# Algorithms for Competitive Programming

### Algebra

* **Fundamentals**
  + [Binary Exponentiation](https://cp-algorithms.com/algebra/binary-exp.html)
  + [Euclidean algorithm for computing the greatest common divisor](https://cp-algorithms.com/algebra/euclid-algorithm.html)
  + [Extended Euclidean Algorithm](https://cp-algorithms.com/algebra/extended-euclid-algorithm.html)
  + [Linear Diophantine Equations](https://cp-algorithms.com/algebra/linear-diophantine-equation.html)
  + [Fibonacci Numbers](https://cp-algorithms.com/algebra/fibonacci-numbers.html)
* **Prime numbers**
  + [Sieve of Eratosthenes](https://cp-algorithms.com/algebra/sieve-of-eratosthenes.html)
  + [Linear Sieve](https://cp-algorithms.com/algebra/prime-sieve-linear.html)
  + [Primality tests](https://cp-algorithms.com/algebra/primality_tests.html)
  + [Integer factorization](https://cp-algorithms.com/algebra/factorization.html)
* **Number-theoretic functions**
  + [Euler's totient function](https://cp-algorithms.com/algebra/phi-function.html)
  + [Number of divisors / sum of divisors](https://cp-algorithms.com/algebra/divisors.html)
* **Modular arithmetic**
  + [Modular Inverse](https://cp-algorithms.com/algebra/module-inverse.html)
  + [Linear Congruence Equation](https://cp-algorithms.com/algebra/linear_congruence_equation.html)
  + [Chinese Remainder Theorem](https://cp-algorithms.com/algebra/chinese-remainder-theorem.html)
  + [Factorial modulo](https://cp-algorithms.com/algebra/factorial-modulo.html)
  + [p](https://cp-algorithms.com/algebra/factorial-modulo.html)
  + [Discrete Log](https://cp-algorithms.com/algebra/discrete-log.html)
  + [Primitive Root](https://cp-algorithms.com/algebra/primitive-root.html)
  + [Discrete Root](https://cp-algorithms.com/algebra/discrete-root.html)
  + [Montgomery Multiplication](https://cp-algorithms.com/algebra/montgomery_multiplication.html)
* **Number systems**
  + [Balanced Ternary](https://cp-algorithms.com/algebra/balanced-ternary.html)
  + [Gray code](https://cp-algorithms.com/algebra/gray-code.html)
* **Miscellaneous**
  + [Enumerating submasks of a bitmask](https://cp-algorithms.com/algebra/all-submasks.html)
  + [Arbitrary-Precision Arithmetic](https://cp-algorithms.com/algebra/big-integer.html)
  + [Fast Fourier transform](https://cp-algorithms.com/algebra/fft.html)
  + [Operations on polynomials and series](https://cp-algorithms.com/algebra/polynomial.html)
  + [Continued fractions](https://cp-algorithms.com/algebra/continued-fractions.html)

### Data Structures

* **Fundamentals**
  + [Minimum Stack / Minimum Queue](https://cp-algorithms.com/data_structures/stack_queue_modification.html)
  + [Sparse Table](https://cp-algorithms.com/data_structures/sparse-table.html)
* **Trees**
  + [Disjoint Set Union](https://cp-algorithms.com/data_structures/disjoint_set_union.html)
  + [Fenwick Tree](https://cp-algorithms.com/data_structures/fenwick.html)
  + [Sqrt Decomposition](https://cp-algorithms.com/data_structures/sqrt_decomposition.html)
  + [Segment Tree](https://cp-algorithms.com/data_structures/segment_tree.html)
  + [Treap](https://cp-algorithms.com/data_structures/treap.html)
  + [Sqrt Tree](https://cp-algorithms.com/data_structures/sqrt-tree.html)
  + [Randomized Heap](https://cp-algorithms.com/data_structures/randomized_heap.html)
* **Advanced**
  + [Deleting from a data structure in O(T(n)log n)](https://cp-algorithms.com/data_structures/deleting_in_log_n.html)

### Dynamic Programming

* **DP optimizations**
  + [Divide and Conquer DP](https://cp-algorithms.com/dynamic_programming/divide-and-conquer-dp.html)
* **Tasks**
  + [Dynamic Programming on Broken Profile. Problem "Parquet"](https://cp-algorithms.com/dynamic_programming/profile-dynamics.html)
  + [Finding the largest zero submatrix](https://cp-algorithms.com/dynamic_programming/zero_matrix.html)

