Searching a number in an input

* Linear Search – Normal search starting from 1st then 2nd then 3rd…
* Binary Search – starts from the middle , divides the input into 2 parts , step 1 again in the correct part

DATA STRUCTURES

* Trees
* Stacks and Queues
* Linked Lists
* Hash Tables

<https://www.youtube.com/watch?v=h2d9b_nEzoA>

* Data Structures: [Topcoder tutorial](https://www.topcoder.com/community/data-science/data-science-tutorials/data-structures/" \t "_blank)

ALGORITHMS

* [Youtube playlist: MIT 6.006 Introduction to Algorithms, Fall 2011](https://www.youtube.com/playlist?list=PLUl4u3cNGP61Oq3tWYp6V_F-5jb5L2iHb)
* [Coursera: Princeton Algorithms course](https://www.coursera.org/course/algs4partI)
* [Coursera: Stanford Algorithms course](https://www.coursera.org/course/algo)
* <https://www.quora.com/Which-are-the-10-algorithms-every-computer-science-student-must-implement-at-least-once-in-life>
* <https://en.wikipedia.org/wiki/List_of_algorithms>
* <http://www.programcreek.com/2012/11/top-10-algorithms-for-coding-interview/>
* <http://blog.hackerearth.com/2015/05/top-7-algorithms-and-data-structures-every-programmer-should-know-about.html>
* <https://www.youtube.com/watch?v=-8T8AXxVFZg&feature=youtu.be>
* <https://www.youtube.com/watch?v=jq5M5Gmf0do&feature=youtu.be>
* Greedy algorithms: [Quora: What is an intuitive explanation of greedy algorithms?](https://www.quora.com/What-is-an-intuitive-explanation-of-greedy-algorithms" \t "_blank)
* Backtracking: [GeeksforGeeks: The Knight's tour problem](http://www.geeksforgeeks.org/backtracking-set-1-the-knights-tour-problem/" \t "_blank), [Sada Kurapati: N Queens problem](https://sadakurapati.wordpress.com/2013/12/10/n-queens-backtracking-algorithm/" \t "_blank)
* Dynamic programming: [Function Space: Fibonacci series and Dynamic programming](http://functionspace.com/articles/32/Fibonacci-series-and-Dynamic-programming) (And I really like [Jonathan Paulson's answer here](https://www.quora.com/How-should-I-explain-dynamic-programming-to-a-4-year-old).)
* Graph theory and some algorithms: [Computer Science Source: Depth/Breadth First Search](https://computersciencesource.wordpress.com/2010/05/18/algorithms-depthbreadth-first-search/) and [Youtube video: Dijkstra's algorithm](https://www.youtube.com/watch?v=gdmfOwyQlcI" \t "_blank) were the first for me.

**Week 1:** Introduction.  Asymptotic analysis including big-oh notation.  Divide-and-conquer algorithms for sorting, counting inversions, matrix multiplication, and closest pair.  
  
**Week 2:**Running time analysis of divide-and-conquer algorithms.  The master method.  Introduction to randomized algorithms, with a probability review.  QuickSort.    
  
**Week 3:**More on randomized algorithms and probability.  Computing the median in linear time.  A randomized algorithm for the minimum graph cut problem.  
  
**Week 4:** Graph primitives.  Depth- and breadth-first search.  Connected components in undirected graphs.  Topological sort in directed acyclic graphs.  Strongly connected components in directed graphs.  
  
**Week 5:** Dijkstra's shortest-path algorithm.  Introduction to data structures.  Heaps and applications.  
  
**Week 6:** Further data structures.  Hash tables and applications.  Balanced binary search trees.

**1: Sorting Algorithms**  
Sorting Algorithm - Bubble Sort ([Sorting Algorithm - Bubble Sort](http://www.ideserve.co.in/learn/bubble-sort))   
Sorting Algorithm - Selection Sort ([Sorting Algorithm - Selection Sort](http://www.ideserve.co.in/learn/selection-sort))   
Sorting Algorithm - Insertion Sort ([Sorting Algorithm - Insertion Sort](http://www.ideserve.co.in/learn/insertion-sort))    
Sorting Algorithm - Heap Sort ([Sorting Algorithm - Heap Sort](http://www.ideserve.co.in/learn/heap-sort))   
Sorting Algorithm - Merge Sort ([Sorting Algorithm - Merge Sort](http://www.ideserve.co.in/learn/merge-sort))  
Pancake Sorting ([Pancake Sorting](http://www.ideserve.co.in/learn/pancake-sorting))

* Sorting algorithms: [Wikipedia list of sorting algorithms](https://en.wikipedia.org/wiki/Sorting_algorithm#Comparison_of_algorithms) (They usually teach a few of them like these in this order: Bubble sort, Insertion sort, Merge sort, Heapsort, Quicksort and Bucket sort, a bit different. Look at visualizations too, like at [sorting.at](http://sorting.at/) they are cool.)

