Data Structures and Algorithms for Online Programming Contest

1. Sorting

i. [Merge Sort](algorithms/Merge_Sort.cpp)

ii. [Quick Sort](algorithms/Quick_Sort.cpp)

iii. [Heap Sort](algorithms/heap_sort.cpp)

iv. [Bubble Sort](algorithms/Bubble_Sort.cpp)

v. [Insertion Sort](algorithms/Insertion_Sort.cpp)

vi [Selection Sort](algorithms/Selection_Sort.cpp)

vii. [Bucket Sort](algorithms/Bucket_Sort.cpp)

viii. [Count Sort](algorithms/Count_Sort.cpp)

ix. [Radix Sort](algorithms/Radix_Sort.cpp)

x. [Pancake Sort](algorithms/Pancake_Sort.cpp)

2. Data Structure

i. [Singly Linked List](algorithms/singly_linked_list.cpp)

ii. [Doubly Linked List](algorithms/doubly_linked_list.cpp)

iii. [Vector](algorithms/vector.cpp)

iv. [Stack](algorithms/stack.cpp)

v. [Queue](algorithms/queue.cpp)

vi. [List](algorithms/list.cpp)

vii. [Hashtable](algorithms/hashtable.cpp)

viii. [HashMap](algorithms/hashmap.cpp)

ix. [HashSet](algorithms/hashset.cpp)

x. [Union Find(Disjoint Set)](algorithms/union_find.cpp)

xi. [Binary Search Tree](algorithms/BST.cpp)

xii. [Segment Tree](algorithms/segment_tree.cpp)

xiii. [Segment Tree (Lazy Propagation)](algorithms/segment_tree_lazy.cpp)

xiv. [2D Segment Tree (Quad tree)](algorithms/2D_segment_tree.cpp)

xv. [Binary Indexed Tree](algorithms/BIT.cpp)

xvi. [2D Binary Indexed Tree](algorithms/2D_BIT.cpp)

xvii. [(AVL Tree) Self Balanced BST](algorithms/AVL.cpp)

xviii. [(Splay Tree) Self Balanced BST](algorithms/splay.cpp)

xix. [Ternary Search Tree](algorithms/ternary_search_tree.cpp)

xx. [Heap (Min)](algorithms/heap.cpp)

3. Binary Search

i. [Binary Search](algorithms/binary_search.cpp)

ii. [Lower Bound](algorithms/binary_search.cpp)

iii. [Upper Bound](algorithms/binary_search.cpp)

iv. [Equal Range](algorithms/binary_search.cpp)

4. Graph Theory

i. [Floyd Warshall's](algorithms/floyd_warshall.cpp)

ii. [Loop Detection](algorithms/topsort.cpp)

iii. [Topological Sort](algorithms/topsort.cpp)

iv. [Strongly Connected Component (Kosaraju)](algorithms/scc.cpp)

v. [Lowest Common Ancestor(sparse table)](algorithms/lca.cpp)

vi. [Articulation Point](algorithms/articulation_point.cpp)

vii. [Bridge](algorithms/bridge.cpp)

viii. [Breadth First Search](algorithms/bfs.cpp)

ix. [Dijkstra](algorithms/dijkstra.cpp)

x. [Bellman Ford's](algorithms/bellman_fords.cpp)

xi. [Kruskal Minimum Spanning Tree](algorithms/kruskal.cpp)

xii. [Minimum Vertex Cover](algorithms/min_vertex_cover.cpp)

xiii. [Maximum Flow (Edmonds Karp's) I](algorithms/max_flow.cpp)

xiv. [Maximum Flow (Edmonds Karp's) II](algorithms/Maximum_Flow_Problem_I_Edmond_Karp.cpp)

xv. [Maximum Bipartite Matching](algorithms/maximum_bipartite_matching.cpp)

xvi. [Stable Marriage Problem](algorithms/stable_marriage_problem.cpp)

xvii. [Heavy Light Decomposition](algorithms/HLD.cpp)

5. Greedy Algorithm

i. [Huffman Coding](algorithms/Huffman_coding.cpp)

6. Dynamic Programming

i. [Coin Change and variants](algorithms/coin_change.cpp)

ii. [Knapsack Problem and variants](algorithms/knapsack.cpp)

iii. [Matrix Chain Multiplication](algorithms/mcm.cpp)

iv. [Longest Increasing Subsequence(O(n^2))](algorithms/lis.cpp)

v. [Longest Increasing Subsequence(O(nlogn))](algorithms/lis.cpp)

vi. [Travelling Salesman Problem](algorithms/tsp.cpp)

vii. [Maximum Sum Subarray(O(n^4) and O(n^3))](algorithms/maximum_sum_subarray.cpp)

viii. [Kadane Algorithm](algorithms/kadane.cpp)

ix. [Maximum Sum Subarray using Kadane(O(n^3))](algorithms/kadane.cpp)

x. [Optimal Binary Search Tree](algorithms/optimal_search_tree.cpp)

xi. [Subset Sum](algorithms/subset_sum.cpp)

xii. [Catalan Number](algorithms/CatalanNumber.cpp)

xiii. [DAG Minimum Path](algorithms/DAG_min_path.cpp)

xiv. [Minimum Cost Path](algorithms/min_cost_path.cpp)

xv. [Digit Dp I](algorithms/Digit_dp_I.cpp)

xvi. [Digit Dp II](algorithms/Digit_dp_II.cpp)

xvii. [Digit Dp III](algorithms/Digit_dp_III.cpp)

xviii. [Digit Dp IV](algorithms/Digit_dp_IV.cpp)

