Algorithms for Competitive Programming

The goal of this project is to translate the wonderful resource http://e-maxx.ru/algo which provides descriptions of many algorithms and data structures especially popular in field of competitive programming. Moreover we want to improve the collected knowledge by extending the articles and adding new articles to the collection.

Changelog

- January 16, 2022: Switched to the MkDocs site generator with the Material for MkDocs theme, which give the website a more modern look, brings a couple of new features (dark mode, better search, ...), makes the website more stable (in terms of rendering math formulas), and makes it easier to contribute.

Full list of updates: Commit History

Articles

Algebra

- **Fundamentals**
  - Binary Exponentiation
  - Euclidean algorithm for computing the greatest common divisor
  - Extended Euclidean Algorithm
  - Linear Diophantine Equations
  - Fibonacci Numbers
- **Prime numbers**
  - Sieve of Eratosthenes
  - Linear Sieve
  - Primality tests
  - Integer factorization
- **Number-theoretic functions**

    - Euler's totient function
    - Number of divisors / sum of divisors
- **Modular arithmetic**
    - Modular Inverse
    - Linear Congruence Equation
    - Chinese Remainder Theorem
    - Factorial modulo p
    - Discrete Log
    - Primitive Root
    - Discrete Root
    - Montgomery Multiplication
- **Number systems**
    - Balanced Ternary
    - Gray code
- **Miscellaneous**
    - Enumerating submasks of a bitmask
    - Arbitrary-Precision Arithmetic
    - Fast Fourier transform
    - Operations on polynomials and series

Data Structures

- **Fundamentals**
    - Minimum Stack / Minimum Queue
    - Sparse Table
- **Trees**
    - Disjoint Set Union
    - Fenwick Tree
    - Sqrt Decomposition
    - Segment Tree
    - Treap

  - Sqrt Tree
  - Randomized Heap
- **Advanced**
  - Deleting from a data structure in O(T(n)log n)

Dynamic Programming

- **DP optimizations**
  - Divide and Conquer DP
- **Tasks**
  - Dynamic Programming on Broken Profile. Problem "Parquet"
  - Finding the largest zero submatrix

String Processing

- **Fundamentals**
  - String Hashing
  - Rabin-Karp for String Matching
  - Prefix function - Knuth-Morris-Pratt
  - Z-function
  - Suffix Array
  - Aho-Corasick algorithm
- **Advanced**
  - Suffix Tree
  - Suffix Automaton
  - Lyndon factorization
- **Tasks**
  - Expression parsing
  - Manacher's Algorithm - Finding all sub-palindromes in O(N)
  - Finding repetitions

Linear Algebra

- **Matrices**
  - Gauss & System of Linear Equations
  - Gauss & Determinant
  - Kraut & Determinant
  - Rank of a matrix

Combinatorics

- **Fundamentals**
  - Finding Power of Factorial Divisor
  - Binomial Coefficients
  - Catalan Numbers
- **Techniques**
  - The Inclusion-Exclusion Principle
  - Burnside's lemma / Pólya enumeration theorem
  - Stars and bars
  - Generating all K-combinations
- **Tasks**
  - Placing Bishops on a Chessboard
  - Balanced bracket sequences
  - Counting labeled graphs

Numerical Methods

- **Search**
  - Ternary Search
  - Newton's method for finding roots
- **Integration**
  - Integration by Simpson's formula

Geometry

- **Elementary operations**
  - Basic Geometry
  - Finding the equation of a line for a segment
  - Intersection Point of Lines
  - Check if two segments intersect
  - Intersection of Segments
  - Circle-Line Intersection
  - Circle-Circle Intersection
  - Common tangents to two circles
  - Length of the union of segments
- **Polygons**
  - Oriented area of a triangle
  - Area of simple polygon
  - Check if points belong to the convex polygon in O(log N)
  - Minkowski sum of convex polygons
  - Pick's Theorem - area of lattice polygons
  - Lattice points of non-lattice polygon
- **Convex hull**
  - Convex hull construction
  - Convex hull trick and Li Chao tree
- **Sweep-line**
  - Search for a pair of intersecting segments
  - Point location in O(log N)
- **Miscellaneous**
  - Finding the nearest pair of points
  - Delaunay triangulation and Voronoi diagram
  - Vertical decomposition
  - Half-plane intersection - S&I Algorithm in O(Nlog N)

Graphs

- **Graph traversal**
  - Breadth First Search
  - Depth First Search
- **Connected components, bridges, articulations points**
  - Finding Connected Components
  - Finding Bridges in O(N+M)
  - Finding Bridges Online
  - Finding Articulation Points in O(N+M)
  - Strongly Connected Components and Condensation Graph
  - Strong Orientation
- **Single-source shortest paths**
  - Dijkstra - finding shortest paths from given vertex
  - Dijkstra on sparse graphs
  - Bellman-Ford - finding shortest paths with negative weights
  - 0-1 BFS
  - D´Esopo-Pape algorithm
- **All-pairs shortest paths**
  - Floyd-Warshall - finding all shortest paths
  - Number of paths of fixed length / Shortest paths of fixed length
- **Spanning trees**
  - Minimum Spanning Tree - Prim's Algorithm
  - Minimum Spanning Tree - Kruskal
  - Minimum Spanning Tree - Kruskal with Disjoint Set Union
  - Second best Minimum Spanning Tree - Using Kruskal and Lowest Common Ancestor
  - Kirchhoff Theorem
  - Prüfer code
- **Cycles**
  - Checking a graph for acyclicity and finding a cycle in O(M)
  - Finding a Negative Cycle in the Graph

- Eulerian Path

- **Lowest common ancestor**
  - Lowest Common Ancestor
  - Lowest Common Ancestor - Binary Lifting
  - Lowest Common Ancestor - Farach-Colton and Bender algorithm
  - Solve RMQ by finding LCA
  - Lowest Common Ancestor - Tarjan's off-line algorithm
- **Flows and related problems**
  - Maximum flow - Ford-Fulkerson and Edmonds-Karp
  - Maximum flow - Push-relabel algorithm
  - Maximum flow - Push-relabel algorithm improved
  - Maximum flow - Dinic's algorithm
  - Maximum flow - MPM algorithm
  - Flows with demands
  - Minimum-cost flow
  - Assignment problem. Solution using min-cost-flow in O (N^5)
- **Matchings and related problems**
  - Bipartite Graph Check
  - Kuhn' Algorithm - Maximum Bipartite Matching
- **Miscellaneous**
  - Topological Sorting
  - Edge connectivity / Vertex connectivity
  - Tree painting
  - 2-SAT
  - Heavy-light decomposition

Miscellaneous

- **Sequences**
  - RMQ task (Range Minimum Query - the smallest element in an interval)
  - Longest increasing subsequence

Contributing