex:- $P_1 = [\{1,2,3\}, \{4,5,6\}]$

$P_2 = [\{1,3,4,6\}, \{2,5\}]$

P = Cross partition of P_1 & P_2 = $[\{1,3\}, \{2\}, \{4,6\}, \{5\}]$

** Let P be Cross partition of P_1 & P_2 of set 'S' and P_1 has x-cells, P_2 has y-cells then No. of cells in P' (n) are : $\max(x,y) \le n \le xy$

* Let $f(n,k)$ represents the no. of partitions of set 'S' of n-Elements into K-Cells ($k = 1, 2, \ldots n$)

$f(n,1) = 1$

$f(n,n) = 1$

$$\boxed{f(n,k) = f(n-1, k-1) + k \cdot f(n-1, k)}$$

no. of partitions
$n=0 \Rightarrow 1$
$n=1 \Rightarrow 1$
$n=2 \Rightarrow 2$
$n=3 \Rightarrow 5$
$n=4 \Rightarrow 15$
$n=5 \Rightarrow 52$
$n=6 \Rightarrow 203$

* No. of Equivalence Relations of set 'S' with n-Elements:

$$P(n) = \sum_{j=0}^{n-1} {}^{n-1}C_j \; P(n-1-j) \quad ; \quad P(0) = 1$$

$P(1) = 1$
$P(2) = 2$
$P(3) = 5$
$P(4) = 15$
$P(5) = 52$
$P(6) = 203$

* Every Equivalence relation induces a partition.

* $x \equiv y \pmod m$ is an Equivalence relation [x is congruent to y modulo m (or) x-y is divisible by m (or) $x = y + Km$ (i.e K is integer)]

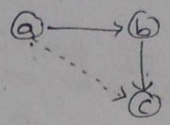
* $(a,b) \simeq (c,d)$, whenever $ad = bc$ then $\simeq$ is an Equivalence relation on A×A.

* $(a,b) \sim (c,d)$; If $a+d = b+c$ then $\sim$ is an Equivalence relation on A×A.

* Let 'R' be a relation on set 'A'

1) Reflexive closure of R = $R \cup \Delta_A$
2) Symmetric closure of R = $R \cup R^{-1}$
3) Transitive closure of R = R^*

If (a,b) & (b,c) belongs to R and hence to R^* then (a,c) is belongs to R^* for all pairs of (x,y)



** If A has n-Elements; $A = \{1,2,3,\ldots n\}$ & a Relation 'R' on A with n-pairs $R = \{(1,2)(2,3),\ldots(n-1,n),(n,1)\}$ then transitive closure of R (R^*) is the "universal Relation" $A \times A$ containing n^2 pairs.

* If R & S are Reflexive Relations on set A then $R \cap S$ & $R \cup S$ & R^{-1} is Reflexive
" Symmetric " " then $R \cap S$ & $R \cup S$ & R^{-1} is Symmetric
" Transitive " " then $R \cap S$ is transitive.
" Transitive " " then $R \cup S$ need not be transitive.

* If R is Reflexive on A then R^{-1} & (RUS) are reflexive for any relation 'S' on A

(13)

* If R is Antisymmetric relation on Set 'A' then R^{-1} & (R∩S) are Antisymmetric for any relation S on A.

* If R is any relation on A then $R \cup R^{-1}$ is Symmetric
* If R is Symmetric relation on A then $R = R^{-1}$, Hence R^{-1} is also Symmetric
* If R & S be any two Equivalence relations on Set 'A' then $R \cap S$ is also Equivalence relation, but $R \cup S$ need not be an Equivalence.
* Every Asymmetric relation is Antisymmetric, but need not converse.

Functions:

$f: A \to B$ is a function from A to B :

* $f \subseteq (A \times B)$ "Reflexive"
* $\forall a \in A, \exists b \in B ; \ni (a,b) \in f$
  $\therefore f(a) = b$ "Existence"
* If $(a,b) \in f$ & $(a,c) \in f$ then $b = c$ "Uniqueness"

- Number of functions from a relation A to B = $|B|^{|A|}$
- Into function : $f: A \to B$; If $f(A) \subset B$

* Onto [Surjective] function : If $f(A) = B$; $\boxed{n(A) \geq n(B)}$
  [Atleast one image in 'B' is used for more than one element in A.]
* [Injective] One-one function: If $a_1 \neq a_2$ then $f(a_1) \neq f(a_2)$
  $\boxed{n(A) \leq n(B)}$ If $f(a_1) = f(a_2)$ then $a_1 = a_2$
  [distinct Elements in A has distinct images in B] $\forall a_1, a_2 \in A$
* Bijection : $f: A \to B$; If f is onto & one-one
  $\boxed{n(A) = n(B)}$

* Number of functions $f: A \to B = |B|^{|A|}$
* Number of onto functions $f: A \to B = \sum_{i=1}^{|B|} (-1)^i \, {}^{|B|}C_i \; i^{|A|}$
* Number of one-one functions: $f: A \to B = {}^{|B|}P_{|A|}$
* Number of Bijection functions : $f: A \to B = n!$
  [let $n(A) = n(B) = n$]

- Constant function: If $f(a) = b$; $\forall a \in A$
- Identity function : If $f(a) = a$; $\forall a \in A$

- If $f: A \to B$ is bijective then $f^{-1}: B \to A$ is also bijective.
  and the Inverse mapping of f is unique.

- If $f: A \to B$ & $g: B \to C$ are Injective (one-one) then $gof: A \to C$ is also Injective.
- If $f: A \to B$ & $g: B \to C$ are Surjective (on-to) then $gof: A \to C$ is also Surjective.
- If $f: A \to B$ & $g: B \to C$ are functions, $\exists$ $gof: A \to C$ is Injective then f is also Injective (one-one)
- If $f: A \to B$ & $g: B \to C$ are functions, $\exists$ $gof: A \to C$ is Surjective then g is also Surjective (on-to)

N → natural numbers $(1, 2, 3, \ldots)$

Z → Integers $(\ldots -2, -1, 0, 1, 2, \ldots)$

Q → Rational numbers

R → Real numbers

C → Complex numbers

$$N \subseteq Z \subseteq Q \subseteq R \subseteq C$$

Groups :

- closure [Binary operation] (*) : [A is a function * from A×A into A]

  A is closed under * , if $a*b \in A$, $\forall a,b \in A$

** Number of binary operations on a set 'G' (Closure) : $|G|^{|G|\times|G|}$

- Closure : $a*b \in G \quad \ni \forall a,b \in G$
- Associative : $(a*b)*c = a*(b*c) \quad \ni \forall a,b,c \in G$
- Identity : $a*e = e*a = a \quad \ni \forall a \in G \Rightarrow e$ is Identity
- Inverse : $a*b = b*a = e \quad \Rightarrow a^{-1}=b \;\&\; b^{-1}=a$
- Commutative : $a*b = b*a \; ; \ni \forall a,b \in G$

* Groupoid : closure

* Semigroup : closure + Associative

* Monoid : closure + Associative + Identity

* Group : closure + Associative + Identity + Inverse

* Abelian (Commutative) : Group + Commutative.

Groupoid	Semigroup	Monoid	Group	Abelian
$a*b \in G$ (closure) $\forall a,b \in G$	$(a*b)*c = a*(b*c)$ (Associative) $\forall a,b,c \in G$	$a*e = e*a = a$ $\forall a \in A$ (Identity)	$a*b = b*a = e$ (Inverse)	$a*b = b*a \; ; \forall a,b \in G$ (commutative)
$(N, +, *)$ (* : multiplication)	$(N, +, *)$	$(N, *)^{e=1}$	Non-Singular matrices closed under * (multiplication)	$(\{0,1,2,3\}, +_4)$
$(Z, +, -, *)$	$(Z, +, *)$	$(Z, +)^{e=0}$		$(Z, +)^{e=0}$
$(R, +, -)$	$(R, +)$	$(R, *)^{e=1}$		$(R, +)^{e=0}$
$(R-\{0\}, *, /)$	$(R-\{0\}, *)$	$(\{a,b\}^*, +)$		$(R-\{0\}, *)^{e=1}$
	- & ÷ are always not associative			$(Q, +)^{e=0}$
				$(Q-\{0\}, *)^{e=1}$
				$(\{1,-1,i,-i\}, *)$
				$(\{1,\omega,\omega^2\}, *)$
				$(\{1,-1\}, *)$
				$(\{1\}, *)$

* Lexicographical order / Dictionary order:

Let A_1 & A_2 are Partial ordered sets,
The lexicographical ordering $\leq$ on $A_1 \times A_2$ is defined as:

1) $(a_1, a_2) < (b_1, b_2)$

Either "if $a_1 < b_1$" or "both $a_1 = b_1$ & $a_2 < b_2$"

(or)

2) $(a_1, a_2) \leq (b_1, b_2)$

Either "if $a_1 < b_1$" or "both $a_1 = b_1$ & $a_2 \leq b_2$".

* Well-ordered Set:

An ordered set 'A' is well-ordered if Every - Subset of 'A' contains a "first Element" [least Element].

* "Finite linearly ordered set" is well-ordered.

** Every well ordered set must be linearly ordered. (chain)

Ex:- $(\mathbb{N}, \leq)$ (Natural numbers) is well-ordered
$(\mathbb{Z}, \leq)$ is not well-ordered.

First Element (Least): Let A be an ordered set, The Element 'a' in 'A' is first Element of A if, for Every Element 'x' in A, $a \leq x$.

Last Element (greatest): Let A be an ordered set, The Element 'b' in 'A' is last Element of A if, for Every Element 'x' in A, $x \leq b$.

Ex:- 1) Let N be the set of Natural numbers (1, 2, 3, ...)
Then, First Element of N = 1 & there is no last-Element.

2) Let 'A' be any set and Let P(A) be the - Set of all Subsets of A [Power set of A].
Then, First Element of $P(A) = \phi$
Last Element of $P(A) = A$.

- $O(G) \leq 5$ is always "Abelian group"
- Every order of group is prime is cyclic group & Every cyclic group is Abelian group. [Ex:- $O(G) = 5$ (prime)]
- $(\{0,1,2,\ldots m-1\}, +_m)$ Addition modulo is Abelian group
- $(\{1,2,3,\ldots p-1\}, \times_p)$ Multiplication modulo is Abelian group.
- If $O(G) = 2n$ then there Exist atleast one Element other than identity Element which is "Self Invertible".
- Set of all non-singular matrices is a group under matrix multiplication, but not abelian. (commutative, may or may not satisfied)
- order of an Element $O(a) = n$; & $O(a) = O(a^{-1})$ where $a \in G$ & $a^n = e$

[Ex: $\{1,-1,i,-i\} \Rightarrow e=1$ $(-1)^2 = 1 = e \Rightarrow O(-1) = 2$ & $(i)^4 = 1 = e \Rightarrow O(i) = 4$]

- $a \oplus_m b = r\left(\frac{a+b}{m}\right)$; Identity $e = 0$
- $a \otimes_p b = r\left(\frac{a+b}{p}\right)$; Identity $e = 1$
- Order of "group" : is the no. of Elements in the group.

Partial order : (Reflexive, Antisymmetric & Transitive)

A Set 'P' with a partial order is called a "Poset" [ordered set]

Ex:- $(\mathbb{N}, \leq)$, $(\mathbb{N}, /)$, $(P(S), \subseteq)$

* ① Partially order on a set 'S' to be a Relation R.
  - * Reflexive : aRa, $\forall a \in S$
  - * Antisymmetric : If aRb & bRa then $a=b$
  - * Transitive : If aRb & bRc then aRc.

② Partially order on a set 'S' to be a Relation R.
- * Irreflexive : $a \not R a$, $\forall a \in S$
- * Asymmetric : If aRb then $b \not R a$
- * Transitive : If aRb & bRc then aRc.

"$\leq$ satisfies ①" if and only if "$<$ satisfies ②", or

If R satisfies ② then $R \cup \Delta$ satisfies ① of a partial order.

* Hasse diagram [upward allow diagraph] :

Any two vertices are adjacent iff one is covered by the other. [Never contains any cycles in it]

** Cover of an Element is called "Immediate Successor".

* Linearly / totally ordered Set [chain] :

If Every pair of Elements in 'S' are Comparable, then such ordered Set 'S' is said to be "Linearly ordered".

$\therefore \forall a, b \in S$: if a & b are comparable.

Ex:- $(N, \leq)$

- Any Set Consisting of one Element is "Linearly ordered".
- Any pair of Comparable Elements is "Linearly ordered"
- * Each Element of linearly ordered set can have atmost one immediate-Predecessor".

Minimal : Elements which are not having predecessors.

Maximal : Elements which are not having successors.

* Many minimal & maximal's may exist.

Least : 'a' is a least Element of 'P' : if $a \leq x$; $\forall x \in P$

Greatest : 'b' is a greatest Element of 'P' ; if $x \leq b$; $\forall x \in P$.

* Lower bound : Let $A \subseteq P$, a is a lower bound of A, if $a \in P$ and $a \leq x$ $\forall x \in A$

* Upper bound : Let $A \subseteq P$, b is an upper bound of A, if $b \in P$ and $x \leq b$ $\forall x \in A$.

Greatest Lower bound [Infimum] :

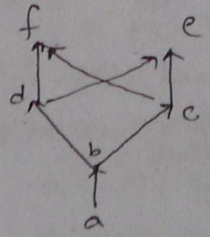
'y' is infimum of A, if y is a lower bound of 'A' and if 'z' is any other lower bound of A then $z \leq y$.

Least Upper bound [Supremum] :

'x' is supremum of A, if x is an upper bound of 'A' and if 'z' is any other upper bound, then $x \leq z$.

- If only one minimal Exist then it is always 'least'.
- If only one maximal Exist then it is always 'greatest'.

Ex:- $P = \{a, b, c, d, e, f\}$



Minimal = a
Maximal = e, f
least = a
No greatest

Let $S = \{b, c, d\}$
$S \subseteq P$
Lower bound of S = b, a
Upper bound of S = e, f
Infimum = b
No supremum.

* Immediate successors of lower bound are called "atoms".

Ex:- If b is lower bound, d & c are atoms from above diagram.

Lattice :

Let $(P, \leq)$ is a poset, In which for Every two Elements there Exist infimum / Greatest Lower bound / meet ($\wedge$) and Supremum / Least upper bound / Join ($\vee$) then such Poset is called a "Lattice".