7. Game Theory

i. [Game Tree(Memorization)](algorithms/game_tree.cpp)

ii. [Nim](algorithms/nim.cpp)

iii. [Misère Nim](algorithms/nim.cpp)

iv. [Nimble Nim](algorithms/nim.cpp)

v. [Poker Nim](algorithms/nim.cpp)

vi. [Prime Power Nim](algorithms/nim.cpp)

vii. [Spagrue Grundy Problem](algorithms/grundy.cpp)

viii. [Grundy Variant: Zero Nim Game](algorithms/grundy.cpp)

ix. [Grundy Variant: Coins on Chessboard](algorithms/grundy.cpp)

x. [Green HackenBush(Colon Principle)](algorithms/hackenbush.cpp)

8. Backtracking

i. [Permutation Generator](algorithms/permutation_generator.cpp)

ii. [N-Queen](algorithms/nqueen.cpp)

iii. [Prime Ring](algorithms/prime_ring.cpp)

9. Hashing

i. [Double Hashing](algorithms/double_hashing.cpp)

ii. [String Hashing by Map](algorithms/map_hashing.cpp)

iii. [Berenstain String Hashing](algorithms/berenstain_hashing.cpp)

iv. [Rolling Hash](algorithms/rabin_karp.cpp)

10. Combinotorics

i. [Factorial](algorithms/factorial.cpp)

ii. [nCr](algorithms/ncr.cpp)

iii. [De-arrangement](algorithms/dearrangement.cpp)

11. String Algorithm

i. [Aho-Corasick Algorithm](algorithms/Aho_Corasick.cpp)

ii. [Knuth-Morris-Pratt's Algorithm](algorithms/kmp.cpp)

iii. [Rabin Karp Pattern Searching](algorithms/rabin_karp.cpp)

iv. [Z Algorithm](algorithms/z.cpp)

v. [Finite Automata Pattern Searching](algorithms/Finite_Automata_Pattern_Searching.cpp)

vi. [Trie (Prefix/Radix Tree)](algorithms/Trie.cpp)

vii. [Longest Common Subsequence](algorithms/lcs.cpp)

viii. [Edit Distance](algorithms/edit_distance.cpp)

ix. [Longest Palindromic Subsequence](algorithms/lps.cpp)

x. [Suffix Array](algorithms/suffix_lcp.cpp)

xi. [Longest Common Prefix](algorithms/suffix_lcp.cpp)

xii. [Minimum Expression](algorithms/Minimum_expression.cpp)

xiii. [Suffix Automata](algorithms/suffix_automata.cpp)

12. Mathematics

i. [Power Function(Big mod)](algorithms/Power.cpp)

ii. [Modular Mutiplicative Inverse(using Big mod)](algorithms/Power.cpp)

iii. [Prime(Sieve of Erathonesis)](algorithms/prime.cpp)

iv. [Segmented Sieve of Erathonesis](algorithms/prime.cpp)

v. [Prime factorization(using Sieve)](algorithms/prime.cpp)

vi. [Prime factorization](algorithms/prime.cpp)

vii. [Primality Test(School method)](algorithms/primality_test.cpp)

viii. [Miller–Rabin Primality Test](algorithms/primality_test.cpp)

ix. [Euler Totient (Phi Function)](algorithms/phi.cpp)

x. [Extended Euclid](algorithms/extended_euclid.cpp)

xi. [Linear Diophatine Equation](algorithms/extended_euclid.cpp)

xii. [Modular Mutiplicative Inverse(using Extended Euclid)](algorithms/extended_euclid.cpp)

xiii. [Matrix Exponentiation](algorithms/matrix_exp.xpp)

xiv. [Floyd Cycle Finding Algorithm](algorithms/floyd_cycle_finding.cpp)

xv. [Big Integer](algorithms/big_integer.cpp)

xvi. [Josephus Recurrence](algorithms/Josephus_Recurrence.cpp)

xvii. [Fast Fourier Transform](algorithms/FFT.cpp)

13. Computational Geometry

i. [Computational Geometry Template](algorithms/computational_geometry_template.cpp)

ii. [Convex Hull (Jarvis's Algorithm or Wrapping)](algorithms/convex_hull_jarvis.cpp)

iii. [Convex Hull (Graham Scan)](algorithms/convex_hull_graham_scan.cpp)

14. Miscellaneous

i. [Template](template.cpp)