### String Processing

* **Fundamentals**
  + [String Hashing](https://cp-algorithms.com/string/string-hashing.html)
  + [Rabin-Karp for String Matching](https://cp-algorithms.com/string/rabin-karp.html)
  + [Prefix function - Knuth-Morris-Pratt](https://cp-algorithms.com/string/prefix-function.html)
  + [Z-function](https://cp-algorithms.com/string/z-function.html)
  + [Suffix Array](https://cp-algorithms.com/string/suffix-array.html)
  + [Aho-Corasick algorithm](https://cp-algorithms.com/string/aho_corasick.html)
* **Advanced**
  + [Suffix Tree](https://cp-algorithms.com/string/suffix-tree-ukkonen.html)
  + [Suffix Automaton](https://cp-algorithms.com/string/suffix-automaton.html)
  + [Lyndon factorization](https://cp-algorithms.com/string/lyndon_factorization.html)
* **Tasks**
  + [Expression parsing](https://cp-algorithms.com/string/expression_parsing.html)
  + [Manacher's Algorithm - Finding all sub-palindromes in O(N)](https://cp-algorithms.com/string/manacher.html)
  + [Finding repetitions](https://cp-algorithms.com/string/main_lorentz.html)

### Linear Algebra

* **Matrices**
  + [Gauss & System of Linear Equations](https://cp-algorithms.com/linear_algebra/linear-system-gauss.html)
  + [Gauss & Determinant](https://cp-algorithms.com/linear_algebra/determinant-gauss.html)
  + [Kraut & Determinant](https://cp-algorithms.com/linear_algebra/determinant-kraut.html)
  + [Rank of a matrix](https://cp-algorithms.com/linear_algebra/rank-matrix.html)

### Combinatorics

* **Fundamentals**
  + [Finding Power of Factorial Divisor](https://cp-algorithms.com/algebra/factorial-divisors.html)
  + [Binomial Coefficients](https://cp-algorithms.com/combinatorics/binomial-coefficients.html)
  + [Catalan Numbers](https://cp-algorithms.com/combinatorics/catalan-numbers.html)
* **Techniques**
  + [The Inclusion-Exclusion Principle](https://cp-algorithms.com/combinatorics/inclusion-exclusion.html)
  + [Burnside's lemma / Pólya enumeration theorem](https://cp-algorithms.com/combinatorics/burnside.html)
  + [Stars and bars](https://cp-algorithms.com/combinatorics/stars_and_bars.html)
  + [Generating all](https://cp-algorithms.com/combinatorics/generating_combinations.html)
  + [K](https://cp-algorithms.com/combinatorics/generating_combinations.html)
  + [-combinations](https://cp-algorithms.com/combinatorics/generating_combinations.html)
* **Tasks**
  + [Placing Bishops on a Chessboard](https://cp-algorithms.com/combinatorics/bishops-on-chessboard.html)
  + [Balanced bracket sequences](https://cp-algorithms.com/combinatorics/bracket_sequences.html)
  + [Counting labeled graphs](https://cp-algorithms.com/combinatorics/counting_labeled_graphs.html)

### Numerical Methods

* **Search**
  + [Ternary Search](https://cp-algorithms.com/num_methods/ternary_search.html)
  + [Newton's method for finding roots](https://cp-algorithms.com/num_methods/roots_newton.html)
* **Integration**
  + [Integration by Simpson's formula](https://cp-algorithms.com/num_methods/simpson-integration.html)