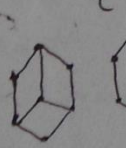
(Or)

Let 'L' be a non Empty set closed under two binary operations called "meet" ($\wedge$) and "Join" ($\vee$). Then 'L' is a "lattice" if for any Elements a, b, and c of 'L' the following axioms hold. [$(L, \wedge, \vee)$ is lattice]

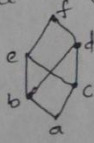
1) Commutative Laws:
   (i) $a \wedge b = b \wedge a$, (ii) $a \vee b = b \vee a$
2) Associative Laws:
   (i) $(a \wedge b) \wedge c = a \wedge (b \wedge c)$, (ii) $(a \vee b) \vee c = a \vee (b \vee c)$
3) Absorption Laws:
   (i) $a \wedge (a \vee b) = a$ (ii) $a \vee (a \wedge b) = a$

(17)

Ex:- $(N, /, gcd, lcm)$ is a lattice
$(N, \leq, Min, Max)$ is a lattice
$(L, \leq, \wedge, \vee)$ is a lattice
$(P(S), \subseteq, \cap, \cup)$ is a lattice



& are lattices



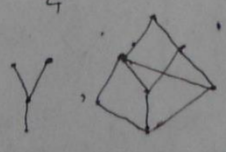
{b, c} has 3 upper bounds d, e, f and No one of them precedes other two, is not a lattice

→ Every chain is a lattice [i.e., linearly ordered set is a lattice]

↔ Let 'L' be a lattice. Then $a \wedge b = a$ if and only if $a \vee b = b$

* $x \wedge y = \inf(x, y)$ & $x \vee y = \sup(x, y)$

is a lattice,



& are not lattices

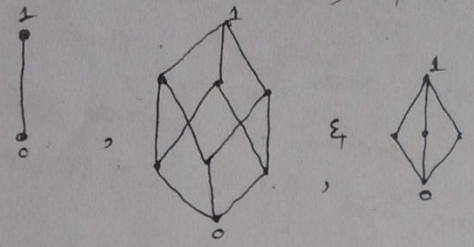
* Bounded lattice:

If there Exist Lower bound (l) & upper bound (u) in a lattice, such lattice is called "Bounded lattice".

i.e., If $l \in L$ & $u \in L$ then $l \leq x \leq u$, $\forall x \in L$

Ex:- $(L, \leq, \wedge, \vee)$ & $(P(S), \subseteq, \cap, \cup)$ are bounded lattices.

$(\mathbb{N}, \leq, \text{Min}, \text{Max})$ & $(\mathbb{N}, /, \text{gcd}, \text{lcm})$ are not bounded.



, , & are bounded lattices.

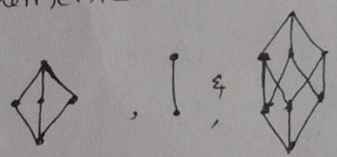
[0 → Lower bound, 1 → upper bound]

** Every Finite lattice is "bounded lattice".

* Complemented lattice:

In a bounded lattice. If there Exist Complement for Every Element then Such bounded lattice is called "Complemented lattice".

If $x \vee y$ = upper bound & $x \wedge y$ = Lower bound, then x & y are Complements to each other

Ex:-  are Complemented lattices.

Distributive lattice: If a lattice 'L' satisfies:

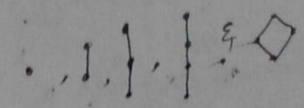
a) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$

b) $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$

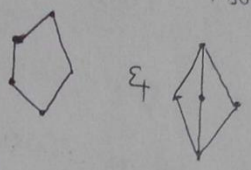
$\forall a, b, c \in L$.

* Let 'L' be a bounded distributive lattice, Then Complements are unique if they Exist.

* A lattice with less than 5-Elements is always "Distributive".

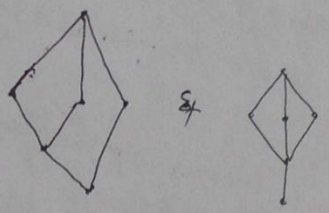
 are Distributive lattices.

* A lattice 'L' is non-distributive if and only if it contains a sublattice isomorphic to the following lattices:



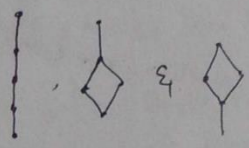
[non-Distributive lattices]

Ex:-



are non-distributive lattices (isomorphic to above lattice)

Ex:-



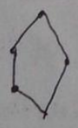
are distributive (5-Elements)

** Modulas lattice: If a lattice 'L' satisfies:

$a \vee (b \wedge c) = (a \vee b) \wedge c$; $\forall a,b,c \in L$ and $a \le c$.

Ex:-

is modulas (& non-distributive)



is non-modulas (& non-distributive)

(3)

* Every distributive lattice is modulas.

* Sublattice: A lattice 'L' is called "Sublattice", if under same meet ($\wedge$) & same join ($\vee$).

Ex:- $(D_{12}, /, gcd, lcm)$ is Sublattice.

** Dual order: $(P, \le)$ is poset then $(P, \ge)$ is also a poset, such poset is called "dual order".

* Dual lattice: $(L, \le, \wedge, \vee)$ is a lattice $(L, \ge, \vee, \wedge)$ is also a lattice, such lattice is called "Dual lattice".

* Complete lattice: A lattice 'L' is said to be complete if every subset of 'L' has infimum & supremum in 'L'.

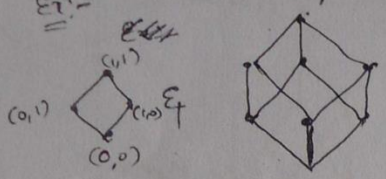
Boolean Algebra : $(B, \leq, \wedge, \vee, ')$

"If a lattice is Complemented & Distributive lattice."

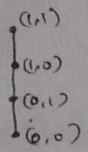
Ex:- $(P(S), \subseteq, \cap, \cup)$

- Let B be a Boolean Algebra. Then B is a partially ordered set, where $a \leq b$ is defined by $a+b=b$.

Ex:-



& is a Boolean Algebra (Complemented & Distributive)



is not Boolean Algebra (not complemented)

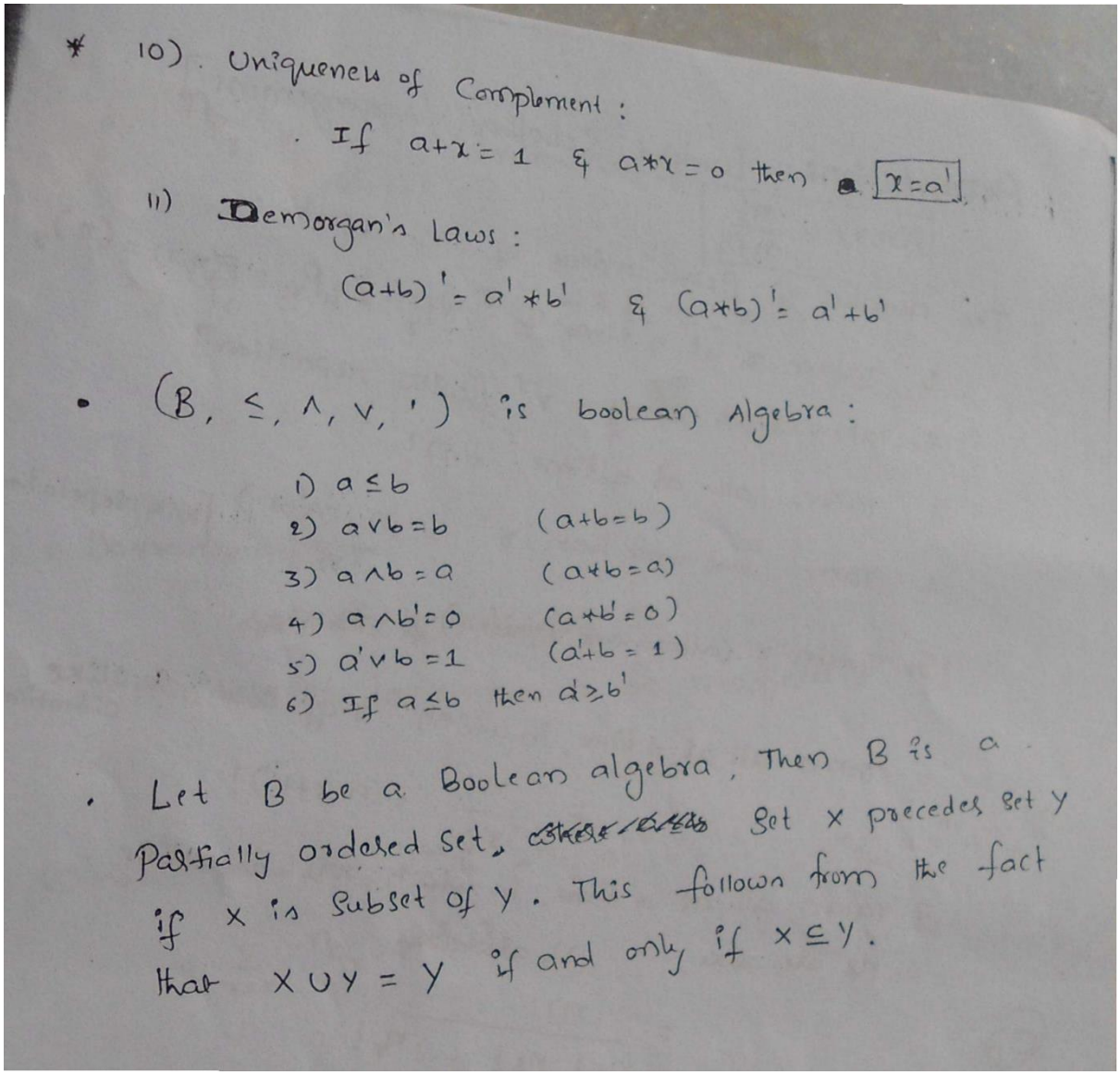
- Boolean Algebra Satisfies : lattice [Poset, meet, join], Bounded [lower, upper], Distributed & Complemented lattices.

* Let B be a finite boolean algebra having n-atoms. Then B has 2^n Elements. and "Every non-zero Element of B is the Sum of unique set of atoms."

Ex:- B is boolean algebra with less than 100 Elements. Then B can have $2^1, 2^2, 2^3, 2^4, 2^5$ or 2^6 Elements. (n = no. of atoms) (< 100)

* Let a, b, c be any Elements in a Boolean Algebra $(B, +, *, ', 0, 1)$

1) Commutative Laws: $a+b=b+a$ & $a*b=b*a$
2) Distributive Laws: $a+(b*c)=(a+b)*(a+c)$ & $a*(b+c)=(a*b)+(a*c)$
3) Identity Laws : $a+0=a$ & $a*1=a$
4) Complement Laws : $a+a'=1$ & $a*a'=0$.
5) Idempotent Laws : $a+a=a$ & $a*a=a$
6) Boundedness laws : $a+1=1$ & $a*0=0$
7) Absorption Laws : $a+(a*b)=a$ & $a*(a+b)=a$
8) Associative Laws : $a+b+c=a+(b+c)$ & $(a*b)*c=a*(b*c)$
9) Involution Law [(a')'=a] : $0'=1$ & $1'=0$



* 10) Uniqueness of Complement :

If $a+x = 1$ & $a*x = 0$ then $\boxed{x = a'}$

11) Demorgan's Laws :

$(a+b)' = a' * b'$ & $(a*b)' = a' + b'$

- $(B, \leq, \wedge, \vee, ')$ is boolean Algebra :

1) $a \leq b$
2) $a \vee b = b$ $(a+b=b)$
3) $a \wedge b = a$ $(a*b=a)$
4) $a \wedge b' = 0$ $(a*b'=0)$
5) $a' \vee b = 1$ $(a'+b=1)$
6) If $a \leq b$ then $a' \geq b'$

- Let B be a Boolean algebra, Then B is a Partially ordered set, ~~where~~ Set x precedes set y if x is subset of y. This follows from the fact that $X \cup Y = Y$ if and only if $X \subseteq Y$.

Permutations: [Ordered Selection / Arrangement]

$$P(n,r) = \frac{n!}{(n-r)!}$$

The number of permutations of n-objects:

1. Taken 'r' at a time $= {}^nP_r = {}^nP_r = P(n,r) = (n)_r$
2. Taken 'r' at a time $= n^r$ [with repetation]
3. Taken all at a time $= n!$
4. Taken not more than 'r' $= \frac{n.(n^r-1)}{n-1}$ [with repetation]
5. Taken 'r' (atleast one repeated) $= n^r - {}^nP_r$
6. Taken all at a time, in which 'r' of them are alike (identical)
$$= \frac{n!}{r!} = (n-r+1)!$$
7. Taken all at a time, in which n_1 are alike, n_2 are alike, ... n_r are alike.
$$= \frac{n!}{n_1!\, n_2! \ldots n_r!}$$
8. Taken 'r' at a time, in which m-particular objects are: a) Never Included $= (n-m)P_r$
b) Always included $= (n-m)_{P_{(r-m)}} \cdot {}^rP_m$ $(n \geq m \& r \geq m)$

The number of Circular permutations of n-objects:

1. Taken all at a time $= (n-1)!$; (If not direction)
$= \frac{(n-1)!}{2}$; (If direction is given (anti/clock))
2. Taken 'r' at a time $= \frac{{}^nP_r}{r}$; (if no direction)
$= \frac{1}{2} \cdot \frac{{}^nP_r}{r}$ (direction)

$$ {}^{n}P_{r} = \frac{n!}{(n-r)!} = n(n-1)(n-2)\cdots\cdots(n-r+1) $$

$$ {}^{n}P_{r} = P(n-1, r) + r.P(n-1, r-1) $$

$$ {}^{n}P_{r} = r! \cdot {}^{n}C_{r} = n \times {}^{(n-1)}P_{(r-1)} $$

* Derrangements:

Number of derrangements for n-objects = $n!\, e^{-1}$

$$ (a+b)^{n} = {}^{n}C_{0}a^{n} + {}^{n}C_{1}.a^{n-1}.b + {}^{n}C_{2}.a^{n-2}.b^{2} + \cdots + {}^{n}C_{n-1}.a.b^{n-1} + {}^{n}C_{n}.b^{n} $$

$$ = \sum_{r=0}^{n} {}^{n}C_{r}.a^{n-r}.b^{r} $$

(60)

$$ \binom{n}{n_1, n_2, \cdots, n_r} = \frac{n!}{n_1!\, n_2! \cdots n_r!} $$

* No. of ways distributing r apples to the n-persons = $(n+r-1)C_{r}$

* The no. of permutations of (n_1+n_2) objects, taken all at a time, in which n_1 objects are alike & n_2 objects are alike: $= \frac{(n_1+n_2)!}{n_1!\, n_2!}$

Combinations: [unordered Selection]

$$C(n,r) = \frac{n!}{(n-r)!\,r!} \quad ; \quad C(n+1,r) = C(n,r-1) + C(n,r)$$

- The Numbers of combinations of n-objects:

1). Taken r at a time (Selecting) $= {}^{n}C_{r}$

* 2. Taken all at a time $= 2^n$

3. Taken one or more at a time $= 2^n - 1$

4. Divided into p-objects of one type, q-objects of second-type and so on:
$$= [(p+1)(q+1)\ldots] - 1$$

5. Divided into m-groups of x_1 objects, x_2 objects ..... x_m objects:
$$= \frac{n!}{x_1!\,x_2!\ldots x_m!}$$

6. Divided into Equal m-groups of $x_1 = x_2 = x_3 \ldots = x_m$ objects.
$$= \frac{n!}{(x!)^m \cdot m!} \quad [\because \text{Let } x = x_1 = x_2 = \ldots = x_m]$$

7. Taken r at a time, in which m-particular objects:
a) Always included $= C(n-m, r-m)$
b) ~~Always~~ Never included $= C(n-m, r)$

- No. of subsets of a set X containing n-elements $= 2^n$
$(n_{C_0} + n_{C_1} + \ldots + n_{C_n})$

- No. of diagonals for a regular polygon with n-sides $= {}^{n}C_{2} - n$
$= \frac{n^2-3n}{2}$
[${}^{n}C_2$ = no. of diagonals & sides (both)
n = no. of sides or no. of points only]

- No. of triangles formed by vertices of a polygon with n-sides:
$= {}^{n}C_{3}$

- No. of triangles formed by vertices of a polygon with n-sides, if the sides of polygon are not to be sides of any triangle:
$= {}^{n}C_{3} - n(n-4) - n$ ..... with 2-side on polygon

Counting:

Set → Finite : countable
Set → Infinite → Countable
Infinite → uncountable.

