

## **Day $-\infty$ to 0:**

Stick to a programming language like C or C++. Make sure that you are comfortable with pointers/objects.

**Day 1:** Understand the concept of [Algorithmic complexity](#). Skip the theory for now, but for every piece of code you write, you should be able to derive both time and space complexity.

**Day 2 - 10:** Let's start with some simple data structures,

1. *Arrays*
2. *Linked Lists*
3. *Strings*
4. *Stacks*
5. *Queues*

Understand their basic operations (*insert, delete, search, traversal*) and their complexity - [Big-O Algorithm Complexity Cheat Sheet](#), and code them all.

**Day 11 - 25:** Let's now learn some simple algorithms,

1. *Sorting* - [Insertion sort](#), [Merge sort](#), [Quick sort](#), [Heap sort](#), [Bucket sort](#), [Counting sort](#), [Radix sort](#), [External sorting](#)
2. *Search* - [Linear search](#), [Binary Search](#) (along with its variants).
3. *Prime Numbers* - [Sieve of Eratosthenes](#), [Primality test](#)
4. *Strings* - [String searching](#), [LCS](#), [Palindrome detection](#)
5. *Miscellaneous* - [Euclidean algorithm](#), [Matrix multiplication](#), [Fibonacci Numbers](#), [Pascal's Triangle](#), [Max Subarray problem](#)

**Day 26 - 50:** Once you are comfortable with everything above, start doing problems from,

1. [Cracking the Coding Interview](#)
2. [Elements of Programming Interviews](#)
3. [Programming Interviews Exposed: Secrets to Landing Your Next Job](#)
4. [GeeksforGeeks](#)
5. [HackerRank](#)
6. [InterviewBit](#)

Stick to chapters of arrays, linked lists, strings, stacks, queues and complexity.

**Day 51 - 60:** Let's learn some non-linear data structures,

*1. Tree*

1. *Binary Tree, Binary Search Tree* - [Tree traversals](#), [Lowest common ancestor](#), [Depth, Height & Diameter](#), [Finding k-th smallest element](#)

*2. Heaps*

2. *Hash table* - [4 sum problem](#), [Checking if sudoku solution is valid](#)
3. *Graph* - [Breadth-first search](#), [Depth-first search](#), [Topological sorting](#), [Minimum spanning tree](#), [Shortest path problem](#),

**Day 61- 90:** Refer to the previous resources and start doing problems from trees, hash tables, heaps and graphs.

**Day 91 - 100:** Understand [Computational complexity theory](#) and [NP-completeness](#), [Knapsack problem](#), [Travelling salesman problem](#), [SAT problem](#) and so on.

**Day 101 -  $\infty$**

: You are now better than most of the CS undergrads. Keep revising the above topics and start competitive programming! Good luck!

Thanks for the A2A [Meghna Bhasin](#)