### Geometry

* **Elementary operations**
  + [Basic Geometry](https://cp-algorithms.com/geometry/basic-geometry.html)
  + [Finding the equation of a line for a segment](https://cp-algorithms.com/geometry/segment-to-line.html)
  + [Intersection Point of Lines](https://cp-algorithms.com/geometry/lines-intersection.html)
  + [Check if two segments intersect](https://cp-algorithms.com/geometry/check-segments-intersection.html)
  + [Intersection of Segments](https://cp-algorithms.com/geometry/segments-intersection.html)
  + [Circle-Line Intersection](https://cp-algorithms.com/geometry/circle-line-intersection.html)
  + [Circle-Circle Intersection](https://cp-algorithms.com/geometry/circle-circle-intersection.html)
  + [Common tangents to two circles](https://cp-algorithms.com/geometry/tangents-to-two-circles.html)
  + [Length of the union of segments](https://cp-algorithms.com/geometry/length-of-segments-union.html)
* **Polygons**
  + [Oriented area of a triangle](https://cp-algorithms.com/geometry/oriented-triangle-area.html)
  + [Area of simple polygon](https://cp-algorithms.com/geometry/area-of-simple-polygon.html)
  + [Check if points belong to the convex polygon in O(log N)](https://cp-algorithms.com/geometry/point-in-convex-polygon.html)
  + [Minkowski sum of convex polygons](https://cp-algorithms.com/geometry/minkowski.html)
  + [Pick's Theorem - area of lattice polygons](https://cp-algorithms.com/geometry/picks-theorem.html)
  + [Lattice points of non-lattice polygon](https://cp-algorithms.com/geometry/lattice-points.html)
* **Convex hull**
  + [Convex hull construction](https://cp-algorithms.com/geometry/convex-hull.html)
  + [Convex hull trick and Li Chao tree](https://cp-algorithms.com/geometry/convex_hull_trick.html)
* **Sweep-line**
  + [Search for a pair of intersecting segments](https://cp-algorithms.com/geometry/intersecting_segments.html)
  + [Point location in O(log N)](https://cp-algorithms.com/geometry/point-location.html)
* **Miscellaneous**
  + [Finding the nearest pair of points](https://cp-algorithms.com/geometry/nearest_points.html)
  + [Delaunay triangulation and Voronoi diagram](https://cp-algorithms.com/geometry/delaunay.html)
  + [Vertical decomposition](https://cp-algorithms.com/geometry/vertical_decomposition.html)
  + [Half-plane intersection - S&I Algorithm in O(Nlog N)](https://cp-algorithms.com/geometry/halfplane-intersection.html)