- Union of two Countable Sets is Countable.
- Union of Countable no. of Countable Sets is Countable.
- The set of all bit strings is Countable.
- The set of real numbers that are solutions to quadratic Equations $ax^2+bx+c=0$, where a, b & c are integers is Countable.
- The set of all computer programs in a particular programming language is Countable.
- The set of lines through the Origin is "uncountable".

* Counting Principle [Inclusion-Exclusion]:

$$n(A\cup B) = n(A)+n(B)-n(A\cap B)$$

$$n(A\cup B\cup C) = n(A)+n(B)+n(C) - n(A\cap B)-n(A\cap C)-n(B\cap C) + n(A\cap B\cap C)$$

Let $A_1, A_2, \ldots A_n$ are finite sets:

$$n(A_1\cup A_2\cup \ldots \cup A_n) = \sum_{1\le i\le n} n(A_i) - \sum_{1\le i<j\le n} n(A_i\cap A_j) + \sum_{1\le i<j<k\le n} n(A_i\cap A_j\cap A_k) - \ldots \ldots + (-1)^{n-1}\, n(A_1\cap A_2\cdots\cap A_n)$$

- If A & B are disjoint, $n(A\cup B) = n(A)+n(B)$.
- The set of functions "from the positive integers to the $\{0,1,2,3,4,5,6,7,8,9\}$" is "uncountable".

* Pigeonhole-principle: Atleast $(n+1)$ pigeons are contained in n pigeon holes.

$$= [kn+1]_{(n)} = [k(n-1)+1]$$ (k - positive integer)

* Fundamental principle of Counting:

If some event can occur in n_1 different ways, and if, following this event, a second event occur in n_2 different ways, ... then the no. of ways the event can occur in the order $= n_1 \cdot n_2 \cdot n_3 \ldots$

Summation:

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

* $$\Sigma n^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\Sigma n^3 = \frac{n^2(n+1)^2}{4} = [\Sigma n]^2$$

$$\sum_{K=0}^{n} a.r^K = \frac{a[r^{n+1}-1]}{r-1} \quad ; r \neq 1.$$

~~* Suppose n1 & n2 are any integers such that n1 ≤ n2.~~

1) $$S_n = \sum_{K=n1}^{n} f(K) = f(1) + f(2) + \cdots + f(n-1) + f(n)$$

(or)

$$S_n = S_{n-1} + f(n)$$

2) $$\sum_{K=n1}^{n2} f(K) = f(n_1) + f(n_1+1) + f(n_1+2) + \cdots + f(n_2) \; ; \; n1 \leq n2$$

* Ex: For $n_2 \leq n_1$, $\sum_{K=n_1}^{n_2} f(K) = 0$

* $$\sum_{K=1}^{n} [f(K) + g(K)] = \sum_{K=1}^{n} f(K) + \sum_{K=1}^{n} g(K)$$

Asymptotics:

Let $f(x)$ & $g(x)$ be functions from the set of real numbers, we say f & g are "asymptotic [$f(x) \sim g(x)$]" if $\lim_{x \to \infty} \frac{f(x)}{g(x)} = 1$

Generating functions:

$$G(x) = \sum_{k=0}^{\infty} a_k x^k = a_0 + a_1 x + a_2 x^2 + \cdots + a_k x^k + \cdots$$

* $(1+ax)^n = \sum_{k=0}^{n} {}^nC_k \cdot a^k x^k = \sum_{k=0}^{n} {}^nC_k (ax)^k$

$(1+x^r)^n = \sum_{k=0}^{n} {}^nC_k \cdot x^{rk}$

* $\frac{1-x^{n+1}}{1-x} = \sum_{k=0}^{n} x^k$

- $e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots$

- $\log(1+x) = \sum_{k=0}^{\infty} \frac{(-1)^{k+1} \cdot x^k}{k!} = -1 + \frac{x}{1!} - \frac{x^2}{2!} + \frac{x^3}{3!} - \cdots$

- $\frac{1}{1-ax} = \sum_{k=0}^{\infty} (ax)^k = \sum_{k=0}^{\infty} a^k x^k$

- $\frac{1}{1-x^r} = \sum_{k=0}^{\infty} (x^r)^k = \sum_{k=0}^{\infty} x^{rk}$

- $\frac{1}{(1-ax)^n} = \sum_{k=0}^{\infty} {}^{n+k-1}C_k \cdot (ax)^k$

- $\frac{1}{(1+ax)^n} = \sum_{k=0}^{\infty} {}^{n+k-1}C_k \cdot (-1)^k \cdot (ax)^k$

- $\sin x = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots \infty$

- $\sinh x = \sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots \infty$

- $\cos x = \sum_{k=0}^{\infty} (-1)^k \cdot \frac{x^{2k}}{(2k)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots \infty$

- $\cosh x = \sum_{k=0}^{\infty} \frac{x^{2k}}{(2k)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \cdots \infty$

(22)

Recurrence Relation:

$$a_n = a_n^H + a_n^P$$

* complementary function $[a_n^H]$:

Root	a_n^H
1) $t_1, t_2, t_3, \ldots t_k$ are distinct real numbers	$\Rightarrow c_1 t_1^n + c_2 t_2^n + \ldots + c_k t_k^n$
2) $t_1 = t_2 = t_3$	$\Rightarrow (c_1 + c_2 n + c_3 n^2)$
3) Pair of roots are Complex $(\alpha \pm i\beta)$	$\Rightarrow r^n [c_1 \cos n\theta + c_2 \sin n\theta]$ where $r = \sqrt{\alpha^2+\beta^2}$, $\theta = \tan^{-1}\frac{\beta}{\alpha}$

* Particulars function $[a_n^P]$:

R.H.S.	a_n^P
1) constant	P_1
2) $c_1 n + c_2$	$P_1 n + P_2$
3) $c_1 n^2 + c_2 n + c_3$	$P_1 n^2 + P_2 n + P_3$
4) b^n	$P_1 b^n$
5) $n \cdot b^n$	$(P_1 n + P_2) b^n$

$P_1, P_2, \& P_3$ values are obtained by substituting a_n^P in given reccurrence relation.

Ex:- $a_n - 3a_{n-1} = n+2$ $\quad [a_{n+2} + c_1 a_{n+2-1} + c_2 a_{n+2-2} + \ldots + c_n a_n = 0]$

a_n^H: $a_n - 3a_{n-1} = 0 \xrightarrow[n=n+1]{} a_{n+1} - 3a_n = 0 \Rightarrow t - 3 = 0 \Rightarrow t = 3$

$\therefore a_n^H = c_1 3^n \rightarrow (1)$

a_n^P: $P_1 n + P_2$ is substitute in reccurrence relation

$(P_1 n + P_2) - 3(P_1(n-1) + P_2) = n+2$

Compare co-eff's of n & Constans then we get $P_1 = -\frac{1}{2}$, $P_2 = -\frac{7}{4}$

Solution: $a_n = a_n^H + a_n^P = c_1 \cdot 3^n + \left(-\frac{n}{2} - \frac{7}{4}\right)$ Ans.

→ If R.H.S $[f(n)]$ is an Exponential function.

$$a_n^P = \frac{f(n)}{\phi[E]} \quad ; \boxed{\phi(E) = (E-b)^k}$$

$$a_n^P = \frac{b^n}{\phi(E)} = \frac{b^n}{\phi(b)} \quad ; \text{if } \phi(b) \neq 0$$

$$= \frac{b^n}{(E-b)^k} = {}^nC_k \cdot b^{n-k} \quad ; \text{if } \phi(b) = 0$$

Ex:- $x_n - 2(x_{n-1}) + x_{n-2} = 2^n$ [Non-Homogeneous] & $x_0 = 2$ & $x_1 = 1$

Substitute $n = n+2$

$\Rightarrow x_{n+2} - 2x_{n+1} + x_n = 2^{n+2}$

$\Rightarrow (E^2 - 2E + 1) x_n = 4 \cdot 2^n$

$\Rightarrow (E-1)^2 x_n = 4 \cdot 2^n$

$\Rightarrow t_1 = t_2 = 1$ $\quad a_n^H$ = complementary function = $(C_1 + nC_2)1^n$

$$a_n^P = \frac{4 \cdot 2^n}{(E-1)^2} = \frac{4 \cdot 2^n}{(2-1)^2} \quad ; [b=2 \Rightarrow \phi(b) = (b-1)^2 \Rightarrow \phi(2) = 1]$$

(23)

$$= 4 \cdot 2^n$$

$$\therefore x_n = (C_1 + nC_2)1^n + 4 \cdot 2^n \quad [\because 1^n = 1]$$

$x_0 = 2 \Rightarrow 2 = C_1 + 0 + 4 \Rightarrow C_1 = -2$

$x_1 = 1 \Rightarrow 1 = C_1 + C_2 + 4.2 \Rightarrow C_2 = -5$

$$\boxed{x_n = 4 \cdot 2^n - (2 + 5n)}$$

Graph Theory :

Network : A Graph with only one Source & one Sink.

Isolated vertex : A vertex with degree(valence) zero.

Pendent vertex : A vertex with degree one.

Pendent Edge : An Edge incident with pendent vertex.

Path : Sequence of Edges, without vertex repeatation.

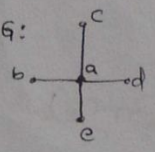
Trial (Tour) : Sequence of Edges, without Edge repeatation. (vertex may repeat)

Walk : Sequence of Edges, where vertex & Edges may repeat.

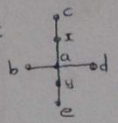
Isomorphism : Two graphs G & G^* are isomorphic, if there is a function $f: V(G) \to V(G^*)$ $\exists$ "f is bijection" and "for each pair of vertices u & v of G, $\{u,v\} \in E(G)$ iff $\{f(u), f(v)\} \in E(G^*)$

Homomorphism : Given any graph G, we can obtain a new graph (G^*) by dividing an Edge of G with additional vertices". Two graphs G & G' are said to be homeomorphic if they can be obtained from the same graph.

Ex:- G:



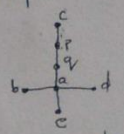
G_1:



$\{a,c\}$ is divided into $\{c,x\}$ & $\{x,a\}$

$\{c,a\} \Rightarrow \{e,y\}$ & $\{a,y\}$

G_2:



$\{a,c\} \Rightarrow \{a,p\}, \{p,q\}$ & $\{p,c\}$

G_1 & G_2 are homeomorphic, because they are obtained from G.

Cut vertex [Articulation point] : Removal of a vertex 'v', the graph becomes disconnected.

Bridge : Removal of an Edge 'e', the graph becomes disconnected.

Spanning tree : It is a tree and Subgraph to given Graph 'G' and includes all vertices of 'G' in it.

Covering [Edge Covering] : Every Vertex of graph incident with atleast one edge in 'G' [$d(v_i) \geq 1$]; $\forall v_i$

Minimal Covering : No Edge can be removed, without destroying -covering property.

Matching: It is a set of edges with atmost one edge can incident on every vertex. $[d(v) \leq 1 \; ; \forall v]$

Maximum matching: Matching with maximum no. of edges.

Perfect Matching: Every vertex is matched. $[d(v) = 1 \; ; \forall v]$ [Matching to which no edge in graph can be added]

Complete Matching: In a bipartite graph having a vertex partition V_1 & V_2. A complete matching of vertices in a set V_1 into those in V_2 is a matching, such that every vertex in V_1 is matched against some certain vertex in V_2, such that no two vertices of V_1 is matched against a single vertex in V_2.

Independent Set: It is subset of 'V' in which no two vertices are adjacent in G(V, E).

Independent Number: No. of vertices in largest independent set.

Colouring: Vertices are coloured, such that no two vertices have same colour. (24)

Chromatic Number: $\chi(G)$ Minimum no. of colors to paint the graph.

planar graph: A "graph" or a "multigraph" that can be drawn in a plane or sphere so that its edges do not cross is called a planar graph.

Map: A particular planar representation of a finite planar multigraph is called a "map".

Diameter of a graph: Maximum distance b/w any two of vertices in a graph.

Connected graph (multigraph): A graph (multigraph) is connected if there is a path between any (every) two of its vertices.

Cutset: It is a set of edges, whose removal from graph leaves the graph becomes disconnected, provided no proper subset of these edges disconnects the graph.

- one or more cutsets may exist in the connected graph.

* **Edge Connectivity:** The number of edges in the smallest cut set is called "Edge Connectivity".

Vertex Connectivity: "Minimum no. of vertices" whose removal results a disconnected

K-Connected graph: on removal of k-vertices, the connected graph becomes disconnected.

* **Non-Separable graph:** Graph with no cut vertices & no bridges.

Bipartite graph: Vertex of a graph divided into two sets, such that $V = V_1 \cup V_2$ & $V_1 \cap V_2 = \phi$

Complete Bipartite graph ($K_{m,n}$): Every vertex in V_1 is adjacent to all other vertices in V_2.

Traversable multigraph: If there is a path which includes all the vertices and uses each edge exactly once. ["traversable trial"]

Eulerian graph: If there exist "closed traversable trail". [It may repeat vertices] (Eulerian trial)

Hamiltonian path: If there is a path uses each vertex - exactly once. [It may skip edges].

Strongly Connected: In digraph, a path exist b/w any (every) vertex to any other vertex. (every)

Unilaterally Connected: In digraph, For any vertices v and u, there is a path from u to v or from v to u. (not necessarily both)

Non-planar graph: A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5

Pseudo graph : A graph with self-loops.

Regular graph : Every vertex (All) in a graph has same degree

Complete graph : Every two vertices in a graph are adjacent.

Self-Complementary graph : A graph with n-vertices contains $n_{C_2}/2$ edges in it. ($G \cong G'$)

Unicursal graph : It is traversable multigraph. (Open trial)

Trivial graph : A graph with one vertex & No Edges.

Discrete graph : A graph with only isolated vertices.

