- Data Structure → way of collecting and organising data.
  - Used to increase efficiency of programs & reduce Complexity of programs.

  Algorithm + data structure = program

- Algorithm: Stepwise solution of a problem
- Abstract data type: Specifies type of data stored, operations supported on them and so on. Provides concept for data abstraction & data hiding.
- Dynamic memory allocation:- Process of manually allocating memory during runtime.

  malloc → used to allocate memory dynamically

  ~~fea~~ calloc → used to allocate multiple blocks of memory

  realloc → change the size of memory block allocated

  free → used to deallocate the dynamically allocated memory

- Algorithm complexity:- Gives the running time and storage space required by the algorithm in terms of input size.

  * Space complexity:- represents the amount of memory space required by the algorithm

  * time complexity: represents the amount of time required by the algorithm to run to completion

- Asymptotic Notation:- Mathematical notation to calculate time complexity of algorithms.

  *(Big Oh Notation):- used to classify algorithms according to how their running time or space requirement grows as the input size grows.

- Stack: Linear data structure based on the concept of LIFO. Elements may be inserted or deleted using push and pop operation from the end.

  Application:

  → Evaluate postfix & prefix exp, Used in recursion

  → Page-visited history on web pages, perform undo in text editors

- Queue: Linear data structure based on the concept of FIFO. ordered collection of objects. Items may be deleted from front end and inserted from rear end. Enqueue & dequeue are used to insert & delete elements.

  Application:

  → Task waiting for printing

  → Time sharing system for use of CPU

  → Task scheduling in operating system

○ Linear queue: Ordered list in which elements are organised in sequential order. Insertion & deletion is fixed: ie front & rear. wastes memory space.

○ Circular queue: Stored data in circular fashion. Capable of inserting & deleting from any point until it is occupied. Makes efficient use of memory space.

○ Priority Queue:- collection of elements such that each elements is assigned a priority. Element with higher priority is processed before element with lower priority. If equal priority, then processed according to order in which they were added to the queue.

Applications: i) CPU scheduling. ii) Priority based jobs. iii) Time sharing system

○ Dequeue: Linear list, in which elements can be added or removed at either end but not middle. Under Double-Ended Queue. It is maintained by a circular array with pointer LEFT & RIGHT which points to the two ends of the queue.

○ Linked List:- Linear collection of data elements whose order is not given by their physical placements in memory. instead each elements points to next. It is basically the collection of nodes which together represents a sequence.

Array	Linked List
→ collection of data elements whose size is fixed (insertion & deletion is difficult)	→ " " " size is not fixed
→ same amount of time reqd to access each element	→ different amount of time
→ If we have to go to a particular elements, then we can reach directly, & it is more expensive	→ if we have to go " " " we have to go through all nodes that come before that node.

○ Singly Linked list: Each node contains two fields: info field & link field
info → used to store data items.
next ~~link~~ → point to the next node in sequence.

○ Doubly Linked List: Node contains three fields.
prev → point to previous node in sequence
info → used to store data
next → point to next node in sequence

○ Circular Linked List: It is a linked list where the link field of last node points to the very first node of the list.

- Searching: A technique that helps to find the place or position of a given element or value in the list.

- Linear/sequential Search:- It is a method of finding an element in the list. It sequentially checks each element in the list until a match is found or the whole list is searched.

- Binary Search: Another method of searching. It works only with sorted list. It compares search element with the middle element in the list.

- Hashing: Efficient searching technique. It calculates the position of the key in the hash table based on value of key. It uses a hash function.

- Hash function:- Transforms a key into the table index value. It It maps the key with corresponding key address or location. It will take any item & returns hash value betn the range 0 to n-1 where n is the size of hash table.

- Hash table:- A data structure used for storing and accessing data very quickly during hashing. Insertion of data is based on key value.

- Types of hash function:
  1) Division:- hash key is based on modulo i.e. remainder.
  2) Mid square: key is squared & mid part is taken as hash key / index
  3) Digital folding:

- Hash collision:- Also known as clash, is a situation when two distinct pieces of data have same hash value..

- Collision resolution:- When hash collision occurs, there must be systematic method for placing the second item in table. This process is called
  - * Open hashing:- Chaining, hashing using bucket (same location is sought)
  - * closed hashing - (open addressing) → Linear probing, Quadratic probing, Double hashing (another location is sought)
  - * Rehashing

→ Chaining:- It allows each slot in table to hold a chain of items. Allows many items to exist at the same location.

→ Linear probing:- If data can't be placed at the index calculated by the hash function, we put it in next empty space in table. It has clustering problem

→ Quadratic probing:- eliminates primary clustering problem. $(h+i^2)$ % table size, h = hash (index) value, i = no. of collision occured

There is no guarantee of finding slot after table is half full

→ Double hashing:- $(h_1(x) + i\, h_2(x))$ % (table size). , $h_1(x) = x \bmod$ size of table, $h_2(x) = R - (x \bmod R)$

→ Rehashing:- Hashing again. Load factor $(\lambda) = \frac{n}{N}$, n = no. of data ele, N = hash table size. λ must be ≤ 1. If $n > N$, then we do rehashing by increasing buckets or modify hash function

1 ○ Recursion: It is the process by which a function calls itself directly or indirectly to work in a smaller part of the program. A problem is solved by dividing it into smaller problems, which are similar in nature to the original problem. These smaller problems are solved & their solutions are applied to get the final form of

○ A function is recursive if i) It calls itself
ii) Function should have base case (stopping

○ Tail recursion: A recursive function is tail recursive when recursive call is the last thing executed by the function. It is better than non-tail recursion, as there is no task left after recursive c

○ Some recursive algorithm: factorial, gcd, Fibonacci series, ToH

○ Applications of recursion: i) Used in Trees
ii) Sorting algorithms uses recursion
iii) searching " " "

○ Advantages & disadvantages

i) The code becomes easier to write.
ii) Reduces unnecessary calling of functions

Disadv:-
i) slower than non-rec. function
ii) requires more memory.
iii) difficult to think recursively

○ Sorting: Process of ~~ordering~~ arranging the elements in specific order.

