Data Structure Workouts

1. Learn what is Data Structure & Algorithms.
2. Learn the basics of Memory Allocation and Memory leak.
3. Learn the concept of Complexity Analysis.
   NB: The complexity of common operations of all data structures should be covered.
4. Learn about Asymptotic analysis (Big-O notation).
5. Learn the concepts of Array. Complete at least three sample workouts & do at least 3 problems from any competitive coding websites (Hacker Rank, Code Chef, Leet code, Algo Expert, etc.)
6. Learn the concepts of the Linked list. Complete at least three sample workouts
   a. Construction of Singly linked list & Doubly linked list.
   b. Convert array to a linked list
   c. Add a node at the end & beginning
   d. Delete node with the value specified
   e. Insert a node after & before a node with x data
   f. Print all elements by order & reverse by order
   g. Write a program to remove duplicates in a sorted singly linked list
7. Learn the concepts of String. Complete at least three sample workouts.
   Eg: Write a function to replace each alphabet in the given string with another alphabet occurring at the n-th position from each of them.
8. Learn about Linear Search & Binary Search. Complete at least 3 sample workouts in each of them
9. Learn the concepts of Recursion. Complete at least 3 sample workouts.
10. Learn about the applications of all structures you covered this week

Write a short description about this task

In order to effectively access and use data, it must be organised and stored in a computer system using a data structure. Data structures like arrays, linked lists, trees, and networks are examples. Each data structure has advantages and disadvantages, and selecting the best one for a given issue can result in algorithms that are more effective. An algorithm is a detailed process or collection of instructions created to address a particular issue. Data saved in data structures can be manipulated, sorted, searched for, and analysed using *algorithms. The ability to solve complex issues quickly depends on the use of efficient algorithms. Data structures and algorithms are used in conjunction to create effective software applications that can quickly and effectively handle challenging problems.*

Write a short description about this task

The act of allocating memory to a programme while it is running is known as memory allocation. The kernel controls memory allocation and deallocation in contemporary operating systems. Memory distribution comes in two flavours: static and dynamic. At compile time, static memory is allocated, and it is given for the duration of the programme. On the other hand, dynamic memory allocation is carried out at runtime and is assigned and deallocated as required. Memory leaks, which occur when memory that has been assigned is improperly deallocated, are a frequent issue in software development. As a result, the programme gradually uses up more and more memory. Numerous problems, such as crashes, speed issues, and other unexpected behavior, may result from this.

Because the code that was meant to deallocate the memory was improperly written or because of programming error, memory leaks happen when the programme fails to release the memory allocated to an object. It's crucial to make sure that all dynamically allocated memory is correctly deallocated when it's no longer required to prevent memory leaks.

Write a short description about this task

The method of evaluating an algorithm's performance in terms of its time and space requirements is known as complexity analysis. It aids in our comprehension of how an algorithm's performance changes as the size of the input grows. The number of operations the algorithm performs and the amount of time needed for each operation are the two key elements that decide how time-complex an algorithm is. On the other hand, the quantity of memory needed to run an algorithm affects how complex it is in terms of space.

```
Arrays
  Accessing an element: O(1)
  Searching for an element: O(n)
  Inserting an element at the end: O(1)
  Inserting an element at the beginning: O(n)
  Deleting an element: O(n)
```

```
Linked Lists
  Accessing an element: O(n)
  Searching for an element: O(n)
  Inserting an element at the end: O(1)
  Inserting an element at the beginning: O(1)
  Deleting an element: O(1)
```

Write a short description about this task

Asymptotic analysis is a way of analysing the performance of an algorithm by looking at how its running time or memory usage scales with the size of the input. It provides a way to compare the efficiency of different algorithms without worrying about the specific hardware or implementation details.

Big-O notation is a commonly used notation in asymptotic analysis. It represents the upper bound of the growth rate of an algorithm's time or space complexity. In other words, it gives us an idea of how fast the algorithm will run as the input size grows larger.

Big-O notations and their meanings:

O(1): constant time - the running time of the algorithm does not depend on the size of the input.

O(log n): logarithmic time - the running time of the algorithm grows logarithmically with the size of the input

O(n): linear time - the running time of the algorithm grows linearly with the size of the input.

O(n log n): linearithmic time - the running time of the algorithm grows at a rate of n log n with the size of the input.

O(n^2): quadratic time - the running time of the algorithm grows at a rate of n squared with the size of the input.

O(2^n): exponential time - the running time of the algorithm grows exponentially with the size of the input.

Write a short description about this task

An array is a data structure that stores a collection of elements, such as numbers, characters, or objects, in contiguous memory locations. Arrays are a fundamental data type and are commonly used to store and manipulate collections of data. Arrays provide efficient random access to individual elements, but may be less efficient for inserting or deleting elements in the middle of the array, especially for large arrays.

Link to the folder containing code and screenshot of the output

Array Workouts

Write a short description about this task

A linked list is a data structure that consists of a sequence of nodes, where each node contains data and a reference or pointer to the next node in the list. Linked lists are used to implement dynamic data structures that can grow or shrink in size during runtime.

Link to the folder containing code and screenshot of the output

linked list

Write a short description about this task

In programming, a string is a sequence of characters, such as letters, numbers, and symbols. Strings are a fundamental data type and are commonly used to store and manipulate text data. There are many built-in string methods that allow you to perform common string operations, such as converting a string to uppercase or lowercase, finding the index of a substring, and replacing a substring with another string.

Link to the folder containing code and screenshot of the output

String Workouts

Write a short description about this task

Linear search and binary search are two popular algorithms used to search for an element in a list or an array. Linear search is a simple algorithm that iterates through the list or array one element at a time until it finds the desired element. Binary search, on the other hand, is a more efficient algorithm that works by repeatedly dividing the list or array in half until the desired element is found.

Link to the folder containing code and screenshot of the output

Binary & Linear search

Write a short description about this task

Recursive function calls keep happening until a predetermined condition, usually a base case, is satisfied. The base case is the problem's most basic iteration that can be resolved without the use of further recurrence. When the function has finished handling the base *case, it passes the outcome to the function that came before it on the call stack so that it can be used to handle more challenging cases.*

Link to the folder containing code and screenshot of the output

Recurrsion

Write a short description about this task

Arrays

- ➢ *Sorting Algorithms*
- ➢ *Searching Algorithms:*
- ➢ *Data Storage:*
- ➢ *Graphics and Image Processing*
- ➢ *Mathematical Applications*
- ➢ *Game Development*
- ➢ *Web Development*
- ➢ *Operating Systems*
- ➢ *Artificial Intelligence and Machine Learning*

Linkedlist

- ➢ *Dynamic Memory Allocation*
- ➢ *Implementing Stacks and Queues*
- ➢ *File Systems*
- ➢ *Music and Video Players*
- ➢ *Web Browsers*
- ➢ *Hash Tables*
- ➢ *Garbage Collection*