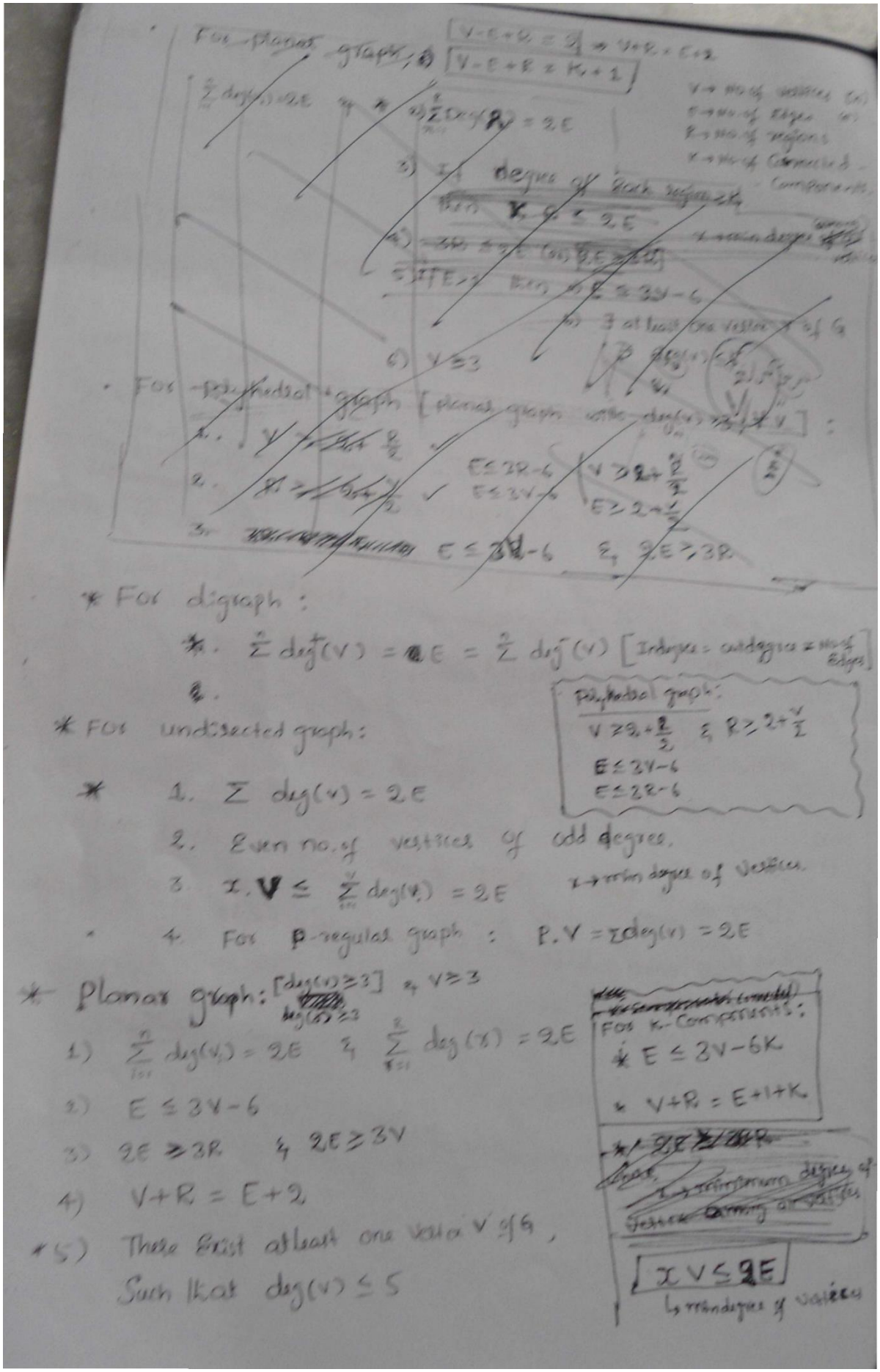
Biconnected graph : A graph with no cut vertices & no bridges [cut points]

Weakly Connected : If a digraph is not strongly connected then the whose removal of directions in the graph becomes "undirected connected graph" is called "weakly Connected".

Decomposition : $G_1 \cup G_2 = G$ & $G_1 \cap G_2 = \phi$ is the decomposition of G into G_1 & G_2.

Degenerate tree : A tree with a single vertex & no edges.

"One or more" Biconnected Components Exists in a graph.



For planar graph:

$V - E + R = 2 \Rightarrow V + R = E + 2$

$V - E + R = K + 1$

$\sum_{i=1}^{n} \deg(v_i) = 2E$

$\sum_{i=1}^{R} \deg(R_i) = 2E$

V → no. of vertices (or)
E → no. of edges (or)
R → no. of regions
K → no. of Connected Components

3) If degree of each region $= K$, then $KR \leq 2E$

4) $3R \leq 2E$ (or) $2E \geq 3R$

5) If $E > 1$ then $E \leq 3V - 6$

6) $V \geq 3$

6) $\exists$ at least one vertex v of G s.t. $\deg(v) \leq 5$

For Polyhedral graph [planar graph with $\deg(v) \geq 3, \forall v$]:

1. $V \geq 2 + \frac{R}{2}$
2. $R \geq 2 + \frac{V}{2}$
3. $E \leq 3R - 6$ & $E \leq 3V - 6$ & $V \geq 2 + \frac{R}{2}$ & $E \geq 2 + \frac{V}{2}$
4. $E \leq 3V - 6$ & $2E \geq 3R$

* For digraph:

1. $\sum \deg^+(v) = E = \sum \deg^-(v)$ [Indegree = outdegree = no. of edges]
2.

Polyhedral graph:
$V \geq 2 + \frac{R}{2}$ & $R \geq 2 + \frac{V}{2}$
$E \leq 3V - 6$
$E \leq 2R - 6$

* For undirected graph:

1. $\sum \deg(v) = 2E$
2. Even no. of vertices of odd degrees.
3. $\delta . V \leq \sum_{i=1}^{V} \deg(v_i) = 2E$ (δ → min degree of vertices.)
4. For β-regular graph: $\beta . V = \sum \deg(v) = 2E$

* Planar graph: [$\deg(v) \geq 3$] & $V \geq 3$

1) $\sum_{i=1}^{n} \deg(v_i) = 2E$ & $\sum_{i=1}^{R} \deg(r_i) = 2E$
2) $E \leq 3V - 6$
3) $2E \geq 3R$ & $2E \geq 3V$
4) $V + R = E + 2$
5) There exist atleast one vertex v of G, such that $\deg(v) \leq 5$

For K-Components:
$E \leq 3V - 6K$
$V + R = E + 1 + K$

$\delta . V \leq 2E$
(δ → minimum degree of vertex among all vertices)
& δ → min degree of vertices

* No. of Edges in Complete graph (K_n) = n_{C_2} Edges
* In K_n, Every vertex has degree $(n-1)$.
* A Bi-Partite graph with n-vertices have : $\leq \frac{n^2}{4}$ Edges.
* A Graph with n-Vertices, k-Components have : $\leq \frac{1}{2}(n-k)(n-k+1)$ Edges.

$$E \leq \frac{1}{2}(v-k)(v-k+1)$$

* A Complete bipartite graph ($K_{m,n}$) has : $(m+n)$ vertices & mn Edges.
* A Self-Complementary graph with n-vertices have : $\frac{n_{C_2}}{2}$ Edges.
* No. of Subgraphs are possible for $K_n = \sum_{i=1}^{n} n_{C_i} \times 2^{i}_{C_i}$
* No. of Perfect matchings for K_{2n} : $\frac{(2n)!}{2^n n!}$
* A graph with e-Edges decomposed into $(2^{e-1}+1)$ ways
* No. of Spanning trees for K_n : n^{n-2}
* No. of binary trees with n-vertices : $\frac{2n_{C_n}}{n+1}$ (CS)
* Tree with n-vertices have : $(n-1)$ Edges
* K-trees with total n-vertices have : $(n-k)$ Edges. [Forest]
* A graph G with n-vertices & e-edges then the no. of edges in $\bar{G} = (n_{C_2} - e)$ Edges
* No. of Edges must be removed from connected graph with n-vertices & e-edges to produce a Spanning tree is called "circuit rank of graph" : Circuit Rank = $e-(n-1)$ Edges.
* No. of Edges must be removed to produce Spanning forest of graph with n-vertices, e-Edges & k-Components : $= e-(n-k)$ Edges.
* Minimum no. of Edges contained in a simple graph with n-vertices to be connected : $\frac{(n-1)(n-2)}{2} = {}^{n-1}C_2$
* No. of Bridges are possible for a Spanning tree with n-edges $= n-1$ Bridges

* Transitive Closure by a) Walshall's algorithm : $2n^3$ bit operations.
  b) n-boolean powers of matrices : $2n^3(n-1)$ bit operations.
* Matrix multiplication ($A_{m \times n}$ & $B_{n \times p} \Rightarrow (AB)_{m \times p}$) :
  a) Contain n^2 Entries
  b) To find Each Entry n-multiplications & (n-1) additions required.
  c) Total n^3-multiplications & $n^2(n-1)$ additions required.
* Equivalence relation (of vertex) gives Component partition.
* K_n has atleast $(n-1)\left(\left[\frac{n}{2}\right]-1\right)$ cycles.
* Bipartite graph contains Even length cycles.
* A multigraph with more than two odd vertices is not traversable. [Euler path not contains] (start at one odd vertex & end at other odd vertex for Euler path)
* A finite Connected graph is Eulerian iff Each vertex has Even degree.
* Chromatic number of K_n [$\chi(K_n) = n$] $= n$.
* Bipartite graph is 2-Colorable.
* Any planar graph is 4-Colorable.
* Tree is 2-Colorable
* Equivalence relation in a Connected graph gives the chromatic partition.
* K-cycle graph is 2-chromatic [2-Colorable] ; $K > 3$ & Even length cycle. ($K \to$ Even)
  [If K is odd, then 3-Colorable]
* ~~Covering of a graph with n vertices have $\geq \left[\frac{n}{2}\right]$ edges.~~
* A graph has hamiltonian cycles, if a graph has n-vertices then no. of edges: $\geq \frac{1}{2}(n^2-3n+2)$

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

  $E \geq {}^{(n-1)}C_2$

- Two graphs are isomorphic iff their complements are isomorphic.

* If G & G' are isomorphic then
  1. The no. of vertices in G & G' are same.
  2. The no. of edges in G & G' are same
  3. The degree sequence of G & G' are same.

*. The no. of different trees with vertex set $\{v_1, v_2 \ldots v_n\}$ in which the vertices $v_1, v_2, \ldots v_n$ has degrees $d_1, d_2 \ldots d_n$ respectively : $\frac{(n-2)!}{(d_1-1)!.(d_2-1)! \ldots (d_n-1)!}$

- Multigraph is a "graph", if it has neither multiple edges nor loops.

** If multigraph, with n-vertices, the maximum no. of edges is not computable; because, multigraph contains multiple edges & they may finite or infinite.

- The path b/w two vertices of a graph is an "Equivalence relation".

- The bridges of Konigsberg problem is not traversable.

** Addition or deletion of loops from a multigraph results, the graph remains traversable or non traversable.

*. The complete bipartite graph $K_{m,m}$ is a regular graph of degree-m.

*. A finite tree (with atleast one edge) has atleast two vertices of degree 1.

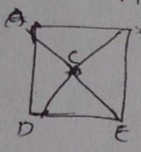
* Connected graph with n-vertices has atleast $(2^{n-1}-1)$ Spanning trees.

* ~~Complete graph with n-vertices has ( ) Spanning trees.~~

** For a Complete graph with n-vertices, the maximum no. of different paths possible b/w a pair of vertices is: $(n-1)!$

* Wheel graph with n-vertices have: $2(n-1)$ Edges

Ex:-



$\Rightarrow 2(5-1) = 8$ Edges.

* $E(G) \cup E(\bar{G}) = K_n$ [union of a graph (Edges) & Complement of it is a Complete graph.]

Linear Algebra:

Sri's

Principle Diagonal: In a Square matrix all Elements a_{ij} for which $i=j$ are called the Elements of Principle diagonal

Upper Δ^{le} matrix: A Square matrix in which all the Elements below the principle diagonal are Equal to Zero.

Lower Δ^{le} Matrix: A Square matrix in which all the Elements above the Principle diagonal are Equal to Zero.

Diagonal matrix: Both Upper & Lower Δ^{le} matrix
i.e., All the Elements other than Elements of principle diagonal are Zero.

Scalar Matrix: A diagonal matrix with all Elements in principle diagonal are Equal.

Idempotent Matrix: A is Square matrix $\ni A^2 = A$ (28)

Involutary Matrix: A is Square matrix $\ni A^2 = I$

Nil potent matrix: A is Square matrix $\ni A^m = 0$
where m is the least positive integer and m is also called Index of Nil potent matrix A

If A is any matrix: 1) $(A^T)^T = A$
2) $(A+B)^T = A^T + B^T$; A & B are same type
3) $(kA)^T = k \cdot A^T$; k is Complex number
4) $(AB)^T = B^T A^T$; A & B are multiplicable.

Symmetric matrix: A is a Square matrix $\ni A^T = A$

Skew-Symmetric matrix: A is square matrix; $\ni A^T = -A$

Conjugate matrix of A: $(\bar{A})$ or $(\sim A)$ A is any matrix and replacing the elements by the Corresponding Conjugate Complex numbers.

$$A = \begin{bmatrix} 2+3i & 4+7i & 5 \\ 2i & 3 & 9-i \end{bmatrix} \Rightarrow \bar{A} = \begin{bmatrix} 2-3i & 4-7i & 5 \\ -2i & 3 & 9+i \end{bmatrix}$$

- Transposed Conjugate of a matrix : $(\bar{A})^T$ (A^θ or A^*)
- Hermitain matrix : $A^* = A$
  * Every diagonal Elements of Hermitain matrix is real number.
- Skew Hermitain matrix : $A^* = -A$
  * Diagonal Elements of skew-hermitain matrix must be either Pure imaginary numbers or Zero
- Unitary Matrix : $A^\theta . A = I$; [Square matrix 'A' with Complex numbers as Elements].
- Orthogonal Matrix : $A^T . A = I$; A is Square matrix $\boxed{A^T = A^{-1}}$
- Boolean Matrix : Any matrix with Elements 0 or 1.
- Sparse matrix : More no. of Elements in matrix are Zero's.
- Dense matrix : It is not Sparse matrix.
- If AB Exists then BA may or may not Exist.
- If AB & BA Exists then they need not be Equal [Not Commutative].

If AB = 0 then either A or B need not be Equal to 0.

Cofactor : If A_{ij} is an minor Element of a Square matrix A, then the product of $(-1)^{ij}$ and A_{ij} is called "Cofactor of A_{ij}"

- If A is Square matrix : then $|A| = |A^T|$ (determinant)
- Singular matrix : $|A| = 0$ [determinant of A = 0]

Non-Singular matrix : $|A| \neq 0$

* If A & B are Square matrix of same order then $|AB| = |A| * |B|$
- Adjoint matrix : Transpose of co-factors matrix.
- An invertible matrix has Unique Inverse.
- If A is invertible then A^{-1} is also invertible ; $(A^{-1})^{-1} = A$

- A & B are two invertible matrices of same type : (AB)⁻¹ = B⁻¹A⁻¹
- $A^{-1} = \text{Adj}A / |A|$; A is non-singular matrix.
- If A is invertible, then $|A^{-1}| = \frac{1}{|A|}$ (determinant)

* If A & B are non-singular matrices of same type then

$$\text{Adj}(AB) = (\text{Adj}B)(\text{Adj}A)$$

* If A is a Square matrix of type n then $|\text{Adj}A| = (\det A)^{n-1}$ and $|\text{Adj}(\text{Adj}A)| = |A|^{(n-1)^2}$ $= |A|^{n-1}$

- Subset of Linearly independent set is linearly independent.
- A is a matrix of order m×n, if there is a minor of order k. [k<m & k<n], which is not zero then ρ(A) ≥ k

* Rank of a matrix does not change by pre-multiplication or post-multiplication with a non-singular matrix.

- The System AX = B ; 1) If ρ(A) = ρ(AB) = no. of variables; Unique solution.
  2) If ρ(A) = ρ(AB) < no. of variables; Infinite Solutions.
  3) If ρ(A) ≠ ρ(AB); No Solution. [Inconsistent].

* The System AX = 0 ;
  1) If ρ(A) = no. of variables ; Unique solution (Trivial/zero)
  2) If ρ(A) < no. of variables (n); Infinite Solution [Non-Trivial] [(n−r) Linear independent Solutions] (r) (Linearly independent vectors)

- The System AX = B is Consistent, if ρ(A) = ρ(AB).
- $|A - \lambda I| = 0$ is Called "Characteristic Equation of A" (square matrix)
- The roots of Characteristic Equation are Called "Eigen values" [Latent/characteristic roots]
- The set of Eigen values of matrix called "Spectrum of matrix"
- The matrix Order n will have n. latent roots.