○ Internal sorting:- Arranging the numbers within the array only, which is in computer's primary memory.

○ External sorting:- Sorting of number from external file by reading from secondary memory.

○ Uses of sorting: Dictionary, index of book, banking applications

* Comparison sorting algorithm: Bubble sort, selection sort, insertion sort

* Divide & conquer algorithm: Quick sort, merge sort, heap sort, shell sort

○ Bubble sort: Simplest sorting alg. works by repeatedly swapping the adjacent elements if they are in wrong order.

○ Selection sort: It is about picking smallest element from given array & placing it into sorted portion in list. Initially 1st element is considered to be min. & compared with other elements.

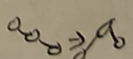
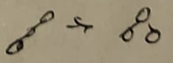
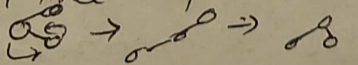
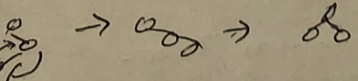
○ Insertion sort: element gets compared & inserted into the correct position in the list. you consider 1 part of list to be sorted & another consider 1st element to be sorted posn, & other elements - - - -

* Divide & Conquer: Important problem solving technique that makes use of recursion.
Divide: smaller problems are solved recursively
Conquer: Solution of original problem is then formed from the sol's of "

○ Quick-sort: Fast D&C algorithm. [partition & sort phase]

○ Merge-sort: Merging two or more sorted files into a third sorted file.

○ Shell short: Improved version of insertion sort. Contains Gap.

- Tree: A tree is a non-linear data structure, which is a collection of nodes linked with each other in a non-cyclic form. A tree has a root, internal & leaf nodes
- Root: A node with no parent.
- Leaf: A node with no children
- Internal node: Node with at least one child.
- Path: Sequence of edges from source to destination node
- Depth of node: Length of path from root to the node.
- Height of node: Length of longest path from node to a leaf node.
- Depth of tree: Depth of node that is maximum among all the nodes
- Height of tree: Height of root node.
- Sibling: Nodes having same parents
- Binary tree: Tree in which each internal node has at most (max) two children.
- Strictly binary tree: Each internal node has either 0 or 2 child.
- Complete binary tree: Nodes at each level have only two child
- Almost complete binary: Nodes are fixed from left to right from the parent node
- Binary Tree Traversal: Visiting each node exactly once in a systematic manner. (1) Pre-order: root, left, right (2) In-order: left, root, right 3) post-order: left, right, root.
- Binary Search Tree: A binary tree that is either empty or every node contains a key value that satisfies:-
  - → All keys in left subtree of root are smaller than root key
  - → " " " right subtree are greater than root node key
  - → The left & right subtree of the root are again BST.
- AVL Tree: Balanced binary tree. It is a bst, where the balance factor at each node is -1, 0, or 1. Balance factor = height of left subtree - height of right subtree.
- Make non avl tree to avl: by rotation:
  1) Left rotation:- when tree becomes right skewed 
  2) Right rotation: when tree " left skewed: 
  3) Left-right rotation: 
  4) Right-Left rotation:- 

- Graph: Graph is a non linear data structure which is a set of vertices(V) connected by edges 'E'. G(V,E). A graph may be directed or undirected. Each edge has one or two vertices associated with it.
- Directed graph: Direction of edges is specified.
- Simple graph: Graph with no multiple edges. or loops
- Multi graph: Graph with multiple edges.
- Pseudo graph: Graph with loops as well as multiple edges
- Mixed graph: Graph with both directed & undirected edges
- Application of graph: i) Model geographic map of cities
  ii) Analyze the electrical circuits
  iii) Finding shortest paths.
- Representation of graph: Adjacency lists & adjacency matrix.
- Adjacent vertices: Two vertices connected with common edge
- Adjacent edges: Two edges going out of a same vertex
- Adjacency list: Specifies the vertices adjacent to each vertex
- Adjacency matrix: n-by-n zero-one matrix $A = [a_{ij}]$
  $a_{ij} : \begin{cases} 1 & \text{if } v_i \,\&\, v_j \text{ are adjacent} \\ 0 & \text{o/w} \end{cases}$
- Incidence matrix:- matrix formed by incidency of edges
- Complete graph: Simple graph that contains exactly one edge between each pair of vertices. K_n.
  K_1 • K_2 — K_3 △ K_4 ⊠
- Connected graph:- An undirected graph is connected if there is a path between every pair of distinct vertices of graph
- Graph Traversal:- Visiting each vertices in a graph exactly once in a systematic manner.
  → Breadth First Search (BFS):- uses Queue.
  → Depth First search (DFS): uses Stack.
- Shortest path:- Shortest path betn any two vertices with minimum weight.
- Relaxation: Process of testing total weight of shortest path betn any 2 vertic
- Dijkstra's Alg:- ~~Short~~ Single Source Shortest path algorithm for +ve weight gr
- Spanning Tree: A subgraph of a graph G with minimum possible n of edges. If G has n vertices, then its spanning tree has n-1 edges. It cannot form cycle, cannot be disconnected.
- Minimum Spanning Tree:- A m.s.t of a weighted graph G is a subgraph of G that includes all vertices of G with min. possi edges & has min. weight than all other possible spanning tree.
- Kruskal's Algorithm:- M.S.T finding algorithm. It may not be connected from beginni
- Prim's algorithm:- It is always connected from beginning to end.