**2: Algorithms on Linked Lists**  
Reverse a Linked List - Iterative ([Reverse a Linked List - Iterative](http://www.ideserve.co.in/learn/reverse-a-linked-list-iterative))   
Reverse a Linked List - Recursive ([Reverse a Linked List - Recursive](http://www.ideserve.co.in/learn/reverse-a-linked-list-recursive))   
Merge two sorted linked lists ([Merge two sorted linked lists](http://www.ideserve.co.in/learn/merge-two-sorted-linked-lists))   
Find intersection of two Linked Lists ([Find intersection of two Linked Lists](http://www.ideserve.co.in/learn/find-intersection-of-two-linked-lists))   
Find intersection of two Linked Lists - O(m + n) Time Complexity and O(1) Space Complexity (

**3: Algorithms on  Arrays**  
Count frequencies of array elements in range 1 to n ([Count frequencies of array elements in range 1 to n](http://www.ideserve.co.in/learn/count-frequencies-of-array-elements))   
Find all permutations of a String ([Find all permutations of a String](http://www.ideserve.co.in/learn/all-permutations-of-a-string))   
Binary Search in a Sorted Array ([Binary Search in a Sorted Array](http://www.ideserve.co.in/learn/binary-search-in-a-sorted-array))   
Leaders in an array ([Leaders in an array](http://www.ideserve.co.in/learn/leaders-in-an-array))   
Find a Peak Element in an array ([Find a Peak Element in an array](http://www.ideserve.co.in/learn/find-a-peak-element-in-an-array))   
Find pivot in a sorted rotated array ([Find pivot in a sorted rotated array](http://www.ideserve.co.in/learn/find-pivot-in-a-sorted-rotated-array))   
Find an element in a sorted rotated array ([Find an element in a sorted rotated array](http://www.ideserve.co.in/learn/find-an-element-in-a-sorted-rotated-array))   
Find element in sorted rotated array without finding pivot ([Find element in sorted rotated array without finding pivot](http://www.ideserve.co.in/learn/find-an-element-in-a-sorted-rotated-array-without-finding-pivot))   
Find duplicates in an integer array ([Find duplicates in an integer array](http://www.ideserve.co.in/learn/find-duplicates-in-an-array))  
Maximum average subarray ([Maximum average subarray](http://www.ideserve.co.in/learn/maximum-average-subarray))   
Maximum subarray sum ([Maximum subarray sum](http://www.ideserve.co.in/learn/maximum-subarray-sum))   
Next greater element in an array ([Next greater element in an array](http://www.ideserve.co.in/learn/next-great-element-in-an-array))   
Fibonacci Number ([Fibonacci Number](http://www.ideserve.co.in/learn/nth-fibonacci-number))   
Rotate an Array ([Rotate an Array](http://www.ideserve.co.in/learn/rotate-an-array))    
Find Majority Element in an Array ([Find Majority Element in an Array](http://www.ideserve.co.in/learn/find-majority-element-in-an-array))   
Find median of two sorted arrays ([Find median of two sorted arrays](http://www.ideserve.co.in/learn/find-median-of-two-sorted-arrays))   
First non-repeating character in a string ([First non-repeating character in a string](http://www.ideserve.co.in/learn/first-non-repeating-character-in-a-string))  
Re-arrange elements in an array to put positive and negative elements in alternate order ([Re-arrange elements in an array to put positive and negative elements in alternate order](http://www.ideserve.co.in/learn/re-arrange-elements-to-put-positive-negative-elements-in-alternate-order))  
Find the next greater number using same digits ([Find the next greater number using same digits](http://www.ideserve.co.in/learn/next-greater-number-using-same-digits))   
Longest Substring with non-Repeating Characters ([Longest Substr...](http://www.ideserve.co.in/learn/longest-substring-with-non-repeating-characters) (more)