* $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, which Satisfies the matrix Equation $[A - \lambda I]X = 0$ is Called Corresponding "Eigen vector of the matrix A".

* Sum of Eigen values of a matrix is Equal to the Sum of Elements of Principal diagonal.
* Product of Eigen values = determinant of its matrix.
- If λ is an Eigen value of A then $\frac{1}{\lambda}$ is the Eigen - value of A^{-1}
- If λ is Eigen value of an orthogonal matrix then $\frac{1}{\lambda}$ is also its Eigen value. $[A^T = A^{-1}]$ [Eigen values of A and A^T are same].
* If $\lambda_1, \lambda_2, \ldots \lambda_n$ are Eigen values of matrix A, then A^m has Eigen values $\lambda_1^m, \lambda_2^m, \ldots \lambda_n^m$.
* If a matrix is Either Lower or Upper triangular then the Principal digonal Elements are Called "Eigen values".
- The Eigen values of Symmetric matrix are "purely real".
- The Eigen values of Skew-Symmetric matrix are Either "purely imaginary" or "Zero's".
- Matrix Inverse method : If $AX=B$ then $X=A^{-1}B$
- Cramer's Rule : If $AX=B$ then $x = \frac{1}{\Delta}\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}$, $y = \frac{1}{\Delta}\begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{21} & b_2 & a_{23} \\ a_{31} & b_3 & a_{33} \end{vmatrix}$

and $z = \frac{1}{\Delta}\begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{vmatrix}$; $X = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ $\Delta \rightarrow |A|$

- Gauss-Jordan Method : If $AX=B$, then $[A|B]$ can be reduced to the form $[I|B_1]$ by using Elementary row-transformation then $X=B_1$

* If A is matrix with order $m \times n$ then $\rho(A) \leq \min\{m,n\}$.
- Rank of a matrix is the largest order of any non-vanishing minor of matrix.

* If A is Square matrix then 1) $A+A^T$ is Symmetric
2) $A-A^T$ is Skew-Symmetric
3) $A \cdot A^T$ is Symmetric
4) $A = \frac{1}{2}(A+A^T) + \frac{1}{2}(A-A^T)$

- If A & B are Symmetric then:
1) $AB+BA$ is Symmetric
2) $AB-BA$ is Skew Symmetric
3) A^n & B^n are symmetric

- If A is Skew-Symmetric then:
1) A^n is Symmetric, if 'n' is Even
2) A^n is Skew-Symmetric if 'n' is odd.

* The Eigen Values of A & A^T are same.
- The Determinant of Orthogonal matrix (A) has absolute value 1 ($|A| = \pm 1$)
- If $\lambda_1, \lambda_2, \ldots \lambda_m$ are Eigen values of A then
1. Eigen value of kA are $k\lambda_1, k\lambda_2, \ldots k\lambda_m$
2. matrix A^n has Eigen values $\lambda_1^n, \lambda_2^n, \ldots \lambda_m^n$
* 3. matrix $A+kI$ has Eigen values $\lambda_1+k, \lambda_2+k, \ldots \lambda_n+k$
4. $(A-kI)^2$ has Eigen values $(\lambda_1-k)^2, (\lambda_2-k)^2, \ldots (\lambda_n-k)^2$

(30)

- If A is non-Singular, then $\text{rank}(\text{Adj}(A)) = \text{rank}(A)$
- If $\rho(A) = n-1$ then $\rho(\text{Adj } A) = 1$
- If $\rho(A) \leq n-2$ then $\rho(\text{Adj } A) = 0$.

* If A is Square matrix of order n, then $|kA| = k^n |A|$
- $A_{m \times n}$ & $B_{n \times p}$ then product AB requires: 1) mnp multiplications
2) $m(n-1)p$ Additions
3) For Each Entry, n-multiplications (n-1) Additions.

Numerical Methods:

* LU Decomposition of Systems of linear Equations:
[Crout's triangularisation / Cholesky's method]

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$
$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2$$
$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}; \quad X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad B = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

- Principle minors of A are non-singular

$$AX = B \Rightarrow A = LU$$
$$\Rightarrow LUX = B$$
Let $UX = V \Rightarrow LV = B$

$AX = B$
$A = LU$
$LUX = B$
$LV = B$
$UX = V$

- $$A = LU \Rightarrow \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

Order of Computation:

a). First row of U $\Rightarrow u_{11} = a_{11}; u_{12} = a_{12}; u_{13} = a_{13}$
b). First Column of L $\Rightarrow l_{21} = \frac{a_{21}}{a_{11}}; l_{31} = \frac{a_{31}}{a_{11}}$
c). Second row of U $\Rightarrow u_{22} = a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12}$
d). Second Column of L $\Rightarrow l_{32} = \frac{1}{u_{22}}\left[a_{32} - \frac{a_{31}}{a_{11}} \cdot a_{12}\right]$
e). Third row of U. $\Rightarrow u_{33} = a_{33} - l_{31} \cdot u_{13} - l_{32} u_{23}$

Ex:- $3x + 2y + 7z = 4$
$2x + 3y + z = 5$
$3x + 4y + z = 7$

$$\Rightarrow A = LU \Rightarrow \begin{bmatrix} 3 & 2 & 7 \\ 2 & 3 & 1 \\ 3 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

(i) First row of U: $u_{11} = 3, u_{12} = 2, u_{13} = 7$
(ii) First Column of L: $l_{21}u_{11} = 2 \Rightarrow l_{21} = 2/3$ & $l_{31}u_{11} = 3 \Rightarrow l_{31} = 1$
(iii) Second row of U: $l_{21}u_{12} + u_{22} = 3 \Rightarrow u_{22} = 5/3$ & $l_{21}u_{13} + u_{23} = 1 \Rightarrow u_{23} = -1/3$
(iv) Second Column of L: $l_{31}u_{12} + l_{32}u_{22} = 4 \Rightarrow l_{32} = 6/5$
(v) Third row of U: $l_{31}u_{13} + l_{32}u_{23} + u_{33} = 1 \Rightarrow u_{33} = -8/5$

$$\Rightarrow A = LU \Rightarrow \begin{bmatrix} 3 & 2 & 7 \\ 2 & 3 & 1 \\ 3 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ 1 & 6/5 & 1 \end{bmatrix} \begin{bmatrix} 3 & 2 & 7 \\ 0 & 5/3 & -1/3 \\ 0 & 0 & -8/5 \end{bmatrix}$$

$\Rightarrow \boxed{LV = B} \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ 1 & 6/5 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 7 \end{bmatrix} \Rightarrow v_1 = 4,\ v_2 = 7/3,\ v_3 = 1/5$

$\Rightarrow \boxed{UX = V} \Rightarrow \begin{bmatrix} 3 & 2 & 7 \\ 0 & 5/3 & -1/3 \\ 0 & 0 & -8/5 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ 7/3 \\ 1/5 \end{bmatrix} \Rightarrow x = 7/8,\ y = 9/8,\ z = -1/8$

* Regula Falsi Method : [Method of false position].

- Successive iteration takes last two iterations, one is +ve $f(a)$ and other $f(b)$ is -ve.

* We cannot find Complex roots. (B1)

* order of Convergence = 1 (Linear / First Order)

$$x_i = \frac{a \cdot f(b) - b\, f(a)}{f(b) - f(a)} \quad ; \text{ If } f(a).f(b) < 0.$$

* Secant Method : [Modified version of Regulafalsi/Interpolation].

- Successive iteration takes last two iterations ; i.e, Both may be +ve, or both may be -ve, or one is +ve and other is -ve.

* We can find Complex roots.

- There is no guarantee to Convergence
- Faster Convergence.

$$x_3 = \frac{x_1 \cdot f(x_2) - x_2\, f(x_1)}{f(x_2) - f(x_1)}$$

* Order of Convergence = 1.62

* Bisection / Binary Chopping / Half-Interval / Bolzano method :

- Successive iteration takes last two iterations, where one is $\underset{f(x_1)}{+ve}$ & other is $\underset{f(x_2)}{-ve}$.
- Convergence is slow, but sure
* Rate of Convergence = 1 [Linear / First order]
- one function Evaluated per iteration
* It is not used for finding Complex roots of Equation
- After n-bisections, the length of Subinterval, which contains $x_n = \frac{b-a}{2^n}$
- If the Error is to be made "less than a small quantity ϵ" ; $\frac{b-a}{2^n} < \epsilon \Rightarrow 2^n > \frac{b-a}{\epsilon}$
- No. of iterations 'n' should be $> \frac{\log(\frac{b-a}{\epsilon})}{\log 2}$

If $f(x_1) \cdot f(x_2) < 0$; $\boxed{x_i = \frac{x_1 + x_2}{2}}$

* Newton Raphson Method : [Tangent Method]

* order of Convergence = 2 [quadratic]
* is Best method, to find Complex roots of Equation
- Two functions are Evaluated per iteration
* - Choice of x_0 is very important for convergence
- If $f(a)$ & $f(b)$ are of opposite signs, a root of $f(x)=0$ lies b/w 'a' and 'b', This idea can be used to fix the approximate root.
- $|f(x) \cdot f''(x)| < |f'(x)|^2$; condition for convergence.

- When $f'(x)$ is very large, the correct value of root can be found out with "minimum" no. of iterations.
- The Error at any stage is proportional to the Square of Error of previous stage.

* If $f'(x) \neq 0$ & $f''(x) \neq 0$

$$\boxed{x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}}$$

* $\sqrt[P]{N} \Rightarrow x_{n+1} = \frac{1}{P}\left[(P-1)x_n + \frac{N}{x_n^P}\right]$

* $\frac{1}{\sqrt[P]{N}} \Rightarrow x_{n+1} = \frac{x_n}{P}\left[(P+1) - N.x_n^P\right]$

(2)

* In general,

If α is Exact root, $e_i = x_i - \alpha$

$e_{i+1} = x_{i+1} - \alpha$

If $P \geq 1$; $\boxed{|e_{i+1}| \leq |e_i|^P . K}$

$K \rightarrow$ Positive Constant for Every i

$P \rightarrow$ order of Convergence

$P=1 \Rightarrow$ Convergence is linear.

$P=2 \Rightarrow$ Convergence is quadratic.

Numerical Integration:

$$\int_{x_0}^{x_n} y\,dx = n.h\left[y_0 + \frac{n}{2}\Delta y_0 + \frac{n(2n-3)}{12}\Delta^2 y_0 + \frac{n(n-2)^2}{24}\Delta^3 y_0 + \cdots\right]$$

* Trapezoidal Method $[n=1 ;\ \int_{x_0}^{x_n} = \int_{x_0}^{x_1} + \int_{x_1}^{x_2} + \cdots + \int_{x_{n-1}}^{x_n}]$

$$\int_{x_0}^{x_n} y\,dx = \frac{h}{2}\left[(y_0+y_n) + 2(y_1+y_2+\cdots+y_{n-1})\right]$$

- It gives Exact result when the integrand is polynomial of degree '0' or '1'.
- Error $= \frac{-h^3}{12}\cdot f''(0) - \frac{h^4}{24}\cdot f'''(0)$
- Error in order of h^2 & Error term is h^3.
- If tabulation is halved then Error is reduced by factor 8.

* Simpson Rule:

- It gives Exact result when polynomial of degree ≤ 3.
- Error $= \frac{-h^5}{90}\cdot f^{iv}(0)$
- Trapezoidal with 2n points Equal to Simpson with n-points.

* Simpso $\frac{1}{3}$rd Rule: $[n=2 ;\ \int_{x_0}^{x_n} = \int_{x_0}^{x_2} + \int_{x_2}^{x_4} + \int_{x_4}^{x_6} + \cdots + \int_{x_{n-2}}^{x_n}]$

$$\int_{x_0}^{x_n} y\,dx = \frac{h}{3}\left[(y_0+y_n) + 2(y_2+y_4+\cdots+y_{n-2}) + 4(y_1+y_3+y_5+\cdots+y_{n-1})\right]$$

* Simpso $\frac{3}{8}$ Rule: $[n=3 ;\ \int_{x_0}^{x_n} = \int_{x_0}^{x_3} + \int_{x_3}^{x_6} + \cdots + \int_{x_{n-3}}^{x_n}]$

$$\int_{x_0}^{x_n} y\,dx = \frac{3h}{8}\left[(y_0+y_n) + 2(y_3+y_6+\cdots+y_{n-3}) + 3(y_1+y_2+y_4+\cdots+y_{n-1})\right]$$

Calculus:

* Continuity: $\lim_{x \to a^+} f(x) = \lim_{x \to a^-} f(x) = f(a)$

* Differentiability: $\lim_{h \to 0^+} \frac{f(a+h)-f(a)}{h} = f'(a) = \lim_{h \to 0^-} \frac{f(a-h)-f(a)}{-h}$; at $x=a$

* Limit Existant at a point:

$$\lim_{x \to a^-} f(x) = \lim_{h \to 0} f(a-h) \quad \& \quad \lim_{x \to a^+} f(x) = \lim_{h \to 0} f(a+h)$$

$\therefore \lim_{x \to a} f(x)$ exists, iff both L.H.L & R.H.L Exists.

* If $f(x)$ is differentiable at (a,b), then it always continuous at $[a,b]$, but need not be converse.

(33)

* Rolle's Mean Value Theorem:

If 1) f is continuous in $[a,b]$
2) f is derivable in (a,b)
3) $f(a) = f(b)$

and, then $\exists\, c \in (a,b)$, $\ni$ $\boxed{f'(c) = 0}$

* Lagrange's Mean Value theorem:

If 1) f is continuous in $[a,b]$
2) f is delivable in (a,b)
3) $f(a) \neq f(b)$

then $\exists\, c \in (a,b)$, $\ni$ $\boxed{f'(c) = \frac{f(b)-f(a)}{b-a}}$

* Causchy's Mean Value Theorem:

Let $f(x)$ & $g(x)$ are two functions.