### Graphs

* **Graph traversal**
  + [Breadth First Search](https://cp-algorithms.com/graph/breadth-first-search.html)
  + [Depth First Search](https://cp-algorithms.com/graph/depth-first-search.html)
* **Connected components, bridges, articulations points**
  + [Finding Connected Components](https://cp-algorithms.com/graph/search-for-connected-components.html)
  + [Finding Bridges in O(N+M)](https://cp-algorithms.com/graph/bridge-searching.html)
  + [Finding Bridges Online](https://cp-algorithms.com/graph/bridge-searching-online.html)
  + [Finding Articulation Points in O(N+M)](https://cp-algorithms.com/graph/cutpoints.html)
  + [Strongly Connected Components and Condensation Graph](https://cp-algorithms.com/graph/strongly-connected-components.html)
  + [Strong Orientation](https://cp-algorithms.com/graph/strong-orientation.html)
* **Single-source shortest paths**
  + [Dijkstra - finding shortest paths from given vertex](https://cp-algorithms.com/graph/dijkstra.html)
  + [Dijkstra on sparse graphs](https://cp-algorithms.com/graph/dijkstra_sparse.html)
  + [Bellman-Ford - finding shortest paths with negative weights](https://cp-algorithms.com/graph/bellman_ford.html)
  + [0-1 BFS](https://cp-algorithms.com/graph/01_bfs.html)
  + [D´Esopo-Pape algorithm](https://cp-algorithms.com/graph/desopo_pape.html)
* **All-pairs shortest paths**
  + [Floyd-Warshall - finding all shortest paths](https://cp-algorithms.com/graph/all-pair-shortest-path-floyd-warshall.html)
  + [Number of paths of fixed length / Shortest paths of fixed length](https://cp-algorithms.com/graph/fixed_length_paths.html)
* **Spanning trees**
  + [Minimum Spanning Tree - Prim's Algorithm](https://cp-algorithms.com/graph/mst_prim.html)
  + [Minimum Spanning Tree - Kruskal](https://cp-algorithms.com/graph/mst_kruskal.html)
  + [Minimum Spanning Tree - Kruskal with Disjoint Set Union](https://cp-algorithms.com/graph/mst_kruskal_with_dsu.html)
  + [Second best Minimum Spanning Tree - Using Kruskal and Lowest Common Ancestor](https://cp-algorithms.com/graph/second_best_mst.html)
  + [Kirchhoff Theorem](https://cp-algorithms.com/graph/kirchhoff-theorem.html)
  + [Prüfer code](https://cp-algorithms.com/graph/pruefer_code.html)
* **Cycles**
  + [Checking a graph for acyclicity and finding a cycle in O(M)](https://cp-algorithms.com/graph/finding-cycle.html)
  + [Finding a Negative Cycle in the Graph](https://cp-algorithms.com/graph/finding-negative-cycle-in-graph.html)
  + [Eulerian Path](https://cp-algorithms.com/graph/euler_path.html)
* **Lowest common ancestor**
  + [Lowest Common Ancestor](https://cp-algorithms.com/graph/lca.html)
  + [Lowest Common Ancestor - Binary Lifting](https://cp-algorithms.com/graph/lca_binary_lifting.html)
  + [Lowest Common Ancestor - Farach-Colton and Bender algorithm](https://cp-algorithms.com/graph/lca_farachcoltonbender.html)
  + [Solve RMQ by finding LCA](https://cp-algorithms.com/graph/rmq_linear.html)
  + [Lowest Common Ancestor - Tarjan's off-line algorithm](https://cp-algorithms.com/graph/lca_tarjan.html)
* **Flows and related problems**
  + [Maximum flow - Ford-Fulkerson and Edmonds-Karp](https://cp-algorithms.com/graph/edmonds_karp.html)
  + [Maximum flow - Push-relabel algorithm](https://cp-algorithms.com/graph/push-relabel.html)
  + [Maximum flow - Push-relabel algorithm improved](https://cp-algorithms.com/graph/push-relabel-faster.html)
  + [Maximum flow - Dinic's algorithm](https://cp-algorithms.com/graph/dinic.html)
  + [Maximum flow - MPM algorithm](https://cp-algorithms.com/graph/mpm.html)
  + [Flows with demands](https://cp-algorithms.com/graph/flow_with_demands.html)
  + [Minimum-cost flow](https://cp-algorithms.com/graph/min_cost_flow.html)
  + [Assignment problem. Solution using min-cost-flow in O (N^5)](https://cp-algorithms.com/graph/Assignment-problem-min-flow.html)
* **Matchings and related problems**
  + [Bipartite Graph Check](https://cp-algorithms.com/graph/bipartite-check.html)
  + [Kuhn's Algorithm - Maximum Bipartite Matching](https://cp-algorithms.com/graph/kuhn_maximum_bipartite_matching.html)
* **Miscellaneous**
  + [Topological Sorting](https://cp-algorithms.com/graph/topological-sort.html)
  + [Edge connectivity / Vertex connectivity](https://cp-algorithms.com/graph/edge_vertex_connectivity.html)
  + [Tree painting](https://cp-algorithms.com/graph/tree_painting.html)
  + [2-SAT](https://cp-algorithms.com/graph/2SAT.html)
  + [Heavy-light decomposition](https://cp-algorithms.com/graph/hld.html)

### Miscellaneous

* **Sequences**
  + [RMQ task (Range Minimum Query - the smallest element in an interval)](https://cp-algorithms.com/sequences/rmq.html)
  + [Longest increasing subsequence](https://cp-algorithms.com/sequences/longest_increasing_subsequence.html)
  + [Search the subsegment with the maximum/minimum sum](https://cp-algorithms.com/others/maximum_average_segment.html)
  + [K-th order statistic in O(N)](https://cp-algorithms.com/sequences/k-th.html)
* **Game Theory**
  + [Games on arbitrary graphs](https://cp-algorithms.com/game_theory/games_on_graphs.html)
  + [Sprague-Grundy theorem. Nim](https://cp-algorithms.com/game_theory/sprague-grundy-nim.html)
* **Schedules**
  + [Scheduling jobs on one machine](https://cp-algorithms.com/schedules/schedule_one_machine.html)
  + [Scheduling jobs on two machines](https://cp-algorithms.com/schedules/schedule_two_machines.html)
  + [Optimal schedule of jobs given their deadlines and durations](https://cp-algorithms.com/schedules/schedule-with-completion-duration.html)
* **Miscellaneous**
  + [Josephus problem](https://cp-algorithms.com/others/josephus_problem.html)
  + [15 Puzzle Game: Existence Of The Solution](https://cp-algorithms.com/others/15-puzzle.html)
  + [The Stern-Brocot Tree and Farey Sequences](https://cp-algorithms.com/others/stern_brocot_tree_farey_sequences.html)