DATA STRUCTURE

# Introduction

Data Structures are building blocks or raw material for any software programs.

# Big O notation

Big O notation is used to measure how running time or space requirements for the program grow as input size grows.

|  |
| --- |
|  |

## Rule Book

### Rule 1 (Always consider the worst case)

### Rule 2 (Remove constants)

### Rule 3 (Different terms for inputs)

### Rule 4 (Drop Non dominants)

## O(n) – Linear Time

|  |
| --- |
|  |

## O(1) – Constant Time

|  |
| --- |
|  |

## O(n^2)

|  |
| --- |
|  |

## O(n!)

It is the most expensive one and worst. For example, we are adding a nested loops for every input we have.

void nFacRuntimeFunc(int n) {

for(int i=0; i<n; i++) {

nFacRuntimeFunc(n-1);

}

}

## O(n log n)

Divide and conquer.

## O(log n)

Binary Search

# Data Structure

## Array

It is nothing but storing data in a sequential order.

|  |
| --- |
|  |

|  |
| --- |
|  |

## Hash Table

Hash Table is nothing storing value via key and value pair. Based on the key we find the value. It uses **Hash Function** to get the index for storing the value in Ram.

In JavaScript: - Object (which uses Hash Table as the underline database)

In Python: - Dictionary (which uses Hash Table as the underline database)

E.g :- basket.grapes = 1000

|  |
| --- |
|  |

|  |
| --- |
|  |

### Hash Collisions

With more data and limited data, we will always have collision. We can avoid collision using Linked list (Separate channing)

|  |
| --- |
|  |

## Linked List

|  |
| --- |
|  |

**Node contains value of the data and a pointer pointing to the next nod**e.

|  |
| --- |
|  |

### Pointer

It is the reference of another place in memory.

### Doubly Link List

|  |
| --- |
|  |

Benefits of Doubly link list is helps to traverse backward. Drawback is it will take some addition memory with respect to singly linked list.

|  |
| --- |
|  |

## Stacks

LIFO – Last in First Out

E.g:- Browser history back and front, undo in text editor

Stack can build with array as well as with Linked list

|  |
| --- |
|  |

## Queue

FIFO – First in first out

E.g:- Many are their, like 1) Movie Ticket app, 2) Uber booking cab, 3) Printer

enqueue: - append a value to the end of the queue.

dequeue: - Remove the value at the start of the queue.

Peek:- return the value at the start of the queue.

Empty:- return a Boolean value of whether the queue is empty or not.

|  |
| --- |
|  |

Queue always try to implement using Linked list instead of array.

|  |
| --- |
|  |

## Tree

In tree a node can point to only child.

Linked list also we can consider as tree.

Eg. DOM

|  |
| --- |
|  |

### Binary Tree

Each node will only have either zero, one or two nodes and each child can only have one parent.

|  |
| --- |
|  |

#### Binary Search Tree

Rules: -

1. All child node in the tree to the right of the root node must be greater than the current node.
2. All child node in the tree to the left of the root node must be less than the current node.
3. A node should have max 2 children.

|  |
| --- |
|  |

Disadvantages it can lead to unbalanced BST.

|  |
| --- |
|  |

|  |
| --- |
|  |

### Heap

A Heap is a special Tree-based data structure in which the tree is a complete binary tree. Generally, Heaps can be of two types:

1. **Max-Heap:** In a Max-Heap the key present at the root node must be greatest among the keys present at all its children. The same property must be recursively true for all sub-trees in that Binary Tree.
2. **Min-Heap:** In a Min-Heap the key present at the root node must be minimum among the keys present at all its children. The same property must be recursively true for all sub-trees in that Binary Tree.

|  |
| --- |
|  |

# Algorithm

## Recursion

A recursion function is a function that refers to it. Recursion is useful in the task where we have repeated subtask to do.

Anything code in recursion can be done iteratively(loop).

|  |
| --- |
|  |

|  |
| --- |
|  |

## Sorting

Sorting is very important for any company like Google, Facebook, Amazon, Netflix etc.. to sort the large data in very efficient way.

|  |
| --- |
|  |

### Bubble Sort

Don’t use bubble sort it only uses of educational purposes to teach sorting.

### Selection Sort

Don’t use selection sort it only uses of educational purposes to teach sorting.

### Insertion Sort

Insertion sort should be used with only of few items or items are almost sort, then it is fast.

It is efficient when the data is almost sorted.

### Merge Sort

Use divide and conquer method due to that Time Complexity is O(nlogn) but the space complexity is O(n).

Worried of worst case scenario then use merge sort.

### Quick Sort

Unlike Merge Sort , Quick sort also uses divide and conquer technique which gives the time complexity of O(n logn) and also space is O(logn)