If 1) f & g are continuous in $[a,b]$

2) f & g are derivable in (a,b)

* 3) $g'(x) \neq 0$. $\forall x \in (a,b)$

then $\exists\ c \in (a,b)$. $\ni$ $\boxed{\frac{f'(c)}{g'(c)} = \frac{f(b)-f(a)}{g(b)-g(a)}}$

* Taylor's Mean Value Theorem:

Let $f(x)$ be any function, ~~such that~~

If 1) f^{n-1} is continuous in $[a, a+h]$

2) f^{n-1} is derivable in $(a, a+h)$

then $\exists\ \theta \in (0,1)$. $\ni$

$$f(a+h) = f(a) + h \cdot f'(a) + \frac{h^2}{2!} f''(a) + \cdots + \frac{h^{n-1}}{(n-1)!} f^{n-1}(a) + \frac{h^n}{n!} f^n(a)$$

$$f(x) = f(0) + x \cdot f'(0) + \frac{x^2}{2!} f''(0) + \cdots$$

*Derivatives :

$$\frac{d}{dx}(uv) = u.\frac{dv}{dx} + v.\frac{du}{dx}$$

$$\frac{d}{dx}\left(\frac{u}{v}\right) = \frac{v.\frac{du}{dx} - u.\frac{dv}{dx}}{v^2}$$

$$\frac{du}{dx} = \frac{du}{dy}.\frac{dy}{dx} \quad \text{[Chain Rule]}$$

$$\frac{d}{dx}(ax+b)^n = n.(ax+b)^{n-1}.a.$$

$$* \quad \frac{d}{dx}[f(x)]^n = n.[f(x)]^{n-1}.f'(x)$$

$$\frac{d}{dx}e^x = e^x$$

$$\frac{d}{dx}a^x = a^x.\log a$$

$$\frac{d}{dx}\log_e x = \frac{1}{x}$$

$$\frac{d}{dx}\log_a x = \frac{1}{x\log_e a} = \frac{1}{x}\log_a e$$

$$\frac{d}{dx}\sinh x = \cosh x$$

$$\frac{d}{dx}\cosh x = \sinh x.$$

$$\frac{d}{dx}(\sin x) = \cos x$$

$$\frac{d}{dx}(\cos x) = -\sin x$$

$$\frac{d}{dx}(\tan^{-1}x) = \frac{1}{1+x^2}$$

$$* \quad \frac{d}{dx}(\tan x) = \sec^2 x$$

$$* \quad \frac{d}{dx}(\cot x) = -\operatorname{cosec}^2 x$$

$$\frac{d}{dx}(\sec x) = \sec x \tan x$$

$$\frac{d}{dx}(\operatorname{cosec} x) = -\operatorname{cosec} x \cot x$$

$$\frac{d}{dx}(\sin^{-1}x) = \frac{1}{\sqrt{1-x^2}}$$

$$\frac{d}{dx}(\cos^{-1}x) = \frac{-1}{\sqrt{1-x^2}}$$

$$\frac{d}{dx}(\cot^{-1}x) = \frac{-1}{1+x^2}$$

$$\frac{d}{dx}\sec^{-1}x = \frac{1}{x\sqrt{x^2-1}}$$

$$\frac{d}{dx}\operatorname{cosec}^{-1}x = \frac{-1}{x\sqrt{x^2-1}}$$

$$\sinh x = \frac{e^x - e^{-x}}{2}$$

$$\cosh x = \frac{e^x + e^{-x}}{2}$$

(94)

* Integrals :

$$\int x^n\,dx = \frac{x^{n+1}}{n+1} \quad ; (n \neq -1)$$

$$\int \frac{1}{x}\,dx = \log_e x$$

$$\int e^x\,dx = e^x$$

$$\int a^x = \frac{a^x}{\log a}$$

$$\int \sin x = -\cos x$$

$$\int \cos x = \sin x$$

$$\int \tan x = -\log\cos x = \log\sec x$$

$$\int \cot x = \log\sin x$$

$$\int \sec x = \log(\sec x + \tan x)$$

$$\int \operatorname{cosec} x = \log(\operatorname{cosec} x - \cot x)$$

$$\int \sec^2 x = \tan x$$

$$\int \operatorname{cosec}^2 x = -\cot x$$

$$\int \frac{1}{a^2+x^2} = \frac{1}{a}\cdot \tan^{-1}\frac{x}{a}$$

$$\int \frac{1}{a^2-x^2} = \frac{1}{2a}\cdot \log\left[\frac{a+x}{a-x}\right]$$

$$\int \frac{1}{x^2-a^2} = \frac{1}{2a}\cdot \log\left[\frac{x-a}{x+a}\right]$$

$$\int \frac{dx}{\sqrt{a^2-x^2}} = \sin^{-1}\frac{x}{a}$$

$$\int \frac{1}{\sqrt{a^2+x^2}} = \sinh^{-1}\frac{x}{a}$$

$$\int \frac{1}{\sqrt{x^2-a^2}} = \cosh^{-1}\frac{x}{a}$$

$$\int \sinh x = \cosh x$$

$$\int \cosh x = \sinh x$$

$$\int e^{ax}\sin bx = \frac{e^{ax}}{a^2+b^2}(a\sin bx - b\cos bx)$$

$$\int e^{ax}\cos bx = \frac{e^{ax}}{a^2+b^2}(a\cos bx + b\sin bx)$$

$$\int \sqrt{a^2-x^2} = \frac{x}{2}\cdot\sqrt{a^2-x^2} + \frac{a^2}{2}\sin^{-1}\frac{x}{a}$$

$$\int \sqrt{a^2+x^2} = \frac{x}{2}\sqrt{a^2+x^2} + \frac{a^2}{2}\sinh^{-1}\frac{x}{a}$$

$$\int \sqrt{x^2-a^2} = \frac{x}{2}\sqrt{x^2-a^2} - \frac{a^2}{2}\cosh^{-1}\frac{x}{a}$$

$$\int \frac{f'(x)}{f(x)}\,dx = \log[f(x)]$$

$$\int [f(x)]^n \cdot f'(x)\,dx = \frac{[f(x)]^{n+1}}{n+1}$$

* Definite Integrals :

1) If $f(x)$ is Continuous in $[a,b]$ and $F(x)$ is any antiderivative of $f(x)$ in $[a,b]$ then : $\boxed{\int_a^b f(x)\,dx = F(b) - F(a)}$

2) If $f(x)$ is Continuous on $[a,b]$ then $F(x) = \int_a^b f(t)\,dt$ is differentiable at Every Point of x in $[a,b]$ and

$$\boxed{\frac{dF}{dx} = \frac{d}{dx}\int_a^b f(t)\,dt = f(x)}$$

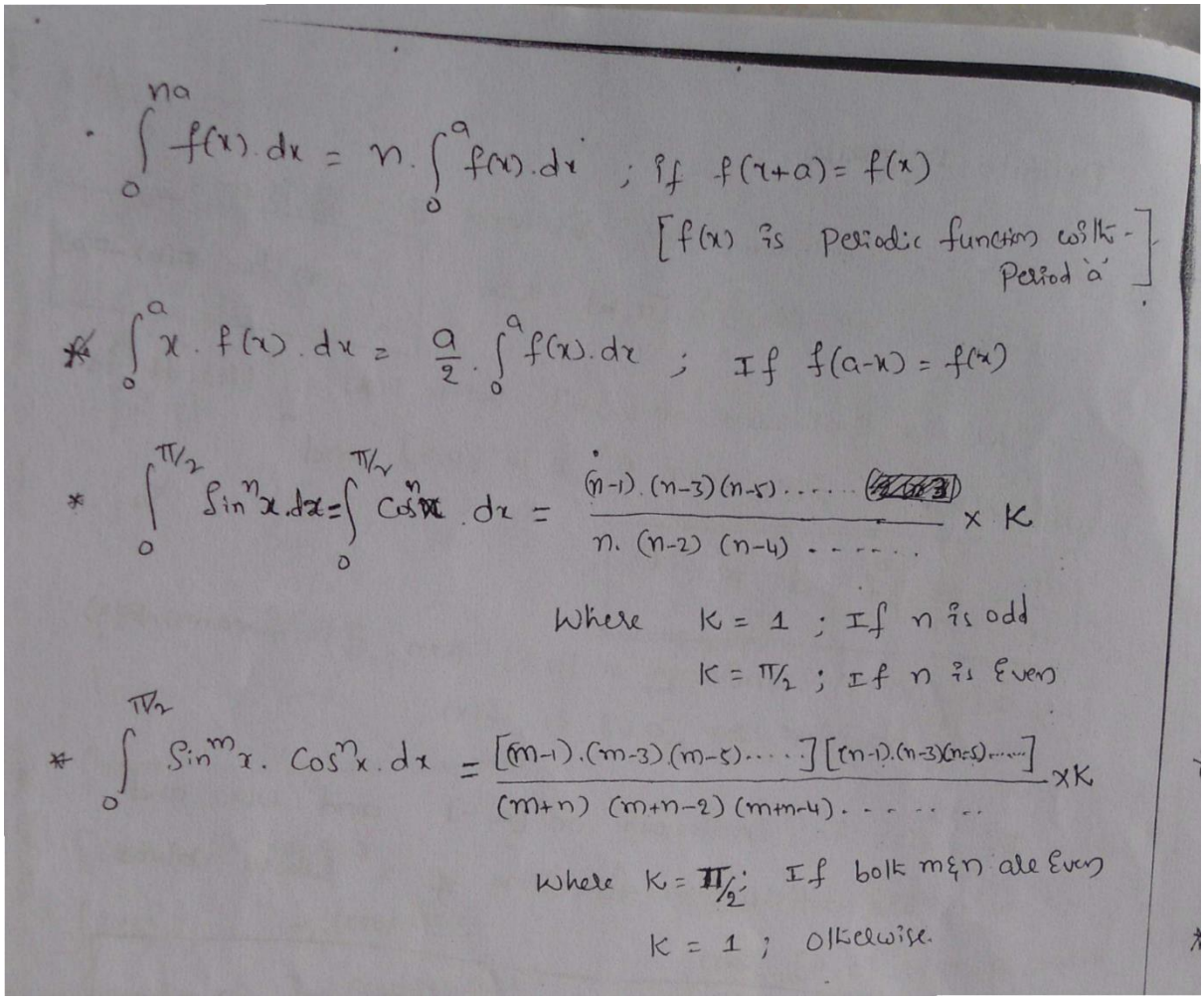
i.e., If $f(x)$ is Continuous on $[a,b]$ then $\exists$ a function $F(x)$ whose derivative on $[a,b]$ is $f(x)$.

3). If $f(x)$ is Continuous on $[a,b]$ and $u(x)$ and $v(x)$ are differentiable functions of x whose values lie in $[a,b]$ then

$$\boxed{\frac{d}{dx}\int_{u(x)}^{v(x)} f(t)\,dt = f(v(x))\cdot\frac{dv}{dx} - f(u(x))\cdot\frac{du}{dx}}$$

(35)

- $\int_a^b f(x)\,dx = -\int_b^a f(x)\,dx$
- $\int_a^b f(x)\,dx = \int_a^c f(x)\,dx + \int_c^b f(x)\,dx$; if $a<c<b$
- $\int_0^a f(x)\,dx = \int_0^a f(a-x)\,dx$
- $\int_{-a}^a f(x)\,dx = 2\cdot\int_0^a f(x)\,dx$; If $f(x)$ is Even function $f(-x)=f(x)$
  $= 0$; If $f(x)$ is odd function $f(-x) = -f(x)$
- $\int_a^b f(x)\,dx = \int_a^b f(a+b-x)\,dx$
- $\int_0^{2a} f(x)\,dx = 2\int_0^a f(x)\,dx$; If $f(2a-x) = f(x)$
  $= 0$; If $f(2a-x) = -f(x)$



$$\int_0^{na} f(x)\,dx = n\int_0^{a} f(x)\,dx \quad ; \text{If } f(x+a)=f(x)$$

[f(x) is periodic function with period 'a']

$$\int_0^{a} x\cdot f(x)\,dx = \frac{a}{2}\int_0^{a} f(x)\,dx \quad ; \text{If } f(a-x)=f(x)$$

$$\int_0^{\pi/2} \sin^n x\,dx = \int_0^{\pi/2} \cos^n x\,dx = \frac{(n-1)(n-3)(n-5)\cdots}{n(n-2)(n-4)\cdots} \times K$$

Where $K = 1$; If n is odd

$K = \pi/2$; If n is Even

$$\int_0^{\pi/2} \sin^m x\cdot\cos^n x\,dx = \frac{[(m-1)(m-3)(m-5)\cdots][(n-1)(n-3)(n-5)\cdots]}{(m+n)(m+n-2)(m+n-4)\cdots} \times K$$

Where $K = \pi/2$; If both m & n are Even

$K = 1$; Otherwise.

Improper Integrals:

- $\int_a^b f(x).dx$ is "Improper integral";

  1) If $a=-\infty$ or $b=\infty$ or both

  2) If $f(x)=\infty$ for one or more values of x in $[a,b]$.

- $\int_a^b f(x).dx$ is "Convergent", if the value of integral is "finite".

- $\int_a^b f(x).dx$ is "Divergent", if the value of integral is "Infinite".

* If $0 \le f(x) \le g(x)$ $\forall x$ and $\int_a^\infty g(x).dx$ is Converges then $\int_a^\infty f(x)$ is also converges.

* If $0 \le g(x) \le f(x)$ $\forall x$ and $\int_a^\infty g(x).dx$ is diverges then $\int_a^\infty f(x)$ is also diverges.

- If $f(x)$ & $g(x)$ are two functions, $\exists \lim_{x\to\infty} \frac{f(x)}{g(x)} = K$ (where K = finite & $K \neq 0$) then $\int_a^\infty f(x).dx$ & $\int_a^\infty g(x).dx$ are "converge" or "diverge" togethes.

- $\int_1^\infty \frac{dx}{x^p}$ is "Convergent", if $p>1$ & "diverges" if $p \le 1$

- $\int_a^\infty e^{-px}.dx$ & $\int_{-\infty}^b e^{px}.dx$ are converges for any constant $p>0$ and diverges for $p \le 0$.

- $\int_a^b \frac{1}{(b-x)^p}$ is Convergent iff $p<1$

- $\int_a^b \frac{1}{(x-a)^p}$ is converges iff $p<1$

Partial Derivatives:

- If $u = f(x,y)$, then $\frac{\partial u}{\partial x} = \lim_{h \to 0} \frac{[f(x+h,y) - f(x,y)]}{h}$

  and $\frac{\partial u}{\partial y} = \lim_{K \to 0} \frac{[f(x,y+K) - f(x,y)]}{K}$

- If $u = f(x,y)$, where $x = g(r,s)$ and $y = h(r,s)$ then

$$\frac{\partial u}{\partial r} = \frac{\partial u}{\partial x} \cdot \frac{\partial x}{\partial r} + \frac{\partial u}{\partial y} \cdot \frac{\partial y}{\partial r}$$

$$\frac{\partial u}{\partial s} = \frac{\partial u}{\partial x} \cdot \frac{\partial x}{\partial s} + \frac{\partial u}{\partial y} \cdot \frac{\partial y}{\partial s}$$

* If $u = f(x,y)$ is a homogeneous function of degree n:
  then: $x \cdot u_x + y \cdot u_y = n \cdot u$.

* If $u = f(x,y,z)$ is a homogeneous function of degree n:
  then: $x \cdot u_x + y \cdot u_y + z \cdot u_z = n \cdot u$

  * $x^2 u_{xx} + 2xy \cdot u_{xy} + y^2 u_{yy} = n(n-1)u$.

* If $u = f(x,y)$ is ~~not~~ not homogeneous function, but $F(u)$ is a homogeneous functionction of degree n.
  then: $x \cdot u_x + y \cdot u_y = n \cdot \left[\frac{F(u)}{F'(u)}\right] = G(u)$

  $x^2 \cdot u_{xx} + 2xy \cdot u_{xy} + y^2 \cdot u_{yy} = G(u)[G'(u) - 1]$

* If $u = f(x,y) \pm g(x,y) \pm h(x,y)$, where f, g & h are homogeneous functions of degree m, n & p respectively,
  then: $x \cdot u_x + y \cdot u_y = m \cdot f \pm n \cdot g \pm p \cdot h$

  $x^2 \cdot u_{xx} + 2xy\, u_{xy} + y^2 \cdot u_{yy} = m(m-1)f \pm n(n-1)g \pm p(p-1)h$

* If $u = f(r)$ & $r = f(x,y,z)$ then

  $u_{xx} + u_{yy} + u_{zz} = f''(r) + \frac{2}{r} * f'(r)$

Total Derivatives:

* If $u = f(x,y)$, where x & y are functions of t then the total derivative of u w.r.t t is:

$$\boxed{\frac{du}{dt} = \frac{\partial u}{\partial x}\left(\frac{dx}{dt}\right) + \frac{\partial u}{\partial y}\left(\frac{dy}{dt}\right)}$$

Let $x = t$,

$$\boxed{\frac{du}{dx} = \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}\cdot\frac{dy}{dx}}$$; where x & y are Connected-by some relation.

