**=> Common techniques when Array given:**

1. **Sliding window using 2 pointers,**purely 2 pointers etc.
2. **Prefix Sum , Hashing**(remember string hashing too) **and PreComputation.**
3. **Sorting** or manipulating the array in any form.

Using Set, Multiset, Ordered Set(can also be done using merge sort (eg:find inversions)),sorting algo procedures.

1. Using Data structures : stack,queue,**heap,**deque etc

2 stacks, 2 heaps , combination of data structures etc

1. Working with **Indices** rather than values.

-> **Advanced techniques for Arrays:**

1. **DP ,** ((imp)Can also imagine as either DFS or BFS) , sometimes **matrix exponentiation is used.**
2. (**imp**)Binary search on the answer.
3. Graphs (BFS(single or multi src), DFS, DSU).
4. Bit Manipulation
5. Segment Tree for Range Queries.

**=> Strings :**

-> String **Hashing**.(permutation hash, anagram hash).

1. String **Reversal & Rotation.**
2. String **Duplication**. aba -> aba aba
3. String **Matching** Techniques.(Rabin Karp, Z-array etc)
4. **(imp)Hashing** for each of 26 characters.
5. **Trie** approach.

**=> Helpful Functions :**

1. s.substr(start\_index, len);
2. next\_permution**[**start\_iter, end\_iter**)**, same way prev\_permutation

Time = O(n) in worst case , O(1) amortized

**=> TREES :**

**For binary tree:**

1. Checking in the **Subtree** rooted at left and subtree rooted at right.
2. Calculating **DEPTH** of tree , and using Depth of individual nodes.
3. All **traversal** (including lvl order,vertical order) and their reverse.
4. **DP on Trees.**
5. Indexing the tree, when go left , index of par + 1, else -1, or depending on question.

**For general m-ary tree :**

1. Bfs and dfs.
2. Recording Start and end dfs times.
3. LCA

**Misc :**

1. Euler Tour