* Sorting algorithms
  + SBI (selection, bubble, insertion)
  + Merge sort
  + Heap sort (learn this later when you understand graphs/trees)
  + Quick sort (mainly, hoare’s partition scheme)
    - Use quick sort everywhere, it worked well for all algorithms.
* Searching algorithms
  + linear
  + binary
    - lower bound
    - upper bound
* String searching
  + Basics
    - string
    - string reverse
    - palindrome
    - substring
    - prefix
    - suffix
    - border
  + Searching for pattern
    - Naive - compare each char of Pattern (P) to each char of text (T)
    - KMP
    - Boyer Moore
    - Others exist but just go thru them
    - All understand the pattern P to search more, in a string T (text), so it is easy to skip some comparisions that are already made.
  + Suffix array - simply, indexes of sorted suffixes of a string
  + LCP array - 1st entry is undefined, other entries are longest common prefix lengths of (1st and 0th index of Suffix array of a string, & so on)
* Greedy
  + Take the 1st instinct, for now whatever is best, take it.
  + (Distra’s algorithm is good example, but will come in Graph Theory)
* Graph Theory
  + Adjacency matrix - simple 2d array - easy but requires a lot of memory
  + Adjacency list - array of (varying length) arrays - saves a lot of memory
  + BFS
  + DFS
    - Pre-order - node, left, then right
    - Post-order - left, right then node
    - In-order - left, node, then right
  + Dijkstra - cool algorithm
  + Floyd Warshal - beautiful algorithm- N power N is now N power 3
  + Bellman Ford - ((forgot!))
  + Kruskal's and Prim's - traverse all nodes of graph once, i.e. minimum spanning tree
* Dynamic programming
  + DONT take the 1st instinct, whatever looks best imm. may not be best when considered full problem.
  + Plain recursion
    - Recursion + Memoization
  + DP
    - + Memoization always
    - 2D Memoization
    - 3D Memoization
  + Learn the different types:
    - Maximization
    - Minimization
    - Number of ways. Generally has % some prime number (like 10 power 9 + 7)
    - Modify the problem, to get to one of the above.
  + See 100 videos on DP
    - Because, though the above will help in DP, the solving part is still tricky.
* Bit manipulation
  + bitwise AND &
  + bitwise OR |
  + bitwise XOR ^
    - This is most used :-), highly useful but torturous too, that is, not much good information on internet.
  + Binary index tree (BIT) / Fenwick tree - Another beautiful tree (actually array) - Store only left subtree’s value (say, sum) at a node - Easy to modify, etc.
* More as I remember, or work on.