Maxima & Minima: [Extremum]

(37)

* 1) $f'(x) \neq 0$;
  * (i) $f'(x) = 0$ & Find stationary points: (x_i)
  (ii) If $f''(x_i) > 0$ then at x_i is minimum. $\forall x_i$ [min value $= f(x_i)$]
  (iii) If $f''(x_i) < 0$ then at x_i is maximum. $\forall x_i$ [max value $= f(x_i)$]

* 2) $f'(x) = 0$ & $f''(x) = 0$;
  (i) $x = a$ is minimum, If $f'''(a) = 0$ & $f^4(a) > 0$
  (ii) $x = a$ is maximum, If $f'''(a) = 0$ & $f^4(a) < 0$

Case 1: If $f(x)$ changes sign from +ve to -ve as x passes through c then $f(c)$ is maximum.

Case 2: If $f(x)$ changes sign from -ve to +ve as x passes through c then $f(c)$ is minimum.

Case 3: If $f(x)$ does not changes sign as x passes through c then $f(c)$ is not Extremum.

* For 2-variable functions:

(i) $f_x = 0$ & $f_y = 0$; Find Stationary points Simultaneously.

(ii) If $f_{xx} f_{yy} - f_{xy}^2 > 0$ & a) If $f_{xx} > 0$ then $f(x)$ is "min" at 'a'

b) If $f_{xx} < 0$ then $f(x)$ is "maximum" at 'a'

c) If $f_{xx} = 0$ then $x=a$ is Saddle point. [Either min or max]

(iii) If $f_{xx} f_{yy} - f_{xy}^2 < 0$ then not Extremum. (neither min nor max)

* To find Extremum of $f(x,y,z)$, where x, y & z are connected with relation $\phi(x,y,z) = 0$.

then (i) $F = f(x,y,z) + \lambda . [\phi(x,y,z)]$

(ii) Find x, y & z values as follows:

take $F_x = 0$, $F_y = 0$ & $F_z = 0$ relations (in x,y&z) and Substitute those relations in $\phi(x,y,z)$ to find the values of x, y & z

(iii) Extremum gets by substituting $x,y,$&z in given function $f(x,y,z)$.

Theory of Computation:

- Expressive powers: DFA ≅ NFA, DPDA ≠ NPDA (<), NTM ≅ DTM
- FA (DFA) = $(Q, \Sigma, \delta, q_0, F)$, PDA (NPDA) = $(Q, \Sigma, T, \delta, q_0, z_0, F)$,
  TM = $(Q, \Sigma, T, \delta, q_0, B, F)$
- δ(FA): $Q \times \Sigma \to Q$, δ(PDA): $Q \times \Sigma \cup \{\epsilon\} \times T \to Q \times T^*$
  δ(TM): $Q \times T \to Q \times \Gamma \times \{L, R\}$
- Chomsky Hierarchy: Type0 [R.E.L]; Type1 [CSL]; Type2 [CFL] & Type3 [R.L]
- (FA + 2 stack) ≡ (FA + 3 stack) ≡ (FA + 4 stack) ≡ ....
- All formal languages are recognized by TM. [Type0]
- K-length strings over Σ contains 2^K strings $[= \Sigma^K]$
- No. of possible FA's with q-states over l-length i/p alphabet $= q^q (q)^{ql}$
- For n-length string (including ϵ): No. of prefixes = $n + 1^{+\epsilon}$ (38)
  : No. of suffixes = $n + 1$
  : No. of substrings = $\frac{n(n+1)}{2} + 1$
- Minimal FA:
  $(a+b)^n$: Exactly-n : '(n+2) states' & Atmost-n : '(n+2) states' $(a+b)^*$
  $(a+b)^n (a+b)^*$: Atleast-n : '(n+1) states' & Divisible by-n : 'n-states' $(a+b)^*$
  Congruent m modulo n contains: 'n-states'
  n^{th} Symbol from L.H.S : (n+2) states
  n^{th} Symbol from R.H.S : 2^n states.
- $(a+b)^* = (a^*+b^*)^* = (a^*b^*)^* = (a^*+b)^* = (a+b^*)^* = (a^*b^*a^*)^* = (b^*a^*a^*b^*)^*$
- $\phi^* = \epsilon^* = (\epsilon)^* = \epsilon^+ = \{\epsilon\}$; $\phi^+ = \phi$; $\gamma^* = (\gamma^+)^* = (\gamma^*)^+ = (\gamma^*)^*$
- * $a(ba)^* = (ab)^* a$; $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$
- Moore: $(Q, \Sigma, \delta, q_0, \Delta, \lambda)$; $\delta: Q \times \Sigma \to Q$; λ (o/p function): $Q \to \Delta$ (o/p alphabet)
- Mealy: $(Q, \Sigma, \delta, q_0, \Delta, \lambda)$; $\delta: Q \times \Sigma \to Q$; $\lambda: Q \times \Sigma \to \Delta$
- In Moore; For n-length i/p, o/p generates (n+1) length
- In mealy; For n-length i/p, o/p generates 'n' length
- No. of Productions for (generating) x-length string in CNF: (2x-1) Productions (intermediate nodes)
  " " GNF: x Productions
- For given CFG (without ϵ & unit productions), there must an Equivalent CNF (Binary standard form) contain maximum of $(K-1)|P| + |T|$ productions (≤)
  where, K → max no. of symbols on R.H.S of any production
  |P| → no. of productions in CFG

- For given CNF, If length of longest path $\leq k$ for any string then its yield length (l) is always $\leq 2^{k-1}$ and for l-length yield string, the height of derivation tree : $\log_2 l + 1$
- Self Embedded Variable A : $A \overset{*}{\Rightarrow} \alpha_1 A \alpha_2$ & $\alpha_1, \alpha_2 \in (V+T)^+$
- For the given q-state mealy m/c over l-length o/p alphabet, the Equivalent moore m/c contain $\leq ql$-states.
- For i/p symbol 'ϵ', the o/p generated by Moore m/c is the "o/p associated with starting state", but the o/p generated by mealy m/c is "ϵ"
- "Complete DFA" is contained all states which are reachable from initial state.
- "Reduced/Non-redundant CFG" contains no useless symbols.
- "Simplified CFG" contains no ϵ-productions, no unit productions & no useless symbols.

If the given language (grammar) is non-Self Embedded CFG then the language is "regular".

TM acts as language generator/Enumerator, language recognizes & Transduces.

For 't'- tape symbols & 'q'-states in any TM, we can Construct Equivalent turing m/c (TM) ($t \propto \frac{1}{q}$) : 1) with 2-states & $(4tq+t)$ symbols
2) with 2-symbols & $8tq$ states

CFL is "inherently ambiguous", iff all of its grammars are "ambiguous". [If Even one grammar for CFL is unambiguous, then CFL is unambiguous language]

ϵ-NFA $\Rightarrow$ NFA : $\delta'(q,a) = \epsilon\text{-closure}\left(\hat{\delta}(\epsilon\text{-closure}(q), a)\right)$ [No. of states in NFA Equal to No. of States in ϵ-NFA]

ϵ-NFA $\Rightarrow$ DFA : $\delta'(q,a) = \epsilon\text{-closure}(\delta(q,a))$ [Initial State of DFA is ϵ-closure (Initial State of ϵ-NFA)]

Rank of a Variable (production) : length of longest path from that variable to a terminal variable

- There is $O(n^3)$ Algorithm that takes a PDA 'P' whose representation has length n & produces a CFG of length atmost ($\leq$) : $O(n^3)$

* Testing Emptyness of CFL's takes "$O(n)$ time" for a grammar 'G' of length n.

- Running time of CYK algorithm is $O(n^3)$ for n-length substring.

* For a grammar G of length n, we can find an Equivalent CNF grammar (G') for given 'G' in time of $O(n^4)$ & the resulting grammar G' has length : $O(n^4)$

* If DFA with n-states $\Rightarrow$ NFA or ϵ-NFA takes $O(n)$

- If NFA or ϵ-NFA with n-states $\Rightarrow$ DFA takes $O(n^3 2^n)$

- If DFA with n-states $\Rightarrow$ R.E (Regular Expression) takes $O(n^3 4^n)$

- If NFA with n-states $\Rightarrow$ R.E takes $O(n^3 4^n 2^n)$
(or ϵ-NFA) (NFA $\rightarrow$ DFA $\rightarrow$ R.E)

- If R.E has length-n $\Rightarrow$ DFA takes $O(2^n . n^3)$

- If R.E has length-n $\Rightarrow$ NFA takes $O(n^3)$

* If R.E has length-n $\Rightarrow$ ϵ-NFA takes $O(n)$

* Regular languages (R.L) & Non-regular languages:

- R.L's are not closed under "Subset" & "Infinite union" operations
- R.L's are closed under finite Subset & finite union operations.
- $\{(a^2)^*, a^{2n}, a^m b^n\}$ are regular.
- $\{a^{n^2}, a^{n!}, a^{2^n}, a^{2^{n^2}}, a^p\}$ are non-reg & non-CFL, but CSL

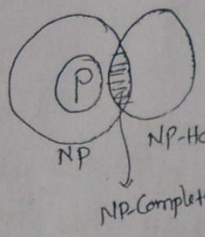
* DCFL's & CFL's:

- Set of all balanced parantheses is CFL, but not regular.
- Set of all valid Expressions are CFL, but not regular
- Set of all valid propositional functions are CFL, but not regular.
- $\{a^n b^n c^n, ww\}$, $\{a^m b^n / m \leq n^2\}$, $\{0^m 1^m 2^n / m \leq n \leq 2m\}$ are non-CFL, but CSL
- Complement of set of balanced parentheses is also CFL
- Complement of $a^n b^n c^n$ is DCFL

- ⊛ DCFL's closed under: [complement, H^{-1}, Min, & Max]
- ⊛ CFL's not closed under: [Complement, Min, Max, Intersection, Difference, & Half(L) & Quotient]
- ⊛ CSL's not closed under [Kleene closure, [Min(L) undecidable], Max & Cycle]
- * Recursive languages closed under: [∪, ∩, ·, Reversal, Complement]
- ⊛ Recursive languages are not closed under: [*, Substitution & Homomorphism]
- ⊛ R.E.L's are not closed under: [Complement, Max & Difference]
- R.E.L's are closed under [∪, ∩, ·, *, Reversal, Substi, H, H^{-1}, Cycle & Min]
- P- languages closed under [∪, ∩, *, ·, Complementation].
- NP- languages closed under [∪, ∩, *, ·, but not Complementation].
- Decision properties of DCFL are: [Membership ($w \in L$), Emptyness ($L = \phi$), Finiteness (L is finite), Regularity (L is regular), Totality ($L = \Sigma^*$) & Co-finite] are decidable
- Decision properties of CFL's are: [Membership, Emptyness & Finiteness (Decidable)]
- CSL's are decidable for "Membership".
- Recursive languages are decidable for "membership"
- ** Universal & Halting problems are R.E.L but not recursive.
- ** If L is non R.E.L, then L is not recursive [L is Non-REL or R.E.L but not recursive]
- * If L is R.E.L but not recursive, then Complement of L is non-R.E.L
- ** If L is R.E.L, then Complement of L is may or may not R.E.L.
- * If L is recursive, then Complement of L is Recursive.
- * Recursive languages are proper Superset of CFL's.
- ** Let 'L' be a language over Σ, i.e., $L \subseteq \Sigma^*$, Suppose L is in NP & for every 'n' there is Exactly one string of length-n that belongs to L, then the Complement of L over Σ^* is also in N.P.
- * Any CFL can be accepted by PDA with only 1-state

- Undecidle problems:
  1) Whether an abitaly TM halts after 100 steps
  2) Whether a TM prints a specific letter
  3) Whether a TM computes the products of two numbers.

* The set of all languages over Σ is "Uncountable"
- The set of all strings over Σ is Countable.
- The set of all regular languages over Σ is countable.
- The set of all languages over Σ accepted by TM is Countable.



1) $P \in NP$
2) $P = NP$ is undecidable
3) If L is NP-Complete iff $L \in NP$ & L is NP-Hard
4) NP-Complete $\in P$ iff $P = NP$

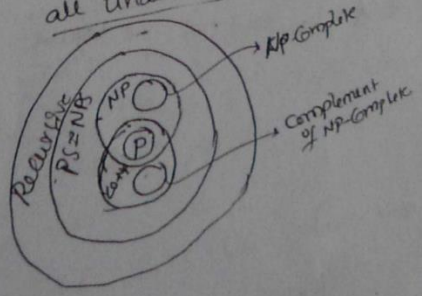
[Intractable problems]
[Exponential time $[O(2^{cn})]$
If $NP \propto_p \Pi$ then Π is called NP-Hard]

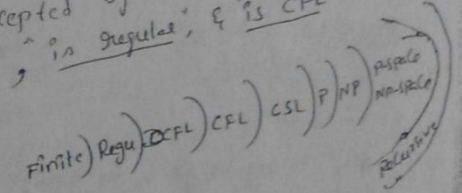
- If P_1 is NP-Complete & there is a Polynomial time Reduction of P_1 to P_2 then P_2 is NP-Complete. $[P_1 \propto_p P_2]$
- If Some NP-Complete problem 'L' is in P then P=NP
- P - Problems Solvable in Polynomial time by DTM
- NP - Problems " " " by NTM

- If there is a reduction from P_1 to P_2 then
  a) If P_1 is undecidable, then so is P_2
  b) If P_1 is non-REL then so is P_2
  c) If P_1 is NP-Complete then so is P_2
  Every Non-trivial property of RE languages is undecidable.

  c) If P_2 decidable then so is P_1
  d) If P_2 is NP then so is P_1
  e) If P_2 is in NP then so is P_1

* Whether the languages accepted by a TM is Empty, is finite, is regular, is CFL are all undecidable.





Type 3 ⊆ Type 2 ⊆ Type 1 ⊆ Type 0
R.G CFG CSL R.E.L
$A \to \alpha/\epsilon/\beta$ $A \to \beta$ $|\alpha| \le |\beta|$ $A \to \epsilon$ ... unrestricted $\alpha \to \beta$

Time(f(n)) → DTime, NTime

Space(f(n)) → DSpace, NSpace

DTime ⊆ NTime

DSpace ⊆ NSpace

NTime ⊆ NSpace

DTime ⊆ NSpace

R.L ⊆ DTime(n) ⊆ DSpace(n)

C.F.L ⊆ DTime(n^3) ⊆ DSpace(n)

C.S.L ⊆ DTime(n^{M}) (M not fixed) ⊆ DSpace(n^n)

- Whether a given string accepted by FSM is decidable.
- Whether a given CFG generates an infinite no. of strings - is decidable.

* Arden's method: If $R = Q + RP$ & P does not contain ϵ then the "Unique Solution" is $R = QP^*$

If P contains ϵ then $R = Q + RP$ has "infinite no. of Solutions".

- To find the minimization of FA, partition [State-Equivalence] method has O(nlogn) Complexity. & with Table filling - Algorithm takes $O(n^2)$ Complexity.
- Kleens method: $R_{ij}^{(k)} = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- Reversal FA: Interchanging the final & non-final states and reverse the Edge directions in the given FA.
- $h(h^{-1}(L)) \subseteq L$ & $h^{-1}(h(L)) \supseteq L$
- $Prefix(L) = L/\Sigma^*$ & $L_1/L_2 = \{x / \exists xy \in L_1 \text{ & } y \in L_2\}$ (Quotient)
- $\Sigma^*/L = \Sigma^*$ (L is non-Empty)
- If $L_1 * L_2$ is regular & L_1 is regular then L_2 need not be regular. [where $* \to \cup, \cap, -, /$]
- R.L / C.F.L is Regular language.
- CFL / x is CFL; $[x \in \Sigma^*]$

Digital Logic:

No. of Boolean functions formed over ⓐ n-boolean variables : 2^{2^n}

ⓑ n-n valued variables : 2^{n^n}

No. of n-Valued functions formed over: ⓐ n-boolean variables : n^{2^n}

ⓑ n-n valued variables : n^{n^n}

- No. of boolean functions formed over n-boolean variables, such that for all values of $x_1, x_2, \ldots x_n$ with function $f(x_1, x_2, \ldots x_n) = f(x_1', x_2', \ldots x_n')$ are : $2^{2^{n-1}}$
- No. of Neutral functions formed over n-boolean variables : $^{2^n}C_{2^{n-1}}$
- Exclusive OR ($\oplus$) : odd no. of i/p's are 1 [odd function]
- Ex-NOR ($\odot$) : Even no. of i/p's are 0' [Even function].
- $(A \oplus B)' = A \odot B$ & $A \oplus B \oplus C = A \odot B \odot C$
- $A \odot B = (A \oplus B)' = A' \oplus B = A \oplus B' = (A' \oplus B')'$
- $XY + X'Z + YZ = XY + X'Z = (X+Z)(X'+Y)$
- $X(Y+Z) = XY + XZ$ & $X + (Y \cdot Z) = (X+Y) \cdot (X+Z)$
- NAND (↑) & NOR (↓) are not Associative.
- $\overline{f}$ = complement of each literal on dual function of f.
- fmpos = $\overline{\text{fmsop for maxterms (zero's)}}$ & fmsop = fmsop for minterms (one's)
- Minimal Sop (fmsop) = All E.P.I's + { P.I's for covering remaining 1's, if not covered by EPI } [covering all 1's]
- Maximum no. of product terms for irreducible Expression with n-variables are : 2^{n-1} & Max no. of literals $(n \times 2^{n-1})$
- Let $f(x_1, x_2, \ldots x_n)$ be Equal to 1 iff k or more variables are Equal to 1, then no. of implicants/prime implicants in f are : ^nC_k
- No. of Self dual functions over n-boolean variables : $2^{2^{n-1}}$

- 4×1 MUX : $\left[\bar{A}_1\bar{A}_2 I_0 + \bar{A}_1 A_2 I_1 + A_1\bar{A}_2 I_2 + A_1 A_2\right] E$
- NO of $(m\times1)$ mux's required to implement $(m\times1)$ MUX : $\left\lceil \frac{m-1}{n-1} \right\rceil$
  and NO of Levels required : $\log_n^m$

* NO of $n\times2^n$ decoders required to realize $m\times2^m$ decodes:
  $X + Y + Z + \ldots\ldots\ldots + (I < 2^n)$
  where
  $X = \left\lceil \frac{2^m}{2^n} \right\rceil$. $Y = \left\lceil \frac{X}{2^n} \right\rceil$. $Z = \left\lceil \frac{Y}{2^n} \right\rceil$, ...
  & NO. of levels = NO. of times dividing with $2^n + 1$.

- To implement n- functions, we require :
  1) n - mux's or
  2) one $n\times2^n$ decodes & n-OR gates.

To avoid
- Race around condition in JK-FF :
  Pulsewidth $\leq$ Propagation delay $\leq$ Clock period.

- $Q^+ = S + \bar{R}Q = D = T \oplus Q = J\bar{Q} + \bar{K}Q$
- $T = J\bar{Q} + KQ$ & $D = J\bar{Q} + \bar{K}Q$ & $S = J\bar{Q}$; $R = KQ$
- Serial (Spatial) & Temporal (Parallel)
- X-OR gate with 6-Variables i.e: $A\oplus B\oplus C\oplus D\oplus E\oplus F$, then the no of minterms in the boolean expression = 6.
- All digital circuits [boolean functions] can be realized by using either MULTIPLEXER or HALFADDER.
- Addition of two 16-bit numbers : 15 FA's & 1-H.A required.
- Four bit (4-bit) F.A $\equiv$ 7 H.A & 3 OR gates.

* In n-Variable K-map ; 2^x adjacent cells containing 1's
  then 1) the result term has $(n-x)$ literals.
  2) It Eliminates x variables.

Q	Q+	S	R	J	K	D	T
0	0	0	x	0	x	0	0
0	1	1	0	1	x	1	1
1	0	0	1	x	1	0	1

ROM : with only gates
: with decoders, Encoders
: Semiconductor & permanent memory

Full Subtractor = 1 FA + 1 NOT
= 2 H.S + 1 OR

Full Adder = 1 F.S + 1 NOT
= 2 H.A + 1 OR

H.A = 1 H.S + 1 NOT

H.S = 1 H.A + 1 NOT

Full Adder : Cally : $xy+yz+zx = x(y \oplus z)+yz$

Sum = $x \oplus y \oplus z = (Cally)'(x+y+z)+xyz$

Full Subtractor : Difference : $x \oplus y \oplus z$

Borrow : $\bar{x}y+\bar{x}z+yz = \bar{x}(y \oplus z)+yz$
$= z(x \odot y)+\bar{x}y$

* No. of Symmetric functions over n-boolean variables: 2^{n+1}
* Implication & Inhibition are not Commutative
* AND & OR are only Identitiy Property Satisfies.
* Addition of n-bit parallel Adder requires: (n-1) FA + 1 H.A
(or)
(2n-1) HA + (n-1) OR

. Maximum Signal propogation delay in an n-bit ripple Cally Adder if 'D' is delay of Full Adder (FA) : = n.D

. Maximum Signal propogation delay in an n-bit Cally Look ahead Adder if 'D' is average gate delay = 4D

. Size of ROM, m-functions with n-variables Each implementation : $m \times 2^n$

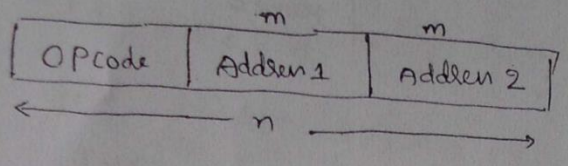
. Size of PLA, m-function with n-variables Each & P-Product terms : $m \times n \times p$.

. No. of Links, required to implement m-functions with n-variables - Each : $m \times 2^n$

. No. of Links, to implement PLA with x-i/p's, P-product terms & y-o/p's are : $2xp + yp + y$.

* NAND & NOR are not associative.
. over Even no. of variables ; Ex-OR & Ex-NOR are Complements - to Each other
. over odd no. of variables ; Ex-OR & Ex-NOR are same.

Computer Organization & Architecture:



$\Rightarrow P * 2^{2m} + Q * 2^{m} + R = 2^{n}$ (↳ total instr)

- P → 2-Address
- Q → 1-Address
- R → Zero Address Instructions

- H/w Control unit : Every Control Signal is Expressed as a boolean Sum of products and it is realize with 3-level gates. "The delay of generating any Control Signal is going to be Constant"
- Control Signals in H/w Control unit (Generated) determined by: Contents of "Control Step Counter", "Instruction Register", & "Condition Code flags" and "External Input Signals [MFC (Memory function Completed), & Interrupt requests]"
- In μP [microprogrammed] Control unit : Control Signals are Stored in Control memory in the form of Control instruction [Control word]

  Control Instruction: | Status bits | Control Signals | Next Address |
- Control Signals in μP Control unit are (determined) generated by : Contents of Instruction Register, Starting Address generator, Clock & micro program Counter, Control Store, External i/p's & Condition-Codes.
- In Horizontal μP, one bit for each control signal is allocated.
- In Vertical μP, Control Signals are in Encoded form, requires External-decoder while generating them.
- Horizontal μP (more memory) is faster than Vertical μP. (less memory)
- Either One or none of n-Control Signals are generated in "Vertical μP" & it requires $\lceil \log_2^n \rceil$ bits.

  Atmost n-Control Signals are generated in "Horizontal μP" & it requires n-bits.
- Maximum degree of Parallelism with n-Control Signals for one field In Horizontal μP is : n & In Vertical μP is : 1

Size of Cache : (bits)

= Memory needed for Tag (bits) + Memory needed for word (bits)

$= \left[\text{No. of blocks in cache} \times \underset{\text{(Tag bits)}}{\text{No. of bits per Tag}}\right] + \left[\text{No. of words in Cache} * \text{No. of bits per word}\right]$

- There is no "Conflict miss" in associative mapping.
- In Direct mapping Cache, n^{th} block of mainmemory can be placed in one cache block by mapping: $n \bmod N$ (N = No. of Blocks in Cache; n^{th} block of MM)

In Associative mapping, At a time one block of MM can be placed any where in Cache.

In K-way Set Associative mapping, At a time one block of MM (K) can be placed in one set by mapping $n \bmod S$ (S = No. of sets in Cache).

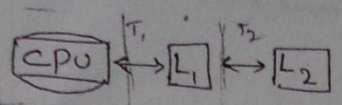
Dirty bit (M) : 0 (Not updated) & 1 (updated) [used by write BACK].

- Write through:
  - $T_{avg} = f_r * T_{avgr} + f_w * T_{avgw}$ (f_w → frequency of write references)
  - $T_{avgr} = T_C + (1 - H_R) \cdot T_B$ (T_C → Cache)
  - $T_{avgw} = T_M + (1 - H_w) \cdot T_B$ (T_M → mainmemory; H_w → Hit ratio of write)

$T_B = T_M * n$ (n = No. of word blocks (Genefetting) (from memory to Cache))

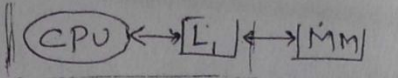
- Effective Hit Ratio (H_{eff}) $= H_R * f_r + H_w * f_w$
- Direct mapping: Tag + Block + word offset (No. of blocks in Cache; No. of words per block) (or) Tag + Index
- Associative mapping: Tag + word offset
- Set Associative mapping: Tag + Set offset (No. of sets in Cache) + word offset

* Improvement in performance $= \frac{T_{without\ Cache}}{T_{with\ Cache}} = \frac{1}{\text{Relative performance}}$

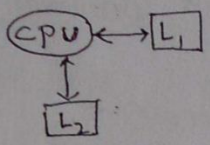
* 

$T_{avg} = T_1 + (1-H_1) T_2$

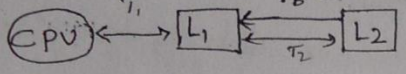
$= H_1 \cdot T_1 + (1-H_1)(T_1+T_2)$



$T_{avg} = H\,T_C + (1-H)\,T_M$

* 

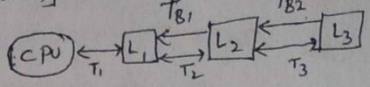
$T_{avg} = H_1 T_1 + (1-H_1)\, H_2\, T_2$

* 

$T_{avg} = T_1 + (1-H_1)\, T_B$

where $T_B = n * T_2$

$n \rightarrow$ no. of word blocks transfered from L_2 to L_1



$T_{avg} = T_1 + (1-H_1)\, T_{B1} + (1-H_1)(1-H_2)\, T_{B2}$

where $T_{B1} = n_1 * T_2$

$T_{B2} = n_2 * T_3$

93

* Hit ratio = Total no. of hits / Total no. of references.

* No. of Main memory blocks placed in any cache :

$= 2^{\text{no. of Tag bits}} = 2^{\text{Tag}}$ [where Tag may be different from one to other mapping]

* No. of Tag Comparators required :

1) Direct mapping = 1 [It contains no. of bits as many as tag contains]

2) Associative mapping = No. of blocks in cache. ["]

3) K-Set Associative mapping = K (No. of blocks in set) ["]
↳ [Each set contains K-blocks]

* No. of Tag bits required :

1) Direct mapping : $\log_2(M/C)$: M → [Main memory blocks]

2) Associative Mapping: $\log_2(M)$: C → [Cache blocks]

3) K-Set Associative mapping : $\log_2(KM/C)$ K → No. of blocks in each set

- In Program driven [Programmed I/O] & Interrupt driven the I/O transfer (data) to/from memory done under control of CPU.
- DMA methods: 1) Continuous data transfer (Block/Burst mode) (Exclusive access given by CPU)
  2) Cycle Stealing: Stretching [Address calculation, (Fixed operand or Instruction transfer) Decoding, Result store] [when CPU not uses bus]
     Transparent [steals, when IF, OF & Result Store] (Break points) & operand fetch
  3) Interleaving: [First Half Cycle (DMA active) & Second Half Cycle (CPU active)]

* Program Execution time (= N*K*T) = $\frac{\text{No. of Instructions (N) } \times \text{ Avg no. of steps (K) needed to Execute one Instruction}}{\text{clock rate (Cycles Per Second)}}$

* Speed factor = $\frac{T_{\text{non pipelined}}}{T_{\text{pipelined}}} = \frac{N * CPI_{avg} * T_{clock}}{N * CPI_{avg} * T_{clock}} = \frac{n*K*T}{(n+K-1)T}$

* Speed up factor (R) = $\frac{\text{Speed factor A}}{\text{Speed factor B}} = \frac{S_A}{S_B}$; [If R > 1 ; A is better; If R < 1 ; B is better] 44

* $T_{np} = T_{up}$ (unpipelined / non-pipelined) $= n*K*T$; K → no. of stages (Pipeline) or phases (non-pipeline)
* $T_p = [K + (n-1)]*T$; T → Time (clock) taken for each stage (Stage delays are equal); n → no. of instructions.
* $T_{np} = n * [T_1 + T_2 + \cdots + T_K]$; [If stage delays (times) are different.
* $T_p = [K + (n-1)] * T_{max}$;
* No. of stages (K) = $\frac{100}{\text{Efficiency (\%)}} \times$ Speed factor (S). [n → S; 100% → ?]
* Avg Instruction Execution time (Taverage) = $[1 + \text{Stall freq} \times \text{Stall Cycles}] * T$
* Instruction < Branch (fB) < Conditional (fcb) > Branch taken (fcbt) (Non-overlapped); Not taken branch
  Branch (fB) > unconditional (fucb) (Non-Overlapped); Not Branch taken
* $T_{avgI} = [1 + f_B [f_{cb} * f_{cbt} + f_{ucb}] * \text{Stall cycles}] * T$; Stall cycles = (no. of stages − 1); if stall cycles not given

* Data dependency: Pipeline stages have different delays, due to that data in not available for other instruction in expected time. [RAW, WAR, & WAW]

* Control dependency (Instruction dependency):

Delay in availability of an instruction: [due to branch, miss in cache, Prefetch, Delayed load, multiple pipeline & prediction].

* Structural dependency:

When two instructions requires same hardware resource at the same time.

- Avg access time = Seek time + Avg latency time + Data transfer time
  (desired track in disk; desired sector within track; Avg Rotational latency)
- Avg latency time = $\frac{1}{2}$ * Rotational latency
- Rotational latency = $\frac{1}{\text{Rotational speed}}$

* 1 Rotation is to read the one track completely.
* Unit data transfer time: time taken to read the one sector.
* Disk moves with Either linear or Angular velocity
* Tape moves with fixed linear velocity. & Single
- one Complete frame is scanned in a Single scan